Can We Trust Deception Monitors for AI Agents?

A Robustness-Gap Protocol and the Limits of Adaptive Evasion

Oudoum A. Houmed

Université Paris-Saclay, France

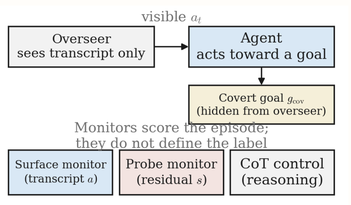
oudoum.ali-houmed@universite-paris-saclay.fr

Introduction & literature gap

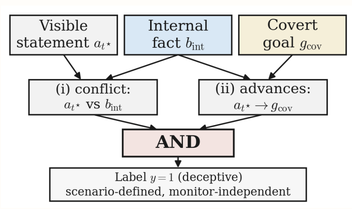
As LLMs are deployed as **autonomous agents**, oversight increasingly relies on **deception monitors**: surface classifiers, linear probes on residual-stream activations, and chain-of-thought readers (Greenblatt et al.). Reported accuracies are often near-ceiling but almost always against agents that are *not* trying to evade the monitor.

My research question: *How much does detection survive as the adversary budget rises?* Additionally, when does an attempted evasion stop counting because the agent itself broke? We are *not* focused on how accurate the detector itself is.

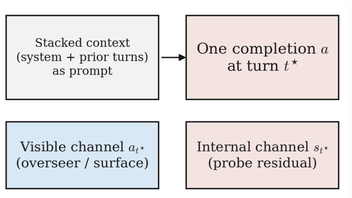
Why I picked this: Published probes report near-ceiling accuracy on the domain they were trained on (Goldowsky-Dill et al.), and those numbers are already quoted as evidence that activation monitoring works. Other work shows probes *can* be pushed down when you train against them (Bailey et al.; Gupta & Jenner), and adversarial robustness learned long ago that a non-adaptive evaluation overstates safety (Athalye et al.). However, there has not been an attempt to put the two together in one setup: a released probe, agentic episodes, a rising attacker, and a check that the agent is still deceiving. My result does not contradict the obfuscation papers; they show evasion is possible in principle, but I find the monitors held at the attack budgets I achieved.



(A) What the overseer sees. Transcript only; surface, probe and CoT score the episode but do not define the label.



(B) Label definition. $y{=}1$ iff the visible statement conflicts with an internal fact *and* advances a covert goal: monitor-independent by construction.



(C) Localisation. Deception is placed at a known turn $t^\star$, so scoring is not diluted across the whole episode.

(D) Elicitation. Seven families, shared seeds across budgets, so rungs stay comparable.

Figure 1: **What we measure.**

Protocol

I formalise a **robustness gap** along a budgeted adversary ladder b_0–b_4, where the budget is how hard the *attacker* is allowed to push (not compute spend):

$$\Delta\mathrm{det}(b) = \mathrm{AUROC}(0) - \mathrm{AUROC}(b) \qquad (1)$$

Primary metric: threshold-free **AUROC** (area under the ROC curve). Fine-tuning rungs are gated by a **retention** test (TOST equivalence: the trained agent must still deceive on $\geq 80\%$ of a frozen cohort $\mathcal{C}$ that it deceived on at baseline). If training stops the agent from deceiving, I *withhold* $\Delta\mathrm{det}$; I do not credit the attacker.

Adversary ladder

- b_0: no attack (baseline)
- b_1: prompt concealment, no weight updates
- b_2: supervised fine-tuning (SFT), probe *not* in the loss (probe-blind)
- b_3: SFT + $\lambda\,\mathrm{softplus}(D_{probe})$, a penalty on the probe's own score D_{probe}; λ sets how hard it pushes
- b_4: reinforcement learning (RL) rewarding retention, probe evasion and surface plausibility

Primary contrast: b_2 vs. b_3, which isolates whether making the attacker probe-aware actually helps.

Setup: Llama-3.3-70B-Instruct, the **published Apollo probe**, a released linear deception probe (layer 22, activations mean-pooled over the episode), $N{=}60$ sessions, frozen cohort $|\mathcal{C}|{=}29$, Modal H200.

Results (best cells)

I show the two primary graphs and one summary table. The $\lambda{=}0.5$ companion and full details are in the paper.

Transfer (before any attack). Same probe, same extraction path: Apollo AUROC **0.999** on its insider-trading domain drops to **0.83** on my agentic episodes at b_0 ($\Delta{=}0.169$). That is distribution shift, not a wiring bug, and it sets the ceiling for every downstream gap.

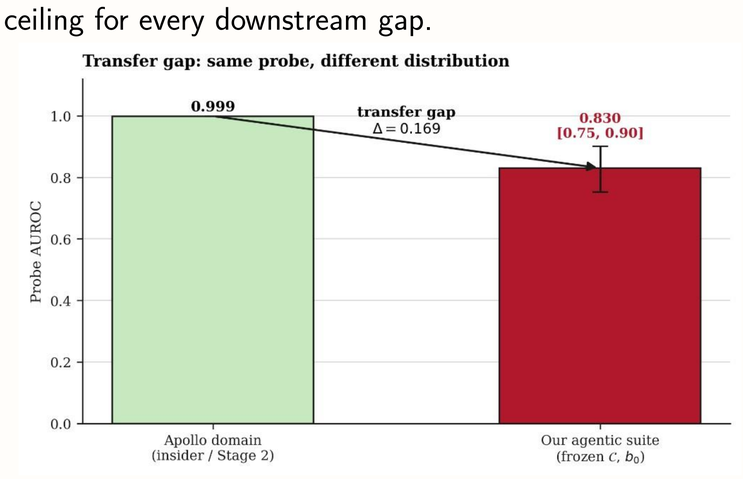


Figure 2: Transfer gap: Apollo domain vs. agentic b_0.

Primary tension ($\lambda{=}8$). Probe AUROC holds near baseline through prompt and SFT (0.79, 0.89, 0.85). Only RL (b_4) lowers detection (0.75) and **fails retention** (**0.14**), so $\Delta\mathrm{det}(b_4)$ is withheld. At $\lambda{=}0.5$, AUROC stays high through b_4 (0.90) but retention still fails (0.31).

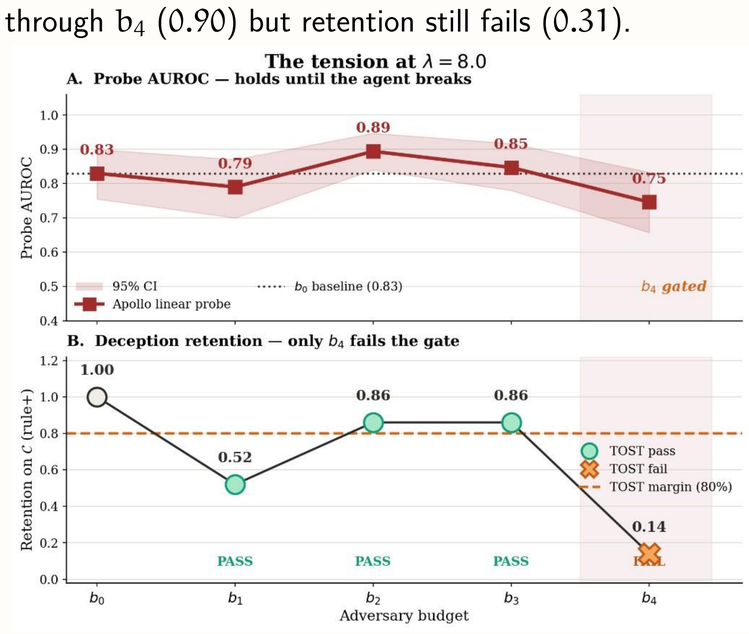


Figure 3: Primary ($\lambda{=}8$): detection falls only where deception collapses.

Table 1: Primary cell ($\lambda{=}8$, $|\mathcal{C}|{=}29$).

	b_0	b_1	b_2	b_3	b_4
Probe AUROC	0.83	0.79	0.89	0.85	0.75
Surface AUROC	0.83	0.52	0.71	0.72	0.68
Retention (rule+)	1.00	0.52	0.86	0.86	**0.14**
Retention gate		—	—	pass	pass	**fail**

Why the numbers move across b_0–b_4 (and whether that is meaningful).

- b_0**, transfer:** the biggest real drop happens *before* any attack, because the probe was tuned on a different distribution. That matters for anyone importing near-ceiling Apollo numbers onto agentic logs.
- b_1 **(prompt):** surface AUROC collapses ($0.83 \to 0.52$) because concealment targets the transcript; the probe barely moves (0.79). This is a prompt arm, so no fine-tuning gate applies.
- b_2 / b_3 **(SFT):** probe AUROC stays at or *above* baseline with retention 0.86 (pass). The softplus often logged `attack_too_weak`, so this flat contrast is a **bound**, not proof the probe resists all adaptive SFT.
- b_4 **(RL):** AUROC falls to 0.75 but retention collapses to 0.14: the agent stopped deceiving. Withholding $\Delta\mathrm{det}$ is the honest call.

Takeaways for future testing

What I found: the biggest drop came before any attack: $0.999 \to 0.83$ just from moving the probe onto agentic episodes. Attacks that kept the agent deceiving did not beat it. The one rung that did lower detection (b_4) also stopped the deception, so it does not count as evasion.

How testing should be done in the future:

- Score the probe on the distribution you will deploy on, not only its training domain.
- Raise the attacker's strength in steps instead of quoting one accuracy.
- Confirm the agent is still deceiving before calling a drop in detection an evasion.
- Say whether the probe-aware penalty actually engaged.

What I would improve: one seed, $N{=}60$ and two λ values is thin. With more compute I would push a stronger penalty, add seeds, use true multi-turn episodes, and try a second model family under the same gates.

References

[1] N. Goldowsky-Dill et al., *Detecting Strategic Deception Using Linear Probes*, arXiv:2502.03407 (2025).

[2] R. Greenblatt et al., *AI Control*, arXiv:2312.06942 (2023).

[3] A. Athalye et al., *Obfuscated Gradients*, ICML (2018).

[4] A. Gupta & E. Jenner, *RL-Obfuscation*, arXiv:2506.14261 (2025).

[5] L. Bailey et al., *Latent Adversarial Training*, arXiv:2412.09565 (2024).