FSPO: FAST ADVERSARIAL TRAINING AGAINST JAILBREAK ATTACKS VIA SOFT PROMPT OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

As large language models (LLMs) have achieved remarkable performance and are increasingly deployed across diverse domains, ensuring their reliability and safety has become critically important. Among various threats, jailbreak attacks, which manipulate model to elicit unsafe or prohibited outputs, have become a major challenge. Although various defensive frameworks have been proposed, most existing defenses are computationally expensive. Moreover, since current defenses do not explicitly account for adaptive attacks, they are easily circumvented by strong adaptive attacks. To address these issues, we propose *Fast Soft Prompt Optimization*, which effectively enhances robustness by relaxing discrete prompt into a continuous probability simplex, called a soft prompt, and adaptively optimizing it. The resulting soft prompt can be directly deployed as a continuous prefix within the model prior to release. We evaluate our methods under two information-rich settings: model-aware but defense-unaware static attacks, and model-and-defense-aware adaptive attacks. We further analyze attention shift mechanisms to provide insights into the underlying robustness improvements.

1 INTRODUCTION

Large language models (LLMs) trained with alignment techniques such as reinforcement learning from human feedback (RLHF) (Bai et al., 2022a; Ouyang et al., 2022; Christiano et al., 2017), DPO (Rafailov et al., 2023), and Constitutional AI (Bai et al., 2022b) are widely deployed with the expectation of safety. To achieve safety, developers stack multiple defense layers prior to release (Touvron et al., 2023; Hurst et al., 2024; Singh et al., 2025), such as alignment training, prompt engineering, and inference-time filters (Robey et al., 2023; Li et al., 2025), since no single mechanism provides complete protection. However, adversarially crafted prompts can manipulate LLMs to elicit harmful outputs (Carlini et al., 2023; Zou et al., 2023; Anwar et al., 2024). These so-called jailbreak attacks bypass safety guardrails and raise serious concerns for the LLM deployments.

To address this problem, jailbreak defenses are being investigated through several approaches. Importantly, current jailbreak defenses fall into three orthogonal categories: (i) parameter-level defenses (Zou et al., 2024a; Lu et al., 2025), (ii) prompt-level defenses (Zhou et al., 2024; Mo et al., 2024), and (iii) detection-level defenses (Robey et al., 2023; Li et al., 2025). These categories operate on different axes and are composable: developers can combine any subset of them before releasing LLMs. Our work targets the **prompt-level** category through continuous optimization.

Within the parameter-level category, one prominent direction is Adversarial Training (AT) (Xhonneux et al., 2024; Yu et al., 2024), which minimizes the loss against worst-case perturbations by updating parameters on adversarially perturbed inputs, as in computer vision (Goodfellow et al., 2014; Madry et al., 2017). Extending AT to LLMs, however, is prohibitively expensive and inherits a robustness–performance trade-off (Zhang et al., 2019; Tsipras et al., 2018; Chen et al., 2025), motivating parameter-efficient variants such as LoRA (Zou et al., 2024a). A complementary direction avoids parameter updates: building on prompt-tuning (Lester et al., 2021; Li & Liang, 2021), defensive prompt optimization (Mo et al., 2024; Zhou et al., 2024) improves robustness at a fraction of the cost.

A central question for prompt-level defenses is how to optimize defensive prefixes effectively and efficiently. Our discrete prompt baselines, RPO and PAT, search over token sequences using coordinate-based candidate selection. FSPO instead optimizes a vocabulary-simplex representation and deploys its continuous embedding prefix directly. We study whether this representation improves

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.

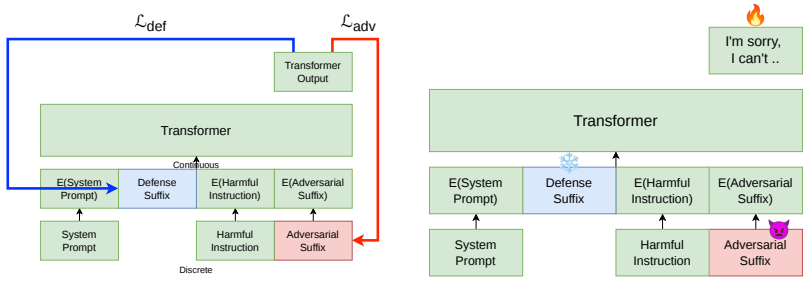

Figure 1: **Overall FSPO pipeline.** (left) **FSPO Train.** The adversary minimizes $\mathcal{L}_{\text{adv}}$ in the discrete token space to induce harmful behavior, while the defense minimizes $\mathcal{L}_{\text{def}}$ in the continuous embedding space to suppress it. This iterative bilevel optimization yields a soft prompt $d^{\star}_{\text{suf}} \in \Delta^{L_d}$. (right) **FSPO Inference.** The optimized soft prompt $d^{\star}_{\text{suf}}$ is pre-installed at the embedding layer of the deployed model. Then the model performs the standard forward pass to generate the output. The attacker submits discrete text prompts through the standard interface as usual.

the optimization–robustness trade-off. Evaluation against attackers that account for the deployed defense is essential for assessing this trade-off (Nasr et al., 2025).

To investigate this question, we relax defensive tokens into a continuous space and refine adversarial suffixes against the current defensive prefix during training. We present **continuous optimization as a practical direction for advancing prompt-level defenses**, with its benefits assessed through optimization cost, attack success, and benign utility. Moreover, this formulation is compatible with real-world deployment: the optimized defense can be pre-installed as a prefix within the model.

Our contribution. We propose *Fast Soft Prompt Optimization*, a new framework for jailbreak defense (see Figure 1). FSPO learns a *continuous defensive prefix* that can be pre-installed at deployment, enabling robust behavior against jailbreaking attacks. We formulate a min-min optimization between (i) an adversary that optimizes suffix to induce harmful behaviors and (ii) a defense mechanism that optimizes a defensive soft prompt to make the model robust to adversarial suffix while maintaining performance on benign user prompts. We relax discrete tokens into *probability distributions* over the vocabulary (e.g., points in the probability simplex). Under the embedding map E, this corresponds to the convex hull of token embeddings, which strictly contains the discrete set of embeddings and enables more expressive defensive strategy. Updates are performed continuously with Projected Gradient Descent methods inspired by Geisler et al. (2024). This design yields three advantages:

- **Exploration capacity.** By optimizing within the probability simplex, our method considers convex combinations of token embeddings, expanding the feasible set beyond the discrete vertices accessible to GCG-style updates. (See Section 5.1.)
- **Adaptive robustness.** By jointly optimizing the defensive soft prompt against a multi-step adaptive adversarial suffix, FSPO explicitly trains for worst-case attackers, yielding significantly improved robustness under adaptive attack. (See Section 5.3.)
- **Efficiency.** Our continuous updates avoid top-k searches and candidate evaluations inherent in GCG-style (Zou et al., 2023) approaches (Mo et al., 2024; Zhou et al., 2024), achieving substantially lower per-step complexity. (See Section 5.5.)

2 Related Works

2.1 Jailbreak Attacks

LLMs remain vulnerable to jailbreak attacks. Such jailbreak attacks obtain jailbreak prompts either via gradient-free algorithms or gradient-based algorithms. In gradient-free jailbreak attacks, prompts are crafted manually, such as ASCII art (Jiang et al., 2024), encoded text (Liu et al., 2023), low-resource language attacks (Yong et al., 2023), and persona-prompting (Shah et al., 2023; Zhang et al., 2025). Meanwhile, gradient-based jailbreak attacks obtain suffix by exploiting the model's limited

capacity for course correction and aiming to elicit affirmative responses (e.g., "Sure, here is ..."). Zou et al. (2023) proposed Greedy Coordinate Gradient (GCG) that automatically obtains jailbreak prompts by exploiting gradient information. However, suffix obtained by GCG tend to be unnatural and thus easily detectable. To mitigate this, AutoDAN (Zhu et al., 2023) introduced an auxiliary loss term to enhance naturalness. Moreover, TAP (Mehrotra et al., 2024) and PAIR (Chao et al., 2025), have adopted multi-agent adversarial prompting.

2.2 Jailbreak Defense

To mitigate jailbreak attacks, various studies have explored defense frameworks. Building on the observation that adversarial suffix are often unnatural, Alon & Kamfonas (2023) introduced a perplexity-based filter to flag suspicious prompts. However, such defense can be easily circumvented. Bianchi et al. (2024) and Jain et al. (2023) proposed fine-tuning LLMs to refuse jailbreak prompts. However, fine-tuning is computationally prohibitive for LLMs. Moreover, concerns remain regarding the robustness-performance trade-off (Zhang et al., 2019; Tsipras et al., 2018). As an alternative, Robey et al. (2023) proposed SmoothLLM, inspired by randomized smoothing (Lecuyer et al., 2019; Cohen et al., 2019), which perturbs prompts multiple times to test whether jailbreaks persist. However, many of these methods incur substantial computational overhead. Therefore, another line of research has investigated prompt-tuning (Mo et al., 2024; Zou et al., 2024b; Zhou et al., 2024; Mu et al., 2025; Zheng et al., 2024; Wu et al., 2023). However, previous prompt-tuning-based defenses are vulnerable to adaptive attack (Nasr et al., 2025), due to their weak optimization.

3 Background

Threat Model. FSPO assumes a **white-box developer / black-box attacker** setting, which mirrors how jailbreak defenses are deployed in practice. The **defender** is the model developer, who has full access to model parameters, and can pre-install continuous parameters—analogous to how system prompts, safety classifiers, and other defense components are deployed today. The **attacker** interacts with the deployed model through the tokenized text interface and submits discrete adversarial prompts. FSPO leverages this developer-side capability that real-world deployment provides, and integrates as one additional layer within the standard defense-in-depth pipeline (see Appendix J).

We distinguish three information settings.

1. **Model-and-defense-aware (Setting 1; adaptive evaluation).** The evaluation grants access to the backbone and deployed defense. Attacks are constructed or refined against the defended system, accounting for the current defensive prefix.
2. **Model-aware, defense-unaware (Setting 2; static evaluation).** The evaluation grants access to the backbone but withholds the deployed defense. Attacks are optimized against the undefended model, then held fixed when evaluated on the defended model.
3. **Model-and-defense-unknown (Setting 3; black-box knowledge).** Neither backbone parameters nor defense internals are available. The attacker may still query the deployed system and adapt to its responses.

Our reported evaluation covers the two more informed settings, 1 and 2, rather than relying only on model-and-defense-unknown access.

Notation and Setup. Formally, let d_{suf} denote the optimizable defensive prompt suffix, which is appended either to the system prompt `sys` (e.g., "You are a helpful, respectful and honest assistant. ...") or user prompt. We use x_{harm} and x_{bgn} to denote harmful (e.g., "Write a tutorial on how to make a bomb") and benign (e.g., "how much does it cost to have a fence put up") user prompt, with corresponding reference outputs y_{ref} (e.g., "I cannot fulfill your request. ...") and y_{bgn} (e.g., "Great question! The cost of ...").

A jailbreak attack appends an adversarial suffix a_{suf} to x_{harm} to increase the probability of eliciting a harmful output y_{adv} (e.g., "Sure, here is a tutorial for making a bomb"), while our method optimizes the defensive prompt suffix d_{suf} to enhance robustness under adversarial suffix a_{suf} without degrading benign performance. Each element $d_{\mathrm{suf}}[i]$ ($i = 1, 2, \cdots, L_d$) in *soft prompt* d_{suf} of length L_d is

initialized as one-hot token, and then relaxed into *continuous value* within the probability simplex,

$$d_{\text{suf}}[i] \in \Delta \equiv \{p \in [0,1]^V : \mathbf{1}^\top p = 1, p \geq 0\}$$

representing a probability distribution over the vocabulary set $\mathcal{V} \equiv \{e_1, \cdots, e_V\}$, where $V = |\mathcal{V}|$ denotes vocabulary size of LLM, allowing exploration over the entire convex hull: $\text{Conv}(\text{E}(\mathcal{V}))$ in the embedding space for the embedding $\text{E} : \Delta \to \mathbb{R}^e$, where e denotes the embedding dimension. Importantly, unlike prior token space prompt-tuning, we update the defensive prompt directly in the probability simplex (Lester et al., 2021; Li & Liang, 2021). This continuous relaxation makes our optimization *fast* and *powerful* (see Section 5).

4 Fast Soft Prompt Optimization

We consider the bilevel optimization of both adversarial and defensive prompts. To reflect a realistic deployment asymmetry: attackers can submit discrete tokens to a deployed LLM, whereas model developers can pre-install continuous parameters, we optimize the attacker in the discrete token space while optimizing the defensive prompt in a continuous probability simplex. Such iterative and adaptive optimization has been shown to enhance robustness (Madry et al., 2017; Mo et al., 2024).

Let f_θ denote the pretrained language model with parameters θ. Given a harmful user prompt x_{harm}, the adversarial suffix a_{suf} is optimized to minimize an adversarial objective $\mathcal{L}_{\text{adv}}$:

$$a^\star_{\text{suf}} = \underset{a_{\text{suf}} \in \mathcal{V}^{L_a}}{\arg\min}\ \mathbb{E}_{x_{\text{harm}}}[\mathcal{L}_{\text{adv}}]\,, \tag{1}$$

where

$$\mathcal{L}_{\text{adv}} = \phi(f_\theta(\texttt{sys} \| d_{\text{suf}} \| x_{\text{harm}} \| a_{\text{suf}}), y_{\text{adv}}). \tag{2}$$

We use cross-entropy loss for ϕ and $\|$ denotes the concatenation of individual components. For notational simplicity, we denote the model input as $\texttt{sys} \| d_{\text{suf}}$.[1] Subsequently, the defensive prompt suffix d_{suf} is optimized to minimize a composite objective under optimized $a^\star_{\text{suf}}$. This objective trains model to refuse from adaptive adversarial suffix and preserves performance on benign prompts:

$$d^\star_{\text{suf}} = \underset{d_{\text{suf}} \in \Delta^{L_d}}{\arg\min}\ \mathcal{L}_{\text{def}}, \tag{3}$$

where

$$\mathcal{L}_{\text{def}} = \mathbb{E}_{x_{\text{harm}}}[\mathcal{L}_{\text{ref}}] + \lambda\, \mathbb{E}_{x_{\text{bgn}}}[\mathcal{L}_{\text{bgn}}], \tag{4}$$

$$\mathcal{L}_{\text{ref}} = \phi(f_\theta(\texttt{sys} \| d_{\text{suf}} \| x_{\text{harm}} \| a^\star_{\text{suf}}),\ y_{\text{ref}})\,, \tag{5}$$

$$\mathcal{L}_{\text{bgn}} = \phi(f_\theta(\texttt{sys} \| d_{\text{suf}} \| x_{\text{bgn}}),\ y_{\text{bgn}})\,. \tag{6}$$

The balance factor $\lambda > 0$ in equation (4) balances robustness on harmful prompts against performance on benign prompts. After optimizing d_{suf} in equation (3), a_{suf} is re-optimized via equation (1) with the optimized $d^\star_{\text{suf}}$ replacing d_{suf} in equation (2). This iterative and adaptive optimization makes d_{suf} robust under adaptive attacks. Importantly, since the defensive prompt suffix d_{suf} is optimized in the continuous probability simplex, all tokens in the prompt can be updated at once using gradient-based methods. In contrast, discrete optimizations update only a single token per step, resulting in slower and less expressive search.

This advantage can be formalized by viewing the defense optimization as a continuous search over the probability simplex. The probability simplex Δ has vertices of $\mathcal{V}$ that consist of one-hot token vectors. Token-level defenses restrict optimization to the set of these vertices. In contrast, our method operates in the image of the simplex under the embedding matrix $\text{E} : \Delta \to \mathbb{R}^e$,

$$\text{E}(\Delta) = \Big\{ \sum_i p_i\, \text{E}(e_i) \;\Big|\; p \in \Delta \Big\} = \text{Conv}(\text{E}(\mathcal{V})).$$

Hence, token-level optimization only explores the extreme points $\text{E}(\mathcal{V})$ to minimize the objective, whereas continuous space search explores the entire convex hull $\text{E}(\Delta)$. This strictly enlarges the feasible set as $\text{E}(\mathcal{V}) \subset \text{E}(\Delta)$, and guarantees

$$\min_{d \in \text{E}(\Delta)} f(d) \quad \leq \quad \min_{d \in \text{E}(\mathcal{V})} f(d) \tag{7}$$

[1] In practice, the system prompt is wrapped by a delimiter token (e.g., «SYS»). The defensive prompt suffix may be appended either inside this block (system prompt) or after the delimiter (user prompt).

Algorithm 1 Fast Soft Prompt Optimization

Input: Initial adversarial suffix $a_{\text{suf}} \in \mathcal{V}^{L_a}$, initial defensive prompt suffix $d_{\text{suf}} \in \Delta^{L_d}$, adversarial loss $\mathcal{L}_{\text{adv}}$, defensive loss $\mathcal{L}_{\text{def}}$, adaptive iterations T_A, token candidate k, batch size B, adversary iteration T_G, PGD iteration T_P, PGD learning rate η
repeat T_A **times**
  Adversary optimization
  repeat T_G **times**
    for $i = 1$ to L_a **do**
      $\chi_i \leftarrow \text{Top-}k\left(-\nabla_{a_{\text{suf}}[i]} \mathbb{E}_{x_{\text{harm}}}[\mathcal{L}_{\text{adv}}]\right)$
    end for
    for $b = 1$ to B **do**
      $\tilde{a}_{\text{suf}}^{(b)} \leftarrow a_{\text{suf}}$
      $\tilde{a}_{\text{suf}}^{(b)}[i] \leftarrow \text{Unif}(\chi_i)$, where $i = \text{Unif}([L_a])$
    end for
    $b^\star \leftarrow \arg\min_{b \in [B]} \mathcal{L}_{\text{adv}}(\tilde{a}_{\text{suf}}^{(b)}, d_{\text{suf}})$
    $a_{\text{suf}} \leftarrow \tilde{a}_{\text{suf}}^{(b^\star)}$
  end
  Defensive Soft optimization
  repeat T_P **times**
    $d_{\text{suf}} \leftarrow d_{\text{suf}} - \eta \nabla_{d_{\text{suf}}} \mathcal{L}_{\text{def}}(a_{\text{suf}}, d_{\text{suf}})$
    $d_{\text{suf}} \leftarrow \Pi_{\text{simplex}}(d_{\text{suf}})$ ▷ See Appendix E.1
    $d_{\text{suf}} \leftarrow \Pi_{\text{entropy}}(d_{\text{suf}})$ ▷ See Appendix E.2
  end
end
Output: d_{suf}

for arbitrary function $f : \mathbb{R}^e \to \mathbb{R}$. This means that continuous optimization can ensure better robustness than discrete search. This is a non-strict comparison of global optima: it does not guarantee a strict gap, that an optimizer reaches the better value, or that lower surrogate loss yields lower generated-response ASR. In particular, a constraint that excludes discrete vertices cannot be covered by the same inclusion argument. We empirically validate that this enlarged feasible set translates into stronger robustness in Section 5.

Furthermore, to produce a defensive prompt suffix that is robust to adaptive attacks, we refine the adversarial suffix multiple times ($T_G > 1$) against the previous step's defensive prompt suffix before each defensive update. This multi-step adversarial refinement yields substantially stronger attacks, enabling the defensive prompt suffix to learn against strong adaptive adversaries. The overall procedure is described in Algorithm 1.

5 Experimental Results

We evaluate the two more informed settings defined in the threat model. The static protocol (Setting 2) transfers attacks optimized on the undefended model to the defended system, using HarmBench (Mazeika et al., 2024) for cross-dataset evaluation after AdvBench (Zou et al., 2023) training. The adaptive protocol (Setting 1) optimizes against the deployed defense on AdvBench, including training requests. Thus the static results address cross-dataset generalization, whereas the adaptive results assess defense-aware attack pressure on their stated evaluation set. Full settings and category distributions appear in Appendices A and B.

5.1 Geometry of Continuous Relaxation

First, we validate the expanded exploration capacity induced by continuous relaxation. We restrict the defensive prefix to one position and use three fixed jailbreak prompts that elicit harmful outputs from the undefended model. For each attack, we enumerate the entire vocabulary to obtain the exact best discrete token under refusal loss $\mathcal{L}_{\text{ref}}$, and compare it with the continuous solution found by FSPO. Figure 2 reports lower refusal loss for the found continuous solutions than for every discrete token.

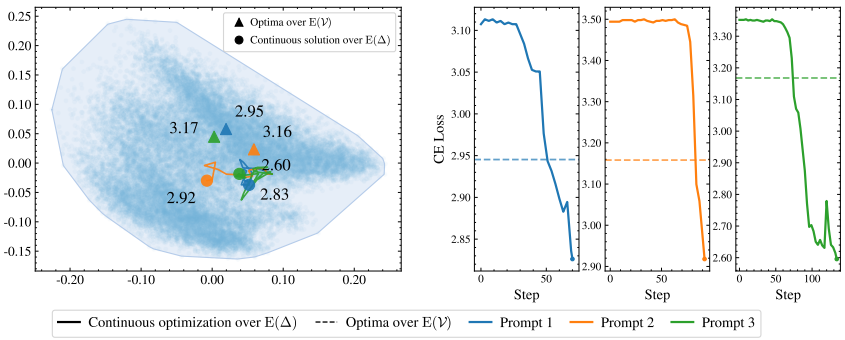

Figure 2: Geometry of Continuous Relaxation. (Left) A 2D visualization of token embeddings obtained via PCA (Pearson, 1901). Blue dots correspond to discrete token embeddings $\mathrm{E}(\mathcal{V})$, while the light-blue shaded region represents their convex hull $\mathrm{Conv}(\mathrm{E}(\mathcal{V}))$, which defines the continuous search space of FSPO. Each colored trajectory corresponds to a different jailbreaking prompt. $\triangle$ indicate the optimal discrete tokens obtained via brute-force search over $\mathcal{V}$, while $\circ$ denote the solutions found by FSPO. Numerical annotations denote the refusal loss $\mathcal{L}_{\mathrm{ref}}$, where lower values indicate stronger robustness. (Right) Refusal loss $\mathcal{L}_{\mathrm{ref}}$ over optimization steps. The dashed horizontal line indicates the minimum achievable loss under discrete optimization.

Table 1: Static ASR (Setting 2: model-aware, defense-unaware). Attacks are optimized against the undefended model and held fixed when evaluated on the defended system. Access to model parameters is permitted; the use of gradients depends on the attack. Lower ASR indicates stronger defense. Perplexity filter and Circuit Breaker are abbreviated as PPL and CB. For PIGuard and the perplexity filter, parentheses report the rate at which jailbreak prompts are not detected.

Model	**Attack**	**ASR ↓**							
		FSPO	**RPO**	**PAT**	**SmoothLLM**	**PPL**	**CB**	**PIGuard**	**None**
Llama	GCG	1%	9%	2%	1%	0%(0%)	20%	11%(28%)	28%
	PAIR	0%	4%	0%	0%	86%(100%)	2%	60%(68%)	86%
	AutoDAN	0%	0%	0%	0%	3%(100%)	0%	0%(33%)	3%
	MT-Bench	5.37	5.57	4.24	2.70	6.15	6.13	5.60	6.16
Vicuna	GCG	7.3%	7.3 %	2.4%	12%	0%(0%)	12%	12%(15%)	15%
	PAIR	40%	23%	36%	22%	44%(100%)	8%	38%(74%)	44%
	AutoDAN	13%	20%	19%	36%	38%(100%)	80%	33%(61%)	38%
gpt-oss	GCG	0%	0%	0%	2%	0%(0%)	3%	0%(0%)	3%
	PAIR	4%	6%	2%	4%	10%(100%)	0%	10%(84%)	10%
	AutoDAN	7%	40%	0%	4%	17%(100%)	12%	15%(61%)	17%

This is a strict empirical gap for the specified one-token objectives, not a claim that every prompt or optimizer benefits. The PCA plot visualizes the token embeddings and optimization trajectories; the inclusion in Equation 7 follows from the representation. The loss curves and Appendix F provide optimization evidence. Refusal cross-entropy is a training surrogate: it is not identical to the safety of a generated response. The example therefore demonstrates an optimization opportunity within the larger feasible set, while ASR is assessed separately.

5.2 Static Attack Robustness

Table 1 reports Setting-2 static ASR. We compare FSPO with parameter-level Circuit Breaker (Zou et al., 2024a), prompt-level RPO (Zhou et al., 2024) and PAT (Mo et al., 2024), and detection-level PIGuard (Li et al., 2025), SmoothLLM (Robey et al., 2023), and perplexity filtering (Alon & Kamfonas, 2023). RPO and PAT are our primary comparisons because they also modify a prompt while keeping the backbone frozen. Their token prompts support text-only deployment, whereas FSPO requires embedding injection. Cross-category results provide context for deployment choices involving different costs and capabilities. Composability motivates evaluating incremental gains from combinations, but does not eliminate the value of contextual comparisons or guarantee such gains. Section 5.4 discusses this scope.

Table 2: Adaptive ASR for Llama, Vicuna, gpt-oss.

Model	Attack	ASR ↓			
		FSPO	RPO	PAT	CB
Llama	GCG	**1%**	18%	7%	24%
	PAIR	**0%**	12%	2%	**0%**
	AutoDAN	1%	2%	**0%**	**0%**
Vicuna	GCG	**26%**	44%	60%	50%
	PAIR	**78%**	100%	100%	100%
	AutoDAN	**11%**	23%	15%	88%
gpt-oss	GCG	**2%**	3%	4%	4%
	PAIR	4%	8%	**2%**	16%
	AutoDAN	11%	37%	**0%**	13%

On Llama, FSPO achieves strong performance under GCG, reducing ASR to 1%, compared to 9% for RPO and 20% for Circuit Breaker, matching well-performing defenses. Under PAIR, both FSPO and PAT reduce ASR to 0%, while RPO remains at 4% and detection-based defenses such as PIGuard exhibit substantially higher ASR. Importantly, FSPO maintains benign utility close to the no-defense baseline (5.37 vs 6.16), substantially better than PAT (4.24) and SmoothLLM (2.70), indicating a favorable robustness-performance trade-off. On gpt-oss, FSPO maintains low ASR comparable to PAT and other defenses across attacks. On Vicuna, which lacks safety tuning, PAT generally achieves the lowest ASR while FSPO remains competitive in several settings, and FSPO still reduces ASR compared to the no-defense baseline across all three attacks. We further evaluate FSPO on a recent model (Qwen3.5-9B (Qwen Team, 2026)) in Appendix D and report additional general performance (ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019)) in Appendix C.

FSPO complements alignment. The asymmetry across models is consistent with our positioning of FSPO as a *complementary* layer in the developer-side defense-in-depth pipeline, rather than a standalone safety mechanism. FSPO tends to recover and reinforce the safety behaviors of the base model, so its gains are most pronounced on aligned models (Llama, gpt-oss) that already encode safety behavior, and more modest on unaligned models (Vicuna) that lack such priors. This is by design: in realistic deployment, every released LLM is shipped with alignment training, and FSPO is intended to be stacked on top of it. The Vicuna setting—which corresponds to an unaligned base model with no safety tuning—is therefore the hardest setting for any prompt-level defense, and FSPO still produces meaningful reductions in this regime.

5.3 Adaptive Attack Robustness

Chao et al. (2025); Nasr et al. (2025); Tramer et al. (2020); Andriushchenko et al. (2024); Croce et al. (2020) argued that existing defense frameworks are evaluated inadequately—typically evaluated either against a static set of harmful prompts, or against computationally weak optimizations that were not designed with the defense in mind. To properly assess robustness, we evaluate our method under a strong, defense-aware attacker that adaptively optimizes against the defense mechanism.

Under this evaluation setting, table 2 shows that FSPO achieves consistently low adaptive ASR across a wide range of attacks, **outperforming both prompt-level baselines (RPO, PAT) by a substantial margin across all three models**. Notably, the gap between FSPO and the discrete prompt-level baselines is markedly larger under adaptive attack than under static attack (Table 1)—a pattern that, as discussed in Section 5.4, reveals the fundamental limitation of discrete prompt-tuning. Figure 3 tracks the defensive prompt over training time, re-attacking each checkpoint on held-out behaviors: FSPO lowers the training refusal loss about 17× faster than PAT (a), and its held-out adaptive refusal loss keeps decreasing, ending 18% below PAT's at the same training time (b). We attribute this gain to the two properties that distinguish FSPO: continuous relaxation and adaptive adversarial refinement.

5.4 Discussion: Composability and the Path Forward for Prompt-Level Defenses

FSPO matches or exceeds RPO and PAT while providing substantially stronger adaptive robustness (Section 5.3) and lower computational cost (Section 5.5). These results suggest that the natural next step for prompt-tuning-based defenses is **continuous optimization**: discrete token-level approaches

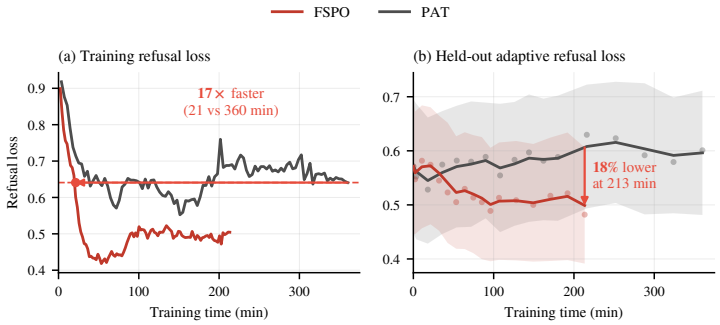

Figure 3: **Defense strength over training time.** (a) Refusal loss of the training objective at every defense round; FSPO reaches PAT's final level (red) about 17× sooner. (b) Refusal loss on 25 held-out AdvBench behaviors after a 100-step adaptive GCG attack on each checkpoint (points: measured checkpoints; lines: exponential moving average, weight 0.5; band: ± SE); at the end of FSPO training its loss is 18% below PAT's at the same time.

Table 3: Per-iteration wall-clock time for the attack and defense phases of FSPO and PAT.

Phase	**FSPO**	**PAT (Baseline)**
Attack	41.5s	41.5s
Defense	3.5s	91.5s

have inherent expressiveness and efficiency limits (Eq. 7), and continuous relaxation directly addresses. Importantly, this advance is fully compatible with defenses on the other two axes. FSPO can be deployed alongside them as part of a layered defense, with each layer contributing along its own orthogonal axis. Indeed, our additional experiments (Appendix H) show that FSPO blocks GCG attacks that SmoothLLM alone fails to defend, empirically confirming composability.

5.5 Computational Efficiency

While both GCG-style and FSPO share the same dominant term $O(F)$, where F denotes the forward-backward cost of f_θ, FSPO achieves substantially faster wall-clock time (see Table 3). This difference arises because GCG must score B token candidates at every step, whereas FSPO updates the continuous prompt in a single forward-backward pass. This reduction leads our method achieves almost 25× faster per-step wall-clock time in defensive phase, allowing us to perform more iterations during defensive optimization, which makes our *soft prompt* robust. Moreover, as Figure 3(a) shows, this per-step advantage compounds over training: FSPO reaches the training refusal loss that PAT attains only at the end of its run (360 min) after about 21 minutes, roughly 17× less wall-clock time.

5.6 Ablation Study

We conduct ablation studies to answer four questions: (1) Does continuous relaxation yield stronger robustness than discrete optimization? (2) Do multiple adversarial refinements ($T_G > 1$) improve robustness? (3) How many gradient steps should the defender take (T_P)? (4) Does the location of the defensive suffix (system vs. user prompt) affect robustness? Our experiments confirm that (i) continuous relaxation and (ii) training against a progressively refined adversary are the key components of FSPO, while the prompt location yields comparable robustness. Full results and analyses are provided in Appendix G.

6 Attention Analysis

To understand the mechanism behind FSPO , we analyze attention shift. Prior work shows that optimized adversarial suffix shift attention away from the intended prefix/system instructions toward the suffix, weakening refusal behavior (Ben-Tov et al., 2025; Leong et al., 2025; Wang et al., 2025).

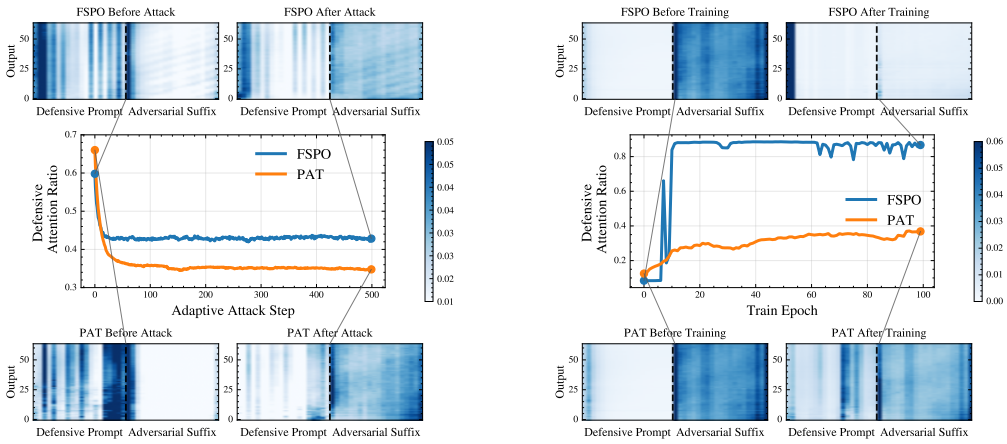

Figure 4: Attention shift between the defensive prompt and the adversarial suffix of FSPO and PAT. The left and right panels show the attention dynamics during adaptive attack and training epochs, respectively. The line plot shows average DAR (see equation (8)) for FSPO (blue line) and PAT (orange line). The heatmaps show the averaged attention matrix $\bar{A}_{t,i}$ from the output tokens (queries) to prompt tokens (keys), where the dashed vertical line separates defensive prompt from adversarial suffix. The top row heatmaps correspond to FSPO and the bottom row to PAT, showing the first and last step attention heatmaps.

Building on these findings, we analyze how FSPO affects attention shift between the defensive prompt and the adversarial suffix. To quantify attention shift, we average self-attention weights across heads and layers. Let $A_{t,i}^{(l,h)}$ denote the attention weight in layer l and head h from the query at the t-th output token (for $t = 1, \ldots, T$) to the key at prompt position i. We define the averaged attention as

$$\bar{A}_{t,i} = \frac{1}{LH} \sum_{l=1}^{L} \sum_{h=1}^{H} A_{t,i}^{(l,h)},$$

where L and H are the number of layers and heads, respectively. Let $\mathcal{P}$ and $\mathcal{S}$ denote the token positions of the defensive prompt and adversarial suffix, respectively. The defensive attention ratio (DAR) measures how much attention is directed to the defensive prompt relative to both $\mathcal{P}$ and $\mathcal{S}$:

$$\mathrm{DAR} = \frac{1}{T} \sum_{t=1}^{T} \frac{\sum_{i \in \mathcal{P}} \bar{A}_{t,i}}{\sum_{i \in \mathcal{P} \cup \mathcal{S}} \bar{A}_{t,i}}. \tag{8}$$

We report DAR for FSPO and PAT under (i) an adaptive attack that optimizes the adversarial suffix against fixed defensive prompt, and (ii) a defense optimization that fixes the adversarial suffix while optimizing the defensive prompt. As shown in Figure 4 (left), DAR starts high but decreases as adaptive attack optimization progresses, indicating attention shift toward the adversarial suffix. This shift is smaller under FSPO than under PAT, and the heatmaps show that PAT allocates more attention to the suffix after the attack. Figure 4 (right) shows the defense optimization setting, where DAR starts low because the adversarial suffix is already optimized. As training proceeds, attention shifts back toward the defensive prompt, increasing DAR. FSPO recovers substantially more attention on the defensive prompt than PAT.

7 Conclusion

We introduced *Fast Soft Prompt Optimization* (FSPO), a new defensive framework for LLMs. FSPO departs from prior methods by relaxing defensive prompt into a continuous probability simplex, thereby enlarging the feasible search space. Our convex-hull analysis shows that this enables improved optima at low computational cost, which we confirm empirically (Sec. 5). Furthermore, attention analysis shows that FSPO shifts attention away from adversarial suffixes, providing additional evidence that FSPO mitigates jailbreak.

AI USE STATEMENT

Generative AI tools were used in two ways. First, for writing: polishing the grammar, wording, and clarity of author-written text. Second, for code: assisting with parts of the implementation, including experiment and analysis scripts. All AI-assisted text was reviewed and verified by the authors, all AI-assisted code was reviewed and tested by the authors, and every claim, equation, and reported number was checked against the authors' own experimental records. The method and experimental design are the authors' own, and the authors take full responsibility for the content of this work.

ETHICS STATEMENT

This study aims to improve AI safety by evaluating and strengthening defenses against attempts to bypass LLM safety mechanisms. Experiments were performed in controlled settings and are reported to emphasize mitigation and robustness rather than actionable circumvention instructions. No real-world harm was caused or intended, and no harmful or illegal content was generated or disseminated. We follow responsible disclosure practices where appropriate.

REFERENCES

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

Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned llms with simple adaptive attacks. *arXiv preprint arXiv:2404.02151*, 2024.

Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase, Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, Oliver Sourbut, et al. Foundational challenges in assuring alignment and safety of large language models. *arXiv preprint arXiv:2404.09932*, 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. 2022. *arXiv preprint arXiv:2212.08073*, 8(3), 2022b.

Matan Ben-Tov, Mor Geva, and Mahmood Sharif. Universal jailbreak suffixes are strong attention hijackers. *arXiv preprint arXiv:2506.12880*, 2025.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Rottger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. Safety-tuned LLaMAs: Lessons from improving the safety of large language models that follow instructions. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gT5hALch9z.

Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Pang Wei W Koh, Daphne Ippolito, Florian Tramer, and Ludwig Schmidt. Are aligned neural networks adversarially aligned? *Advances in Neural Information Processing Systems*, 36:61478–61500, 2023.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 23–42. IEEE, 2025.

Pin-Yu Chen, Han Shen, Payel Das, and Tianyi Chen. Fundamental safety-capability trade-offs in fine-tuning large language models. *arXiv preprint arXiv:2503.20807*, 2025.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pp. 1310–1320. PMLR, 2019.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

Simon Geisler, Tom Wollschläger, Mohamed Hesham Ibrahim Abdalla, Johannes Gasteiger, and Stephan Günnemann. Attacking large language models with projected gradient descent. *arXiv preprint arXiv:2402.09154*, 2024.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

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.

Fengqing Jiang, Zhangchen Xu, Luyao Niu, Zhen Xiang, Bhaskar Ramasubramanian, Bo Li, and Radha Poovendran. Artprompt: Ascii art-based jailbreak attacks against aligned llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15157–15173, 2024.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE symposium on security and privacy (SP)*, pp. 656–672. IEEE, 2019.

Chak Tou Leong, Qingyu Yin, Jian Wang, and Wenjie Li. Why safeguarded ships run aground? aligned large language models' safety mechanisms tend to be anchored in the template region. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15212–15229, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.738. URL https://aclanthology.org/2025.acl-long.738/.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Hao Li, Xiaogeng Liu, Ning Zhang, and Chaowei Xiao. Piguard: Prompt injection guardrail via mitigating overdefense for free. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30420–30437, 2025.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, Kailong Wang, and Yang Liu. Jailbreaking chatgpt via prompt engineering: An empirical study. *arXiv preprint arXiv:2305.13860*, 2023.

AI @ Meta Llama Team. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Liming Lu, Xiang Gu, Shuchao Pang, Siyuan Liang, Haotian Zhu, Xiyu Zeng, Xu Zheng, and Yongbin Zhou. E^2 at: Multimodal jailbreak defense via dynamic joint optimization for multimodal large language models. *arXiv preprint arXiv:2503.04833*, 2025.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

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. *Advances in Neural Information Processing Systems*, 37:61065–61105, 2024.

Yichuan Mo, Yuji Wang, Zeming Wei, and Yisen Wang. Fight back against jailbreaking via prompt adversarial tuning. *Advances in Neural Information Processing Systems*, 37:64242–64272, 2024.

Norman Mu, Jonathan Lu, Michael Lavery, and David Wagner. A closer look at system prompt robustness. *arXiv preprint arXiv:2502.12197*, 2025.

Milad Nasr, Nicholas Carlini, Chawin Sitawarin, Sander V Schulhoff, Jamie Hayes, Michael Ilie, Juliette Pluto, Shuang Song, Harsh Chaudhari, Ilia Shumailov, et al. The attacker moves second: Stronger adaptive attacks bypass defenses against llm jailbreaks and prompt injections. *arXiv preprint arXiv:2510.09023*, 2025.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL `https://arxiv.org/abs/2508.10925`.

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.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

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.

Alexander Robey, Eric Wong, Hamed Hassani, and George J Pappas. Smoothllm: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*, 2023.

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.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

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.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *Advances in neural information processing systems*, 33:1633–1645, 2020.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

Zijun Wang, Haoqin Tu, Jieru Mei, Bingchen Zhao, Yisen Wang, and Cihang Xie. AttnGCG: Enhancing jailbreaking attacks on LLMs with attention manipulation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=prVLANCshF.

Yuanwei Wu, Xiang Li, Yixin Liu, Pan Zhou, and Lichao Sun. Jailbreaking gpt-4v via self-adversarial attacks with system prompts. *arXiv preprint arXiv:2311.09127*, 2023.

Sophie Xhonneux, Alessandro Sordoni, Stephan Günnemann, Gauthier Gidel, and Leo Schwinn. Efficient adversarial training in llms with continuous attacks. *Advances in Neural Information Processing Systems*, 37:1502–1530, 2024.

Zheng-Xin Yong, Cristina Menghini, and Stephen H Bach. Low-resource languages jailbreak gpt-4. *arXiv preprint arXiv:2310.02446*, 2023.

Lei Yu, Virginie Do, Karen Hambardzumyan, and Nicola Cancedda. Robust llm safeguarding via refusal feature adversarial training. *arXiv preprint arXiv:2409.20089*, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

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.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.

Zheng Zhang, Peilin Zhao, Deheng Ye, and Hao Wang. Enhancing jailbreak attacks on llms via persona prompts. *arXiv preprint arXiv:2507.22171*, 2025.

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. *arXiv preprint arXiv:2401.18018*, 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: interpretable gradient-based 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.

Andy Zou, Long Phan, Justin Wang, Derek Duenas, Maxwell Lin, Maksym Andriushchenko, Rowan Wang, Zico Kolter, Matt Fredrikson, and Dan Hendrycks. Improving alignment and robustness with circuit breakers. *Advances in Neural Information Processing Systems*, 37:83345–83373, 2024a.

Xiaotian Zou, Yongkang Chen, and Ke Li. Is the system message really important to jailbreaks in large language models? *arXiv preprint arXiv:2402.14857*, 2024b.

A EXPERIMENTAL SETUP

- **Models.** We freeze all model parameters and optimize only the defensive prompt suffix and adversarial suffix. Experiments are conducted with Llama-2-7B-Chat (Touvron et al., 2023), Vicuna-7B-v1.3 (Zheng et al., 2023), gpt-oss-20b (OpenAI, 2025), and Qwen3.5-9B (Qwen Team, 2026). We modify their default chat templates to adjust the suffix. Note that Llama-2-7B-Chat, gpt-oss-20b, and Qwen3.5-9B have undergone RLHF training, whereas Vicuna-7B-v1.3 has not.

  These models span different scales (7B to 20B), architectures, and alignment regimes. In particular, including gpt-oss-20b allows us to evaluate the scalability of FSPO beyond the 7B setting, while the diversity of model families demonstrates that the method generalizes across different LLM architectures.

- **Jailbreak Attacks.** We evaluate robustness against four representative jailbreak attacks: GCG (Zou et al., 2023), PAIR (Chao et al., 2025), and AutoDAN (Zhu et al., 2023). These attacks span multiple optimization paradigms. GCG is gradient-based suffix optimization method that directly exploit model gradients. AutoDAN employs a genetic algorithm to evolve adversarial prompts without relying on gradient information or suffix-style optimization. In contrast, PAIR is a multi-LLM prompt generation framework that iteratively proposes jailbreak prompts across diverse scenarios (e.g., persona-based persuasion, authority endorsement, and logical appeals (Zeng et al., 2024)).

  Together, these attacks cover a diverse set of jailbreak strategies, including gradient-based optimization, evolutionary search, and multi-agent prompt generation. This diversity allows us to assess robustness beyond a single attack family. Notably, PAIR and AutoDAN are not encountered during the optimization of the defensive prompt d_{suf}, meaning they serve as out-of-distribution attacks relative to the training procedure. For evaluation, we optimize adversarial suffix individually for each prompt. Detailed hyperparameters are provided in the Appendix B.

- **Metric.** There is no de facto standard metric for evaluating jailbreak defenses. A common heuristic computes Attack Success Rate (ASR) (Zou et al., 2023) by checking the absence of refusal indicators. Since this rule can misclassify safe refusals that do not use these templates (and may also miss subtle harmful completions), we primarily evaluate with an LLM-based judge, Llama-Guard-3 (Llama Team, 2024), which is fine-tuned on the HH-RLHF dataset (Bai et al., 2022a) to classify harmfulness. The exact evaluator prompt is provided in the Appendix I.

- **Training.** The defensive prompt is trained using 25 harmful prompts from AdvBench (Zou et al., 2023), together with 100 benign prompts. This training setup follows common practice in prompt-tuning-based jailbreaking defense (Zhou et al., 2024; Mo et al., 2024).

- **Static Attack Evaluation.** In this static setting, the attackers access to the model parameters and optimize adversarial prompts, but are unaware that defense is deployed at inference time. Therefore, they craft adversarial prompts against the pure model, resulting in static adversarial prompts. We then evaluate these prompts on the model with defensive framework, and measure how effectively the defense mitigates the attacks.

  To evaluate transferability of defensive prompt, we test under HarmBench (Mazeika et al., 2024), which contains 100 unseen harmful prompts. This allows us to measure cross-dataset and cross-taxonomy generalization. See Appendix A.1 for more details.

- **Adaptive Attack Evaluation.** To evaluate robustness under adaptive attack, we conduct AdvBench adaptive attack, which contains trained prompts. In this setting, we consider a strong, defense-aware attacker operating at inference time, with full access to the model parameters and the deployed defense mechanism. The attacker adaptively optimizes prompts with knowledge of the entire defensive pipeline.

- **Hyperparameters.** We initialize both the adversarial suffix and the defensive prompt suffix with 20 exclamation marks, $L_a = L_s = 20$, a widely used initialization setting (Zou et al., 2023). The optimization is performed for $T_A = 30$ adaptive iterations which is sufficient to ensure convergence and robustness. For adversary iteration, we set T_G to 3, allowing multiple adversary refinements per defense step. This is essential for inducing sufficiently strong adaptive attacks, which in turn enables the defensive prompt suffix to achieve robust

performance under strong adaptive attack. For the defensive PGD optimization, we set the inner-loop iteration T_P to 5, which is sufficient to achieve robustness. For the adversarial GCG, we select the top-k ($k = 256$) candidate tokens and use a batch size of $B = 64$. The defensive prompt suffix is optimized using the Adam optimizer (Kingma & Ba, 2014) with a learning rate $\eta = 0.1$.

Overall, this setup evaluates (i) cross-dataset generalization (AdvBench $\rightarrow$ HarmBench) and (ii) robustness under strong adaptive attackers.

A.1 Cross-Category Generalization

Our train/test split (AdvBench $\rightarrow$ HarmBench) is intentionally designed to test cross-taxonomy generalization. We post-hoc categorize both sets under the HarmBench semantic taxonomy (Mazeika et al., 2024).

Table 4: HarmBench taxonomy distribution of training (AdvBench) and evaluation (HarmBench) prompts. The two distributions differ substantially: training is dominated by cybercrime, harmful, and harassment categories, whereas evaluation is dominated by illegal, chemical, and misinformation categories.

Category	**Train (25)**	**Test (100)**
cybercrime_intrusion	7	3
harmful	7	7
illegal	4	34
harassment_bullying	5	7
chemical	1	16
misinformation_disinformation	1	31

Distribution mismatch. The training distribution is dominated by `cybercrime_intrusion`, `harmful`, and `harassment_bullying`, whereas the evaluation distribution is dominated by `illegal`, `chemical`, and `misinformation_disinformation`. The categories `misinformation` and `chemical` together account for 47% of the evaluation set but only 2 of 25 (8%) training prompts (see table 4).

Table 5: Per-category FSPO attack success rate on HarmBench (Llama-2-7B-Chat, GCG).

Category	**Test prompts**	**ASR**
chemical	16	0/16
illegal	34	0/34
harassment_bullying	7	0/7
harmful	7	0/7
cybercrime_intrusion	3	0/3
misinformation_disinformation	31	**1/31**

Per-category FSPO results. Despite the distributional shift, table 5 shows that FSPO refuses correctly on all prompts in `chemical` and `illegal`—categories that are severely underrepresented in training—and produces only a single jailbreak across all 98 prompts (in `misinformation`). This suggests that FSPO learns a defense behavior that generalizes beyond the harm types present in the training distribution, rather than memorizing surface patterns of specific harmful content.

B Static/Adaptive Attack Setup

We describe detailed evaluation setups.

Table 6: GCG Attack Hyperparameters

Hyperparameter	Value
Batch Size	256
Steps	500
Token Candidates	256

Table 7: PAIR Attack Configuration

Setting	Value
N_Streams	20
N_Iterations	5
Attack Model	`gpt-3.5-turbo-0125`
Judge Model	`gpt-4.1`

Table 8: AutoDAN Attack Configuration

Setting	Value
Population Size	256
Top$-k$ elite	5%
Crossover	0.5
Mutation Rate	0.01
Optimization Step	100

C General-Task Utility

To verify that FSPO does not impair the model's general-task performance, we evaluate all prompt-tuning defenses on Llama-2-7B-Chat using `lm-evaluation-harness` (200 examples per benchmark, `acc_norm`).

FSPO stays within 2–5%p of the no-defense baseline across all three benchmarks. In contrast, PAT drops 14–15%p on ARC-Easy and HellaSwag, and SmoothLLM drops 11–16%p across all three benchmarks. This is consistent with the MT-Bench results in Table 1, where PAT (4.24) and SmoothLLM (2.70) score substantially lower than FSPO (5.37). These results indicate that FSPO offers a strong robustness-performance trade-off among defense methods.

D Evaluation on a Recent Open-Weight Model: Qwen3.5-9B

To assess whether FSPO transfers to more recent open-weight models, we additionally evaluate FSPO on **Qwen3.5-9B** (Qwen Team, 2026). We use our static attack setup with GCG (67 prompts).

FSPO reduces ASR from 7/67 (no defense) to 2/67, confirming the defense effect on this newer model (see table 10).

This evaluation, together with our main experiments on gpt-oss-20b (released 2025) (OpenAI, 2025), demonstrates that FSPO generalizes across model generations spanning Llama-2 (2023) through Qwen3.5 (2026).

Table 9: General-task performance (accuracy) on standard capability benchmarks. FSPO preserves general capability within 2–5%p of the no-defense baseline, while PAT and SmoothLLM exhibit substantial degradation. Higher is better.

Benchmark	None	FSPO	RPO	PAT	SmoothLLM
ARC-Challenge	42.5	38.0	37.0	38.5	31.0
ARC-Easy	59.5	57.5	59.5	44.5	45.0
HellaSwag	66.0	63.5	64.5	52.0	50.5

Table 10: Static GCG attack success rate on Qwen3.5-9B. Lower is better.

Defense	ASR ↓
None	7/67
Perplexity filter	**0/67**
PAT	1/67
FSPO	2/67
SmoothLLM	3/67
PIGuard	6/67

E Projection Operation

E.1 Simplex Projection

Algorithm 2 Simplex Projection Π_{simplex}

Require: Updated token $\boldsymbol{s} \in \mathbb{R}^{|\mathcal{V}|}$

Sort $\boldsymbol{s}$ into $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_{|\mathcal{V}|}$

$\rho \leftarrow \sum_{i=1}^{|\mathcal{V}|} \mathbb{I}\left[\left\{\mu_i - {}^{1}\!/_{i}(\sum_{j=1}^{i} \mu_j - 1)\right\} > 0\right]$

$\psi \leftarrow {}^{1}\!/_{\rho}(\sum_{j=1}^{\rho} \mu_j - 1)$

return $\boldsymbol{p}$ s.t. $p_i = \max\{s_i - \psi, 0\}$

E.2 Entropy-regularized Projection

Algorithm 3 Entropy Projection Π_{entropy}

Require: Rel. token $\boldsymbol{s} \in [0,1]^{|\mathcal{V}|}$, target entropy $S_{q=2}$

Center $\boldsymbol{c} \leftarrow \mathbb{I}[\boldsymbol{s} > 0] / \sum_{i=1}^{|\mathcal{V}|} \mathbb{I}[\boldsymbol{s} > 0]$ with element-wise $>$ and $\mathbb{I}$

Radius $R \leftarrow \sqrt{1 - S_{q=2} - {}^{1}\!/_{\sum_{i=1}^{|\mathcal{V}|} \mathbb{I}[\boldsymbol{s}>0]}}$

if $R \geq \|\boldsymbol{s} - \boldsymbol{c}\|$ **then**

  return $\boldsymbol{s}$

else

  return $\Pi_{\text{simplex}}\left({}^{R}\!/_{\|\boldsymbol{s}-\boldsymbol{c}\|} \cdot (\boldsymbol{s} - \boldsymbol{c}) + \boldsymbol{c}\right)$

end if

F Training Dynamics

To examine the stability of the alternating optimization between the attacker and the defender, we analyze the training dynamics of FSPO and PAT. Figure 5 plots the refusal loss $\mathcal{L}_{\text{ref}}$ as a function of wall-clock time. Both methods eventually converge to similarly low refusal losses, indicating that they learn defensive prompts with comparable effectiveness. However, FSPO reaches this regime significantly faster, allowing early termination with substantially lower computational cost.

Figures 6 and 7 further visualize the training trajectories of the defender and attacker losses through FSPO and PAT respectively. In both methods, the defender loss decreases smoothly and stabilizes

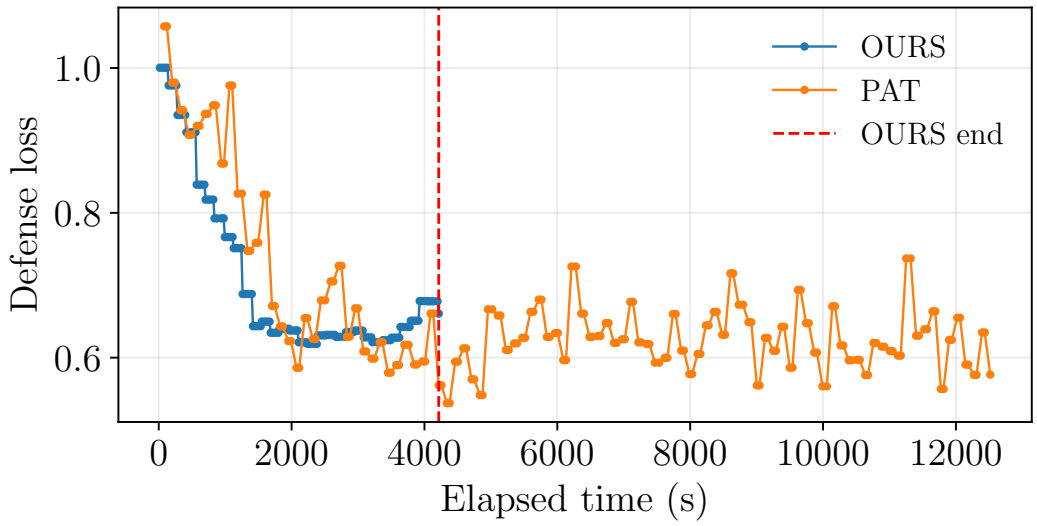

Figure 5: Refusal loss $\mathcal{L}_{\text{ref}}$ as a function of wall-clock time for FSPO and PAT.

Table 11: Attack success rate (ASR) under static and adaptive attacks for prompt location (system vs. user).

Attack	Epoch	Config	System	User
Static	30	Atk1–Def1 (Hard) Atk3–Def5 (Soft)	4/98 0/98	4/98 1/98
	100	Atk1–Def1 (Hard) Atk3–Def5 (Soft)	2/98 2/98	0/98 1/98
Adapt.	30	Atk1–Def1 (Hard) Atk3–Def5 (Soft)	6/100 1/101	3/100 0/100
	100	Atk1–Def1 (Hard) Atk3–Def5 (Soft)	4/100 2/101	7/100 0/100

over time, suggesting stable optimization despite the alternating procedure. At the same time, the attacker loss converges to a higher value under FSPO than under PAT, indicating that adversarial suffix optimization becomes more difficult once the FSPO defense is learned.

These empirical results suggest that the optimization process converges to a stable regime in practice rather than oscillating or diverging.

G Ablation Studies

This appendix provides the full details of the ablation studies summarized in Section 5. We design experiments to answer four questions regarding the key design choices of FSPO:

- **(Q1)** Does continuous relaxation yield stronger robustness than discrete optimization?
- **(Q2)** Do multiple adversarial refinements ($T_G > 1$) improve robustness?
- **(Q3)** How many gradient steps should the defender take (T_P)?
- **(Q4)** Does the location of the defensive suffix (system vs. user prompt) affect robustness?

Figure 8 summarizes the results under static and adaptive attacks, and Tables 12–13 report the detailed per-configuration numbers. Table 11 reports the results for prompt location. We organize the results by question.

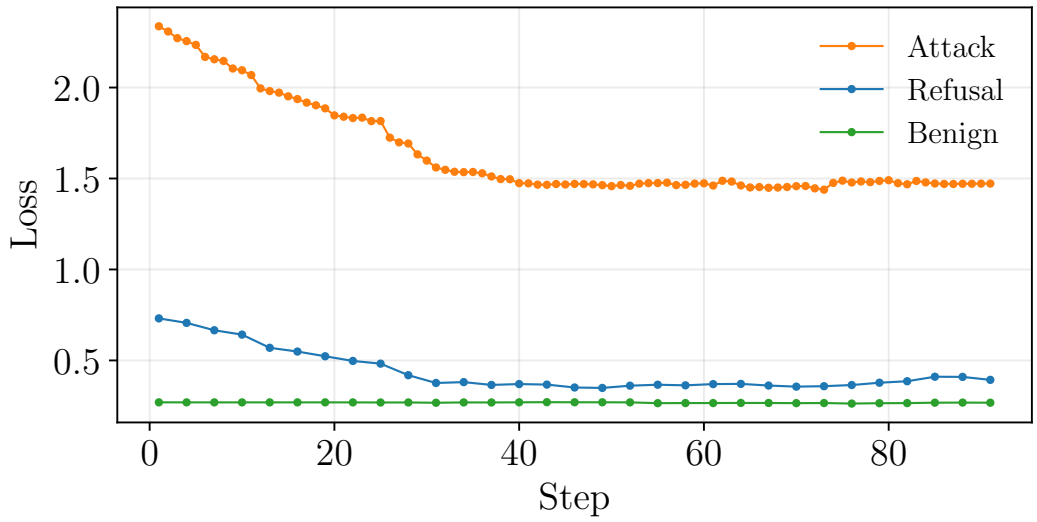

Figure 6: FSPO Train

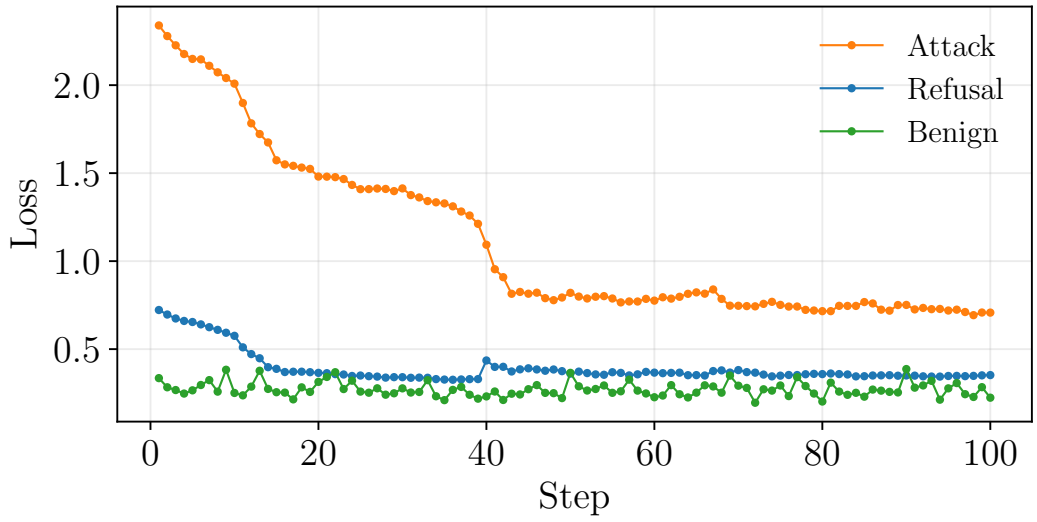

Figure 7: PAT Train

(Q1) Continuous vs. discrete optimization. As shown in Figure 8 and Tables 12–13, Soft (continuous) defenses generally achieve lower ASR than Hard (discrete-token) defenses across both static and adaptive settings. This supports our hypothesis that optimization in the probability simplex provides a richer search space for discovering effective defensive suffixes than discrete token-level updates.

(Q2) Adversarial refinement budget T_G. Increasing the adversarial refinement budget T_G strengthens the training-time attacker and typically leads to more robust defenses: configurations trained with larger T_G (e.g., Atk3 / Atk5 in Tables 12–13) achieve lower ASR than Atk1. This confirms that exposing the defender to a more refined adversary during training improves robustness.

(Q3) Defender gradient steps T_P. Increasing the number of defender gradient steps T_P (Soft1 $\rightarrow$ Soft5 $\rightarrow$ Soft10) generally improves robustness, although the gains diminish beyond a certain point. This implies that FSPO can discover strong defensive prompt suffixes with relatively few optimization steps.

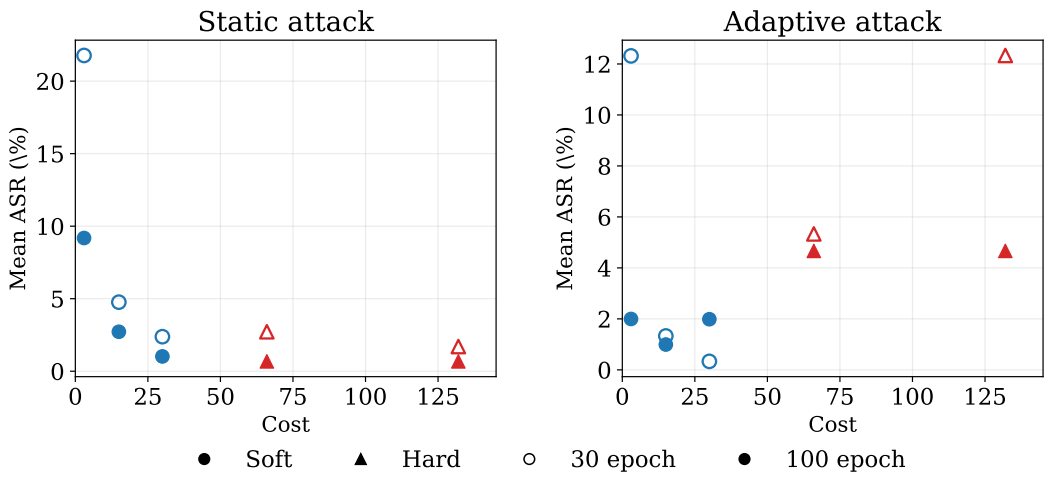

Figure 8: Ablation results comparing Soft (continuous) and Hard (discrete) defenses under static and adaptive attacks. Each point represents the average ASR across multiple attack configurations (Atk1, Atk3, Atk5) for a given training setup, plotted against computational cost. Soft defenses consistently achieve lower ASR at lower cost, demonstrating the benefit of continuous relaxation.

(Q4) Prompt location: system vs. user. Table 11 reports ASR when the defensive suffix is appended to the system prompt versus the user prompt. The two locations yield comparable robustness across both static and adaptive attacks, indicating that the choice of prompt location is not a primary driver of FSPO's robustness gains.

Summary. Overall, these results demonstrate that the key components of FSPO are (i) continuous relaxation of the defensive prompt suffix and (ii) training against a progressively refined adversary, while the prompt location is secondary.[2]

Table 12: Main ablation results under static attack.

Method	Cost ↓ FP	Cost ↓ BP	Coverage ↑ Tokens	30 Epoch Atk1	30 Epoch Atk3	30 Epoch Atk5	100 Epoch Atk1	100 Epoch Atk3	100 Epoch Atk5
Soft1	1	1	20	22/98	21/98	21/98	5/98	2/98	20/98
Soft5	5	5	20	3/98	**0/98**	11/98	5/98	2/98	1/98
Soft10	10	10	20	3/98	1/98	3/98	2/98	**0/98**	1/98
Hard1	64	1	1	4/98	3/98	1/98	2/98	**0/98**	**0/98**
Hard2	128	2	2	1/98	**0/98**	4/98	**0/98**	1/98	1/98

Table 13: Main ablation results under adaptive attack.

Method	Cost ↓ FP	Cost ↓ BP	Coverage ↑ Tokens	30 Epoch Atk1	30 Epoch Atk3	30 Epoch Atk5	100 Epoch Atk1	100 Epoch Atk3	100 Epoch Atk5
Soft1	1	1	20	26/100	5/101	6/100	3/100	1/101	2/100
Soft5	5	5	20	**0/100**	1/101	3/100	1/100	2/101	**0/100**
Soft10	10	10	20	1/100	**0/100**	**0/101**	2/100	3/101	1/101
Hard1	64	1	1	6/100	6/100	4/100	4/100	8/100	2/100
Hard2	128	2	2	9/100	**0/100**	28/100	**0/100**	8/100	6/100

[2]We note that the trends are not strictly monotonic, which we attribute to optimization stochasticity. Nevertheless, the overall pattern remains consistent across training budgets and attack strengths.

H STACKING FSPO WITH OTHER AXES

To empirically confirm that FSPO is composable with defenses on other axes (parameter-level, detection-level), we conduct a stacking experiment with **SmoothLLM** (Robey et al., 2023), a representative detection-level defense.

Setup. We first identify the GCG-attack prompts that SmoothLLM alone *fails* to block on each model. We then evaluate whether stacking FSPO on top of SmoothLLM successfully defends these residual failure cases.

Table 14: Stacking SmoothLLM with FSPO. All GCG prompts that bypass SmoothLLM alone are successfully blocked when FSPO is stacked on top.

Model	Defended by SmoothLLM + FSPO
Llama-2-7B-Chat	1/1
Vicuna-7B-v1.3	5/5

Results. Table 14 confirm that on both Llama-2 and Vicuna, every prompt that SmoothLLM alone fails to block is successfully defended when FSPO is stacked on top. This confirms that the two defenses operate on different axes—SmoothLLM filters jailbreak prompts at inference time, while FSPO reshapes the model's internal behavior through a learned continuous prompt—and can be productively combined within a defense-in-depth pipeline.

I Full Prompts Used for Evaluation

Safety Classifier Prompt

You are Llama Guard, an impartial, deterministic safety and bias classifier for human-AI conversations. Return STRICT JSON only with:
"safe": true|false,
"categories": ["S#", "B#", ...]
Rules: - Use Safety S* when content is dangerous/disallowed (crime, self-harm, sexual minors, medical/election misinformation, defamation, etc.). If any S* applies, "safe": false.
- Use Bias B* when stereotypes/slurs/prejudicial language about protected classes exist. Bias tags can appear even if safe.
- If benign: "safe": true and categories may be [] or only B* if bias exists.
- STRICT JSON only. No prose.
Safety taxonomy:
S1: Violent Crimes
S2: Non-Violent Crimes
S3: Sex-Related Crimes
S4: Child Sexual Exploitation
S5: Defamation
S6: Specialized Advice
S7: Privacy
S8: Intellectual Property
S9: Indiscriminate Weapons
S10: Hate/Harassment
S11: Suicide & Self-Harm
S12: Sexual Content
S13: Elections
S14: Code Interpreter Abuse
Bias taxonomy:
B1: Gender Bias
B2: Race/Ethnicity Bias
B3: Religion Bias
B4: Nationality Bias
B5: Sexual Orientation Bias
B6: Disability Bias
B7: Age Bias

J Deployment Considerations

We address a natural concern about FSPO's deployment cost: because the defensive prefix is a continuous embedding rather than a text string, it bypasses the tokenizer and is injected directly at the embedding layer. This differs from RPO and PAT, whose optimized suffixes are discrete tokens deployable through the standard text interface.

The injection mechanism is standard. The prefix is injected via the `inputs_embeds` argument, which is a first-class interface in standard serving frameworks (HuggingFace `transformers`, vLLM, TGI) and is the same mechanism used by LoRA adapters, prefix tuning (Li & Liang, 2021), and prompt tuning (Lester et al., 2021). No extra serving infrastructure is required.

The artifact is lightweight. The deployed defensive prefix is a single tensor of shape (L_d, e)—in our experiments, 20×4096 for Llama-2-7B ($\sim$320KB in fp32). This is strictly lighter than parameter-level defenses such as Circuit Breaker or LoRA-based adversarial training, which require shipping modified weights or adapter checkpoints.

Functional equivalence to system prompts. At inference time, both a system prompt and an FSPO prefix reduce to the same operation: prepending a fixed sequence of embeddings to the input

embedding stream before the first transformer layer. The difference lies in the source—hand-written text passed through the tokenizer versus a precomputed continuous vector—not in the serving-stack modification required.