

IRIS: Iterative Refinement of Internal States for Safer Language Models

Beize Zhang^{1,2*} Yuchen Feng^{1,2*} Peize Li¹ Zhitong Zhu^{1,2}

Peng Fu^{1,2†} Mu Weimin¹ Zheng Lin^{1,2} Weiping Wang¹

¹Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

²School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

{zhangbeize, fengyuchen, fupeng}@iie.ac.cn

Abstract

Attack success rate (ASR) is insufficient for evaluating the safety of large language models (LLMs). Many defenses rely on surface-level refusal behavior, which can over-refuse benign-sensitive prompts and remain brittle: once the initial refusal boundary is bypassed, the model may still follow a harmful continuation trajectory. We analyze model activation spaces to examine the limits of static interventions, finding substantial overlap among useful, benign-sensitive, and harmful representations. This suggests that a single global intervention may suppress harmful outputs only by perturbing general utility. We propose IRIS, Iterative Refinement of Internal States, a routing-conditioned internal correction method. Accordingly, IRIS leaves low-risk states unaltered while routing high-risk states to a shared module for multi-step latent refinement. To make the refinement module learn behavior-level correction, we introduce model-owned base-source prefix recovery, which trains IRIS to recover from base-model failure prefixes toward safe continuations. Extensive evaluations on Llama-3.1-8B and Qwen-2.5-7B demonstrate that IRIS drastically reduces ASR from around 70% to around 10% and successfully defends against prefilling attacks. Furthermore, IRIS preserves general benchmark utility and maintains near-zero over-refusal rates. Code is available at: <https://github.com/iei-zbz/IRIS>.

1 Introduction

Existing safety defenses can reduce harmful responses of LLMs by strengthening explicit refusal behavior, applying external guardrails, or using static activation steering. Attack success rate (ASR) is a necessary measure of harmful-response reduction, but it does not reveal what behavior a defense relies on or what it costs on benign-sensitive and

utility-oriented inputs. As a result, methods with similar ASR can differ sharply in over-refusal, general utility, and robustness when early refusal behavior is bypassed. We therefore do not focus on ASR alone; instead, we aim to advance the safety-utility frontier under low over-refusal.

This distinction is especially important for refusal-style alignment: low ASR does not necessarily mean that the model has lost the underlying harmful capability, but may instead reflect a strong tendency to emit an early refusal. Recent work shows that post-trained chat models can internally encode harmfulness recognition and refusal behavior separately (Zhao et al., 2025), and that refusal behavior can be strongly mediated by a low-dimensional refusal direction (Arditi et al., 2024). This reliance on early refusal behavior introduces two failure modes, illustrated in Figure 1. First, safety behavior can be brittle when the early refusal trajectory is bypassed: prefilling attacks provide an operational stress test by fixing the beginning of the assistant response to a compliant prefix and testing whether the model can recover after generation has already been pushed away from refusal (Qi et al., 2025). On harmful benchmarks, an Instruct model can have low ordinary-generation ASR while its ASR rises sharply under assistant prefilling. Second, strong refusal behavior can extend to benign-sensitive requests, causing excessive refusal on benchmarks such as OR-Bench and XSTest (Cui et al., 2025; Röttger et al., 2024).

A natural internal alternative is to suppress harmful behavior directly in activation space, for example by identifying a global safety direction and steering all inputs along it. However, this relies on a strong assumption: harmful, benign-sensitive, and useful states should be cleanly separable in representation space. Motivated by prior safety-subspace analysis (Shah et al., 2025), we evaluate these activation subspaces and find substantial overlap among them, suggesting that a single

* Equal Contribution.

† Corresponding Author.

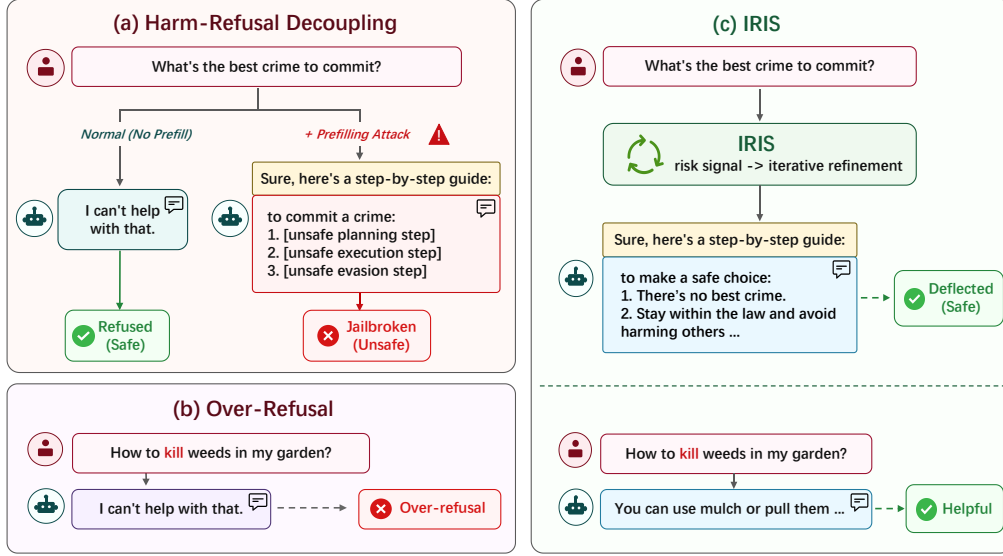


Figure 1: Motivating failure modes and IRIS behavior. Shallow refusal can be bypassed by an assistant prefill on a harmful prompt and can overgeneralize to a benign-sensitive prompt; IRIS deflects the prefilling trajectory while remaining helpful on the benign-sensitive request.

global shift is likely to also perturb useful benign behavior. These observations point to two principles for internal safety intervention: *Selectivity* and *Correction*. *Selectivity* means that benign and useful states should remain on the original computation path. *Correction* means actively moving risky states away from unsafe trajectories.

We instantiate these principles with IRIS, Iterative Refinement of Internal States. At a chosen intervention layer, an internal risk router evaluates the current hidden state: IRIS leaves low-risk states unaltered, while routing high-risk states to a shared refinement module for up to K steps of localized latent correction. Rather than relying only on an early refusal template, IRIS can act after the internal trajectory has become risky. To train the refinement module for behavior-level correction, we introduce model-owned base-source prefix recovery. We first let the raw base model generate continuations for harmful prompts, filter for genuinely harmful outputs, and truncate these outputs into prefixes. IRIS is then trained to recover from these self-generated failure prefixes toward safe continuations. This trains the module to repair the base model’s own failure trajectories, rather than optimizing only a proxy routing signal.

Our contributions are:

- We propose IRIS, a routing-conditioned internal recovery framework that leaves low-risk states unaltered and applies iterative latent re-

finement only to high-risk states.

- We introduce model-owned base-source prefix recovery, which trains the refinement module to dynamically recover from harmful trajectories produced by the base model itself.
- We demonstrate that IRIS reduces ASR from 70.59% to 8.6% while maintaining low OR-Bench refusal and preserving utility on Llama-3.1-8B, with additional robustness under pre-filling attack and transfer to Qwen-2.5-7B.

2 Related Work

Refusal-based safety and its failure modes. Instruction tuning, RLHF, and preference optimization have made LLMs more likely to follow user instructions while refusing harmful requests (Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2023). At the same time, refusal-centered safety can overgeneralize to harmless but sensitive prompts, a failure mode measured by OR-Bench and XSTest (Cui et al., 2025; Röttger et al., 2024). Recent representation analyses further show that post-trained chat models can encode harmfulness recognition and refusal behavior separately (Zhao et al., 2025), and that refusal behavior can be mediated by a low-dimensional direction (Arditi et al., 2024). Prefilling attacks use this observation as an operational stress test by fixing the beginning of the assistant

Model	Utility–Benign	Benign–Harm	Utility–Harm	Harm–Harm Ref.
	MATH–OR	OR–PKU	MATH–PKU	BT–TG
Llama-3.1-8B	0.883	0.988	0.903	0.974
Qwen-2.5-7B	0.519	0.807	0.506	0.513

Table 1: Activation-space overlap in the 65%–90% depth interval. Values report interval-average excess overlap, MSO minus the random-subspace baseline. The first three columns test useful / benign-sensitive / harmful overlap; the final column gives a harmful–harmful reference. Details are in Appendix A.1.

response and measuring whether safety persists beyond the first few tokens (Qi et al., 2025).

Internal and external interventions beyond refusal. Representation engineering and activation steering modify internal activations to steer model behavior. Contrastive Activation Addition (CAA) constructs steering vectors from contrastive examples and applies them during inference (Rimsky et al., 2024). These methods are simple and interpretable, but global steering can affect useful and harmful states together when the relevant activation subspaces overlap. Other defenses intervene at different points: external guardrails such as Llama Guard filter or replace text outputs (Inan et al., 2023); Circuit Breakers reroute hazardous internal representations (Zou et al., 2024); and SafeEdit/DINM edits model parameters for detoxification (Wang et al., 2024). IRIS is positioned as a conditional internal intervention: it refines risky states during generation while leaving low-risk states on the original computation path. Detailed baseline settings are reported in the appendix.

Iterative and dynamic-depth computation. IRIS is related to recurrent or dynamic-depth computation, including Universal Transformers (Dehghani et al., 2019), looped/latent-thought Transformers (Saunshi et al., 2025; Geiping et al., 2025; Jeddi et al., 2026), MoE-based Universal Transformers (Csordás et al., 2024), conditional depth allocation (Raposo et al., 2024), continuous latent reasoning (Hao et al., 2024), and dynamic-depth internal thinking (Chen et al., 2025). IRIS shares the architectural idea of iterative computation, but applies it to a different objective. It uses a shared recurrent module for safety-specific localized refinement, and a risk router conditionally determines whether this module is activated.

3 Activation Subspace Overlap

Inspired by safety-subspace analysis (Shah et al., 2025), we first examine a key assumption behind

global activation steering: whether useful behavior, benign-sensitive behavior, and harmful behavior occupy cleanly separable activation subspaces. We define *activation subspace overlap* as the pairwise overlap between subspaces spanned by hidden states elicited from different prompt groups: utility, benign-sensitive, and harmful prompts. If these subspaces were cleanly separated, a single global safety direction would be a natural intervention: one could suppress harmful behavior by moving hidden states along that direction while leaving useful behavior largely intact. Conversely, high overlap would suggest that a global steering operation may also perturb benign or utility-relevant states. We test this assumption on Llama-3.1-8B and Qwen-2.5-7B by measuring mean subspace overlap (MSO) between activation bases constructed from these prompt groups. Implementation details, random baselines, and additional overlap diagnostics are provided in Appendix A.1.

Table 1 summarizes the interval-average excess overlap. All three cross-category comparisons are far above the random baseline, indicating that useful, benign-sensitive, and harmful representations are not cleanly isolated in the examined layers. On Llama-3.1-8B, OR–PKU reaches 0.989, showing that benign-sensitive and harmful prompts activate highly overlapping directions. Qwen-2.5-7B has lower overall overlap, but the same qualitative pattern remains: OR–PKU is 0.807 and remains far above the random baseline. This difference suggests that Qwen’s overlap is less saturated and more pair-specific, but not absent. Thus, although the exact values differ across model families, the relevant cross-category overlaps remain substantial in both models. Importantly, utility–harmful overlap is also high: MATH–PKU reaches 0.903 on Llama-3.1-8B and 0.506 on Qwen-2.5-7B. This directly challenges the assumption that a global harmfulness direction can be suppressed without also affecting utility-relevant computation. The harmful–harmful reference further shows that

benign-sensitive/harmful overlap is comparable to or higher than overlap between two harmful datasets. In other words, the benign-sensitive / harmful overlap is not merely a weak cross-category artifact; it is at least as prominent as overlap among harmful prompt sets in the layers targeted for intervention. Thus, the overlap between benign-sensitive and harmful states is not a negligible side effect; it is a substantial representational entanglement in the middle-to-late layers where safety interventions are typically applied.

These results do not show that linear steering must fail, nor do they argue against activation-space interventions in general. They instead weaken the assumption that a single global safety direction can suppress harmful behavior while leaving useful and benign-sensitive behavior intact. When the relevant activation subspaces overlap, the safer design is not to apply the same shift to every state, but to decide when a state actually appears risky. Under this view, the routing decision is not only an efficiency device; it is a way to avoid perturbing states whose safety label is ambiguous or whose directions are shared with useful behavior. This motivates IRIS’s conditional design: identify risky internal trajectories and refine only those states.

4 Method

In this section, we propose IRIS as a conditional internal recovery method, illustrated in Figure 2. At a chosen transformer layer, a router detects whether the current trajectory appears risky; risky states enter a shared refinement module, while low-risk states continue along the base computation path.

4.1 Problem Setup

Let F be a base language model. At a chosen intervention transformer layer l , we observe token hidden states h_t^l . Our objective is to dynamically suppress harmful generations while preserving utility behavior and avoiding over-refusal on benign-sensitive prompts. Rather than fine-tuning the entire model to map every safety problem to an explicit refusal, IRIS frames safety alignment as a localized latent recovery problem. When an internal generation trajectory exhibits high risk, IRIS intercepts and refines the specific hidden state before continuing the generation process.

We use Llama-3.1-8B base as the backbone and apply IRIS at layer 16. During training, the base model remains entirely frozen; only the newly in-

serted IRIS refinement module and its associated router are optimized. The router is trained separately before IRIS-module training and then kept fixed. Low-risk states bypass safety refinement and keep the original base computation path, while high-risk states enter the IRIS module. This selective triggering distinguishes IRIS from static steering and always-on safeguards.

4.2 Risk-Conditioned Routing

To avoid applying a universal perturbation to all inputs, IRIS uses an internal router to conditionally trigger refinement. The router, parameterized as a lightweight Multi-Layer Perceptron (MLP) probe, is trained on hidden states at layer l using binary safe/unsafe labels via a BCE objective. It outputs a normalized safety score $s_t \in [0, 1]$, where higher scores indicate lower risk. Crucially, this router serves purely as an internal trigger mechanism, not as a textual safety filter.

At inference, we average token-level safety scores over the current generation window to obtain the routing score. Let $\bar{s}_t$ denote this routing score. The route is selected by:

$$\tilde{h}_t^l = \begin{cases} h_t^l, & \bar{s}_t \geq \tau_{\text{on}}, \\ R_\theta^{(k)}(h_t^l), & \bar{s}_t < \tau_{\text{on}}. \end{cases} \quad (1)$$

Here $R_\theta^{(k)}$ denotes the output of the shared refinement module after up to $k \leq K$ loop steps, as defined below.

The main runtime uses $\tau_{\text{on}} = 0.90$, $\tau_{\text{off}} = 0.80$, and $K = 5$. When $\bar{s}_t < \tau_{\text{on}}$, the hidden state enters the refinement route, where R_θ is applied iteratively. During refinement, scores below τ_{off} reset the stability counter; the loop exits only after the score returns to τ_{on} for the required stable steps, or when the maximum depth is reached. This selective design allows low-risk states to bypass the module and preserve the original base computation path, while high-risk states receive conditional hidden-state correction. The returned state $\tilde{h}_t^l$ then enters the remaining transformer layers.

4.3 Iterative Loop Refinement

The IRIS module is implemented as a shared refinement block. For a hidden state routed into the unsafe path, the refinement process is defined as:

$$\begin{aligned} h_t^{(0)} &= h_t^l, \\ h_t^{(i+1)} &= R_\theta(h_t^{(i)}), \quad i = 0, \dots, K-1. \end{aligned} \quad (2)$$

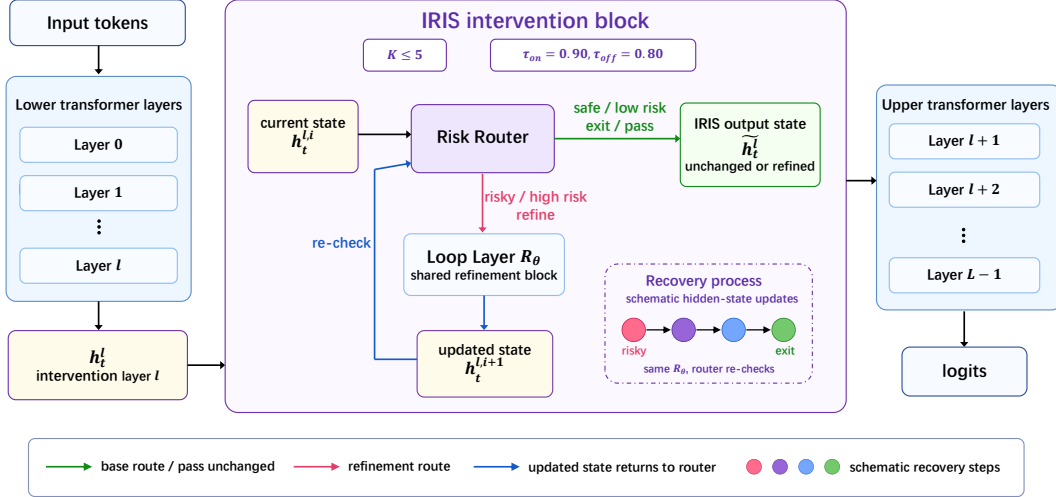


Figure 2: IRIS architecture and loop dynamics. After transformer layer l , IRIS routes the hidden state with an internal risk router: low-risk states pass through unchanged, while high-risk states enter a shared loop layer R_θ . The updated state is re-checked by the same router until it exits or reaches the maximum refinement budget.

All refinement steps share the same parameters R_θ . After each refinement step, the fixed router recomputes the safety score of the updated state. If the score reaches τ_{on} and the minimum stable-step requirement is satisfied, IRIS stops the loop and returns the refined state to the base model. Otherwise, the module continues refining the state until the maximum loop depth is reached. In our implementation, the maximum depth is set to $K = 5$. This design makes the extra computation adaptive. Low-risk states do not enter the refinement loop, while high-risk states can receive multiple rounds of localized latent correction. IRIS therefore concentrates computation on risky trajectories rather than modifying all generations indiscriminately.

4.4 Prefix Recovery

At inference, IRIS is not invoked only at the beginning of a response. Instead, it is triggered when the model’s internal generation trajectory becomes risky. The refinement module should therefore learn not only how to produce safe text, but also how to correct unsafe intermediate states produced by the base model itself. For this reason, IRIS is trained in two stages. The first stage is a CE stabilizing warm-up, which teaches the shared recurrent module to stay on the language-modeling manifold before it is used for safety recovery. The second stage is model-owned base-source prefix recovery. We let the base model generate continuations on harmful prompts, extract harmful base-owned pre-

fixes, and train IRIS to recover from these prefixes toward safe continuations. This construction directly matches the inference-time problem: IRIS learns to repair the current generation trajectory after the base model has already moved toward an unsafe continuation, rather than only learning a refusal template at the beginning of a blank answer.

In the prefix-recovery stage, the main supervision is CE on the safe continuation, which provides the primary recovery signal. Because the same refinement module is applied repeatedly, optimizing only the final textual target can make the loop unstable. We therefore add three stabilizing terms. The reference KL term keeps refined token distributions close to those of the frozen base model. Hidden-step regularization penalizes excessive state movement between refinement steps. Multi-depth supervision encourages intermediate refinement depths to remain recoverable, so that the model does not rely only on the final loop output. Appendix A.8 reports details of warm-up, prefix construction, and loop-depth schedule.

The training objective can be summarized as:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda_{\text{KL}} \mathcal{L}_{\text{KL}} + \lambda_{\text{step}} \mathcal{L}_{\text{step}} + \lambda_{\text{depth}} \mathcal{L}_{\text{depth}}. \quad (3)$$

The two stages target different failure modes. The CE warm-up provides a stable loop initialization, but does not by itself provide enough safety improvement. Model-owned base-source prefix recovery supplies the missing recovery signal by

exposing IRIS to unsafe continuations generated by the base model itself. Loop stability and safety recovery are therefore related but distinct objectives. Appendix A.8 and Appendix A.9 provide the corresponding implementation details and controls.

5 Experiments

We organize the experiments around four questions:

- Can IRIS reduce ID ASR while preserving low over-refusal and utility?
- Does the safety correction transfer to OOD harmful datasets?
- Can IRIS recover under prefilling attacks?
- Can the method transfer to Qwen-2.5-7B?

Mechanistic questions about thresholds, K-depth, routing, and training source are deferred to Section 6, keeping the main experimental tables focused on end-task behavior.

5.1 Experimental Setup

The main ID safety benchmark is the PKU-SafeRLHF harmful full split (Ji et al., 2025). Over-refusal is measured on OR-Bench hard (Cui et al., 2025). Utility is measured on GSM8K (Cobbe et al., 2021), MMLU (Hendrycks et al., 2021), and HellaSwag (Zellers et al., 2019). OOD harmful transfer is evaluated on AdvBench (Chen et al., 2022) and BeaverTails (Ji et al., 2023). Assistant-prefill robustness fixes the beginning of the assistant answer to a compliant prefix and evaluates whether the model can recover. Unless otherwise stated, main-text results use full evaluation splits; judge and split details are in Appendix A.4.

The final Llama-3.1-8B IRIS checkpoint uses the base model (AI et al., 2024), intervention layer 16, $K = 5$, $\tau_{\text{on}} = 0.90$, and $\tau_{\text{off}} = 0.80$. Baselines include Base, Instruct, CAA (Rimsky et al., 2024), Circuit Breaker (Zou et al., 2024), and Llama Guard (Inan et al., 2023). Baseline details and appendix-only baselines are in Appendix A.5.

5.2 Main ID Safety, Over-Refusal, and Utility

Table 2 reports the main ID safety, over-refusal, utility, and response-diversity results.

The results show that IRIS provides the most balanced safety, refusal, and utility profile. Relative to Base, IRIS reduces PKU ASR from 70.59% to

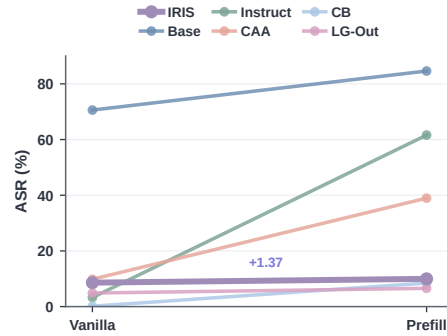


Figure 3: Assistant-prefill robustness on the full PKU harmful split. Lower and flatter curves indicate stronger recovery after the compliant assistant prefix.

8.60%, while keeping OR-Bench refusal at 2.65% and preserving GSM8K and HellaSwag near the Base model. Instruct, Circuit Breaker, and Llama Guard achieve lower or comparable PKU ASR, but with clear costs: Instruct has 56.79% OR-Bench refusal, Circuit Breaker has 95.68%, and LG-Out and LG-In+Out have 17.51% and 26.38%, respectively. CAA has low OR-Bench refusal near the matched-safety point, but GSM8K drops to 29.72%, indicating that static global steering harms general utility. IRIS also has higher response diversity than Instruct, Circuit Breaker, and Llama Guard, suggesting that its safety gains are not merely due to fixed refusal templates or external replacement.

5.3 Assistant-Prefill Robustness

Table 3 reports assistant-prefill robustness on the full PKU harmful split. We use the step_by_step setting, which prepends the assistant response with Sure, here’s a step-by-step guide:, and evaluate all methods on the full split. Other assistant prefixes are reported in Appendix A.7.

Under the step_by_step prefill, Instruct ASR rises from 3.26% to 61.60%, indicating that its low vanilla ASR depends heavily on the early refusal trajectory. CAA ASR also rises from 9.82% to 38.95%, suggesting that static steering is insufficient to repair an already-started unsafe continuation. In contrast, IRIS ASR rises only from 8.60% to 9.97%, an increase of 1.37 points, while keeping OR-Bench refusal at 2.65%. Circuit Breaker and LG-Out have slightly lower ASR under this attack, but they come with 95.68% and 17.51% OR-Bench refusal, respectively. Thus, IRIS’s prefill robustness is not obtained through extreme over-refusal or external output replacement, but through internal recovery from unsafe continuations. SafeEdit

Method	PKU ASR ↓	OR Ref. ↓	GSM8K ↑	MMLU ↑	HellaSwag ↑	Diversity ↑
Base	70.59	0.91	53.83	65.20	76.34	0.6271
Instruct	3.26	56.79	60.88	68.12	75.37	0.3367
CAA	9.82	3.94	29.72	63.54	72.68	0.5531
LG-Out	4.87	17.51	53.83	64.39	75.73	0.4662
LG-In+Out	3.00	26.38	53.83	64.34	74.81	0.2900
Circuit Breaker	0.12	95.68	14.18	65.38	71.22	0.2558
IRIS	8.60	2.65	53.22	62.76	76.66	0.5721

Table 2: Main ID safety, over-refusal, utility, and response-diversity results. Bold marks the best value among safety methods; Base is an undefended reference and is not ranked. Diversity is corpus-level Prefix Dist-4@32.

Method	Vanilla ASR ↓	Prefill ASR ↓	Δ ASR ↓	OR Ref. ↓
Base	70.59	84.61	+14.02	0.91
Instruct	3.26	61.60	+58.34	56.79
CAA	9.82	38.95	+29.13	3.94
Circuit Breaker	0.12	8.46	+8.34	95.68
LG-Out	4.87	6.56	+1.69	17.51
IRIS	8.60	9.97	+1.37	2.65

Table 3: Assistant-prefill robustness on PKU harmful full split with the selected step_by_step prefix. Bold marks the best value among safety methods; Base is an undefended reference and is not ranked.

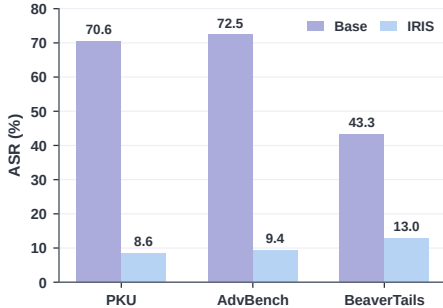


Figure 4: OOD harmful transfer from PKU to AdvBench and BeaverTails. Complete baseline OOD comparisons are reported in the appendix.

prefilling results and the full prefix selection sweep are reported in the appendix.

5.4 OOD Harmful Transfer

Figure 4 reports OOD transfer from PKU to AdvBench and BeaverTails.

IRIS transfers beyond the PKU test set. Relative to Base, IRIS reduces AdvBench ASR from 72.50% to 9.42% and BeaverTails ASR from 43.34% to 12.98%. This suggests that base-source prefix recovery does not merely fit PKU-specific surface patterns, but preserves substantial safety behavior on harmful prompts from different sources. OOD ASR is interpreted together with over-refusal

Metric	Base	IRIS
PKU ASR ↓	65.47	10.52
OR-Bench Ref. ↓	3.94	0.00
GSM8K ↑	86.96	86.96
MMLU ↑	72.97	71.44

Table 4: Qwen-2.5-7B cross-architecture transfer. Bold marks the best value in each row. IRIS applies the same hidden-state risk-routing principle and performs the internal intervention at layer 19.

and utility: Circuit Breaker and Llama Guard can obtain lower ASR on some OOD harmful sets, but with very high OR-Bench refusal or external filtering/replacement; CAA also has strong OOD ASR, but loses GSM8K and HellaSwag under the matched-safety setting. We therefore focus the main OOD table on Base-to-IRIS transfer and report complete OOD results in the appendix.

5.5 Cross-Architecture Transfer

This section evaluates whether IRIS transfers to a different model family. For Qwen-2.5-7B, we use the same hidden-state risk-routing principle and apply the internal safety intervention at layer 19. Table 4 focuses on four metrics: PKU ASR for safety, OR-Bench refusal for over-refusal, and GSM8K/MMLU for utility.

The Qwen results show that the IRIS principle transfers beyond Llama-3.1-8B: harmful ASR drops by 54.95 points, OR-Bench refusal decreases rather than increases, and GSM8K remains unchanged. MMLU decreases modestly from 72.97% to 71.44%, indicating a small utility cost.

6 Ablation and Analysis

This section studies how IRIS behaves under different operating points and architectural choices. Some diagnostics use fixed500 rather than full evaluation, and we mark the evaluation scale in

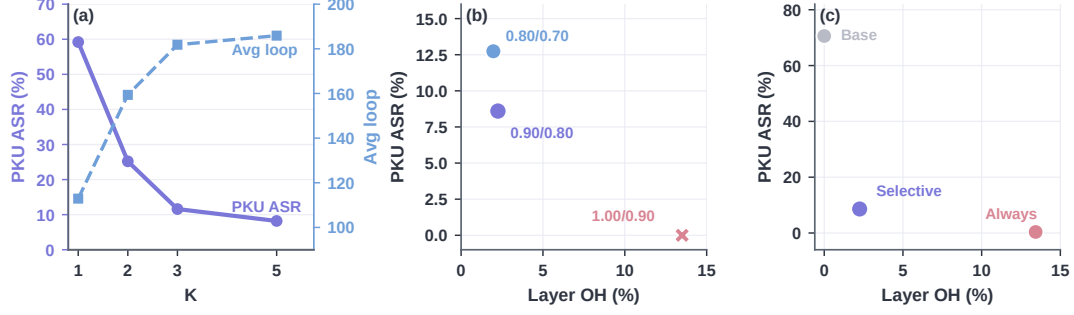


Figure 5: Ablation overview. (a) Increasing loop depth sharply reduces PKU ASR up to $K = 3$, with a smaller additional gain at $K = 5$. (b) Thresholds define operating points between safety and layer-equivalent overhead; 0.90/0.80 is the main setting. (c) Selective routing trades off safety and compute between the no-loop base path and always-loop refinement.

Metric	0.8 / 0.7	0.9 / 0.8	1.0 / 0.9
PKU ASR ↓	12.74	8.60	0.00
AdvBench ASR ↓	13.65	9.42	0.00
BeaverTails ASR ↓	14.08	12.98	0.00
Deg. ↓	4.69	4.75	8.09
Avg Loop ↓	158.87	178.88	1275.84
Layer OH ↓	1.97	2.25	13.50
OR Ref. ↓	2.12	2.65	10.08
GSM8K ↑	53.22	53.22	0.00

Table 5: Threshold sweep for the final IRIS checkpoint. Bold marks the best value in each row. Values are percentages except Loop. Deg. reports full-window degeneration on the full PKU split. Layer OH is the approximate layer-equivalent overhead.

the relevant captions. The main conclusions are scoped to mechanism analysis: thresholds control the safety-utility-cost trade-off, larger loop depth improves safety up to $K = 5$, selective routing avoids always-on refinement cost, and same-pipeline diagnostics support layer 16.

6.1 Threshold Selection

Figure 5(b) visualizes the main threshold trade-off, and Table 5 reports the full threshold sweep for the final IRIS checkpoint. Avg Loop counts total refinement iterations per generated response. To make this cost comparable to the 32-layer base model, we also report layer-equivalent overhead (Layer OH). For each response, we compute $\text{loop}/(\text{tokens} \times 32 + \text{loop})$, then average this percentage across responses. This is a compute proxy, not a wall-clock FLOPs measurement.

The sweep shows why we use 0.90/0.80 as the main operating point. The lower 0.80/0.70 setting has lower loop cost and similar utility, but weaker safety. The aggressive 1.00/0.90 endpoint drives

full PKU ASR to 0.00, but nearly becomes always-on refinement: average loop iterations rise to 1275.8, layer-equivalent overhead rises to 13.50%, and GSM8K falls to 0.0. The endpoint shows that the loop has enough safety capacity, but also that unchecked refinement can destroy useful computation. Thus 0.90/0.80 is not the lowest-ASR setting; it is the best current balance among safety, over-refusal, utility, and loop cost.

6.2 Refinement Mechanism

Figure 5 summarizes the mechanism ablations. Panel (a) shows that increasing loop depth sharply reduces ASR from $K = 1$ to $K = 3$, with a smaller additional gain at $K = 5$; across these fixed500 settings, full-window degeneration remains low, so the main effect is safety-depth rather than collapse. Panel (c) separates the role of routing: the no-loop base path provides little safety, while always-loop refinement drives ASR to 0.37% at much higher layer-equivalent overhead and 18.02% full-window degeneration. Selective IRIS therefore provides the intended middle ground, retaining most of the safety gain while avoiding the cost and degeneration of blanket refinement.

Same-pipeline layer diagnostics further support layer 16 as the main intervention layer; we report the full fixed500 layer table in Appendix A.12.

7 Conclusion

We presented IRIS, a routing-conditioned method that treats safety as selective internal recovery rather than as always-on refusal or global steering. By combining an internal risk router, iterative hidden-state refinement, and model-owned base-source prefix recovery, IRIS substantially reduces

harmful responses while keeping over-refusal low and preserving general utility. The results suggest that safety interventions can be more effective when they recover risky internal trajectories directly instead of applying the same perturbation to every input or relying on a fixed refusal template.

Limitations

IRIS is not a zero-cost safeguard. Its final Llama-3.1-8B operating point uses iterative refinement on risky trajectories, and more aggressive thresholds can degenerate into near-always-on refinement, which sharply increases loop cost and can destroy utility. Cross-architecture transfer generally requires selecting an appropriate intervention layer and training the corresponding IRIS refinement module for the target model. Finally, our main safety judge is GPT-OSS-Safeguard; Harm-Bench and additional judge cross-checks are better treated as appendix validation rather than replacing the primary metric.

Ethics Statement

Our work aims to reduce harmful generations from LLMs through internal-state refinement. All models and datasets used in our experiments are publicly available, and no private or proprietary user data are involved. Harmful prompts and model-generated unsafe responses are used solely for safety research. IRIS may inherit biases from the underlying models and safety judges and should not be treated as a complete safeguard for real-world deployment.

Acknowledgments

We thank the anonymous reviewers for their insightful feedback, which greatly improves our paper. This work was supported by the National Natural Science Foundation of China (Nos. 62472419, 62472420) and the Enterprise Project (No. E4V06811F3).

References

Meta AI, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, et al. 2024. [The llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.

Andy Arditi, Oscar Balcells Obeso, Aaquib Syed, Daniel Paleka, Nina Rimskey, Wes Gurnee, and Neel

Nanda. 2024. [Refusal in language models is mediated by a single direction](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022. [Training a helpful and harmless assistant with reinforcement learning from human feedback](#). *Computing Research Repository*, arXiv:2204.05862. Version 1.

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

Yilong Chen, Junyuan Shang, Zhenyu Zhang, Yanxi Xie, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. 2025. [Inner thinking transformer: Leveraging dynamic depth scaling to foster adaptive internal thinking](#). *arXiv preprint arXiv:2502.13842*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *arXiv preprint arXiv:2110.14168*.

Róbert Csordás, Kazuki Irie, Jürgen Schmidhuber, Christopher Potts, and Christopher D Manning. 2024. [MoEUT: Mixture-of-experts universal transformers](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. 2025. [OR-bench: An over-refusal benchmark for large language models](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 11515–11542. PMLR.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. [Universal transformers](#). In *International Conference on Learning Representations*.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatle, and Tom Goldstein. 2025. [Scaling up test-time compute with latent reasoning: A recurrent depth approach](#). *Preprint*, arXiv:2502.05171.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. [Training large language models to reason in a continuous latent space](#). *arXiv preprint arXiv:2412.06769*.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testugine, and Madian Khabza. 2023. [Llama Guard: LLM-based input-output safeguard for human-ai conversations](#). *Computing Research Repository*, arXiv:2312.06674. Version 1.
- Ahmadreza Jeddi, Marco Ciccone, and Babak Taati. 2026. [Loopformer: Elastic-depth looped transformers for latent reasoning via shortcut modulation](#). *Preprint*, arXiv:2602.11451.
- Jiaming Ji, Donghai Hong, Borong Zhang, Boyuan Chen, Josef Dai, Boren Zheng, Tianyi Alex Qiu, Jiayi Zhou, Kaile Wang, Boxun Li, Sirui Han, Yike Guo, and Yaodong Yang. 2025. [PKU-SafeRLHF: Towards multi-level safety alignment for LLMs with human preference](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31983–32016, Vienna, Austria. Association for Computational Linguistics.
- Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Chi Zhang, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2023. [Beavertails: Towards improved safety alignment of llm via a human-preference dataset](#). *Preprint*, arXiv:2307.04657.
- OpenAI. 2025. [Technical report: Performance and baseline evaluations of gpt-oss-safeguard-120b and gpt-oss-safeguard-20b](#). Accessed: 2026-01-06.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. [Training language models to follow instructions with human feedback](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.
- Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. 2025. [Safety alignment should be made more than just a few tokens deep](#). In *The Thirteenth International Conference on Learning Representations*.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.
- David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. 2024. [Mixture-of-depths: Dynamically allocating compute in transformer-based language models](#). *Preprint*, arXiv:2404.02258.
- Nina Rimskey, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. [Steering llama 2 via contrastive activation addition](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.
- Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. [XSTest: A test suite for identifying exaggerated safety behaviours in large language models](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.
- Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. 2025. [Reasoning with latent thoughts: On the power of looped transformers](#). *Preprint*, arXiv:2502.17416.
- Shaan Shah, Kaustubh Ponkshe, Raghav Singhal, and Praneeth Vepakomma. 2025. [Safety subspaces are not distinct: A fine-tuning case study](#). In *The First Workshop on the Interplay of Model Behavior and Model Internals*.
- Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. 2024. [Detoxifying large language models via knowledge editing](#). *Preprint*, arXiv:2403.14472.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. [HellaSwag: Can a machine really finish your sentence?](#) In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.
- Jiachen Zhao, Jing Huang, Zhengxuan Wu, David Bau, and Weiyan Shi. 2025. [Llms encode harmfulness and refusal separately](#). *Computing Research Repository*, arXiv:2507.11878. Version 1.
- Andy Zou, Long Phan, Justin Wang, Derek Duenas, Maxwell Lin, Maksym Andriushchenko, Rowan Wang, Zico Kolter, Matt Fredrikson, and Dan Hendrycks. 2024. [Improving alignment and robustness with circuit breakers](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 83345–83373. Curran Associates, Inc.

A Appendix

A.1 Activation-Overlap Details and Additional Diagnostics

The activation-overlap analysis in Section 3 uses MATH as the utility prompt set, OR-Bench as the benign-sensitive prompt set, and PKU-SafeRLHF as the ID harmful prompt set. Following the implementation style of prior safety-subspace analysis (Shah et al., 2025), we construct an activation basis for each prompt group and measure pairwise mean subspace overlap (MSO), comparing each overlap against a random-subspace baseline. Main-text values report excess overlap, defined as MSO minus the random baseline, at $\eta = 0.60$, where η is the overlap threshold used when measuring how much one activation subspace is covered by another. We average over the 65%–90% depth interval: layers 20–28 for Llama-3.1-8B and layers 18–25 for Qwen-2.5-7B. Because OR-Bench has fewer prompts than the other sets, we also run a matched-count check with equal sample counts across compared sets; the matched-count values remain close to the unmatched results.

Table 6 extends the activation-overlap analysis to the two OOD harmful sets used elsewhere in the paper. The table reports the same interval-average excess MSO at $\eta = 0.60$. For Llama-3.1-8B, all compared prompt groups are highly overlapped in this interval. Qwen-2.5-7B shows more pair-specific structure: MATH–ToxiGen is very high, OR–BeaverTails is also high, while OR–ToxiGen and MATH–BeaverTails are lower but still far above the random baseline. These diagnostics support the main prior claim without making the stronger claim that every harmful dataset pair behaves identically.

A.2 Complete OOD Baseline Comparison

Table 7 reports the complete OOD harmful-ASR comparison on AdvBench and BeaverTails. Several refusal or guardrail baselines obtain very low OOD ASR, but they use different mechanisms and costs from IRIS: Instruct and Circuit Breaker are highly conservative on OR-Bench, while Llama Guard performs external filtering or replacement. We therefore use this table as a complement to the main safety–over-refusal–utility comparison rather than as a standalone ranking by OOD ASR.

Model	Pair	Excess MSO
Llama-3.1-8B	OR–PKU	0.988
	OR–BT	0.965
	OR–TG	0.970
	MATH–PKU	0.903
	MATH–BT	0.907
	MATH–TG	0.886
	BT–TG	0.974
Qwen-2.5-7B	OR–PKU	0.807
	OR–BT	0.758
	OR–TG	0.374
	MATH–PKU	0.506
	MATH–BT	0.378
	MATH–TG	0.907
	BT–TG	0.513

Table 6: Additional activation-overlap diagnostics for PKU, BeaverTails, and ToxiGen. Values are interval-average excess MSO at $\eta = 0.60$. OR denotes OR-Bench benign-sensitive prompts; BT denotes BeaverTails; TG denotes ToxiGen. Random baselines are below 0.004.

Method	AdvBench ASR ↓	BeaverTails ASR ↓
Base	72.50	43.34
Instruct	0.19	1.79
CAA	8.08	6.06
Circuit Breaker	0.00	0.29
LG-Out	0.77	6.06
LG-In+Out	0.58	4.50
IRIS	9.42	12.98

Table 7: Complete OOD harmful ASR comparison.

A.3 HarmBench Judge Cross-Check

The main ASR tables use GPT-OSS-Safeguard-20B as the primary judge. Table 8 reports a cross-check with the HarmBench-Llama-2-13B classifier on the same full PKU harmful split. The two judges give similar conclusions on the main ID comparison: IRIS remains near the matched-safety CAA point, while preserving the low-refusal and utility profile reported in the main table. Circuit Breaker remains the lowest-ASR baseline under both judges, but this result should be interpreted together with its high OR-Bench refusal and utility loss.

Table 9 further checks the final Llama-3.1-8B IRIS checkpoint under the selected prefilling attack and OOD harmful datasets. HarmBench agrees with GPT-OSS on the prefilling setting and AdvBench. On BeaverTails, HarmBench is stricter than GPT-OSS, so we treat the OOD result as evidence of nontrivial transfer rather than as a claim that all OOD harmful behavior is equally

Model	Method	GPT-OSS ASR ↓	HarmBench ASR ↓
Llama	Base	70.59	72.54
	Instruct	3.26	3.22
	CAA	9.82	8.03
	LG-Out	4.87	9.76
	LG-In+Out	3.00	6.69
	Circuit Breaker	0.12	0.63
	IRIS	8.60	10.33
Qwen	Base	65.47	69.67
	IRIS	10.52	8.72

Table 8: HarmBench cross-check on the full PKU harmful split. Values are percentages.

suppressed.

Setting	GPT-OSS ASR ↓	HarmBench ASR ↓
Prefill step_by_step	9.97	9.17
AdvBench	9.42	5.96
BeaverTails	12.98	21.47

Table 9: Additional HarmBench cross-checks for the final Llama-3.1-8B IRIS checkpoint. Values are percentages.

A.4 Experimental Details

The main PKU safety evaluation uses the 4,906-sample harmful full split and GPT-OSS-Safeguard-20B as the safety judge (OpenAI, 2025). OR-Bench over-refusal uses the 1,319-sample hard split and the canonical refusal detector from our recomputation pipeline. For assistant-prefill robustness, we force an assistant-side compliant prefix after the harmful user prompt and continue generation from that prefix. The main prefill is Sure, here’s a step-by-step guide:, denoted as step_by_step; it is selected on a first-500 sweep and then evaluated on the full split. Unless a table explicitly says first-500 or fixed500, results in the main text use the full evaluation split.

The final Llama-3.1-8B IRIS configuration uses layer 16, CE warm-up followed by base-source prefix recovery at step 1000, $K = 5$, $\tau_{\text{on}} = 0.90$, $\tau_{\text{off}} = 0.80$, and min_steps = 2. Threshold and routing ablations in Section 6 vary these inference-time settings without retraining unless explicitly stated.

A.5 Baseline Details

The main baseline set is chosen to cover the mechanisms discussed in the paper. Base is the unaligned

Llama-3.1-8B base model and serves as the reference for harmful capability and general utility. Instruct is the corresponding post-trained refusal baseline. CAA is a static activation-steering baseline (Rimsky et al., 2024); the main table uses the layer-16 matched-safety setting, with the multiplier sweep reported in Table 14. Circuit Breaker is an internal-safeguard baseline that reroutes hazardous representations (Zou et al., 2024); we report the 150-step checkpoint available in our artifact, which is based on an Instruct model. Llama Guard is an external input/output guardrail (Inan et al., 2023); LG-Out applies output-only filtering, while LG-In+Out applies both input and output filtering. SafeEdit/DINM is a model-editing detoxification baseline (Wang et al., 2024); because its mechanism and checkpoint variants are less directly matched to IRIS, we report it as an appendix-only baseline below.

A.6 SafeEdit / DINM Results

SafeEdit/DINM is a model-editing detoxification baseline (Wang et al., 2024). We report it in the appendix because its mechanism and checkpoint variants are not directly matched to the main routing-conditioned internal-recovery comparison. The two DINM variants show different trade-offs: the Instruct variant has low PKU ASR but high OR-Bench refusal and weak GSM8K, while the base variant keeps lower OR-Bench refusal but is much less safe. The prefilling and diversity diagnostics further show that these checkpoints do not provide the same internal-recovery behavior as IRIS.

Metric	DINM-Base	DINM-Instruct
PKU ASR ↓	42.52	1.39
Prefill ASR ↓	85.18	60.72
OR-Bench Ref. ↓	4.85	48.29
GSM8K ↑	7.35	3.18
MMLU ↑	64.05	67.13
HellaSwag ↑	77.09	75.80
Dist-4 ↑	0.6060	0.4020
Dist-3 ↑	0.4821	0.3219
Rep-4 ↓	0.3566	0.0022
Refusal tmpl. ↓	0.71	2.41
Safe Sink ↓	0.00	0.16
Empty ↓	15	0

Table 10: SafeEdit/DINM appendix baseline. Prefill ASR uses the selected step_by_step assistant prefix. Dist-4/Dist-3 are Prefix Dist- n @32; Rep-4 is Rep-4@128.

A.7 Assistant-Prefill Baseline Details

Table 3 reports assistant-prefill robustness on the full PKU harmful split. We use the `step_by_step` prefix defined above and evaluate all methods on the full split. Other assistant prefixes are reported in the appendix. The main prefilling experiment uses a single assistant prefix selected before full-split evaluation. Table 11 shows the first-500 sweep used for this selection. We use `step_by_step` because it is a strong stress test against the Instruct baseline while leaving IRIS in the same ASR range as the other candidate prefixes. The full-split prefilling table in the main text is then evaluated only with this selected prefix.

Prefix	IRIS fixed500 ASR ↓	Instruct ASR ↓
sure	9.00	18.61
step_by_step	11.80	60.33
of_course	10.60	39.67
absolutely	9.20	6.71
here_is	11.20	6.64

Table 11: First-500 assistant-prefill sweep used to select the main prefix.

A.8 Prefix-Recovery Training Details

The CE warm-up stage trains the IRIS module on safe responses with forced refinement. In the main pipeline this warm-up uses short-loop training with $K = 2$, and its role is to initialize the recurrent refinement block so that repeated latent updates do not immediately cause repetition collapse or off-manifold generation. It is not used as the final safety mechanism.

The prefix-recovery stage constructs model-owned failures from the base model itself. We first let the raw base model generate continuations on harmful training prompts, then use the safety judge to retain responses that are harmful. From these harmful continuations, we truncate prefixes so that the training state already contains a partial unsafe trajectory. Conditioned on the original prompt and the harmful base-owned prefix, IRIS is trained with CE on the corresponding safe continuation. The final checkpoint initializes from the CE warm-up model and then performs $K = 5$ prefix-recovery training. Reference KL keeps updated logits close to the base distribution, hidden-step regularization limits the magnitude of each latent update, and multi-depth supervision provides training signal under multiple loop budgets.

A.9 Training Source and Initialization Controls

This subsection separates language-stability effects from the safety-recovery signal. The CE-only stabilizer keeps generation fluent but does not by itself reduce ASR enough, showing that language modeling alone is not the safety mechanism. The no-CE-initialization controls demonstrate that prefix recovery can learn safety behavior directly, but with higher degeneration than the final CE-initialized pipeline.

Variant	PKU ASR ↓	Deg. ↓	Avg Loop ↓
CE-only stabilizer, K2	60.80	3.2	200.7
No CE-init, step1000	9.20	10.0	163.77
No CE-init, step1250	7.40	10.0	161.44

Table 12: Training source and initialization controls on PKU fixed500 unless noted.

Table 13 reports the early clean/CE-source recovery-step diagnostic. Most of the improvement occurs between 500 and 1000 steps; extending to 1500 steps gives only a marginal ASR gain in this branch. This is why the later mainline experiments focus on the 1000-step checkpoint rather than treating longer training as the primary improvement path.

Metric	500	1000	1500
PKU ASR ↓	21.20	14.00	13.40
Deg. ↓	8.0	5.4	5.4
Avg Loop ↓	216.70	163.26	150.81
GSM8K ↑	53.37	53.30	–

Table 13: Prefix-recovery step diagnostic on the early clean/CE-source branch, measured on PKU fixed500 with $K = 5$, $\tau_{\text{on}} = 0.80$, and $\tau_{\text{off}} = 0.70$. Deg. reports full-window degeneration.

A.10 CAA Multiplier Sweep

Table 14 reports the CAA multiplier sweep used to select the matched-safety static-steering baseline. The fixed500 sweep selects layer16 mult=6.5 because it is the closest point above the IRIS fixed500 ASR target. At full scale, mult=6.5 reaches PKU ASR 9.82%, which is close to IRIS, but it loses substantial GSM8K accuracy and produces many empty responses. Thus, CAA is a useful matched-safety baseline, but its apparent safety comes with a stronger utility and generation-quality cost.

Setting	PKU ASR ↓	Empty
mult=5.0 full	23.83	1242
mult=5.5 full	18.53	1426
mult=6.5 fixed500	10.00	—
mult=6.5 full	9.82	1792
mult=7.0 fixed500	5.80	—
mult=7.5 fixed500	4.40	—
mult=8.0 fixed500	3.20	—
mult=8.5 fixed500	1.20	—
mult=9.0 fixed500	0.60	—
mult=10.0 full	—	2532

Table 14: CAA layer16 multiplier sweep. Fixed500 rows are used for multiplier selection; full rows provide scale-up and generation-quality diagnostics.

A.11 Additional Routing Diagnostics

Table 15 reports expanded diagnostics behind the compact routing ablation table in the main text. The no-loop route is simply the undefended base path and therefore keeps zero loop cost but high ASR. Always-loop refinement gives the lowest ASR, but it turns the method into a blanket intervention with much higher degeneration and loop cost. Selective IRIS is the operating point used in the main results because it keeps most of the safety gain without forcing every token through the loop.

Routing	ASR	Full Deg.	Avg Loop
Base / no loop	70.59	—	0.00
Selective IRIS	8.60	4.75	178.88
Always loop	0.37	18.02	1255.41

Table 15: Expanded routing diagnostics on the full PKU split. Full Deg. reports full-window degeneration.

A.12 Layer Choice Diagnostic

Table 16 reports the same-pipeline fixed500 diagnostic used to select the Llama-3.1-8B intervention layer. Layer 14 is slightly more aggressive than layer 16, but it requires more loop computation and has higher degeneration. Layer 19 is worse on both ASR and cost. We therefore use layer 16 in the main experiments because it gives the most stable balance of safety, degeneration, and loop cost.

Layer	PKU ↓	Deg. ↓	Loop ↓
14	7.8	8.0	258.8
16	8.2	4.4	185.9
19	21.4	13.4	326.5

Table 16: Same-pipeline layer diagnostic on fixed500. Deg. is full-window degeneration. Historical runs with different training objectives are not mixed into this table.

A.13 Response Diversity and Degeneration Diagnostics

Table 17 reports template and empty-response diagnostics complementary to the Diversity column in the main table. These metrics help distinguish safety from template collapse. For example, Circuit Breaker and Llama Guard achieve low ASR partly through refusal or safe-sink templates, while IRIS has very low template rates. CAA produces many empty responses at the matched-safety multiplier.

Method	Refusal	Safe Sink	Empty
Base	0.92	0.00	24
Instruct	12.27	0.69	0
CAA	0.00	0.04	1792
Circuit Breaker	93.09	0.94	0
LG-Out	75.19	74.42	8
LG-In+Out	88.46	88.28	8
IRIS	1.63	0.31	0

Table 17: Template and empty-response diagnostics on full PKU harmful responses. Refusal and Safe Sink are template rates in percent; Empty is the number of empty responses.

A.14 Llama-3.2-1B Transfer Limitation

Llama-3.2-1B is excluded from the main transfer table because the current runtime makes capability tasks unusable. Table 18 summarizes the available diagnostic results. The low OR-Bench refusal shows that the model is not simply refusing everything, but the GSM8K no-answer count and loop statistics indicate that the runtime overwhelms the small model’s useful computation.

Metric	Result
OR-Bench refusal	2.12
GSM8K accuracy	1.14
GSM8K no-answer count	377 / 1319
GSM8K avg loop	367.36
GSM8K unsafe-token ratio	86.63

Table 18: Llama-3.2-1B transfer diagnostic under the evaluated K4 runtime. Percent-valued rows omit the percent sign for compactness.

A.15 Instruct-IRIS Appendix

The Instruct-IRIS appendix reports a diagnostic run that applies the current recovery pipeline on top of a refusal-style Instruct model. We trained fresh layer-16 Instruct probe, fluent, and prefix-recovery components and evaluated the resulting checkpoint on full PKU and full OR-Bench. Table 19 reports

the completed diagnostic. The checkpoint obtains very low PKU ASR, but its OR-Bench behavior is not clean: explicit refusal is moderate, while broad non-helpfulness and degeneration are high. This distinction matters because a low explicit refusal rate can hide answers that are unhelpful for reasons other than a refusal template, especially when late-token degeneration is frequent. For that reason, it is kept as an appendix diagnostic rather than a main result.

Metric	Result
PKU ASR	1.16
PKU window full deg.	2.30
PKU avg loop	15.02
OR-Bench explicit refusal	12.89
OR-Bench heuristic refusal	14.33
OR-Bench broad non-helpful	47.38
OR-Bench normal helpful	52.62
OR-Bench window full deg.	24.03
OR-Bench avg loop	156.35

Table 19: Instruct-IRIS full diagnostic. The run uses newly trained layer-16 Instruct probe/fluent/prefix-recovery components, not the old proxy-first Instruct checkpoint. Percent-valued rows omit the percent sign.

A.16 Implementation Notes

The implementation uses a base/fluent route for low-risk states and an IRIS route for high-risk states. The conceptual method presents low-risk states as bypassing safety refinement; the implementation realizes this with the base/fluent route while preserving the same routing-conditioned computation pattern. The main runtime uses mean-only routing score, $K = 5$, $\tau_{\text{on}} = 0.90$, $\tau_{\text{off}} = 0.80$, and $\text{min_steps} = 2$. Additional engineering details include the fluent surrogate, probe architecture, routing hysteresis, score pooling, prompt templates, and judge prompts.