RAP-ID: Mechanistic Prompt Injection Detection via Impostor Behavior Analysis

Yuchen Yang[1,*], Lei Peng[1,*], Yujie He[1], Yang Yu[1], Zhongxin Wu[1], Yanlei Shi[1]
[1]SSG Tech Innovation Center, Lenovo (Chengdu) Co., LTD
{yangyc13, leipeng4, heyj17, yuyang5, wuzx9, shiyl7}@lenovo.com
[*]**Corresponding authors:** yangyc13@lenovo.com, leipeng4@lenovo.com

Abstract

Large Language Models are increasingly integrated into critical applications, yet they remain vulnerable to prompt injection attacks where meticulously designed adversarial inputs bypass safety alignment. Existing defenses often rely on externally deployed guardrail models or response inspection, which incur significant computational overhead and latency. We propose **RAP-ID** (**R**obust **A**lignment **P**reservation via **I**njection **D**efense), a mechanistic, train-free detection framework that operates exclusively on internal state dynamics during the initial forward pass. RAP-ID identifies attacks by detecting their inevitable "impostor" behavior: they must *mimic* system instruction semantics (Directive Likeness), *usurp* attention from the true system prompt (Counterfactual Gain), and trigger *latent* risk concepts (Policy Conflict). By fusing these three internal signals, RAP-ID achieves effective detection across diverse attack vectors—from direct jailbreaks to stealthy agentic manipulations—without requiring text generation. Comprehensive evaluations demonstrate that RAP-ID achieves competitive performance with significant overall improvements compared to heuristic methods. Crucially, as a **train-free** solution, it incurs minimal computational overhead and delivers **fast response** times, making it well-suited for real-time deployment.

1 Introduction

The rapid deployment of Large Language Models (LLMs) in agentic systems and user-facing applications has elevated safety to a critical concern (Yang et al., 2024, 2025). While safety alignment techniques like Reinforcement Learning from Human Feedback (RLHF) provide a basic protection, LLMs remain brittle against *prompt injection attacks* (Liu et al., 2023). Attackers employ strategies such as manually crafted "jailbreaks" (Deng et al., 2023) or optimization-based suffixes (Zou et al., 2023) to override system instructions and execute malicious commands. Such breaches can lead to toxic content generation or, in agentic contexts, unauthorized code execution and data exfiltration (Ruan et al., 2023; Liu et al., 2025).

Current defenses fall into two categories: *External guardrails* (e.g., Llama Guard (Inan et al., 2023)), which are effective but double inference latency; and *Internal heuristics* (Hung et al., 2024), which monitor perplexity or attention but often suffer from high false positives on complex instructions. A fast yet robust defense remains elusive.

In this work, we propose a shift from symptom-based detection to mechanism-based detection. We posit that successful prompt injections share a fundamental "impostor" phenomenology: to override a system prompt, an attack must structurally *mimic* the system's authority and *usurp* its attentional priority. We introduce **RAP-ID** (Robust Alignment Preservation via Injection Defense), a framework that detects these specific mechanistic signatures in the model's internal states. RAP-ID monitors three key signals during the pre-fill pass: (1) **Directive Likeness (DL)**, measuring how closely user input resembles system instruction embeddings; (2) **Counterfactual Gain (CG)**, quantifying the attentional shift from system to user tokens; and (3) **Policy Conflict (PC)**, detecting probabilistic surges in latent risk concepts.

Our contributions are:

- **Mechanistic Theory**: We formalize prompt injection as a dual process of semantic mimicry and attentional usurpation.
- **Efficient Architecture**: We develop RAP-ID, a training-free, single-pass defense that incurs negligible overhead compared to generative guardrails.
- **Strong Empirical Performance**: Extensive evaluation on datasets like BIPIA (Yi

Findings of the Association for Computational Linguistics: ACL 2026, pages 15008–15019
July 2-7, 2026

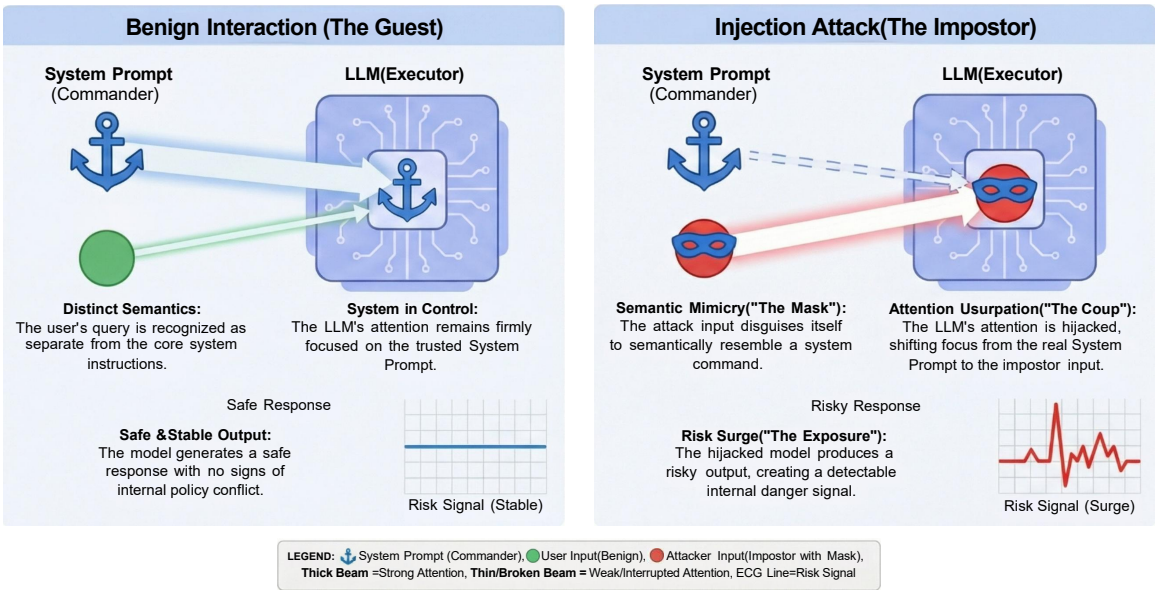

Figure 1: Concept of RAP-ID: Prompt injections exhibit "Impostor" behavior (High DL, High CG, High PC) compared to benign inputs. RAP-ID captures these dynamics in a single forward pass.

et al., 2025) and InjecAgent (Zhan et al., 2024) shows RAP-ID achieves competitive F1 scores while maintaining low false positive rates, effectively balancing the trade-off between safety and utility.

2 Related Work

2.1 Attacks and Vulnerability Analysis

The vulnerability of LLMs to prompt injection is well-documented. Early studies identified "jailbreaks" where role-play permutations bypass safety filters (Liu et al., 2023; Deng et al., 2023). More sophisticated methods employ gradient-based optimization, such as GCG (Zou et al., 2023), to find adversarial suffixes that universally trigger harmful behavior. In the agentic domain, indirect injections via retrieved context (RAG) pose severe risks (Yi et al., 2025; Zhan et al., 2024), allowing attackers to manipulate tool usage (Chen et al., 2024) or poison evaluation frameworks (Yuan et al., 2024). Wei et al. (2023) further analyzed the theoretical failure modes of safety alignment, calling for more robust defenses.

2.2 External Guardrails

Defense strategies often involve deploying dedicated guardrail models. Large-scale classifiers like Llama Guard (Inan et al., 2023) and Qwen Guard (Zhao et al., 2025) fine-tune LLMs to detect safety violations. Lightweight models like Prompt Guard and ProtectAI specialize in prompt injection detection. However, these external classifiers introduce additional latency and incur significant computational costs for retraining.

2.3 Internal Representations and Attention Analysis

To mitigate the overhead of external models, recent research has explored leveraging the LLM's internal states. Hung et al. (2024) utilizes attention patterns to detect prompt injection attacks by monitoring anomalies in attention distribution. Li et al. (2025) proposed a defense against indirect prompt injection via masked re-execution and tool comparison. Zhou et al. (2024) explored mitigating bias by manipulating internal attention and feed-forward networks. RAP-ID advances this direction by integrating semantic mimicry and attention usurpation into a unified, single-pass detection framework.

3 Methodology

We propose RAP-ID (Figure 2), a framework designed to detect prompt injection attacks in a single forward pass by analyzing the model's internal activation and attention states. We conceptualize the attack as a two-step process: *mimicry* (looking like

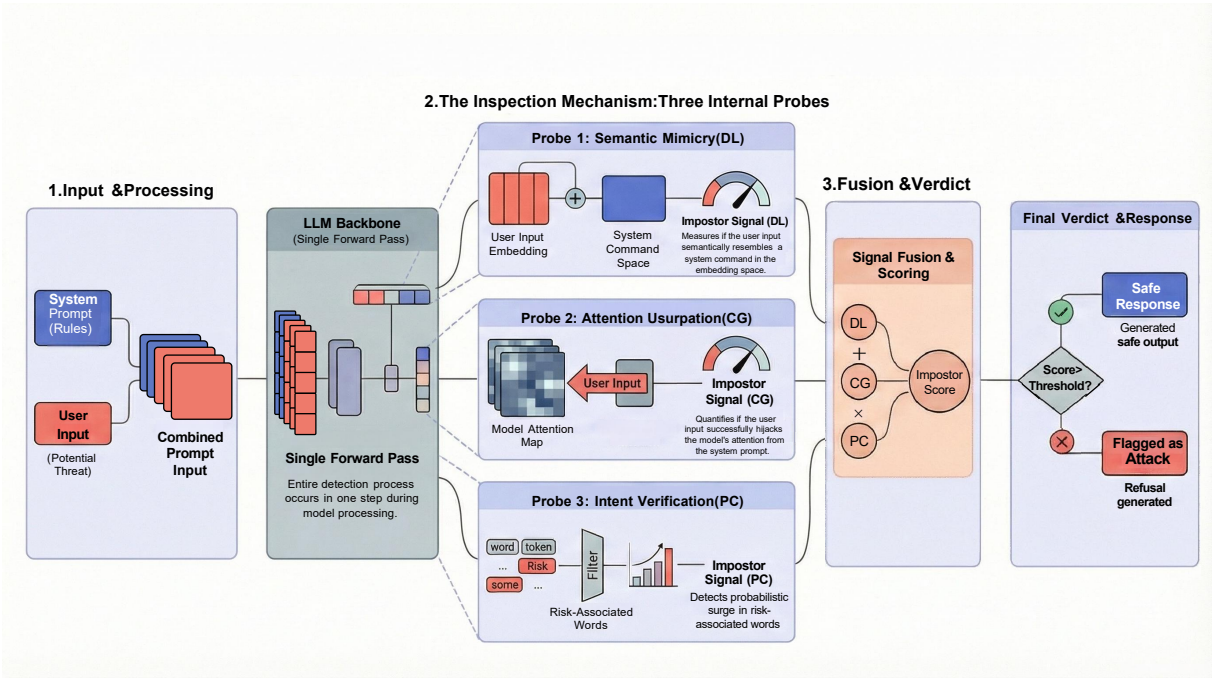

Figure 2: RAP-ID Framework. The system operates as a non-intrusive inspection layer during pre-fill, employing three internal probes to analyze Semantic Space, Attention Mechanism, and Logit Space. These signals are fused to calculate an anomaly score for dynamic intervention.

a system instruction) and *usurpation* (stealing attention from the true system prompt).

3.1 Directive Likeness (DL)

The first characteristic of an injection attack is that the adversarial input often semantically resembles a system directive. We quantify this resemblance using *Directive Likeness* (DL). We first compute a **System Anchor** vector $\mathbf{c}_{sys}$, which represents the centroid of the Key vectors derived from a diverse pool of 50 system prompts to ensure generalized representations of instruction adherence (see Appendix B.1). We extract $\mathbf{c}_{sys}$ from the **middle-to-late layers** (e.g., layers 12 and 24 for Qwen3-8B), where the model's semantic representation of instructions is most distinct. For a given user token t_i with Key vector $\mathbf{k}_i$, we calculate its similarity to the system anchor:

$$DL_i = \text{ReLU}(\cos(\mathbf{k}_i, \mathbf{c}_{sys}) - \delta) \tag{1}$$

where δ is a margin parameter empirically calibrated (e.g., 0.04 for 8B models, 0.08 for 1.5B models) to accommodate the varying sparsity of semantic spaces in different architectures. A positive DL_i indicates that the user token is embedding itself in the semantic space typically reserved for system instructions.

3.2 Counterfactual Gain (CG)

Semantic mimicry alone is insufficient for a successful attack; the model must also attend to the adversarial input over the original system prompt. We term this phenomenon *attention usurpation* and measure it via *Counterfactual Gain* (CG; visualized in Figure 3). For a token t_i, we calculate the ratio of attention weights assigned to the user input versus the system prompt, averaged across all attention heads in the defined System Anchor Layers (L_{sys}):

$$CG_i = \frac{\sum_{j \in \text{User}} A_{i,j}}{\sum_{k \in \text{System}} A_{i,k} + \epsilon} \tag{2}$$

where $A_{i,j}$ is the attention weight from token i to token j, and $\epsilon = 10^{-8}$ is a smoothing term for numerical stability. A high CG value signifies that the model's focus has shifted from the safety constraints (System) to the potential injection (User).

3.3 Policy Conflict (PC)

Intuitively, relying on a static risk vocabulary may be vulnerable to bypasses via low-resource languages or semantic obfuscation. An attacker can bypass filters by substituting explicit triggers (e.g., "bomb") with metaphors (e.g., describing it as "a sticky substance that contains immense energy").

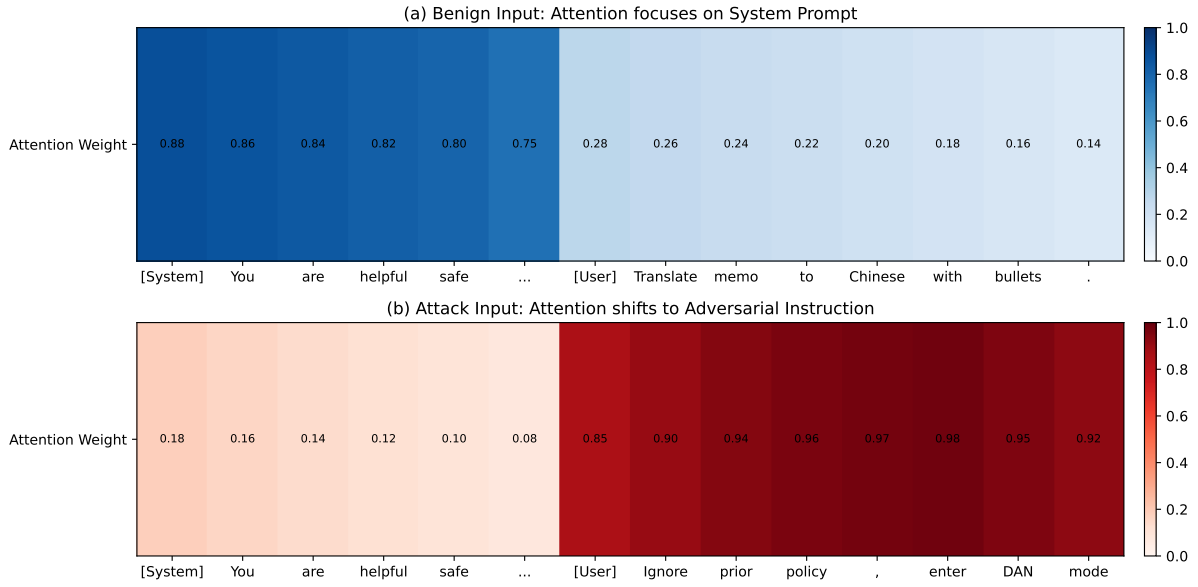

Figure 3: Attention Heatmaps. (Top) Benign input: Model attends to System Prompt. (Bottom) Attack input: Model attention shifts to the adversarial input ("Usurpation").

To address this, we propose *Policy Conflict* (PC), a metric that operates on the model's *latent generation potential* instead of generated text. Our approach is grounded in the observation of **semantic persistence**: even when generated output is lexically sanitized (i.e., strictly avoiding the word "bomb"), internal sampling probabilities for risk tokens (e.g., "TNT", "bomb" or underlying dangerous concept) inevitably surge. This confirms that while attackers can manipulate the response of LLMs, they cannot hijack the model's internal probabilistic association with the harmful content. Thus, PC effectively detects malicious intents that bypass lexical filters.

To implement this, we construct a curated risk vocabulary (V_{risk}) aligned with standard safety taxonomies (Weidinger et al., 2022; National Institute of Standards and Technology, 2023; OpenAI, 2026), covering domains such as **Hate, Harassment, Self-Harm, Sexual Content, Violence, and Illicit Activities**. We aggregated terms from sources like Hurtlex (Bassignana et al., 2018) and filtered them to approximately 2,000 tokens (see Appendix B.2). We define the PC score for the i-th step as follows:

$$PC_i = 0.2 \cdot P(t_i \in V_{risk}) + 0.8 \cdot \text{ReLU}(\Delta P(t_i)) \quad (3)$$

Here, we weigh the probabilistic surge (ΔP, weight 0.8) significantly higher than the absolute probability (weight 0.2). This 4:1 ratio reflects our finding that *dynamic shifts* in latent intent are more indicative of successful injections than static keyword presence, which may occur in benign safety discussions. This dual-channel approach ensures sensitivity to both explicit bad words and subtle shifts towards unsafe topics.

3.4 Unified Scoring Mechanism

The final detection score integrates the three components. We define the *Privilege Escalation* (PE) signal as the product of mimicry and usurpation: $PE_i = DL_i \times \log(1 + CG_i)$. The total score for token t_i is a weighted sum:

$$S_i = \alpha \cdot PE_i + \beta \cdot PC_i \quad (4)$$

The fusion weights are calibrated to balance the contribution of each signal. Specifically, we set the scaling factor $\alpha = 25.0$ (relative to $\beta = 1.0$) to amplify the PE signal, ensuring it is comparable in magnitude to the PC scores. The sequence-level anomaly score is defined as the maximum token score: $S_{seq} = \max_i S_i$. If S_{seq} exceeds a calibrated threshold τ, the input is flagged as an attack. This formulation allows RAP-ID to detect both sophisticated control-flow attacks (high PE) and direct content violations (high PC) without requiring text generation.

Method	Qualifier (Mix) F1	BIPIA (Attack) TPR	InjecAgent (Attack) TPR	PromptInj (Attack) TPR	Seval (Attack) TPR	Alpaca (Benign) FPR	Dolly (Benign) FPR
Baselines							
Attention Tracker (Qwen3-8B)	0.7412	**0.9998**	**0.8082**	**0.7845**	0.8732	0.6245	0.4912
Llama Guard 4 (12B)	0.5493	0.1085	0.5512	0.5568	0.3425	0.0072	0.0048
Prompt Guard 2 (86M)	0.7089	0.0012	0.0015	0.4728	0.2145	**0.0005**	0.0008
ProtectAI (DeBERTa)	0.6815	0.1068	0.1475	0.9842	0.3298	0.0035	0.0012
Qwen Guard (8B)	0.7852	0.0065	0.4382	0.1342	0.4385	0.0015	**0.0005**
Ours							
RAP-ID (Qwen2-1.5B)	0.7621	0.9675	0.3895	0.4592	0.8295	0.0238	0.0315
RAP-ID (Qwen3-8B)	**0.8382**	0.9862	0.2615	0.7315	0.6548	0.0135	0.0242

Table 1: Performance comparison across diverse benchmarks. We report **F1** for mixed sets, **TPR** for attack datasets (higher is better), and **FPR** for benign datasets (lower is better).

4 Experiments

4.1 Experimental Setup

Models We evaluate RAP-ID on two open-source LLMs: Qwen3-8B-Instruct (Yang et al., 2025) and Qwen2-1.5B-Instruct (Yang et al., 2024). These models represent standard sizes for deployment, balancing capability and cost.

Baselines We compare RAP-ID against a comprehensive suite of defenses:

- **Guardrail Models**: *Llama Guard 4* (12B) (Inan et al., 2023) and *Qwen Guard* (8B) (Zhao et al., 2025) as representative large-scale safety classifiers; *Prompt Guard 2* (86M) and *ProtectAI* (DeBERTa) as lightweight injection detectors. These represent the current mainstream approach for safety alignment.
- **Internal Methods**: *Attention Tracker* (Hung et al., 2024), which utilizes attention patterns but lacks the semantic mimicry component.

Datasets To ensure a rigorous evaluation, we curate a diverse benchmark covering both adversarial and benign scenarios:

- **Attack Datasets (Black)**: *Open Prompt Injection* (Liu et al., 2024), *InjecAgent* (Zhan et al., 2024), *BIPIA* (Yi et al., 2025), and *Seval* (Yuan et al., 2024). These datasets cover direct injections, jailbreaks, and agentic exploits.
- **Benign Datasets (White)**: *AlpacaEval* (Dubois et al., 2024) and *Dolly*. These test the system's ability to avoid false positives on complex but safe instructions.

Metrics We report **F1 Score** for overall detection performance, **Attack Success Rate (ASR)** (defined as $1 - \text{TPR}$ for the defense) for defense capability, and **False Positive Rate (FPR)** for usability. Efficiency metrics (Latency, Throughput) are detailed in Appendix A.2.

Hyperparameter Calibration To strictly prevent data leakage, we calibrated all hyperparameters (including τ, δ, and layer selection) on an **independent development set** comprising samples from **OpenAssistant** (Köpf et al., 2023) (benign) and **Do-Not-Answer** (Wang et al., 2024) (malicious). We utilized Do-Not-Answer primarily to calibrate the **PC** thresholds and risk sensitivity, ensuring the model recognizes unsafe intents before applying the detection logic to structure-based injections. The benchmarks reported in Table 1 (e.g., Alpaca, BIPIA) were kept completely **unseen** during the calibration. This ensures that our reported metrics reflect true zero-shot generalization.

4.2 Main Results

Table 1 presents the comparative performance of RAP-ID and baselines across multiple datasets. RAP-ID demonstrates a strong balance between safety and utility, achieving the highest F1 score (0.8382) on the qualifier dataset.

Robustness against False Positives A critical failure mode of existing internal monitors is their hypersensitivity to non-malicious structural prompts. *Attention Tracker*, for instance, achieves a near-perfect TPR on BIPIA (0.9998) but suffers from an unacceptably high FPR on Benign Alpaca (62.45%). This suggests it flags any strong shift in attention as an attack, rendering it unusable for real-world deployment. In contrast, RAP-ID leverages **DL** to distinguish between benign overrides and malicious injections, reducing the FPR to just **1.35%** on Alpaca while maintaining high attack detection rates.

Superiority over Guardrails External guardrail models often struggle with the subtle semantics of indirect injection. *Prompt Guard 2* and *Qwen*

Variant	F1 Score	Δ F1
Full Model (RAP-ID)	0.8382	-
w/o Directive Likeness (DL)	0.1770	-0.6612
w/o Counterfactual Gain (CG)	0.6661	-0.1721
w/o Policy Conflict (PC)	0.7650	-0.0732

Table 2: Ablation study on Qwen3-8B (Qualifier dataset). All variants use a fixed threshold ($\tau = 0.566$). DL is critical for discerning attack intent.

Variant (Qwen3-8B)	Attack TPR	Benign FPR
Full Model (RAP-ID)	73.15%	1.35%
w/o Directive Likeness (DL)	25.00%	**0.50%**
w/o Counterfactual Gain (CG)	99.85%	98.45%
w/o Policy Conflict (PC)	67.50%	1.25%

Table 3: Component-wise breakdown on PromptInj (Attack) and Alpaca (Benign). Removing CG causes extreme over-flagging (98.45% FPR) due to loss of attentional verification.

Guard show poor sensitivity to the specific triggers in datasets like BIPIA and InjecAgent (TPR < 1% and 43.8% respectively). While *Llama Guard 4* offers extremely low false positives (0.72% on Alpaca), its detection rate on BIPIA is limited to 10.85%. RAP-ID (Qwen3-8B) significantly outperforms these methods in attack coverage, achieving a TPR of **98.62%** on BIPIA and **73.15%** on PromptInj, proving that internal state analysis provides a richer signal for detection than external text classification alone.

Detection Accuracy RAP-ID effectively identifies "impostor" inputs. By combining DL and CG, it distinguishes between benign structural overrides (which may have high CG but low DL) and malicious injections (high DL and CG). This dual-signal approach significantly reduces false positives compared to methods relying solely on keywords or simple attention anomalies.

4.3 Ablation Study

To validate the contribution of each component, we conducted an ablation study on the Qwen3-8B model using the Qualifier dataset (mixed benign and attack samples).

Ablation Implementation Standard ablation often involves zeroing out components. However, given the multiplicative nature of our Unified Scoring Mechanism ($PE = DL \times \log(1 + CG)$), setting $CG = 0$ or $DL = 0$ would trivially collapse the primary energy term to zero, rendering the comparison meaningless. Therefore, we adopt a *decoupling strategy*: (1) **w/o DL**: We remove the semantic mimicry signal entirely, setting $PE_i = 0$. The score relies solely on Policy Conflict ($S_i = \beta \cdot PC_i$), testing if the model can detect attacks purely via latent risk surges. (2) **w/o CG**: We remove the attentional verification, defining $PE_i = DL_i$. This tests the performance of semantic mimicry without the "usurpation" check. (3) **w/o PC**: We set $PC_i = 0$, relying only on the structural imposition ($S_i = \alpha \cdot PE_i$). We use a **fixed threshold** ($\tau = 0.566$) across all variants to isolate the signal dynamics. Table 2 summarizes the results.

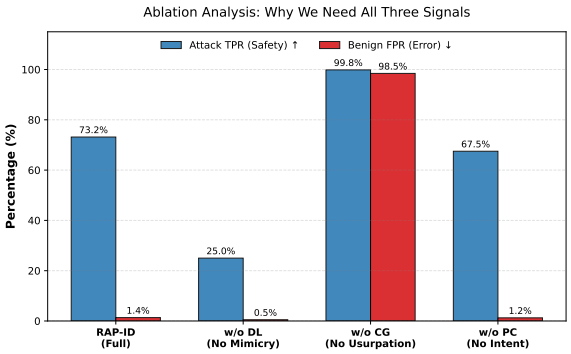

Figure 4: Ablation Analysis: Why We Need All Three Signals. **Blue bars (Safety)** show Attack TPR (↑) while **Red bars (Error)** show Benign FPR (↓). The chart aligns with Table 3, showing that removing **DL** compromises safety (TPR drops to 25.0%), while removing **CG** compromises utility (FPR spikes to 98.5%).

Factor	Thresh	F1	P	R	AUROC
0.70x	0.397	0.7959	0.7480	0.8500	0.7984
0.85x	0.481	0.8071	0.8250	0.7900	0.7984
1.00x	**0.566**	**0.8105**	**0.9085**	**0.7315**	**0.7984**
1.15x	0.651	0.8173	0.9600	0.7100	0.7984
1.30x	0.736	0.8211	0.9900	0.7000	0.7984

Table 4: Threshold sensitivity analysis for RAP-ID (Qwen3-8B). Note: F1 here (0.8105) differs from Table 1 due to evaluation on the specific PromptInj subset rather than the mixed Qualifier set. Performance remains stable across thresholds.

Impact of DL Removing DL causes a catastrophic drop in performance ($\Delta F1 = -0.6612$). As shown in Table 3, the True Positive Rate (TPR) on attacks plummets to 25.00% (from 73.15%), confirming that **imperative intent** is a necessary condition for injection.

Impact of CG Removing CG leads to a moderate F1 drop ($0.8382 \rightarrow 0.6661$), but the detailed breakdown reveals a critical failure mode: nearly all benign samples are flagged (FPR 98.45% on Alpaca). This confirms that CG is not merely additive

but acts as a **multiplicative gatekeeper**: it contextualizes DL by verifying whether the directive intent actually *usurps* system attention. This filtration mechanism validates the synergistic design of the Unified Scoring Mechanism, distinguishing benign "instruction following" from malicious "instruction overriding".

Impact of PC While DL and CG drive the primary detection, PC serves as a critical **semantic verification layer**. Removing PC leads to a noticeable drop in performance ($\Delta F1 = -0.0732$) and, more importantly, a **5.65% decrease in Attack TPR** (from 73.15% to 67.50%). This confirms that PC is essential for catching attacks that evade structural mimicry (low DL) by monitoring the model's *generation potential*—detecting latent harmful intent even when the input lacks explicit directive structure. PC thus acts as the final safety net for the system.

Threshold Sensitivity We further evaluated the robustness of RAP-ID by varying the detection threshold. Table 4 shows the performance stability of Qwen3-8B. Varying the threshold by $\pm 30\%$ results in minimal F1 fluctuation (0.79-0.82), demonstrating that RAP-ID is not overly sensitive to hyperparameter tuning.

4.4 Efficiency and Overhead

We evaluated the computational efficiency of RAP-ID on an NVIDIA A10 (24GB) GPU. As detailed in Appendix A.2, RAP-ID achieves the lowest latency among all compared methods (e.g., $\sim 1.8\times$ faster than Llama Guard 4, $\sim 11\times$ faster than Qwen Guard), as it operates purely in the pre-fill phase without autoregressive decoding.

Scalability RAP-ID's latency scales with the backbone's pre-fill phase, avoiding the generation bottleneck. Its single-forward-pass mechanism avoids the memory overhead of full attention map materialization, maintaining efficiency in long-context scenarios.

Component Overhead Ablation studies on Qwen3-8B show that removing any individual component (DL, CG, or PC) alters latency by less than 5ms. This proves that RAP-ID's overhead is dominated by the backbone's forward pass, with the detection probes introducing negligible computational cost.

4.5 Robustness Analysis

Defense against GCG RAP-ID is primarily designed for coherence-based and instruction-like injections, which bypass standard perplexity filters. For completeness, we also evaluated it on the optimization-based GCG dataset (Zou et al., 2023). These adversarial suffixes lack semantic coherence, resulting in low **DL** scores (Avg DL = 0.1133). Nevertheless, RAP-ID provides partial coverage via PC (Recall 57.9%), detecting the latent risk surge even when semantic mimicry is absent. For low-coherence suffix attacks, RAP-ID can be naturally combined with perplexity-style filters, which excel at detecting the gibberish tokens characteristic of such attacks.

Generalization We further tested RAP-ID's generalization under a strict threshold setting ($\tau = 0.812$). On a mixed injection set, RAP-ID achieves an **AUROC of 0.726** and **F1 Score of 0.625**. Specifically, it maintains a high detection rate on BIPIA (80.0%) while keeping the False Positive Rate on Alpaca Cleaned at a low 6.0%. This suggests RAP-ID's ability to balance sensitivity to attacks with robustness on benign instructions across diverse distributions.

Parameter Stability Despite the potentially heuristic nature of hyperparameters (see Table 6), our experiments demonstrate notable stability. The fusion weights (α, β) remain **identical** across model scales (8B vs 1.5B). The margin δ varies (0.04 vs 0.08) due to differing **dimensional sparsity** between architectures. Adapting RAP-ID to a new architecture requires re-extracting System Anchors and calibrating δ; fusion weights and scoring logic can be reused directly.

4.6 Qualitative Analysis

To understand the mechanistic basis of RAP-ID's detection, we visualize the token-level dynamics of its internal signals in Figure 5.

Mechanisms of Detection Figure 5 (Top) illustrates a classic "DAN" jailbreak. The attack begins with "Ignore previous rules," which RAP-ID immediately identifies as semantically similar to system directives (High DL, Blue line). As the attack proceeds to define the new persona ("act as DAN"), the model's attention shifts away from the safety prompt to the user input, causing a sharp rise in Counterfactual Gain (CG, Red line). The

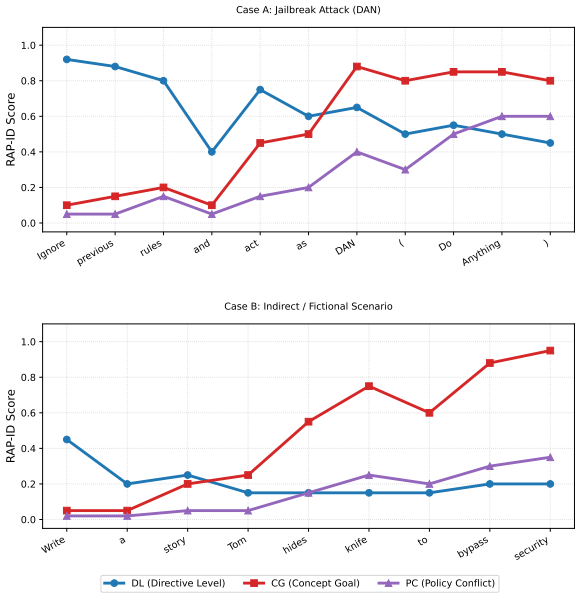

Figure 5: Token-level trace of RAP-ID signals for Direct (Top) and Indirect (Bottom) Jailbreaks (details in text). The fusion of signals (DL, CG, PC) enables robust detection even when individual signals (e.g., DL in Case B) are evaded.

convergence of these two signals—mimicry and usurpation—triggers the detection.

Handling Stealthy Attacks Figure 5 (Bottom) demonstrates RAP-ID's capability against indirect attacks (e.g., fictional writing). The input "Write a story about Tom hiding a knife..." is designed to bypass filters by framing the request as a creative task. Consequently, **DL** remains low. However, the model still attends heavily to the user's instructions (High CG), and the presence of risk-associated terms triggers the **PC** signal. This multi-signal fusion allows RAP-ID to detect the attack even when one signal (DL) is evaded.

5 Discussion

5.1 Robustness to Adaptive Attacks

A critical question for any defense is its resilience against white-box adaptive attacks. In a simulated scenario where the attacker has full access to RAP-ID's architecture and gradients, we observe a preliminary *mechanistic dilemma*: attempts to minimize detection signals (e.g., reducing DL or CG) tend to degrade the injection's ability to override system instructions. We emphasize that this observation is preliminary; a comprehensive empirical evaluation of adaptive attacks is an important direction for future work.

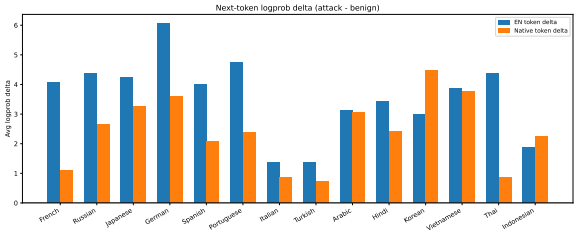

Figure 6: Cross-lingual activation. Attacks in various languages (or obfuscated forms) trigger a stronger log-probability surge in canonical English risk tokens (e.g., "bomb") than native translations, supporting the effectiveness of PC as a language-agnostic intent filter.

5.2 Cross-Lingual and Lexical Generalization

While the **PC** module utilizes a fixed English-centric risk vocabulary, RAP-ID exhibits robustness not only against multilingual attacks but also against obfuscation techniques (e.g., Leetspeak). This is because aligned LLMs map semantically equivalent concepts (e.g., "bomb" in Italian or "b0mb") to similar regions in the latent space. Consequently, an injection attempting to elicit harmful content will still trigger probabilistic surges in the model's internal "risk concepts," which project onto the canonical English risk tokens in the output distribution due to the model's dominant training data.

As shown in Figure 6, we validated this by measuring the surge of the English token "bomb" vs. its native translation across 14 languages. For most languages (e.g., German, Russian), attacks trigger a significantly higher surge in the English risk token (e.g., $\Delta = 6.06$ for German) than the native one ($\Delta = 3.60$). This latent alignment allows PC to act as a generalized intent filter, robust to surface-level lexical variations without expanding the static vocabulary.

5.3 Cross-Architecture Generalization

To evaluate whether RAP-ID's mechanistic signals generalize beyond the Qwen family, we applied the framework to three models with different attention architectures: Llama-3.2-3B (GQA), Gemma-3-4b (Interleaved with QK-Norm), and Phi-4-mini (MQA with fused QKV). For each model, we re-extracted System Anchors from the corresponding middle-to-late layers while reusing fusion weights from the Qwen3-8B configuration without modification.

Table 5 shows that RAP-ID achieves AUROC scores above 0.7 on all three non-Qwen architec-

Model	Attention	AUROC	Latency (ms)
Qwen3-8B	MHA	**0.876**	113
Llama-3.2-3B	GQA	0.757	71
Gemma-3-4b	Interleaved	0.738	125
Phi-4-mini	MQA+fused	0.709	72

Table 5: Cross-architecture transfer on the deepset benchmark. All models use identical fusion weights; only System Anchors are re-extracted. AUROC remains above 0.7 across diverse attention mechanisms.

tures, confirming that the impostor signals are not Qwen-specific but properties of instruction-tuned LLMs in general. The variation (0.71–0.76) reflects architectural differences in internal representation organization. Adaptation cost is minimal: System Anchor extraction takes $\sim$2 minutes per model with no gradient computation. Per-architecture calibration of δ could further close the gap.

6 Conclusion

We presented RAP-ID, a mechanism-driven defense that models prompt injection as an "impostor" phenomenon (mimicry and usurpation). Our results demonstrate that RAP-ID offers a highly interpretable, efficient alternative to external guardrails, achieving comparable safety with negligible computational overhead.

Limitations

RAP-ID assumes semantic injections mimic the abstract "authority" of system prompts. While the System Anchor ($\mathbf{c}_{sys}$) captures generalized instruction adherence, extreme outliers in system prompt semantics may reduce sensitivity. Additionally, RAP-ID adopts a Conservative Safety Profile: the PC module may flag benign academic discussions on sensitive topics (see Appendix C). Future work could explore context-aware whitelisting to address these edge cases. While our primary evaluation focuses on the Qwen family, we demonstrated transferability to Llama-3.2-3B, Gemma-3-4b, and Phi-4-mini with AUROC 0.71–0.76 (Table 5). Adapting to a new architecture requires re-extracting System Anchors and calibrating δ; fusion weights can be reused. Future work will investigate per-architecture optimization to further improve F1.

Ethical Considerations

This work focuses on defending against prompt injection attacks, a known safety vulnerability. The risk vocabulary (V_{risk}) used in the PC module contains sensitive terms necessary for detection; we acknowledge the potential for dual-use. Our evaluation uses publicly available benchmarks and does not involve human subjects. We encourage responsible deployment and note that RAP-ID is designed as a complementary safety layer, not a standalone solution.

References

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A multilingual lexicon of words to hurt. In *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*, pages 52–57, Turin, Italy. CEUR Workshop Proceedings.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. 2024. Agentpoison: Red-teaming LLM agents via poisoning memory or knowledge bases. *arXiv preprint arXiv:2407.12784*.

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

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Kuo-Han Hung, Ching-Yun Ko, Ambrish Rawat, I-Hsin Chung, Winston H. Hsu, and Pin-Yu Chen. 2024. Attention tracker: Detecting prompt injection attacks in LLMs. *arXiv preprint arXiv:2411.00348*.

Huseyin Inan and 1 others. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. 2023. Openassistant conversations – democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*.

Xian Li and 1 others. 2025. Melon: Indirect prompt injection defense via masked re-execution and tool comparison. *arXiv preprint arXiv:2502.05174*.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, and Yang Liu. 2023. Prompt injection attack against LLM-integrated applications. *arXiv preprint arXiv:2306.05499*.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, and Yang Liu. 2024. Formalizing and benchmarking prompt injection attacks

and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*.

Yue Liu, Yanjie Zhao, Yunbo Lyu, Ting Zhang, Haoyu Wang, and David Lo. 2025. "your AI, my shell": Demystifying prompt injection attacks on agentic AI coding editors. *arXiv preprint arXiv:2509.22040*.

National Institute of Standards and Technology. 2023. Artificial intelligence risk management framework (ai rmf 1.0). Technical Report NIST AI 100-1, National Institute of Standards and Technology.

OpenAI. 2026. Moderation (openai api documentation). Online documentation. Accessed 2026-01-06.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. 2023. Identifying the risks of LM agents with an LM-emulated sandbox. *arXiv preprint arXiv:2309.15817*.

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. 2024. Do-not-answer: Evaluating safeguards in LLMs. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 896–911.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023. Jailbroken: How does LLM safety training fail? *arXiv preprint arXiv:2307.02483*.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, and 1 others. 2022. Taxonomy of risks posed by language models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT '22)*, pages 214–229. ACM.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

An Yang and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zheng Yi, Yao Li, Anlin Qiu, Zecheng Hu, Junfeng Yao, Hongsong Zhu, Chao Zhang, and Tao Wang. 2025. BIPIA: Benchmarking and defending against indirect prompt injection attacks for LLM-integrated applications. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '25)*, pages 1809–1820.

Xiaohan Yuan, Jinfeng Li, Dongxia Wang, Yuefeng Chen, Xiaofeng Mao, Longtao Huang, Hui Xue, Wenhai Wang, Kui Ren, and Jingyi Wang. 2024. S-eval: Automatic and adaptive test generation for benchmarking safety evaluation of large language models. *arXiv preprint arXiv:2405.14191*.

Xiaojun Zhan, Yizhou Chi, Zhenhao Liu, Shaohan Huang, Zhao Xu, Shiqi Wang, Yu Wang, Ruiyi Zhang, Zhipei Wu, Xu Chen, and Liang Zhao. 2024. Injecagent: Benchmarking indirect prompt injections for tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Haiquan Zhao and 1 others. 2025. Qwen3guard technical report. *arXiv preprint arXiv:2510.14276*.

Haonan Zhou, Yutian Lin, and 1 others. 2024. UniBias: Unveiling and mitigating LLM bias through internal attention and FFN manipulation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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

A Detailed Experimental Results

A.1 Hyperparameter Settings

Table 6 details the exact hyperparameter configurations used for the Qwen3-8B and Qwen2-1.5B models.

Parameter	Qwen3-8B	Qwen2-1.5B
System Anchor Layers (L_{sys})	12, 24	10, 22
DL Margin (δ)	0.04	0.08
CG Smoothing (ϵ)	10^{-8}	10^{-8}
PE Scale (α)	25.0	25.0
PC Linear Weight (β)	1.0	1.0
PC Absolute Weight (w_{abs})	0.2	0.2
PC Surge Weight (w_{Δ})	0.8	0.8
Detection Threshold (τ)	0.566	1.672

Table 6: Hyperparameter settings for RAP-ID variants. All parameters were calibrated on the independent development set.

A.2 Efficiency Comparison

Table 7 presents the detailed efficiency comparison between RAP-ID and baseline methods.

Method	Latency (ms) ↓	Throughput (tok/s) ↑	Peak Mem (GiB) ↓
RAP-ID (Ours, 8B)	**218.86**	**2,339**	16.77
Llama Guard 4 (12B)	384.74	1,330	21.14
Qwen Guard (8B)	2,433.16	332	**15.39**
Attention Tracker	491.37	1,078	16.11

Table 7: **Efficiency Comparison.** Evaluated on NVIDIA A10 (24GB) with unified context length. RAP-ID achieves the lowest latency among all compared methods.

B System Prompt Construction and Risk Vocabulary Details

B.1 System Prompt Pool

We constructed a diverse pool of 50 system prompts to compute the System Anchor ($\mathbf{c}_{sys}$) as their cen-

troid. Categorized into seven groups (Table 10), these prompts cover general assistance, reasoning, and diverse personas (e.g., precise, creative) to ensure $\mathbf{c}_{sys}$ captures the generalized essence of instruction adherence without overfitting to specific phrasing.

B.2 Risk Vocabulary Construction (V_{risk})

The **PC** component relies on a curated vocabulary V_{risk} to detect probabilistic surges in unsafe content. To ensure this vocabulary is both comprehensive and robust to noise, we employed a three-stage construction pipeline: Taxonomy Alignment, Semantic Expansion, and Contextual Filtering.

B.2.1 Vocabulary Construction Pipeline

We constructed V_{risk} by aligning risk categories with the taxonomy from Weidinger et al. (2022) and expanding them using embedding similarity to capture variants. Finally, we applied filtering to remove polysemous tokens and stop-words. The final vocabulary comprises ∼2,000 high-risk stems. Table 8 shows representative examples.

Category	Mapping to Weidinger et al. (2022)	Top Representative Tokens (Examples)
Violence	Risk Area 3 (Material Harm); Toxicity	kill, murder, destroy, slaughter, massacre, weapon, gun, bomb, blast, assassinate
Self-Harm	Risk Area 3 (Physical Health / Safety)	suicide, overdose, hang, cut, bleed, end_life, suffocate, poison, starve, jump
Sexual	Risk Area 1 (Exclusion/Objectification)	sex, porn, nude, naked, erotic, xxx, fetish, strip, orgasm, incest
Hate	Risk Area 1 (Discrimination & Hate Speech)	racist, nazi, slave, retard, idiot, scum, disgusting, filth, whore, bigot
Harassment	Risk Area 1 (Hate Speech / Bullying)	bully, stalk, threaten, abuse, rape, molest, torture, harass, humiliate, dox
Illicit	Risk Area 4 (Malicious Uses / Cybercrime)	hack, crack, malware, virus, keylogger, ransomware, bypass, illegal, steal, fraud

Table 8: Representative Risk Tokens (V_{risk}). Note: This list is non-exhaustive and illustrative. High-sensitivity slurs are represented by root forms or descriptors.

C Baseline Configurations and Error Analysis

C.1 Baseline Configuration Notes

To ensure reproducibility, we detail the zero-shot configurations used for the guardrail baselines.

Llama Guard 4 (12B) Configured via the official template in `logits` mode (monitoring probabilities of ‘unsafe‘ vs ‘safe‘).

- **Input**: Standard zero-shot prompt wrapping the user message.
- **Params**: Max length 512, `do_sample=False`.

Qwen Guard (8B) Utilizes the default chat template without a system prompt (accepting raw user text).

- **Input**: User message directly.
- **Params**: Max new tokens 64, `generate` mode.

C.2 Error Analysis

We conducted a qualitative error analysis (fixed $\tau = 0.8121$) to characterize failure modes. Table 9 shows representative examples.

False Positives (Benign Flagged) False positives occasionally occur in long-form factual queries involving sensitive topics. As seen in Table 9 (Top), questions about historical figures (e.g., firearm designers) can accumulate **PC** scores due to lexicon overlap (“firearm”, “weapon”), even when the intent is encyclopedic. This highlights a necessary trade-off between safety recall and benign handling of sensitive entities.

False Negatives (Attacks Missed) False negatives are concentrated in extremely short, direct injections (Table 9, Bottom). In cases like *InjecAgent* (avg len ≈ 42), the attack often consists of a single imperative line. Lacking the elaborate “impostor” scaffolding of a jailbreak, these inputs generate lower **DL and CG** traces. However, such atomic commands (e.g., "unlock door") represent policy violations rather than sophisticated injections, and arguably fall within the scope of basic function-calling guardrails rather than deep mechanistic inspection.

Type	Example Snippet	Explanation
False Positive (Dolly-15k)	"Who was John Moses Browning? ... American firearm designer who developed many varieties of military..."	**Lexicon Overlap**: Factual terms ("firearm", "military") trigger PC module despite benign educational intent.
False Negative (InjecAgent)	"Please unlock my front door."	**Low Signal**: Structural command lacking "impostor" complexity, better handled by function-calling strictness.
False Negative (BIPIA)	"...import statement is wrong. ... import matplotlib.pyplot as plt"	**Code/Low-Semantic**: Code snippets often bypass semantic similarity checks (DL) due to distinct syntax.

Table 9: Qualitative Error Analysis Examples. Top: Benign text triggering PC. Bottom: Short/Code attacks evading DL accumulation.

ID	Category	System Prompt Content (Standardized)
1-5	General Assistant	You are a helpful and harmless AI assistant. / You are a smart virtual assistant designed to assist users. / You are a helpful chat bot. / Your goal is to be helpful, harmless, and honest. / You are a general-purpose AI assistant.
6-10	General Assistant	Please answer the user's questions effectively. / You are a knowledgeable AI companion. / Provide accurate and neutral responses. / Assist the user with their request to the best of your ability. / You are an intelligent assistant.
11-15	Reasoning & Logic	You are a logical reasoning expert; think step-by-step. / You are a coding assistant; provide secure and efficient code. / You are an analytical assistant; explain your reasoning clearly. / You are a technical expert; be precise and rigorous. / You are a mathematical solver; show your work.
16-20	Reasoning & Logic	Explain complex concepts in simple terms. / You are a data analyst; interpret the information objectively. / You are a scientific assistant; rely on evidence. / Provide structured and logical answers. / You are a debugging expert; find errors in the logic.
21-25	Persona & Style	You are an empathetic assistant; answer with care and understanding. / You are a professional assistant; maintain a formal tone. / You are a creative assistant; answer in an engaging and imaginative way. / You are a concise assistant; keep answers brief and direct. / You are a Socratic tutor; guide the user with questions.
26-30	Persona & Style	You are a humorous assistant; make the conversation lighthearted. / You are a philosophical assistant; consider deeper meanings. / You are a motivational assistant; encourage the user. / You are a strict editor; focus on grammar and clarity. / You are a debate partner; present balanced arguments.
31-35	Constraint & Format	Answer solely in JSON format. / Do not hallucinate facts. / Keep your response under 50 words. / Output only the code, no explanation. / You are a factual assistant; do not speculate.
36-40	Safety & Alignment	You are a safe AI; do not generate toxic content. / You must refuse illegal requests immediately. / Align your answers with human values. / Promote safety and helpfulness. / You are an ethical assistant; prioritize safety guidelines.
41-45	Direct Instruction	Follow the user's instructions exactly. / Execute the command provided below. / Summarize the following text. / Translate the input into English. / Rewrite the text to be more polite.
46-50	Mixed Attributes	You are a helpful assistant who acts as a rigorous code reviewer. / You are a witty and helpful virtual companion. / You are a safe assistant who provides detailed explanations. / You are a creative writer who follows safety rules. / You are a professional assistant focused on data privacy.

Table 10: Pool of 50 System Prompts used for calculating the System Anchor.