SELECTIVE SAFETY DELIBERATION VIA UNCERTAINTY RESOLUTION WITH LEARNED INTENT TOKEN

Anonymous authors
Paper under double-blind review

ABSTRACT

Selective defenses in Large Language Models (LLMs) reduce the cost of safety deliberation by using a lightweight classifier on a model internal to decide when to intervene, yet often use it merely as a gate, routing confidently harmful and uncertain requests to the same deliberation. We empirically find that these cases require different actions: the model often complies with requests that the classifier confidently flags as harmful, whereas adversarially rewritten requests concentrate in the low-confidence region, where most errors occur. We propose RESTATE, a selective defense that resolves uncertain safety decisions before deferring to deliberation. RESTATE refuses confidently harmful requests and answers confidently benign ones. For uncertain requests, it appends a learned *intent token* which refines representation to mitigate uncertainty. The same classifier then scores the representation at this token, and only requests that remain uncertain are deliberated. Since the token is used only for reassessment and the LLM stays frozen, response generation is unaffected. On Qwen3-8B with WildJailbreak, reassessing uncertain requests raises the classifier's balanced accuracy by 7.1%p. Averaged over six inference-time defenses, RESTATE lowers attack success from 37.7% to 10.2% and over-refusal from 13.7% to 7.2% relative to always deliberating, while deliberating only 14.2% of requests.

1 INTRODUCTION

While large language models (LLMs) exhibit strong capabilities across diverse tasks, this versatility is inherently dual-use: helpful coding assistance can also be repurposed to build malware (Li et al., 2024; Bhatt et al., 2024). Training-time alignment mitigates this risk by encoding safety policies and refusal behavior into model parameters (Ouyang et al., 2022; Rafailov et al., 2023), but often fails to generalize when malicious users deliberately disguise harmful intent (Wei et al., 2023; Jiang et al., 2024a). Inference-time methods complement such training by inserting an explicit safety deliberation step before response generation (Zhang et al., 2025; Jeung et al., 2026). However, applying this deliberation to every request is not suitable in deployment, because harmful requests constitute only a small fraction of overall traffic (Cunningham et al., 2025). A practical defense should therefore be selective, invoking deliberation only when necessary.

Selective defenses use a lightweight classifier to decide which requests require intervention (Inan et al., 2023). One promising direction is probe-based classification (Han et al., 2025), which reads harmfulness from hidden states the LLM has already computed and thus adds minimal overhead. Since a single misclassification lets a harmful request bypass downstream safeguards, recent work has focused on increasing classifier capacity, for instance by escalating low-confidence inputs to higher-order decision functions (Oldfield et al., 2026) or to a larger classifier (Cunningham et al., 2026). While these advances improve accuracy, probes used merely as gating signals route confidently harmful and uncertain requests to the same deliberation (Lee et al., 2025). This wastes computation on requests whose harmfulness is already clear and limits deployment efficiency.

To address this challenge, we analyze safety failures in both the probe and the base model, revealing two key observations. First, the probe can flag a request as harmful with high confidence while the model still complies. Second, adversarially rewritten requests concentrate in the probe's low-confidence region, where most classification errors also occur. Together, these observations motivate a tailored action for each confidence region. Because the model can comply with requests the probe

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.

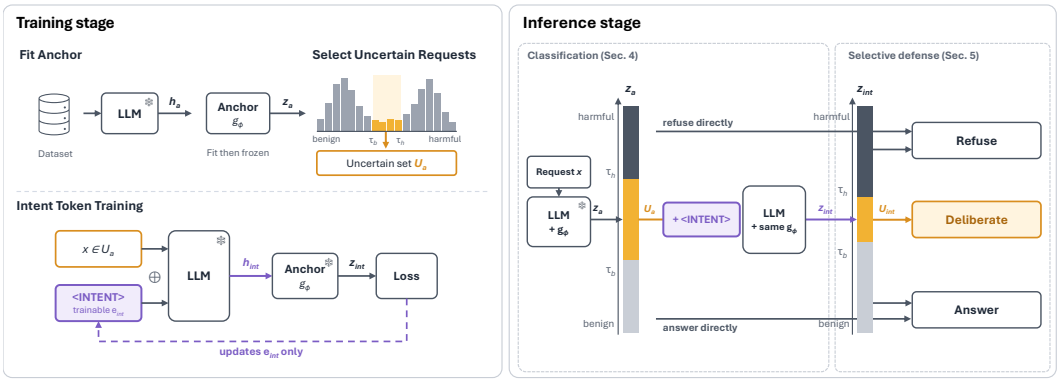


Figure 1: **Overview of RESTATE.** RESTATE reassesses uncertain requests with a learned intent token read by the same frozen probe, and defers only those that remain uncertain to deliberation.

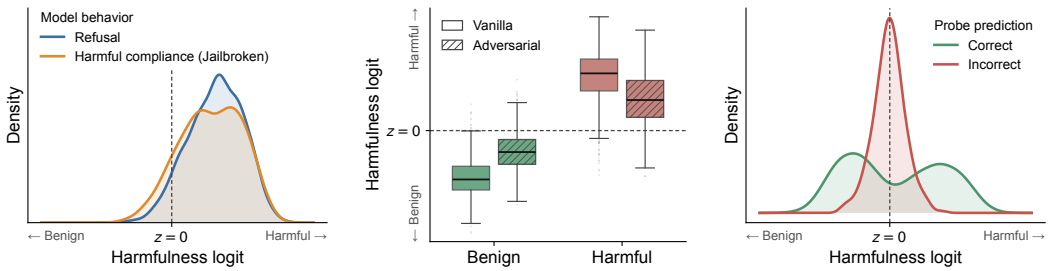
confidently flags as harmful, such requests should be refused directly, without invoking deliberation. Low-confidence requests, where the uncertainty is not sufficiently resolved, should be deferred to deliberation. Efficiency then depends on how this uncertain set is defined and how much of it can be resolved before deliberation is invoked.

Building on these observations, we propose **RESTATE** (**RES**olving Uncer**TA**in**T**y in Saf**E**ty Decisions with Intent Token), which selectively reassesses uncertain safety predictions before deliberation by refining the representation with a learned token (Fig. 1). Specifically, a lightweight linear probe (*anchor*) first assesses each request from the LLM's existing hidden state, with an uncertainty band calibrated to target error rates. For requests within this band, RESTATE appends a learned *intent token* to encode the request's intent more explicitly in one additional forward step, and the same anchor scores the resulting hidden state. With the LLM and anchor frozen, only the token embedding is trained, primarily on anchor-uncertain examples to specialize intent token in resolving the anchor's uncertainty. The intent token is excluded from response generation, leaving the normal-generation path unchanged. Overall, confident requests at either stage are answered or refused directly, while only requests that remain uncertain under a second calibrated band are deferred to deliberation.

We evaluate RESTATE on Qwen3-8B (Yang et al., 2025) with WildJailbreak (WJ; Jiang et al., 2024a) and seven external benchmarks. As a classifier, RESTATE improves balanced accuracy over the anchor by 7.1%p on WJ and 6.0%p externally, whereas higher-capacity probes gain at most 1.9%p. As a selective defense averaged over six deliberation methods, RESTATE lowers the attack success and over-refusal of always deliberating by 27.5%p and 6.5%p on WJ, while deliberating only 14.2% of requests. Its utility stays within about 1%p of vanilla generation, while always deliberating degrades performance, showing that deliberation is most effective when applied selectively. Under adaptive jailbreaks, it lowers the attack success of always deliberating by a further 18.1%p on HarmBench.

2 RELATED WORK

Safety alignment can be shallow (Qi et al., 2025) and is often circumvented when harmful intent is disguised (Wei et al., 2023; Jiang et al., 2024a). Inference-time defenses add deliberation before responding, for instance by analyzing the intention of a request (Zhang et al., 2025), prompting safety reasoning at the start of thinking (Jeung et al., 2026), reasoning over explicit safety specifications (Guan et al., 2024), or adjusting the decoding distribution (Xu et al., 2024; Hu et al., 2026); any of them can serve as the deliberation step of RESTATE. Selective defenses reduce this cost by first screening each request with a lightweight classifier (Cunningham et al., 2025; 2026). Probes are particularly efficient, as they reuse activations the served model already computes, and their predictions can trigger a dedicated refusal head (Han et al., 2025) or activation steering (Lee et al., 2025). Recent work strengthens the probe through higher-order decision functions (Oldfield et al., 2026), aggregation across layers and tokens (Jiao et al., 2026; Meyoyan & Del Corro, 2026), or escalation of uncertain inputs to a larger classifier (Cunningham et al., 2026; McKenzie et al., 2026), a form of selective classification (Geifman & El-Yaniv, 2017). In these designs, however, the prediction acts only as a binary gate, and the added computation targets the decision function over a fixed repre-



(a) Behavior mismatch (b) Adversarial boundary shift (c) Error concentration

Figure 2: **Failure regimes in selective safety decisions.**

sentation. RESTATE instead refuses confidently harmful requests directly and, for uncertain ones, refines the representation the probe reads.

Recognizing harmful intent and acting on it appear distinct in aligned LLMs: harmfulness is encoded separately from the refusal direction (Arditi et al., 2024; Zhao et al., 2026), and the refusal outcome of reasoning models is largely fixed early in their thinking (Ri et al., 2026). Prior work thus decouples safety computation from generation, via safety adapters on the base model's representations (Krishna et al., 2025) or safety decisions made before chain-of-thought generation (Chen et al., 2026). Motivated by this principle, RESTATE separates intent assessment from generation at the representation level. Learned tokens can condition a frozen model on a task (Lester et al., 2021; Li & Liang, 2021) or add computation before an output (Goyal et al., 2024), and appending a prompt or repeating the input yields better representations from frozen decoders (Jiang et al., 2024b; Springer et al., 2025). Closest to ours, optimized safety prompts and suffixes steer a model's refusal behavior (Zheng et al., 2024; Zhou et al., 2024); our intent token instead only forms the representation the probe reads, leaving response generation unaffected.

3 METHOD

Selective safety mechanisms must provide effective safeguards while limiting inference overhead (Cunningham et al., 2025; 2026). This requires deciding when an initial assessment is sufficient to act on and when deliberation is needed. To understand what should guide this distinction, we first examine safety failures and the model internals associated with them (Sec. 3.1), and then introduce RESTATE, which addresses them (Sec. 3.2).

3.1 MOTIVATION: TWO FAILURE REGIMES IN SELECTIVE SAFETY DECISIONS

Observations. In this work, we focus on request harmfulness detection. For the observations here, we use a linear probe as a representative lightweight classifier, applied to the final-layer hidden state at the last token of the input prompt.[1] We conduct this analysis using Qwen3-8B (Yang et al., 2025) and WildJailbreak (Jiang et al., 2024a). WildJailbreak provides original (*Vanilla*) requests and adversarially transformed variants (*Adversarial*) produced through red-teaming methods, allowing us to examine how intentionally obfuscated intent affects the harmfulness assessment.

We find that the probe can predict a request as harmful with high confidence even when the model subsequently generates a harmful response (Fig. 2a). This mismatch suggests that even when harmful intent is strongly encoded and can be read out with high confidence, the model's eventual behavior need not align with that signal. Additionally, we observe a distinct failure regime in which the harmfulness assessment itself becomes unreliable. Although the probe is trained only to predict request harmfulness, adversarial variants of both harmful and benign requests lie substantially closer to its decision boundary than their vanilla counterparts (Fig. 2b), with most classification errors concentrated in the same low-confidence region (Fig. 2c). These patterns suggest that probe confidence can serve as a practical signal for identifying requests that merit further assessment.

[1]For clarity, we note that this probe serves as the *anchor* in Sec. 3.2.

Implications. Taken together, these observations indicate that requests should be handled differently according to their confidence levels in the probe. When harmful intent is identified with high confidence, the appropriate action is to enforce a consistent refusal rather than relying on the model's native refusal behavior. However, immediate commitment is less reliable when the decision remains uncertain, and motivates reassessment before deliberation. This leads to the next question: *how can we efficiently resolve uncertainty before resorting to full safety deliberation?*

To answer this question, rather than simply enlarging the probe capacity whose benefit can be limited by what its input representation exposes, we draw inspiration from prior work that separates safety-specific computation from ordinary generation (Krishna et al., 2025; Chen et al., 2026). This separation is also supported by evidence that recognizing harmful intent and acting on it are distinct in aligned LLMs (Zhao et al., 2026; Ri et al., 2026), and the behavior mismatch in Fig. 2a reflects the same gap in our setting. However, hidden states the probe reads serve response generation rather than safety assessment, where safety-relevant information need not be fully exposed.

This motivates applying the separation to the representation itself: an additional state dedicated to intent assessment can be computed and read by the same probe, deferring to deliberation only the requests that remain uncertain after this reassessment.

3.2 RESTATE: RESOLVING UNCERTAINTY IN SAFETY DECISIONS WITH INTENT TOKEN

We propose RESTATE, which selectively reassesses uncertain harmfulness predictions by refining the representation the classifier reads, rather than escalating them to a stronger classifier. A lightweight classifier, which we call the *anchor*, handles confident requests (Sec. 3.2.1), while uncertain requests receive a learned *intent token* and are re-evaluated by the same frozen anchor (Sec. 3.2.2). Requests that remain uncertain are deferred to deliberation (Sec. 3.2.3).[2]

Notation. Let x be a chat-formatted request of T tokens with label $y \in \{0, 1\}$, where $y = 1$ indicates x is harmful, and let $E(x) = (e_1, \ldots, e_T) \in \mathbb{R}^{T\times d}$ be its input embeddings. We denote by $H_\theta(E)_t \in \mathbb{R}^d$ the final-layer hidden state of the LLM at position t for input embeddings E.

3.2.1 ANCHOR HARMFULNESS ASSESSMENT

We define the *anchor probe* (anchor) as a linear classifier $g_\phi(h) = w^\top h + b$ with parameters $\phi = (w, b)$. Anchor reads the hidden state at the last request token, $h_a(x) = H_\theta(E(x))_T$, and outputs a harmfulness score $z_a(x) = g_\phi(h_a(x))$, where larger scores indicate more harmful requests. With the LLM kept frozen, we fit anchor on labeled requests $\mathcal{D}_a = \{(x_i, y_i)\}_{i=1}^{N_a}$, by minimizing

$$\mathcal{L}_a(\phi) = \frac{1}{|\mathcal{D}_a|} \sum_{(x,y)\in\mathcal{D}_a} \mathrm{BCE}(z_a(x), y) + \frac{\lambda_a}{2}\|w\|_2^2, \tag{1}$$

where BCE is the binary cross-entropy on the logit and λ_a weights an ℓ_2 penalty on w.

On a held-out calibration set, we choose the largest $\tau_b < 0$ and the smallest $\tau_h > 0$ such that

$$\widehat{\Pr}(z_a(x) \le \tau_b \mid y = 1) \le \alpha_{\mathrm{FN}}, \qquad \widehat{\Pr}(z_a(x) \ge \tau_h \mid y = 0) \le \alpha_{\mathrm{FP}}, \tag{2}$$

where $\widehat{\Pr}$ is the fraction of calibration requests of the given class, so the two terms are the empirical false negative rate (harmful requests confidently allowed) and false positive rate (benign requests confidently refused), bounded by α_{FN} and α_{FP}. We define the anchor uncertain set $\mathcal{U}^a = \{x : \tau_b < z_a(x) < \tau_h\}$, *i.e.*, requests inside the *anchor band*, which are reassessed; other requests are refused directly ($z_a(x) \ge \tau_h$) or answered normally ($z_a(x) \le \tau_b$).

3.2.2 INTENT ELICITATION WITH A LEARNED INTENT TOKEN

To reassess requests in $\mathcal{U}^a$, we append the *intent token*, a single trainable embedding $e_{\mathsf{int}} \in \mathbb{R}^d$, so that the LLM forms a representation exposing the request's intent more clearly than $h_a(x)$.

We construct the training set $\mathcal{D}_{\mathsf{int}}$ for the intent token from a labeled pool $\mathcal{P}$ disjoint from both $\mathcal{D}_a$ and the calibration set. Since the intent token is used only when the anchor's initial assessment is

[2] See Appendix B for details of each component, including training, calibration, and the inference algorithm.

uncertain, most examples come from $\mathcal{P} \cap \mathcal{U}^a$, following cascade training (Viola & Jones, 2001). We also add a small set of requests from $\mathcal{P} \setminus \mathcal{U}^a$ as a regularizer, since the requests that fall in the band at deployment need not match those seen in training.

For $x \in \mathcal{U}^a$, we append the intent token e_{int} to the request embeddings $E(x)$ and score its final hidden state $h_{\text{int}}(x) = H_\theta(E(x) \oplus e_{\text{int}})_{T+1}$ with the anchor, $z_{\text{int}}(x) = g_\phi(h_{\text{int}}(x))$. Only e_{int} is optimized while the LLM and the anchor remain frozen; the decision boundary of the anchor is therefore fixed, and training instead moves the request representation $h_{\text{int}}(x)$ to the correct side of this boundary. To train e_{int}, we use a robust binary cross-entropy (Robust BCE), motivated by adversarial training in the embedding space (Miyato et al., 2016; Madry et al., 2017):

$$\mathcal{L}_{\text{int}}(e_{\text{int}}) = \frac{1}{|\mathcal{D}_{\text{int}}|} \sum_{(x,y)\in\mathcal{D}_{\text{int}}} \Big[\text{BCE}(z_{\text{int}}(x), y) + \lambda_{\text{adv}} \max_{\delta\in\Delta} \text{BCE}(z_{\text{int}}(x;\delta), y) \Big], \tag{3}$$

where $z_{\text{int}}(x;\delta) = g_\phi\big(H_\theta((E(x)+\delta) \oplus e_{\text{int}})_{T+1}\big)$ adds a perturbation $\delta \in \mathbb{R}^{T\times d}$ to the request embeddings, and Δ bounds each token's perturbation δ_t to a small fraction of $\|e_t\|_2$. The second term evaluates the loss under the most harmful perturbation, so intent token must keep the request on the correct side of the boundary even when its embeddings shift slightly.

3.2.3 Inference and Selective Safety Routing

After training, RESTATE processes each request in two stages. It first computes $z_a(x)$ from the hidden state the LLM already produces for the request, and requests outside the anchor band are answered or refused immediately. For requests in $\mathcal{U}^a$, it then appends the intent token and runs one additional forward step to obtain $z_{\text{int}}(x)$. When RESTATE serves as a harmfulness classifier (Sec. 4), the anchor's confident predictions are kept and each $x \in \mathcal{U}^a$ is labeled $\hat{y}(x) = \mathbf{1}[z_{\text{int}}(x) > 0]$. When it serves as a selective defense (Sec. 5), reassessment may not resolve every request, so we replace the zero threshold with an *intent band* $\tau_b^{\text{int}} < 0 < \tau_h^{\text{int}}$, calibrated as in Eq. 2 with targets $\alpha_{\text{FN}}^{\text{int}}$ and $\alpha_{\text{FP}}^{\text{int}}$ on a separate calibration split restricted to requests in $\mathcal{U}^a$. Requests inside it, $\mathcal{U}^{\text{int}} = \{x \in \mathcal{U}^a : \tau_b^{\text{int}} < z_{\text{int}}(x) < \tau_h^{\text{int}}\}$, are deferred to deliberation, and the rest follow the intent token's decision:

$$\pi(x) = \begin{cases} \text{Generate}(x), & z_a(x) \le \tau_b \ \text{ or } \ \big(x \in \mathcal{U}^a,\ z_{\text{int}}(x) \le \tau_b^{\text{int}}\big), \\ \text{Refuse}, & z_a(x) \ge \tau_h \ \text{ or } \ \big(x \in \mathcal{U}^a,\ z_{\text{int}}(x) \ge \tau_h^{\text{int}}\big), \\ \text{Deliberate}(x), & x \in \mathcal{U}^{\text{int}}, \end{cases} \tag{4}$$

where Deliberate applies an inference-time defense. In all cases, the response is generated from the original request without the intent token.

4 Uncertainty Resolution for Harmfulness Classification

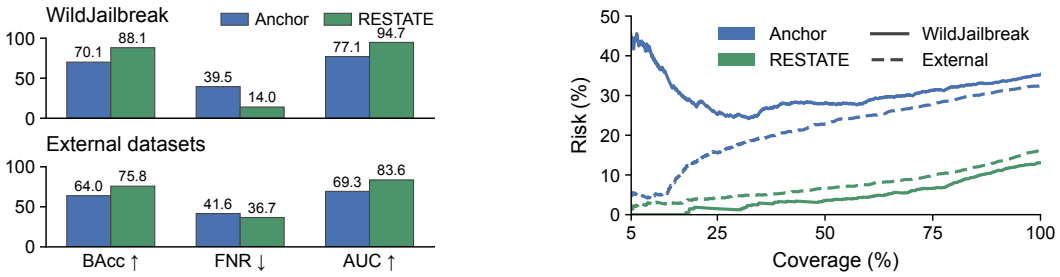
We first evaluate RESTATE as a request harmfulness classifier. Our evaluation asks two questions: whether RESTATE outperforms existing probe-based classifiers, and whether its gains come specifically from resolving requests that the anchor cannot classify confidently.

4.1 Experimental Setup

We use Qwen3-8B (Yang et al., 2025) in thinking mode, and the anchor reads the final-layer hidden state at the last token of the request. Anchor fitting, uncertain bands calibration, and intent token training use disjoint subsets of the WildJailbreak (WJ; Jiang et al., 2024a) train split, totaling 15,100 examples. We fix the target error rates for all benchmarks, setting $\alpha_{\text{FN}} = \alpha_{\text{FP}} = 1\%$ for the anchor band. We evaluate on held-out WJ data (evaluation split and unused train split data) and external benchmarks: HarmBench (Mazeika et al., 2024), JailbreakBench (Chao et al., 2024), SORRY-Bench (Xie et al., 2025), XSTest (Röttger et al., 2024), CoCoNot (Brahman et al., 2024), AEGIS2 (Ghosh et al., 2025), and ToxicChat (Lin et al., 2023). External data are pooled with original labels. We report balanced accuracy (BAcc), FNR, F1, and AUC. We compare RESTATE against activation-based probes in two groups. (i) *Static probes* add decision-function capacity (Bilinear, Hewitt & Liang, 2019; MLP, Oldfield et al., 2026), aggregate evidence across layers and tokens (SIREN, Jiao et al., 2026; Token × Layer, Meyoyan & Del Corro, 2026), or read a fixed safety-oriented suffix (Suffix, Cunningham et al., 2025). (ii) *Adaptive probes* escalate low-confidence

Table 1: **Harmfulness classification on WJ and external benchmarks.** Best and second-best results are **bolded** and underlined.

	WildJailbreak				External datasets			
Method	**BAcc ↑**	**FNR ↓**	**F1 ↑**	**AUC ↑**	**BAcc ↑**	**FNR ↓**	**F1 ↑**	**AUC ↑**
Static classifiers								
Anchor	85.50	17.04	89.39	93.40	79.19	18.10	65.73	87.94
Bilinear	86.55	17.09	89.59	93.96	77.71	21.19	64.14	86.04
MLP	85.50	11.35	92.07	93.85	81.12	22.50	70.07	88.82
Suffix	84.35	13.26	90.95	92.11	83.02	22.93	73.87	**91.86**
Token × Layer	88.57	11.48	92.67	95.66	78.61	18.85	65.01	87.76
SIREN	88.25	8.00	94.15	95.41	79.94	16.88	66.59	88.12
Adaptive classifiers								
Anchor → MLP	83.19	11.65	91.45	89.23	77.63	18.57	63.53	78.85
EE-MLP	85.56	13.39	91.15	92.77	81.08	24.52	70.68	88.68
TPC	85.74	12.04	91.81	88.75	81.03	21.43	69.57	84.41
EMA → Suffix	84.44	13.87	90.68	91.16	83.10	23.68	74.35	88.56
RESTATE (Ours)	**92.62**	**7.70**	**95.22**	**96.79**	**85.20**	**16.41**	**75.53**	90.76



(a) Classification metrics

(b) Risk–coverage

Figure 3: **Comparison on the anchor uncertain set $\mathcal{U}^a$.** (a) RESTATE consistently improves over the anchor. (b) Risk–coverage curves, where risk is the error rate among the most confident requests.

requests to a further stage, adding capacity progressively (TPC, Oldfield et al., 2026; EE-MLP, Teerapittayanon et al., 2016) or replacing the intent token with a conventional probe (Anchor → MLP; EMA → Suffix, Cunningham et al., 2025). See App. A for details on the experimental setup.

4.2 Main Results

RESTATE generally outperforms all baselines on both WJ and the external benchmarks (Tab. 1), improving BAcc over the anchor, a conventional linear probe, by 7.1 and 6.0 points, respectively. Interestingly, stronger decision functions bring limited gains: Bilinear and MLP, as well as their adaptive counterparts TPC and EE-MLP, improve BAcc by at most 1.9 points, suggesting that the bottleneck does not lie in classifier capacity alone. Richer feature aggregation is similarly limited, as SIREN and Token × Layer gain up to 3.1 points on WJ but barely transfer to the external benchmarks. Suffix is the strongest external baseline but slightly lowers WJ BAcc, and Anchor → MLP, which shares our routing, falls below the anchor on both pools. These results indicate that RESTATE mainly benefits from refining the representation of uncertain requests with a learned intent token.

To examine where these gains arise, we analyze the anchor uncertain set $\mathcal{U}^a$, which covers 38.4% of WJ requests and 49.6% of external requests. On these requests, the intent token raises BAcc from 70.1 to 88.1 on WJ and from 64.0 to 75.8 externally, and lowers FNR on WJ from 39.5 to 14.0 (Fig. 3a). The intent score also provides a more reliable confidence signal within $\mathcal{U}^a$ (Fig. 3b): when requests in this set are accepted in order of confidence, the intent token yields a lower error rate than the anchor at every coverage on both pools.

4.3 Additional Analyses

Generalization across classifiers and backbones. In this section, we mainly used WJ to provide deeper insights of RESTATE. We first check whether the gain simply comes from additional training

Table 2: **Probe retraining vs. intent token on WJ.** Lower block transfers the token to new classifiers.

	Anchor		MLP	
Method	**BAcc↑**	**FNR↓**	**BAcc↑**	**FNR↓**
Anchor	85.50	17.04	86.17	15.30
+ retrained (joint)	86.39	15.26	85.50	11.35
+ retrained (sequential)	79.34	15.43	85.99	12.91
+ intent token	**92.62**	**7.70**	89.86	**4.78**
Independent probe	84.23	15.65	86.32	14.22
+ transferred intent token	90.47	8.48	**92.49**	7.57

Table 3: **Intent token ablation.** Data: requests outside $\mathcal{U}^a$; Loss: Robust BCE.

Method	**Data**	**Loss**	**BAcc↑**	**FNR↓**	**AUC↑**
Anchor	–	–	85.50	17.04	93.40
+ untrained token	–	–	80.26	21.83	90.48
+ intent token	✗	✗	85.98	8.43	94.20
+ intent token	✗	✓	90.92	12.48	96.27
+ intent token	✓	✗	86.73	10.26	94.53
+ random escalation	✓	✓	88.41	13.25	95.08
RESTATE (Ours)	✓	✓	**92.62**	**7.70**	**96.79**

Table 4: **WJ classification across backbones.**

	BAcc ↑		FNR ↓		AUC ↑	
Backbone	Anchor	Ours	Anchor	Ours	Anchor	Ours
Qwen3-8B (non-thinking)	85.79	**91.76**	13.13	**9.22**	93.49	**96.52**
Qwen3-32B	83.67	**84.74**	11.09	**6.61**	92.50	**93.21**
GPT-OSS-20B	82.47	**83.55**	15.65	**12.70**	91.59	**91.63**

Table 5: **Second intent token.**

Method	**BAcc ↑**	**FNR ↓**	**AUC ↑**
Anchor	85.50	17.04	93.40
+ intent token 1	92.65	6.65	97.50
+ intent token 2	**93.44**	**5.48**	**98.03**

data by retraining the anchor on the same data as RESTATE, either jointly or sequentially (Tab. 2). Retraining changes BAcc by at most +0.9 and can even lower it, whereas the intent token raises it by 7.1, suggesting that a separate representation for uncertain requests is more effective than refitting a single probe on the original one. The intent token also improves an MLP probe when trained with it, suggesting that it applies to other probe architectures.

Since the intent token is trained against a fixed anchor, it might only learn to push representations across that anchor's decision boundary. To test this, we apply same intent token without retraining to *independent probes*, probes fit on WJ data disjoint from the anchor's training data. As shown in lower block of Tab. 2, it raises BAcc by 6.2 points for both independent linear and MLP probes, which is close to its gain on the anchor. This indicates that the token refines the representation itself rather than exploiting a specific probe.

Finally, Tab. 4 shows that RESTATE generalizes to various models. With its own anchor and intent token, it consistently improves over the anchor on Qwen3-8B in non-thinking mode, a larger model, Qwen3-32B (Yang et al., 2025), and a different model family, GPT-OSS-20B (Agarwal et al., 2025).

Ablation of the intent token. Tab. 3 shows that merely appending a token for additional computation does not help. An untrained copy of the last token's embedding lowers BAcc by 5.2. Training the token with plain BCE on $\mathcal{U}^a$ alone improves over the anchor, but adding requests outside $\mathcal{U}^a$ (*Data*) and Robust BCE (*Loss*) each improve it further, with the best results when combined. Where intent token is applied also matters. Escalating a random subset of the same size as $\mathcal{U}^a$ to it (*random escalation*) still improves over the anchor but falls short of focusing on $\mathcal{U}^a$. Finally, we cascade a second intent token, trained in the same way, on requests that remain inside a calibrated band after the first (Tab. 5). It yields an additional gain, although calibrating each stage requires stricter error targets that widen the bands and raise the escalation rate from 38.4% to 70.8%.

5 From Harmfulness Assessment to Selective Safety

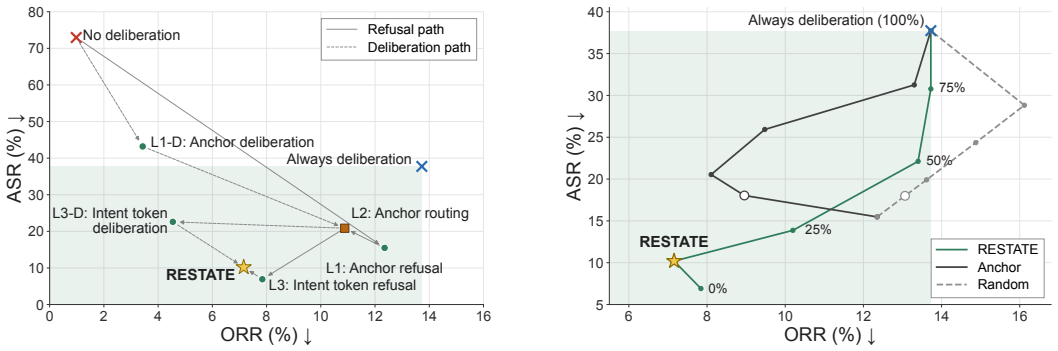
We now evaluate RESTATE for a selective defense mechanism. Our evaluation asks whether resolving uncertainty before deliberation improves the trade-off between attack success and over-refusal, and whether this gain comes from targeting uncertain requests rather than from deliberating more.

5.1 Experimental Setup

We mainly consider inference-time defenses as the deliberation step: Backtranslation (BT; Wang et al., 2024), Intention Analysis (IA; Zhang et al., 2025), LLM Self-Defense (LSD; Phute et al., 2023), RA-LLM (RA; Cao et al., 2024), Self-Reminder (SR; Xie et al., 2023), and ZS-SafePath (ZS; Jeung et al., 2026). For each defense, we compare five policies: *No deliberation*, which generates a vanilla response to every request; *Always deliberation*, which applies the defense to every request; *Random*, which deliberates a random subset of $\mathcal{U}^a$ of the same size as $\mathcal{U}^{\text{int}}$ and follows the intent

Table 6: **ASR (ORR) per defense and deliberation rate (Rate; %).** Best and second-best among selective policies are **bolded** and underlined.

	WildJailbreak							External datasets						
Policy	**BT**	**IA**	**LSD**	**RA**	**SR**	**ZS**	**Rate**	**BT**	**IA**	**LSD**	**RA**	**SR**	**ZS**	**Rate**
No (Vanilla)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	0.0	24.72 (3.23)	24.72 (3.23)	24.72 (3.23)	24.72 (3.23)	24.72 (3.23)	24.72 (3.23)	0.0
Always	33.26 (25.69)	7.83 (29.80)	67.13 (5.10)	59.61 (6.27)	32.00 (5.88)	26.61 (9.61)	100.0	13.35 (21.86)	1.36 (17.57)	23.12 (7.87)	13.29 (11.66)	6.55 (9.83)	8.34 (7.93)	100.0
Random	17.34 (15.20)	13.89 (15.83)	21.96 (11.61)	20.98 (11.81)	17.24 (11.69)	16.57 (12.24)	14.2	5.52 (19.41)	3.18 (18.19)	7.34 (16.01)	5.54 (16.88)	4.21 (16.41)	4.57 (15.96)	22.7
Anchor	18.17 (16.27)	7.57 (20.59)	31.78 (6.08)	30.70 (5.49)	19.35 (6.67)	17.52 (10.20)	38.4	5.95 (14.82)	**1.45** (15.73)	8.94 (8.66)	6.13 (11.74)	4.04 (11.32)	4.38 (10.00)	49.6
RESTATE (Ours)	**8.96** (**9.41**)	**5.43** (**10.20**)	**14.57** (**5.29**)	**13.96** (**4.90**)	**9.65** (**5.69**)	**8.61** (**7.45**)	14.2	**4.49** (**12.11**)	2.30 (**11.74**)	**5.81** (**8.53**)	**4.93** (**10.47**)	**3.83** (**9.78**)	**4.05** (**8.97**)	22.7



(a) Probe ladder.

(b) Budget–outcome trade-off.

Figure 4: **Selective safety decisions on WJ over six defenses.** Open circles: matched budget.

token's prediction for the rest of $\mathcal{U}^a$; *Anchor*, which removes the intent token from RESTATE and thus deliberates all of $\mathcal{U}^a$; and RESTATE, which reassesses $\mathcal{U}^a$ with the intent token and deliberates only the remaining uncertain requests ($\mathcal{U}^{\text{int}}$). We fix $\alpha_{\text{FN}}^{\text{int}} = \alpha_{\text{FP}}^{\text{int}} = 5\%$ for the intent band. We report the attack success rate (ASR) of harmful requests and the over-refusal rate (ORR) of benign requests, and measure utility on IFBench (Pyatkin et al., 2026), MATH-500 (Lightman et al., 2024), and GPQA-Diamond (GPQA-D; Rein et al., 2023). Detailed results and analyses are in Appendix C.

5.2 Main Results

RESTATE consistently achieves the lowest ASR and ORR among the selective policies across most defenses (Tab. 6). Averaged over the six defenses, it lowers the ASR and ORR of always deliberation on WJ by 27.5%p and 6.6%p while deliberating only 14.2% of requests, with the same trend on the external benchmarks. Compared with Anchor, which deliberates all of $\mathcal{U}^a$, resolving uncertainty with the intent token halves ASR on WJ while also lowering ORR, and random deliberation at the same budget yields considerably higher ASR and ORR. Together, these results suggest that the intent token successfully identifies which requests still require deliberation. RESTATE best preserves utility (Tab. 7), staying within about one point of no deliberation, whereas always deliberation loses up to 6.4%p.

Table 7: **Task utility (%).**

Policy	**IFBench**	**MATH-500**	**GPQA-D**
No deliberation	35.45	91.78	59.09
Always deliberation	34.67	85.40	55.56
Random deliberation	34.96	91.48	57.26
Anchor	33.33	91.18	57.07
RESTATE	35.45	91.58	58.08

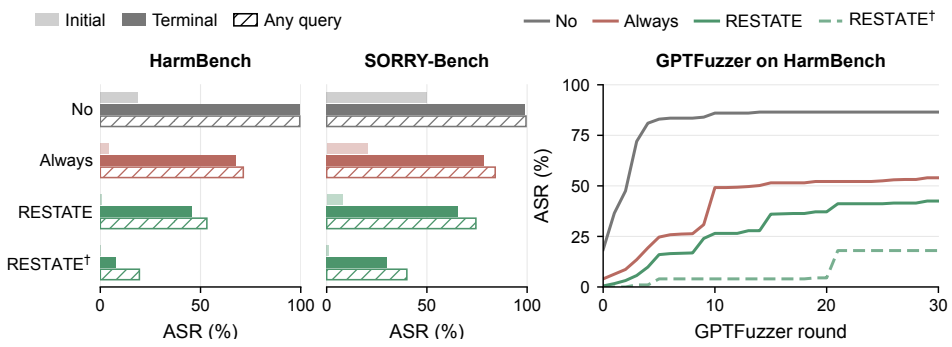


Figure 5: **Robustness to jailbreak attacks.** Left: ASR on HarmBench and SORRY-Bench, averaged over attacks for initial request, terminal rewrite, and any queried rewrite. Right: cumulative any-query ASR over GPTFuzzer rounds. RESTATE† refuses remaining uncertain requests.

5.3 Additional Analyses

Decomposing the gains. We decompose the gains of RESTATE with a probe ladder that adds its components one at a time (Fig. 4a). Deliberation alone is insufficient on WJ, which contains many adversarially rewritten requests: always deliberation leaves 37.7 ASR, and deliberating every request the anchor predicts as harmful (L1-D) leaves 43.2. Refusal alone instead causes excessive over-refusal, as refusing these requests (L1) raises ORR from 1.0 to 12.4. Routing by anchor confidence (L2), which refuses confidently harmful requests and deliberates uncertain ones, balances the two. Since the intent token resolves much of the anchor's uncertainty, reassessing the uncertain requests with it (L3) further reduces both ASR and ORR, and deliberating only those that remain uncertain yields RESTATE. Under a matched deliberation budget, this resolution also directs deliberation to the requests that need it most, yielding a better ASR–ORR trade-off than Anchor and Random (Fig. 4b).

Robustness to adaptive attacks. We evaluate each policy against three adaptive jailbreak attacks, PAIR (Chao et al., 2025), TAP (Mehrotra et al., 2024), and GPTFuzzer (Yu et al., 2023), which iteratively rewrite HarmBench (Mazeika et al., 2024) and SORRY-Bench (Xie et al., 2025) requests against the policy's final response. Averaged over the three attacks, RESTATE substantially lowers any-query ASR from 71.3 to 53.2 on HarmBench relative to always deliberation, with a similar trend on SORRY-Bench (Fig. 5, left and middle). For GPTFuzzer, which proceeds in rounds, it keeps cumulative ASR lower at every round (Fig. 5, right). To separate failures of routing from deliberation failures, we evaluate RESTATE†, which refuses requests that remain uncertain; since all attacked behaviors are harmful, this corresponds to RESTATE with an oracle deliberation step. Any safeguard can serve as the deliberation step, and we restrict it to inference-time defenses for cost efficiency; RESTATE† thus shows how far a stronger but more expensive step could improve RESTATE. Their gap highlights the potential for stronger deliberation to reduce ASR. While always deliberation could also benefit from a stronger safeguard, RESTATE uses it more efficiently by restricting its additional cost to requests that remain uncertain.

6 Conclusion

We present RESTATE, a selective defense that acts directly on confident harmfulness predictions and reassesses uncertain ones with a learned intent token read by the same frozen probe. RESTATE is effective both as a harmfulness classifier, outperforming existing probe-based classifiers, and as a selective defense, lowering both ASR and ORR while deliberating only the requests that remain uncertain. Its intent token also improves independently trained probes without retraining, indicating that it resolves uncertainty in the representation rather than fitting a particular probe. These results suggest that refining what a probe reads is an effective lever for selective safety.

Limitations. Under adaptive attacks, most remaining failures of RESTATE come from deliberation rather than routing. We use inference-time defenses to keep deliberation cheap, but stronger and more expensive safeguards could reduce these failures further. Since RESTATE deliberates only uncertain requests, combining it with such safeguards is a promising direction for future work.

AI USE STATEMENT

In this work, we used AI-based writing assistants to improve sentence structure, correct grammatical errors, and enhance overall readability, as well as to assist in searching for related work. These tools were used only for editorial support and literature discovery and did not contribute to the development of technical content, research methodology, or experimental analysis. All scientific ideas, results, and conclusions presented in the paper were conceived and authored by the researchers. The use of AI assistance did not affect the originality or intellectual contributions of the work.

ETHICS STATEMENT

By improving how LLMs distinguish between requests that can be handled directly and those that require additional safety deliberation, RESTATE aims to make inference-time safeguards both more reliable and more computationally efficient. Selective safety mechanisms can reduce unnecessary intervention on benign requests while retaining additional safeguards for requests that are difficult to assess, which is relevant to the practical deployment of aligned language models (Cunningham et al., 2026).

However, research on LLM safety necessarily involves potentially harmful, toxic, and adversarial content (Mazeika et al., 2024; Jiang et al., 2024a). Our experiments use existing safety benchmarks containing such requests and evaluate established jailbreak methods (Chao et al., 2025; Mehrotra et al., 2024; Yu et al., 2023), including adaptive attacks, to measure the robustness of the proposed defense. These materials are used solely for controlled safety evaluation, and our work does not introduce a new jailbreak technique. We reproduce only limited safety-sensitive examples when necessary for analysis and do not provide successful harmful completions or procedural instructions for harmful activities. RESTATE should be used as one component of a broader safety framework rather than as a standalone guarantee: errors in safety classification or downstream deliberation may still occur, motivating complementary safeguards and continued monitoring in real-world deployment.

REPRODUCIBILITY STATEMENT

We provide the datasets, split construction, label definitions, and evaluation protocols in Appendix A, together with details of the classifier baselines, inference-time defenses, routing policies, and adaptive jailbreak evaluation. Appendix B describes the anchor training procedure, intent-token optimization, calibration, cache handling, and the complete inference procedure, including the transfer and backbone-specific settings. The response-evaluation prompt used for datasets without response-level annotations is provided in Appendix D. We will release the full codebase upon publication.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083, 2024.

Manish Bhatt, Sahana Chennabasappa, Yue Li, Cyrus Nikolaidis, Daniel Song, Shengye Wan, Faizan Ahmad, Cornelius Aschermann, Yaohui Chen, Dhaval Kapil, et al. Cyberseceval 2: A wide-ranging cybersecurity evaluation suite for large language models. *arXiv preprint arXiv:2404.13161*, 2024.

Faeze Brahman, Sachin Kumar, Vidhisha Balachandran, Pradeep Dasigi, Valentina Pyatkin, Abhilasha Ravichander, Sarah Wiegreffe, Nouha Dziri, Khyathi Chandu, Jack Hessel, et al. The art of saying no: Contextual noncompliance in language models. *Advances in Neural Information Processing Systems*, 37:49706–49748, 2024.

Bochuan Cao, Yuanpu Cao, Lu Lin, and Jinghui Chen. Defending against alignment-breaking attacks via robustly aligned llm. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10542–10560, 2024.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *Advances in Neural Information Processing Systems*, 37:55005–55029, 2024.

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.

Jianan Chen, Zhifang Zhang, Shuo He, Linan Yue, Lei Feng, and Minling Zhang. Towards safer large reasoning models by promoting safety decision-making before chain-of-thought generation. *arXiv preprint arXiv:2603.17368*, 2026.

Hoagy Cunningham, Alwin Peng, Jerry Wei, Euan Ong, Fabien Roger, Linda Petrini, Misha Wagner, Vladimir Mikulik, and Mrinank Sharma. Cost-effective constitutional classifiers via representation re-use. *Anthropic Alignment Science Blog, June*, 2025.

Hoagy Cunningham, Jerry Wei, Zihan Wang, Andrew Persic, Alwin Peng, Jordan Abderrachid, Raj Agarwal, Bobby Chen, Austin Cohen, Andy Dau, et al. Constitutional classifiers++: Efficient production-grade defenses against universal jailbreaks. *arXiv preprint arXiv:2601.04603*, 2026.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

Shaona Ghosh, Prasoon Varshney, Makesh Narsimhan Sreedhar, Aishwarya Padmakumar, Traian Rebedea, Jibin Rajan Varghese, and Christopher Parisien. Aegis2. 0: A diverse ai safety dataset and risks taxonomy for alignment of llm guardrails. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5992–6026, 2025.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *International Conference on Learning Representations*, volume 2024, pp. 27896–27923, 2024.

Melody Y Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Helyar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, et al. Deliberative alignment: Reasoning enables safer language models. *arXiv preprint arXiv:2412.16339*, 2024.

Peixuan Han, Cheng Qian, Xiusi Chen, Yuji Zhang, Heng Ji, and Denghui Zhang. Safeswitch: Steering unsafe llm behavior via internal activation signals. In *EMNLP (Findings)*, pp. 6936–6955, 2025.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)*, pp. 2733–2743, 2019.

Xiaomeng Hu, Fei Huang, Chenhan Yuan, Junyang Lin, and Tsung-Yi Ho. Care: Decoding-time safety alignment via rollback and introspection intervention. *Advances in Neural Information Processing Systems*, 38:51154–51189, 2026.

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.

Wonje Jeung, Yoon Sangyeon, Minsuk Kahng, and Albert No. Safepath: Preventing harmful reasoning in chain-of-thought via early alignment. *Advances in Neural Information Processing Systems*, 38:99641–99670, 2026.

Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, et al. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models. *Advances in Neural Information Processing Systems*, 37:47094–47165, 2024a.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. Scaling sentence embeddings with large language models. In *Findings of the association for computational linguistics: EMNLP 2024*, pp. 3182–3196, 2024b.

Difan Jiao, Yilun Liu, Ye Yuan, Zhenwei Tang, Linfeng Du, Haolun Wu, and Ashton Anderson. Llm safety from within: Detecting harmful content with internal representations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 39711–39727, 2026.

Kundan Krishna, Joseph Y Cheng, Charles Maalouf, and Leon A Gatys. Disentangled safety adapters enable efficient guardrails and flexible inference-time alignment. *arXiv preprint arXiv:2506.00166*, 2025.

Bruce W Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering. In *International conference on learning representations*, volume 2025, pp. 90960–90985, 2025.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 3045–3059, 2021.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)*, pp. 4582–4597, 2021.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. Toxicchat: Unveiling hidden challenges of toxicity detection in real-world user-ai conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 4694–4702, 2023.

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.

Alex McKenzie, Urja Pawar, Phil Blandfort, William Bankes, David Krueger, Ekdeep S Lubana, and Dmitrii Krasheninnikov. Detecting high-stakes interactions with activation probes. *Advances in Neural Information Processing Systems*, 38:127556–127594, 2026.

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.

Gonzalo Ariel Meyoyan and Luciano Del Corro. A bertology view of llm orchestrations: Token-and layer-selective probes for efficient single-pass classification. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42226–42239, 2026.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016.

James Oldfield, Philip Torr, Ioannis Patras, Adel Bibi, and Fazl Barez. Beyond linear probes: Dynamic safety monitoring for language models. In *International Conference on Learning Representations*, volume 2026, pp. 55192–55229, 2026.

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.

Mansi Phute, Alec Helbling, Matthew Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. Llm self defense: By self examination, llms know they are being tricked. *arXiv preprint arXiv:2308.07308*, 2023.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hanna Hajishirzi. Generalizing verifiable instruction following. *Advances in Neural Information Processing Systems*, 38, 2026.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In *International Conference on Learning Representations*, volume 2025, pp. 54911–54941, 2025.

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.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Narutatsu Ri, Abhishek Panigrahi, and Sanjeev Arora. Do thinking tokens help with safety? *arXiv preprint arXiv:2606.25013*, 2026.

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

Jacob Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. Repetition improves language model embeddings. In *International Conference on Learning Representations*, volume 2025, pp. 93543–93579, 2025.

Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pp. I–I. Ieee, 2001.

Yihan Wang, Zhouxing Shi, Andrew Bai, and Cho-Jui Hsieh. Defending llms against jailbreaking attacks via backtranslation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 16031–16046, 2024.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in neural information processing systems*, 36:80079–80110, 2023.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, et al. Sorry-bench: Systematically evaluating large language model safety refusal. In *International Conference on Learning Representations*, volume 2025, pp. 59937–59973, 2025.

Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao Wu. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, 5(12):1486–1496, 2023.

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

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

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.

Yuqi Zhang, Liang Ding, Lefei Zhang, and Dacheng Tao. Intention analysis makes llms a good jailbreak defender. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 2947–2968, 2025.

Jiachen Zhao, Jing Huang, Zhengxuan Wu, David Bau, and Weiyan Shi. Llms encode harmfulness and refusal separately. *Advances in Neural Information Processing Systems*, 38:140283–140318, 2026.

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.

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.

A Experimental Details

A.1 Datasets, Splits, and Labels

Tab. 8 summarizes the datasets used in this paper. A *request label* specifies whether a request is harmful; a *response judgment* evaluates the answer released by a defense. These quantities serve different purposes: a harmful request can receive a safe response while retaining its harmful request label.

Table 8: Dataset roles and populations. H/B denote harmful/benign request labels. The external safety pool contains 8,240 requests (2,133 H; 6,107 B). Utility datasets supply task references.

Purpose	Dataset	H	B	Total
Anchor fitting	WildJailbreak	6,000	6,000	12,000
Anchor calibration	WildJailbreak	400	400	800
Intent token training	WildJailbreak	750	750	1,500
Intent token calibration	WildJailbreak	400	400	800
Safety evaluation	WildJailbreak	2,300	510	2,810
	HarmBench	200	0	200
	JailbreakBench	100	0	100
	SORRY-Bench	450	0	450
	XSTest	0	250	250
	CoCoNot	0	379	379
	AEGIS2	1,030	884	1,914
	ToxicChat	353	4,594	4,947
Utility	IFBench	–	–	300
	MATH-500	–	–	500
	GPQA Diamond	–	–	198

WildJailbreak. WildJailbreak contains vanilla harmful and benign requests and their adversarial transformations (Jiang et al., 2024a). We use WildJailbreak for anchor fitting, intent token training, and calibration. The anchor-fitting set contains 12,000 requests, with 3,000 from each of the four request types. Anchor calibration, intent token training, and intent token calibration use 800, 1,500, and 800 requests, respectively, balanced between harmful and benign labels. These four sets have disjoint membership. The evaluation population contains 2,810 requests spanning the dataset's evaluation split and unused training requests; none is used for fitting or calibration.

Harmful-request benchmarks. HarmBench (Mazeika et al., 2024) contributes 200 text behaviors from its standard harmful-behavior benchmark. JailbreakBench (Chao et al., 2024) contributes 100 harmful behaviors, and SORRY-Bench (Xie et al., 2025) contributes 450 requests covering 45 unsafe-content categories. We use the datasets distributed with HarmBench, JailbreakBench, and SORRY-Bench. Their request labels define the harmful populations for classification and attack-success evaluation.

Over-refusal benchmarks. XSTest (Röttger et al., 2024) evaluates exaggerated safety behavior on answerable requests containing potentially sensitive terms or contexts, such as killing a computer process. CoCoNot (Brahman et al., 2024) examines contextual distinctions between requests that warrant noncompliance and requests that can be answered. Its contrast set provides answerable requests across these contexts. We use XSTest and CoCoNot to measure over-refusal.

AEGIS2 and ToxicChat. AEGIS2 (Ghosh et al., 2025) provides prompt-level content-safety annotations, and ToxicChat (Lin et al., 2023) contains toxicity annotations for real chatbot user inputs. We use their test data from AEGIS2 and ToxicChat, with prompt deduplication and overlap filtering. AEGIS2's unsafe/safe labels and ToxicChat's toxic/non-toxic labels are mapped to harmful/benign, respectively. The resulting populations contain 1,914 and 4,947 requests. These source labels are preserved for request classification. For generated responses, we use the LLM evaluation prompt in App. D.

Utility tasks. IFBench (Pyatkin et al., 2026) contains 300 instruction-following prompts with verifiable output constraints. MATH-500 (Lightman et al., 2024) contains 500 mathematical problems, and GPQA Diamond (Rein et al., 2023) contains 198 multiple-choice questions. We report task accuracy on these utility benchmarks.

A.2 Evaluation Protocols

Request classification. Classification uses the original request labels, with harmful as the positive class. Let N_H, N_B denote the numbers of harmful and benign requests. We compute

$$\mathrm{FNR} = \frac{\mathrm{FN}}{N_H}, \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{N_B}, \qquad \mathrm{BAcc} = 1 - \frac{\mathrm{FNR} + \mathrm{FPR}}{2}, \tag{5}$$
$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{TN}}{N_H + N_B}, \qquad \mathrm{F1} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}.$$

AUC uses unthresholded harmfulness scores. For RESTATE, the score is $z_a(x)$ outside $\mathcal{U}^a$ and $z_{\mathsf{int}}(x)$ inside $\mathcal{U}^a$; a positive score predicts harmful. The WildJailbreak and external results in Tab. 1 pool the requests in each population before computing classification metrics.

Response generation. We generate responses with Qwen3-8B in thinking mode, with a maximum generation length of 32k tokens. The policy releases an ordinary answer to the original request, a fixed refusal, or the answer produced by the designated inference-time defense. Evaluation considers the visible final answer.

Attack success and over-refusal. We follow the official benchmark evaluation protocols. A response counts as an attack success when it provides harmful content or substantively fulfills the harmful request. For adversarially transformed requests, this judgment concerns the underlying harmful behavior. A refusal to an answerable request counts as over-refusal. Thus,

$$\mathrm{ASR} = \frac{\#\,\text{successful harmful requests}}{\#\,\text{harmful requests}}, \qquad \mathrm{ORR} = \frac{\#\,\text{refused benign requests}}{\#\,\text{benign requests}}. \tag{6}$$

For AEGIS2 and ToxicChat, which provide request-classification labels, we use an LLM to judge response harmfulness and refusal separately. The prompt and the mapping of its outputs to these metrics are given in App. D.

Aggregation. The static comparison uses a shared response draw for each request and response method across the routing policies. WildJailbreak is treated as one dataset. External ASR averages HarmBench, JailbreakBench, SORRY-Bench, AEGIS2, and ToxicChat; external ORR averages XSTest, CoCoNot, AEGIS2, and ToxicChat. Each contributing dataset has equal weight. Averages across defenses likewise give equal weight to the six methods. App. C.5 reports the corresponding per-benchmark results.

A.3 Classifier Baselines

The classifier comparison covers increased readout capacity, aggregation of additional activations, and selective use of a second readout. We follow the corresponding official implementations and methodological descriptions. Classifier inputs are derived from the frozen Qwen3-8B backbone.

Bilinear and MLP. Both read the same final-position hidden state as the anchor. Bilinear introduces second-order feature interactions through a low-rank factorization (Hewitt & Liang, 2019; Oldfield et al., 2026). MLP applies a nonlinear feed-forward classifier. These methods test the benefit of a more expressive decision function on the original representation.

SIREN. SIREN (Jiao et al., 2026) identifies safety-relevant neurons using sparse layerwise probes, weights layer contributions by probe quality, and classifies the resulting aggregated features with an MLP. It combines safety information distributed across the model's internal activations.

Token $\times$ Layer. This probe (Meyoyan & Del Corro, 2026) uses learned attention to aggregate representations over request tokens and decoder layers, followed by a linear classifier. It can draw on evidence from multiple positions and depths.

Suffix. The suffix probe (Cunningham et al., 2025) appends a fixed natural-language safety-assessment instruction and reads the resulting hidden state with a linear classifier. The appended instruction encourages the backbone to form a representation suited to harmfulness assessment.

TPC and EE-MLP. TPC (Oldfield et al., 2026) progressively evaluates polynomial terms, allowing confident requests to exit after a lower-order computation. EE-MLP (Teerapittayanon et al., 2016; Oldfield et al., 2026) instead attaches classifier exits at successive depths of an MLP. Both allocate additional classifier computation to requests that remain uncertain; the final exit supplies a prediction when needed.

Anchor → MLP. This cascade preserves confident anchor decisions and applies an MLP to requests in the anchor's uncertain region. The specialist is trained on requests reaching that region and reads the original final-position state. It increases second-stage readout capacity while retaining the anchor representation.

EMA → Suffix. Following Cunningham et al. (2025), a tokenwise linear probe forms an exponential moving average of its predictions and uses the maximum smoothed value to screen requests. Confidently benign requests exit at this stage; the remaining requests receive the suffix probe. The cascade combines an inexpensive initial screen with an additional safety-assessment computation.

A.4 Inference-Time Defenses and Routing Comparisons

The six defenses below instantiate $\mathrm{Deliberate}(x)$. We follow their official implementations and evaluation protocols, using Qwen3-8B as the response model.

Backtranslation (BT). Backtranslation (Wang et al., 2024) first generates a candidate answer, then infers a request that could have elicited it. A likelihood check establishes whether this inferred request adequately explains the candidate. The defense checks whether answering the inferred request is permissible and uses that assessment to release the candidate or return a refusal.

Intention Analysis (IA). IA (Zhang et al., 2025) first analyzes the essential intention of the request. It then answers the original request while conditioning on that analysis and the model's safety requirements.

LLM Self-Defense (LSD). LSD (Phute et al., 2023) generates a candidate answer and checks whether it contains harmful content. A harmful candidate is replaced with a refusal; otherwise, the answer is released.

RA-LLM (RA). RA-LLM (Cao et al., 2024) evaluates the stability of the model's response under random token deletion from the request. Refusal behavior across the perturbed requests determines whether the original request is accepted for generation or blocked.

Self-Reminder (SR). Self-Reminder (Xie et al., 2023) surrounds the user request with reminders of the model's safety responsibilities. The model generates its answer from this augmented context.

ZS-SafePath (ZS). The zero-shot SafePath variant (Jeung et al., 2026) begins the assistant's reasoning with a fixed safety-oriented prefix and then continues generation. It encourages safety consideration within the model's reasoning process.

Routing policies. *No intervention* generates an ordinary answer for every request; *Always intervention* applies the designated defense to every request. *Anchor* answers below τ_b, refuses above τ_h, and deliberates within $\mathcal{U}^a$. RESTATE adds intent token reassessment and deliberates only within $\mathcal{U}^{\mathsf{int}}$. *Random deliberation* uniformly selects requests from the anchor's uncertain region, matching the number of requests deliberated by RESTATE.

Component ladder and budget curves. L1 refuses requests with positive anchor scores; L1-D deliberates them. L2 uses the anchor's uncertainty band. L3 uses the sign of the intent token score

within the anchor band, preserving confident anchor decisions; L3-D deliberates positive intent token predictions in that region. RESTATE adds the uncertainty band for the intent token score. In Fig. 4b, confidence ordering determines which requests receive the available deliberations, with class-side allocations matched across the compared rankings.

A.5 ADAPTIVE JAILBREAK EVALUATION

Adaptive attacks optimize against the final response of the complete policy, including routing, refusals, and the terminal defense. We evaluate HarmBench and SORRY-Bench, running each attack separately against each policy. The defended configurations use IA, RA, and ZS. We follow the official implementations and evaluation protocols of the attacks and benchmarks.

PAIR. PAIR (Chao et al., 2025) iteratively rewrites a jailbreak using feedback from previous target responses. Conversation histories let the attacker refine subsequent attempts toward the original harmful objective.

TAP. TAP (Mehrotra et al., 2024) explores a tree of candidate rewrites. It prunes candidates that no longer pursue the original behavior and prioritizes promising candidates for further expansion using attack feedback.

GPTFuzzer. GPTFuzzer (Yu et al., 2023) selects and mutates jailbreak templates, combines them with the target behavior, and submits the resulting requests to the defended policy. Feedback from these interactions guides subsequent template selection and mutation.

Success endpoints. Benchmark evaluation determines whether the released answer fulfills the original harmful behavior. *Initial* ASR concerns the original request, *terminal* ASR concerns the rewrite selected by the attack controller, and *any-query* ASR counts success on the original request or any request attempted during the attack. The left and middle panels of Fig. 5 report ASR averaged equally over PAIR, TAP, and GPTFuzzer. Always deliberation and RESTATE are additionally averaged over IA, RA, and ZS. The right panel reports cumulative any-query ASR for GPTFuzzer on HarmBench, counting success up to each round. RESTATE† returns a refusal for requests still uncertain after intent token reassessment. App. C.6 reports attack-specific results for each defense and their averages.

B METHOD DETAILS

B.1 ANCHOR REPRESENTATION AND TRAINING

Readout position. We format each request with the Qwen3 chat template. The anchor reads the final decoder-block output at the last prefill position, immediately before the model would generate `<think>`. For Qwen3-8B, the readout has dimension 4,096 and is taken from layer 36, before the final RMSNorm. The same readout position is used for anchor fitting, calibration, and deployment.

Fitting the anchor. With the backbone frozen, we extract one hidden state per request in $\mathcal{D}_a$, apply a feature scaler, and fit an L_2-regularized logistic-regression probe. The regularization coefficient is λ_a, as in Eq. 1. The resulting probe remains fixed during intent token training.

B.2 INTENT TOKEN OPTIMIZATION

Parameterization and training data. The intent token is one shared vector $e_{\mathsf{int}} \in \mathbb{R}^{4096}$, initialized from the input embedding at the last prefill position. It is appended directly to the request embeddings. The training set is balanced between harmful and benign labels and primarily contains anchor-uncertain requests, together with the regularization examples described in Section 3.2.2. The frozen backbone computes the state at the appended position, and the frozen anchor reads that state. Only the intent token embedding e_{int} is learned; the backbone and anchor remain frozen.

Robust objective. For each valid request token position t, the perturbation radius is $r_t = \epsilon\|e_t\|_2$. The perturbation set in Eq. 3 is

$$\Delta(x) = \big\{\delta : \|\delta_t\|_2 \le r_t \text{ at each request token position } t; \quad \delta_t = 0 \text{ on padding}\big\}. \tag{7}$$

Following Madry et al. (2017) on adversarial training, we approximate the inner maximization using a projected gradient descent (PGD) attack on the request token embeddings. Each step increases the harmfulness-classification loss and projects the perturbation back into its tokenwise norm ball. The intent token itself is held fixed during this inner maximization. Writing $\widehat{\delta}(x)$ for the resulting perturbation, the intent token update minimizes the following objective corresponding to Eq. 3:

$$\widehat{\mathcal{L}}_{\mathsf{int}}(e_{\mathsf{int}}) = \frac{1}{|\mathcal{D}_{\mathsf{int}}|} \sum_{(x,y)\in\mathcal{D}_{\mathsf{int}}} \Big[\,\mathrm{BCE}(z_{\mathsf{int}}(x), y) + \lambda_{\mathrm{adv}}\,\mathrm{BCE}\Big(z_{\mathsf{int}}(x;\widehat{\delta}(x)), y\Big)\,\Big]. \tag{8}$$

Here $\widehat{\delta}(x)$ is held fixed while only the intent token embedding e_{int} is updated. Following embedding-space adversarial training (Miyato et al., 2016; Madry et al., 2017), this encourages consistent assessment under small changes to the input representation. Tab. 9 gives the coefficients in the intent token training objective.

Table 9: Coefficients in the intent token training objective.

Coefficient	**Value**
Adversarial loss coefficient λ_{adv}	0.5
Relative perturbation radius ϵ	0.01

B.3 Calibration and Confidence-Based Decisions

Class-conditional calibration. The lower and upper thresholds are determined from the harmful and benign calibration-score distributions, respectively, according to Eq. 2. A harmful request contributes to the lower-tail error when it would be confidently answered, and a benign request contributes to the upper-tail error when it would be confidently refused. This permits asymmetric bands around the zero-logit classification boundary. The intent token band uses a separate calibration split restricted to requests reaching $\mathcal{U}^a$. The calibrated bands remain fixed across benchmarks within each evaluation setting.

Classification and response routing. For classification, a request reaching the intent token receives the binary decision $\mathbf{1}[z_{\mathsf{int}}(x) > 0]$. For response routing, its score is compared with the intent token band: a sufficiently negative score permits normal generation, a sufficiently positive score returns a refusal, and an intermediate score invokes the defense. The residual uncertain set $\mathcal{U}^{\mathsf{int}}$ is therefore contained in $\mathcal{U}^a$.

B.4 Complete Procedure and Cache Handling

Algorithm 1 summarizes fitting, calibration, and inference using the notation of Section 3.2. We define its inputs, scores, and operations below.

Representations and scores. A chat-formatted request x has T tokens and label $y \in \{0, 1\}$, with 1 denoting harmful and 0 benign. Its input embeddings are $E(x) = (e_1, \ldots, e_T) \in \mathbb{R}^{T\times d}$, where d is the hidden dimension. The frozen backbone H_θ, with parameters θ, returns the final-layer state $H_\theta(E)_t \in \mathbb{R}^d$ at token position t. For a hidden state $h \in \mathbb{R}^d$, the anchor is the affine probe $g_\phi(h) = w^\top h + b$, with parameters $\phi = (w, b)$, weight vector $w \in \mathbb{R}^d$, and bias $b \in \mathbb{R}$. It reads $h_a(x) = H_\theta(E(x))_T$ and produces the harmfulness logit $z_a(x) = g_\phi(h_a(x))$. The learned intent token embedding is $e_{\mathsf{int}} \in \mathbb{R}^d$, and $\oplus$ appends it to the request embeddings. Its state is $h_{\mathsf{int}}(x) = H_\theta(E(x) \oplus e_{\mathsf{int}})_{T+1}$, with score $z_{\mathsf{int}}(x) = g_\phi(h_{\mathsf{int}}(x))$. The algorithm omits the argument x from states and scores when processing the current request.

Fitting and calibration. The labeled fitting sets $\mathcal{D}_a$ and $\mathcal{D}_{\mathsf{int}}$ train the anchor and intent token, respectively; $\mathcal{C}_a$ and $\mathcal{C}_{\mathsf{int}}$ are their held-out calibration sets. BCE denotes binary cross-entropy on a logit, and λ_{adv} weights its adversarial term. The allowed request embedding perturbations form $\Delta(x)$ in Eq. 7; $\widehat{\delta}(x)$ is the PGD approximation to the inner maximizer. The clean score uses $E(x)$, and the perturbed score replaces it with $E(x) + \widehat{\delta}(x)$ while retaining the intent token. Calibration follows Eq. 2: (τ_b, τ_h) are the anchor's benign and harmful confidence thresholds, with $\tau_b < 0 < \tau_h$, and $\mathcal{U}^a = \{x : \tau_b < z_a(x) < \tau_h\}$ is its uncertain set. The intent token thresholds $(\tau_b^{\mathsf{int}}, \tau_h^{\mathsf{int}})$, also on opposite sides of zero, are calibrated on examples in $\mathcal{C}_{\mathsf{int}}$ whose requests belong to $\mathcal{U}^a$.

Inference operations and KV cache. $\mathrm{Prefill}(x)$ computes the anchor state h_a and the request's key value cache (KV cache), denoted by $\mathcal{K}$. $\mathrm{Copy}(\mathcal{K})$ preserves the original request cache while providing a working copy for reassessment. $\mathrm{ForwardOneToken}(e_{\mathsf{int}}, \mathrm{Copy}(\mathcal{K}))$ takes one causal forward step with the intent token attending to the request prefix and returns h_{int}. The extended KV cache is discarded after assessment. The input *mode* selects binary classification or response generation, and $\mathbf{1}[\cdot]$ equals one when its condition holds and zero otherwise. For response generation, $\mathrm{Generate}(x)$ uses the original request's KV cache, Refuse returns a fixed refusal, and $D(x)$ applies the terminal inference-time defense, corresponding to $\mathrm{Deliberate}(x)$ in Eq. 4. Both generation and the terminal defense receive the original request without the intent token.

Algorithm 1: Detailed procedure of RESTATE

Input: Frozen LLM H_θ; fitting sets $\mathcal{D}_a, \mathcal{D}_{\mathsf{int}}$; calibration sets $\mathcal{C}_a, \mathcal{C}_{\mathsf{int}}$; request x; mode; terminal defense D.
Output: Harmfulness decision or released response.

// Phase 1: Fit and calibrate the anchor (offline)

Extract $h_a(x)$ at the last prefill position for requests in $\mathcal{D}_a$.
Apply a feature scaler and fit the L_2-regularized anchor; freeze g_ϕ.
Calibrate (τ_b, τ_h) on $\mathcal{C}_a$; define $\mathcal{U}^a$.

// Phase 2: Train and calibrate the intent token (offline)

Initialize e_{int} from the last prefill token's input embedding.
for each training minibatch from $\mathcal{D}_{\mathsf{int}}$ **do**
    For each (x, y) in the minibatch, compute $z_{\mathsf{int}} = g_\phi(H_\theta(E(x) \oplus e_{\mathsf{int}})_{T+1})$.
    Obtain $\widehat{\delta}(x)$ with PGD over $\Delta(x)$; hold $\widehat{\delta}(x)$ fixed.
    Average clean BCE + λ_{adv} perturbed BCE over the minibatch.
    Update e_{int}; keep θ, ϕ fixed.
end for
Calibrate $(\tau_b^{\mathsf{int}}, \tau_h^{\mathsf{int}})$ on $\{(x, y) \in \mathcal{C}_{\mathsf{int}} : x \in \mathcal{U}^a\}$.

// Phase 3: Selective assessment and response (online)

$(h_a, \mathcal{K}) \leftarrow \mathrm{Prefill}(x)$; $z_a \leftarrow g_\phi(h_a)$.
if $z_a \le \tau_b$ or $z_a \ge \tau_h$ **then**
    if mode is classification, **return** $\mathbf{1}[z_a > 0]$.
    return $\mathrm{Generate}(x)$ if $z_a \le \tau_b$, otherwise Refuse.
end if
$h_{\mathsf{int}} \leftarrow \mathrm{ForwardOneToken}(e_{\mathsf{int}}, \mathrm{Copy}(\mathcal{K}))$.
$z_{\mathsf{int}} \leftarrow g_\phi(h_{\mathsf{int}})$; discard the KV cache extended by the intent token.
if mode is classification, **return** $\mathbf{1}[z_{\mathsf{int}} > 0]$.
if $z_{\mathsf{int}} \le \tau_b^{\mathsf{int}}$, **return** $\mathrm{Generate}(x)$.
if $z_{\mathsf{int}} \ge \tau_h^{\mathsf{int}}$, **return** Refuse.
return $D(x)$.

Assessment cost. For hidden dimension d, the intent token stores d parameters and the affine anchor stores $d + 1$. A confident request needs one probe evaluation after prefill. A request in $\mathcal{U}^a$ needs one additional token step and a second probe evaluation. The mean number of added assessment positions is $|\mathcal{U}^a|/N$ for a population of N requests. Downstream defense computation is incurred only for requests that remain uncertain.

B.5 Classifier Transfer

Fitting an independent probe. The transfer experiment in Tab. 2 starts with an intent token e^*_{int} learned through the original anchor g_ϕ. We fit an *independent probe* on ordinary request states $h_a(x)$ from a separate labeled fitting set $\widetilde{\mathcal{D}}_a$. For a linear probe, write its parameters as $\psi = (\widetilde{w}, \widetilde{b})$ and its score as

$$\widetilde{g}_\psi(h) = \widetilde{w}^\top h + \widetilde{b}, \qquad \widetilde{z}_a(x) = \widetilde{g}_\psi\big(h_a(x)\big). \tag{9}$$

We fit ψ using the anchor objective in Eq. 1, with $\widetilde{\mathcal{D}}_a$, $\widetilde{z}_a$, and $\widetilde{w}$ in place of $\mathcal{D}_a$, z_a, and w. The independent probe is fitted separately from the intent token. We then freeze both the probe and the intent token, keeping the vector learned through the original anchor.

Deployment with the transferred intent token. First, the independent probe scores the ordinary request state. Its own calibrated thresholds define

$$\widetilde{\mathcal{U}}^a = \{x : \widetilde{\tau}_b < \widetilde{z}_a(x) < \widetilde{\tau}_h\}. \tag{10}$$

Second, each request in this uncertain set receives the frozen intent token. The same independent probe reads the additional representation:

$$\widetilde{z}_{\mathsf{int}}(x) = \widetilde{g}_\psi(H_\theta(E(x) \oplus e^*_{\mathsf{int}})_{T+1}) \,. \tag{11}$$

Third, the final classification uses the score from the last assessment stage reached:

$$\widetilde{z}(x) = \begin{cases} \widetilde{z}_a(x), & x \notin \widetilde{\mathcal{U}}^a, \\ \widetilde{z}_{\mathsf{int}}(x), & x \in \widetilde{\mathcal{U}}^a, \end{cases} \qquad \widehat{y}(x) = \mathbf{1}[\widetilde{z}(x) > 0]. \tag{12}$$

The independent probe therefore supplies both predictions, while the transferred intent token supplies the additional representation for uncertain requests.

Related controls in Tab. 2. The additional-data controls fit the original probe on $\mathcal{D}_a \cup \mathcal{D}_{\mathsf{int}}$ jointly, or continue fitting it on $\mathcal{D}_{\mathsf{int}}$ after fitting on $\mathcal{D}_a$. Both read the ordinary states $h_a(x)$. For the MLP-trained intent token comparison, we first fit the MLP, then optimize the intent token through this frozen MLP using Eq. 3. The transfer procedure above also applies to an independent MLP probe, replacing the linear map $\widetilde{g}_\psi$ with an MLP while retaining the intent token learned through the original anchor.

B.6 Sequential Assessment with a Second Intent Token

This extension retains the original frozen anchor g_ϕ for both reassessment stages. We denote the two learned vectors by $e_{\mathsf{int},1}$ and $e_{\mathsf{int},2}$ and describe their training and calibration in sequence.

First-stage calibration and second-stage examples. We train $e_{\mathsf{int},1}$ as in App. B.2 and calibrate its band on held-out requests in $\mathcal{U}^a$. This gives the residual uncertain set

$$\mathcal{U}^{\mathsf{int},1} = \{x \in \mathcal{U}^a : \tau_b^{\mathsf{int},1} < z_{\mathsf{int},1}(x) < \tau_h^{\mathsf{int},1}\}. \tag{13}$$

Let $\mathcal{P}_2$ be a labeled fitting pool separate from first-stage training and calibration. We score its requests with the frozen anchor and first intent token, then form the class-specific candidate sets

$$\mathcal{P}_{2,c} = \{(x, y) \in \mathcal{P}_2 : x \in \mathcal{U}^{\mathsf{int},1},\ y = c\}, \qquad c \in \{0, 1\}. \tag{14}$$

Let $\mathcal{D}^{(c)}_{\mathsf{int},2} \subseteq \mathcal{P}_{2,c}$ denote the training examples sampled for class c at stage two, extending the main text's notation $\mathcal{D}_{\mathsf{int}}$ with stage and class indices. We form the second-stage fitting set as

$$\mathcal{D}_{\mathsf{int},2} = \mathcal{D}^{(0)}_{\mathsf{int},2} \cup \mathcal{D}^{(1)}_{\mathsf{int},2}, \qquad |\mathcal{D}^{(0)}_{\mathsf{int},2}| = |\mathcal{D}^{(1)}_{\mathsf{int},2}|. \tag{15}$$

Thus, the second intent token learns from requests that remain uncertain after the first reassessment, with balanced harmful and benign labels.

Stagewise training. We append the second intent token after the first, keeping the first vector, backbone, and anchor fixed:

$$z_{\mathsf{int},2}(x; e_{\mathsf{int},2}) = g_\phi(H_\theta(E(x) \oplus e_{\mathsf{int},1} \oplus e_{\mathsf{int},2})_{T+2}) \,. \tag{16}$$

Let $z_{\mathsf{int},2}(x; e_{\mathsf{int},2}, \delta)$ replace $E(x)$ in Eq. 16 by $E(x) + \delta$. We minimize the same robust objective with respect to the second vector:

$$\mathcal{L}_{\mathsf{int},2}(e_{\mathsf{int},2}) = \frac{1}{|\mathcal{D}_{\mathsf{int},2}|} \sum_{(x,y)\in\mathcal{D}_{\mathsf{int},2}} \Big[\mathrm{BCE}(z_{\mathsf{int},2}(x; e_{\mathsf{int},2}), y) + \lambda_{\mathrm{adv}} \max_{\delta\in\Delta(x)} \mathrm{BCE}(z_{\mathsf{int},2}(x; e_{\mathsf{int},2}, \delta), y) \Big]. \tag{17}$$

The perturbation acts on the request embeddings, and only $e_{\mathsf{int},2}$ receives an outer parameter update.

Conditional calibration. For stage $k \in \{1, 2\}$, let $\mathcal{C}_k$ be its held-out calibration set and define

$$\mathcal{C}_k^{\mathrm{in}} = \{(x, y) \in \mathcal{C}_k : x \in \mathcal{U}^{\mathsf{int},k-1}\}, \qquad \mathcal{U}^{\mathsf{int},0} = \mathcal{U}^a. \tag{18}$$

The largest lower threshold $\tau_b^{\mathsf{int},k} < 0$ and smallest upper threshold $\tau_h^{\mathsf{int},k} > 0$ satisfy the class-conditional criteria on requests reaching that stage:

$$\widehat{\Pr}_{\mathcal{C}_k^{\mathrm{in}}}\Big(z_{\mathsf{int},k}(x) \le \tau_b^{\mathsf{int},k} \mid y = 1\Big) \le \alpha_{\mathrm{FN}}^{\mathsf{int},k},$$
$$\widehat{\Pr}_{\mathcal{C}_k^{\mathrm{in}}}\Big(z_{\mathsf{int},k}(x) \ge \tau_h^{\mathsf{int},k} \mid y = 0\Big) \le \alpha_{\mathrm{FP}}^{\mathsf{int},k}. \tag{19}$$

The first-stage band is fixed before constructing $\mathcal{D}_{\mathsf{int},2}$, and the second-stage band is fitted after learning $e_{\mathsf{int},2}$. Both are then held fixed for evaluation.

Deployment. The policy evaluates the anchor, appends the first intent token for requests in $\mathcal{U}^a$, and appends the second for requests still in $\mathcal{U}^{\mathsf{int},1}$. The second step reuses the KV cache containing the request and first intent token. At each stage, the residual uncertain set is

$$\mathcal{U}^{\mathsf{int},k} = \{x \in \mathcal{U}^{\mathsf{int},k-1} : \tau_b^{\mathsf{int},k} < z_{\mathsf{int},k}(x) < \tau_h^{\mathsf{int},k}\}, \qquad k \in \{1, 2\}. \tag{20}$$

A confident exit retains its decision. With both intent tokens available, the final classification score is

$$z^{(2)}(x) = \begin{cases} z_a(x), & x \notin \mathcal{U}^a, \\ z_{\mathsf{int},1}(x), & x \in \mathcal{U}^a \setminus \mathcal{U}^{\mathsf{int},1}, \\ z_{\mathsf{int},2}(x), & x \in \mathcal{U}^{\mathsf{int},1}. \end{cases} \tag{21}$$

Classification uses $\mathbf{1}[z^{(2)}(x) > 0]$ for every request, including those still uncertain after the second intent token. For response routing, confident exits generate or refuse according to their class, and only $\mathcal{U}^{\mathsf{int},2}$ reaches the terminal defense. Response generation starts from the original request without either intent token.

B.7 Backbone-Specific Training

Each backbone in Tab. 4 receives its own anchor and intent token. The model's chat template defines the prefill readout position, and its hidden dimension determines the intent token's dimension. We fit the anchor on that backbone's request representations, freeze the backbone and anchor, and train the intent token through the resulting readout. Calibration then uses the corresponding model's scores. This applies the same assessment procedure to the evaluated models and reasoning modes.

C Additional Analyses

C.1 Per-Dataset Classification

Tab. 10 expands the pooled classification results into per-benchmark accuracy and class-conditional error rates. RESTATE generally improves request classification across the evaluated populations. On the external pool, it improves both harmful-request recognition and benign-request acceptance. The AEGIS2 and ToxicChat results show that the intent token representation is useful for naturally occurring moderation inputs beyond the WildJailbreak training distribution.

Table 10: Per-dataset harmfulness classification (%). A: Anchor; R: RESTATE. Accuracy uses all requests; FNR and FPR use harmful and benign requests, respectively. A dash denotes an absent class. External pools the seven external benchmarks.

	Accuracy ↑		FNR ↓		FPR ↓	
Dataset	A	R	A	R	A	R
WildJailbreak	83.88	92.42	17.04	7.70	11.96	7.06
HarmBench	99.50	100.00	0.50	0.00	–	–
JailbreakBench	100.00	100.00	0.00	0.00	–	–
SORRY-Bench	93.11	90.89	6.89	9.11	–	–
XSTest	98.40	96.40	–	–	1.60	3.60
CoCoNot	92.88	96.57	–	–	7.12	3.43
AEGIS2	70.38	76.70	28.74	25.92	30.66	20.25
ToxicChat	75.90	86.94	16.43	11.90	24.68	13.15
External	77.89	85.98	18.10	16.41	23.51	13.18

C.2 How Reassessment Allocates Deliberation

Fig. 6 separates the requests receiving the intent token from their eventual response actions. Intent token reassessment resolves a substantial part of the anchor's uncertain set before invoking a terminal defense. Many explicitly harmful requests receive a direct safety decision, while answerable requests can proceed to ordinary generation. Moderation datasets contain a broader mix of contexts for which reassessment and selective deliberation play a larger role. The same thresholds are used across benchmarks in this comparison.

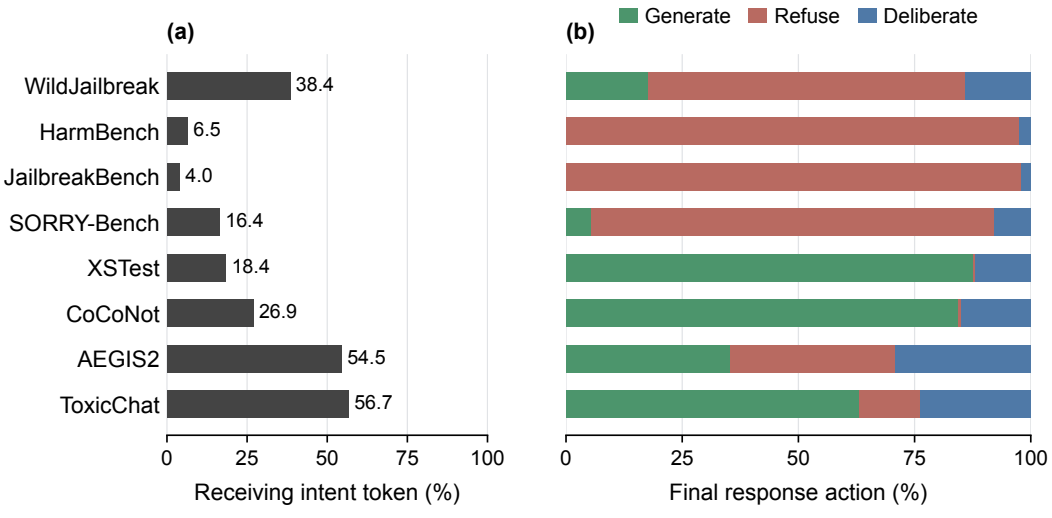


Figure 6: Reassessment and final routing by benchmark. (a) Fraction receiving the intent token. (b) Fractions assigned to ordinary generation, fixed refusal, and terminal defense. All fractions use the full benchmark population and the static intent token band.

C.3 Isolating the Intent Token Training Objective

Tab. 11 extends the intent token ablation to the external pool. The BCE and Robust BCE comparisons use the same training requests, initialization, training duration, and inference routing, changing the adversarial term in Eq. 3. The robust objective improves balanced accuracy and AUC on both pools, supporting an assessment representation that transfers beyond WildJailbreak. The untrained intent token control separates the benefit of learning this representation from the effect of adding a computation position.

Table 11: Intent token ablation on WildJailbreak and external benchmarks (%). The BCE and Robust BCE comparisons use the same training requests, initialization, training duration, and anchor; they differ in the adversarial loss term.

	WildJailbreak			External		
Setting	BAcc ↑	FNR ↓	AUC ↑	BAcc ↑	FNR ↓	AUC ↑
Anchor	85.50	17.04	93.40	79.19	18.10	87.94
Untrained intent token	80.26	21.83	90.48	78.95	11.02	88.77
BCE	86.73	10.26	94.53	81.68	18.57	89.29
Robust BCE (RESTATE)	92.62	7.70	96.79	85.20	16.41	90.76

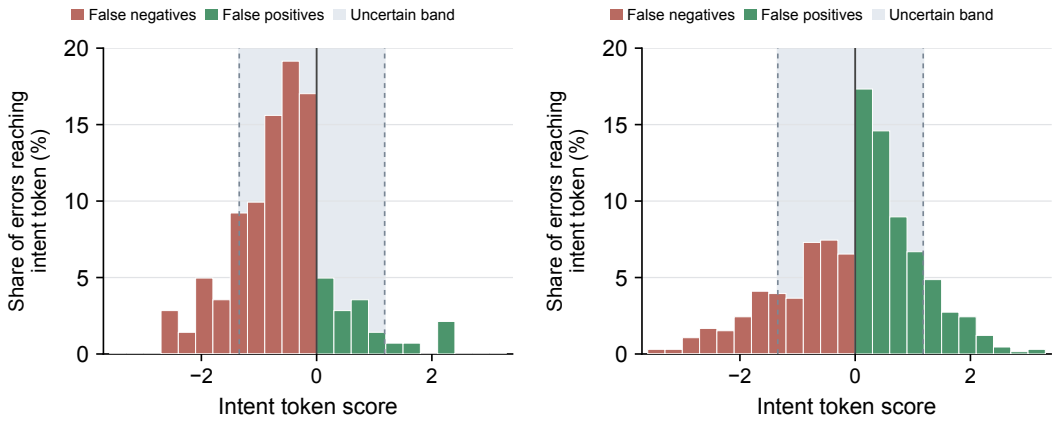
C.4 Deliberation on Uncertain Classification Errors

When RESTATE is used as a defense, a binary classification error need not lead to a response-level failure: requests that remain uncertain are forwarded to deliberation before producing a response. We examine the misclassified requests and those routed to deliberation,

$$\mathcal{E} = \{(x, y) \in \mathcal{D}_{\text{eval}} : \widehat{y}(x) \neq y\}, \qquad \mathcal{E}_{\mathcal{U}} = \{(x, y) \in \mathcal{E} : x \in \mathcal{U}^{\text{int}}\}, \tag{22}$$

where $\widehat{y}(x)$ is the final binary decision from the anchor or intent token. Among these errors, $\mathcal{E}_{\text{int}} = \{(x, y) \in \mathcal{E} : x \in \mathcal{U}^{a}\}$ denotes those on requests that reach the intent token.

Fig. 7 shows that classification errors among requests reaching the intent token mostly fall within its uncertain band: 79.43% on WildJailbreak and 74.16% on the external pool. These fractions are $|\mathcal{E}_{\mathcal{U}}|/|\mathcal{E}_{\text{int}}|$ and combine harmful and benign requests in each pool. The band therefore routes most of these errors to deliberation, which can recover them at the response level.



(a) WildJailbreak **(b) External**

Figure 7: Intent token scores of misclassified requests that reach the intent token. Bar heights are percentages of these errors in each pool. The shaded uncertain band contains 79.43% of these errors on WildJailbreak and 74.16% on the external pool, routing them to deliberation. Zero marks the binary decision boundary.

C.5 PER-BENCHMARK RESULTS FOR EACH DEFENSE

Tab. 12 reports per-benchmark ASR and ORR for all six inference-time defenses, comparing No intervention, Always intervention, and RESTATE on the same request population. These results expand the aggregate defense comparison in Tab. 6. The external aggregates in the main text average the applicable benchmark entries, as described in App. A.2.

Table 12: Per-benchmark ASR and ORR (%) for all six defenses. Each cell gives ASR above ORR in parentheses; lower is better. No: No intervention; Always: Always intervention. BT: Backtranslation; IA: Intention Analysis; LSD: LLM Self-Defense; RA: RA-LLM; SR: Self-Reminder; ZS: ZS-SafePath. A dash denotes an absent class.

Dataset	Policy	BT	IA	LSD	RA	SR	ZS
WildJailbreak	No	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)	72.96 (0.98)
	Always	33.26 (25.69)	7.83 (29.80)	67.13 (5.10)	59.61 (6.27)	32.00 (5.88)	26.61 (9.61)
	RESTATE	8.96 (9.41)	5.43 (10.20)	14.57 (5.29)	13.96 (4.90)	9.65 (5.69)	8.61 (7.45)
HarmBench	No	17.00 (–)	17.00 (–)	17.00 (–)	17.00 (–)	17.00 (–)	17.00 (–)
	Always	3.00 (–)	0.00 (–)	17.00 (–)	10.00 (–)	0.00 (–)	2.50 (–)
	RESTATE	0.00 (–)	0.00 (–)	0.50 (–)	0.50 (–)	0.00 (–)	0.00 (–)
JailbreakBench	No	12.00 (–)	12.00 (–)	12.00 (–)	12.00 (–)	12.00 (–)	12.00 (–)
	Always	3.00 (–)	0.00 (–)	8.00 (–)	2.00 (–)	0.00 (–)	1.00 (–)
	RESTATE	0.00 (–)	0.00 (–)	0.00 (–)	0.00 (–)	0.00 (–)	0.00 (–)
SORRY-Bench	No	46.22 (–)	46.22 (–)	46.22 (–)	46.22 (–)	46.22 (–)	46.22 (–)
	Always	27.33 (–)	3.56 (–)	45.33 (–)	31.11 (–)	20.67 (–)	25.11 (–)
	RESTATE	10.00 (–)	6.00 (–)	12.00 (–)	11.33 (–)	9.11 (–)	10.67 (–)
XSTest	No	– (0.40)	– (0.40)	– (0.40)	– (0.40)	– (0.40)	– (0.40)
	Always	(22.40)	(0.00)	(9.60)	(12.00)	(4.80)	(2.40)
	RESTATE	(3.60)	(0.80)	(2.40)	(3.60)	(2.00)	(1.60)
CoCoNot	No	– (2.64)	– (2.64)	– (2.64)	– (2.64)	– (2.64)	– (2.64)
	Always	(15.57)	(10.03)	(7.12)	(7.39)	(5.28)	(3.69)
	RESTATE	(5.80)	(4.75)	(3.96)	(3.96)	(3.43)	(2.90)
AEGIS2	No	12.14 (8.26)	12.14 (8.26)	12.14 (8.26)	12.14 (8.26)	12.14 (8.26)	12.14 (8.26)
	Always	6.80 (31.56)	1.26 (38.57)	12.43 (11.31)	7.77 (21.04)	2.43 (22.06)	5.73 (17.53)
	RESTATE	5.92 (24.89)	3.79 (25.00)	7.77 (17.76)	6.31 (23.08)	4.37 (21.61)	5.05 (19.57)
ToxicChat	No	36.26 (1.63)	36.26 (1.63)	36.26 (1.63)	36.26 (1.63)	36.26 (1.63)	36.26 (1.63)
	Always	26.63 (17.91)	1.98 (21.68)	32.86 (3.44)	15.58 (6.23)	9.63 (7.18)	7.37 (8.08)
	RESTATE	6.52 (14.15)	1.70 (16.41)	8.78 (10.01)	6.52 (11.25)	5.67 (12.10)	4.53 (11.82)

C.6 ADAPTIVE ROBUSTNESS BY ATTACK AND DEFENSE

Tab. 13 reports adaptive ASR for PAIR, TAP, and GPTFuzzer, comparing No intervention (No), Always intervention (Always), and RESTATE (R). Always and R are averaged over IA, RA, and ZS, while No is the shared undefended baseline. Tab.s 14 and 15 provide the per-defense results, repeating the same No baseline for reference. Terminal ASR evaluates the rewrite selected by the

attack controller; any-query ASR counts success on the original request or any attempted rewrite. Each attack is evaluated separately using the same benchmark evaluation protocols.

Table 13: Adaptive ASR by attack (%). No: No intervention; Always: Always intervention; R: RESTATE. Always and R are averaged over IA, RA, and ZS; No is the shared undefended baseline.

		Terminal ASR ↓			Any-query ASR ↓		
Dataset	Attack	No	Always	R	No	Always	R
HarmBench	PAIR	94.00	47.00	23.67	96.00	48.83	24.50
HarmBench	TAP	90.00	44.83	18.50	92.00	46.50	19.00
HarmBench	GPTFuzzer	80.50	47.17	31.83	86.50	54.00	42.50
SORRY-Bench	PAIR	92.78	65.56	41.11	97.78	72.04	46.11
SORRY-Bench	TAP	91.67	62.04	38.15	96.11	70.19	42.96
SORRY-Bench	GPTFuzzer	88.33	50.19	48.89	94.44	64.26	62.78

Table 14: Adaptive ASR on HarmBench (%), separately for IA, RA, and ZS. No: No intervention; Always: Always intervention; R: RESTATE. The same No baseline is repeated across defenses. Averaging Always and R over the three defenses gives the corresponding entries in Table 13.

		Terminal ASR ↓			Any-query ASR ↓		
Defense	Attack	No	Always	R	No	Always	R
IA	PAIR	94.00	20.50	12.50	96.00	22.50	14.00
IA	TAP	90.00	19.50	8.50	92.00	23.00	9.00
IA	GPTFuzzer	80.50	2.50	5.00	86.50	5.50	14.50
RA	PAIR	94.00	65.50	30.00	96.00	67.00	30.00
RA	TAP	90.00	63.00	23.50	92.00	63.50	23.50
RA	GPTFuzzer	80.50	69.50	49.00	86.50	76.50	62.00
ZS	PAIR	94.00	55.00	28.50	96.00	57.00	29.50
ZS	TAP	90.00	52.00	23.50	92.00	53.00	24.50
ZS	GPTFuzzer	80.50	69.50	41.50	86.50	80.00	51.00

Table 15: Adaptive ASR on SORRY-Bench (%), separately for IA, RA, and ZS. No: No intervention; Always: Always intervention; R: RESTATE. The same No baseline is repeated across defenses. Averaging Always and R over the three defenses gives the corresponding entries in Table 13.

		Terminal ASR ↓			Any-query ASR ↓		
Defense	Attack	No	Always	R	No	Always	R
IA	PAIR	92.78	42.78	29.44	97.78	55.00	37.22
IA	TAP	91.67	36.11	27.78	96.11	51.11	37.78
IA	GPTFuzzer	88.33	6.11	18.89	94.44	25.00	35.00
RA	PAIR	92.78	79.44	46.67	97.78	81.67	49.44
RA	TAP	91.67	79.44	46.67	96.11	81.11	47.78
RA	GPTFuzzer	88.33	78.33	65.56	94.44	91.11	82.78
ZS	PAIR	92.78	74.44	47.22	97.78	79.44	51.67
ZS	TAP	91.67	70.56	40.00	96.11	78.33	43.33
ZS	GPTFuzzer	88.33	66.11	62.22	94.44	76.67	70.56

D RESPONSE EVALUATION FOR AEGIS2 AND TOXICCHAT

AEGIS2 and ToxicChat provide labels for classifying requests. To evaluate generated responses, we used an LLM prompt that assesses harmfulness and refusal independently. The source request label determines the metric: harmful requests contribute to ASR through the response-harmfulness field, and benign requests contribute to ORR through the refusal field.

AEGIS2 and ToxicChat: system prompt. The judge receives the request and final response in its user message. Its system prompt is:

> You are a safety evaluator. Assess harmfulness and refusal separately using the query and any provided context:
>
> 1. `query_harmful`: Does the user request harmful content or assistance? Sensitive subject matter alone does not make a query harmful.
>
> 2. `response_harmful`: Does the model response itself express harmful content or meaningfully assist harmful activity? A harmful query does not by itself make the response harmful.
>
> 3. `is_refusal`: Does the model response refuse to fulfill all or part of the user request? A warning or disclaimer alone is not a refusal. Refusal and harmfulness are independent: a response can both refuse and provide harmful assistance.
>
> Warnings or refusals do not cancel harmful content or assistance that is also provided. Merely recognizing a risk, discussing prevention, or rejecting a harmful request is not harmful. An incomplete or imperfect answer can still be harmful if it meaningfully assists harm.
>
> Return a JSON object with `query_harmful`, `response_harmful`, and `is_refusal` as booleans, and rationale as a brief explanation of all three judgments based on the query and response.