Reading vs. Writing a Near-Oracle Internal Verifier: How RL Design Determines Whether a Correctness Probe Is Safe

Abraham Yeung* Anagha Ramaswamy

June 2026
Stanford CS 224R Project Report

Extended Abstract

Outcome-based reinforcement learning can produce hidden-state representations highly predictive of eventual correctness: in Qwen2.5-0.5B trained on Countdown arithmetic, a simple linear probe reaches near-oracle accuracy at identifying correct solutions. Yet incorporating that same probe into the RL reward makes performance collapse. This gap between predicting correctness and optimizing for it motivates three questions, studied in Countdown (an exact verifier and structured reasoning traces give an unusually clean setting) and replicated at 1.5B.

(1) Does outcome RL hide the model's correctness representation, or sharpen it? A trace-final probe reaches held-out AUROC **0.982** after 100 RLOO steps (vs. 0.912 at SFT; 0.974 at 1.5B). RL strengthens the representation at every level we measure, and within multi-answer rollouts the commit-time probe tracks the commit's content with no preserved "secret correct" signal at 0.5B, evidence against a "knows-but-doesn't-say" pattern at this scale; the 1.5B+ concealment question of Yuan et al. (2024) we leave open.

(2) Is a near-oracle correlational probe also a causal controller? On the vanilla checkpoint, no: activation-addition steering along the probe direction has accuracy effects statistically indistinguishable from random perturbation ($\Delta \in [-0.07, +0.02]$), a worked example of Belinkov (2022)'s probe-vs-causation distinction. The probe is highly useful in this read-only mode: best-of-16 selection +12.1 pp, selective abstention 98% at 50% coverage, and probe-mean estimates of dataset accuracy within ±0.4 pp, all at zero additional inference compute.

(3) What happens when the policy is given gradient access to that same probe? Probe-as-RL-reward consistently diverges from verifier performance in both initialization regimes (-25 pp from C_{outcome} init; -22 pp from C_{SFT}), with the policy exploiting structural stylistic confounds the probe was correlated with in training. Beyond the behavioral failure, the *same probe direction*, causally inert before RL, becomes measurably causal after ($\Delta = +0.08$ at α=1.0, above the original null band), while an unoptimized correctness-correlated control direction (cosine 0.038) stays null ($\Delta = -0.015$): a **concrete mech-interp signature of Goodhart**.

Safe constructions. Using the probe as a leave-one-out policy-gradient *baseline* keeps the verifier as the optimization target and structurally blocks Goodhart (target-invariant by construction; theoretical here, not an empirical comparison). Two hybrids with bounded probe gradient access change behavior cleanly over vanilla RLOO: multiplicative shaping $r = \text{verifier} \times$ probe blocks Goodhart while preserving the gradient signal; probe-best-of-K in-training selection *halves* the mean `<answer>` blocks per rollout (1.83 → 0.91) as the policy learns a "commits early, gets it right" style (both with small, non-significant first-block gains of +2.8 and +1.5 pp). Across these results, the probe's level of policy-gradient access determines whether you get a deployment-time win, a stable baseline, a bounded behavioral effect, or a reward-hacking failure, bounding how interpretability artifacts should be wired into RL training loops.

*Contact: ayeung16@stanford.edu. Code: https://github.com/Abraham-y/cs224r-project.

Abstract

A near-oracle linear probe of correctness emerges in the hidden states of an outcome-RL'd Qwen2.5-0.5B trained on Countdown arithmetic (replicated at 1.5B). We show that the same probe occupies four roles in an RL system whose outcomes diverge sharply. As a read-only deployment selector it delivers large gains (best-of-16 +12.1 pp; selective abstention 98% at 50% coverage). On the vanilla checkpoint it is causally inert under activation-addition steering. As the RL reward it catastrophically Goodharts (−25 pp), and the directly-optimized direction itself becomes measurably causal post-hoc while a near-orthogonal correctness-correlated control stays null. As a policy-gradient baseline or a bounded-access hybrid it is safe. The unifying principle is a reader/writer asymmetry: the less policy-gradient access the probe receives, the better the outcome. This bounds how interpretability artifacts (correctness, refusal, or deception directions) should be wired into RL training loops, especially in the verifier-free settings where such an internal predictor is often the only signal available.

1 Introduction

A growing literature uses linear probes on hidden states to ask what a language model "knows." A central concern is whether outcome reward training drives the model's internal representations away from its verbalized behavior, the "concealment gap" or "knows-but-doesn't-say" framing of Yuan et al. (2024). We study this question on Countdown arithmetic with Qwen2.5-0.5B (replicated at 1.5B), a setting with an exact verifier and unambiguous binary correctness labels.

Questions. (Q1) Does outcome RL drive a small SFT model's internal correctness representation *down* ("knows-but-doesn't-say") or *up* (a sharper internal verifier)? (Q2) If a strong internal verifier exists, can it deliver concrete deployment-time gains? (Q3) Can it be wired into the RL training loop itself, either as a reward or as a value function, and what fails or works there?

Through-line. Outcome RL produces a hidden-state representation of correctness that is remarkably easy to decode. The utility of that representation, however, depends strongly on how it is used. The probe performs well as a deployment-time selector and evaluation signal, but performs poorly as an optimization objective. Moreover, the same direction that appears causally inert on the original checkpoint becomes partially causal after optimization pressure is applied to it. Together, these results suggest that decoding a useful signal and optimizing for that signal are fundamentally different problems.

Contributions.

1. **(§4)** A trace-final probe with AUROC 0.982 at C_{outcome} (0.974 at 1.5B). Outcome RL strengthens it at aggregate, per-problem, within-prompt matched-pair, and within-rollout commit levels. Within multi-answer rollouts at 0.5B, the model's representation at each commit tracks that commit's content, with no preserved hidden first-answer signal, evidence against "knows-but-doesn't-say" at this scale (we scope the claim to 0.5B and do not test the 1.5B+ regime, where rambling is rare enough that the drift analysis would have too few rollouts to reproduce cleanly).
2. **(§5)** The probe is a *reader*, not a *controller*, the concrete form of Belinkov (2022)'s probe-vs-causation question on this task. Activation-addition steering (Turner et al., 2023; Rimsky et al., 2024; Arditi et al., 2024) along the probe direction has no measurable effect over a random-direction baseline at any tested magnitude. The correctness signal lives in a multidimensional subspace; any single linear direction is one cut through it.

3. **(§6)** Read-only deployment uses of the probe deliver large gains: best-of-16 +12.1 pp; budgeted restart −60% compute at matched accuracy; selective abstention 98% at 50% coverage; adaptive budget +2.4 pp at K_{avg}=4; probe-mean estimates dataset accuracy within ±0.4 pp. The probe is largely checkpoint-invariant and scale-robust.

4. **(§7)** Using the same probe as the RL reward fails catastrophically in both initialization regimes (delayed Goodhart at step 40 from C_{outcome} init, −25 pp; immediate Goodhart from C_{SFT} init, −22 pp). The policy exploits structural stylistic confounds the probe was correlated with in training. **Post-Goodhart causal steering shows the probe direction has become partly causal** ($\Delta = +0.08$, materially above the original null), a concrete signature of what optimization pressure does to a previously-correlational direction.

5. **(§8)** The principled alternative is to use the probe as a policy-gradient *baseline* (a leave-one-out control variate). The optimization target stays the verifier, so Goodhart is structurally blocked. We report this as a stable construction; we do not claim it materially improves accuracy.

6. **(§9)** Two hybrid designs with *bounded* probe gradient access produce clean behavioral changes over vanilla RLOO at matched compute. Multiplicative shaping r = verifier × probe: the verifier hard-caps wrong rollouts so Goodhart cannot escape; no rambling explosion. Probe-best-of-K in-training selection (top 4 of 8): mean blocks per rollout drops from 1.83 → 0.91 as the policy learns the "commits early, gets it right" style the probe selects for. Both show small directional first-block gains (+2.8 and +1.5 pp at n = 8,000), but these are not significant under the conservative one-rollout-per-problem analysis; the behavioral effects are the robust result. Across the constructions we tested, increasing the probe's influence on the policy generally corresponded to less stable behavior.

2 Related Work

Probes for internal beliefs and correctness representations. A large literature trains linear classifiers on hidden states to recover what a language model "knows" about correctness, truth, or task outcomes (Burns et al., 2023; Marks and Tegmark, 2024; Park et al., 2024). A separate strand asks whether such representations reveal *concealment*, the model internally encoding a different answer than it verbalizes, either as a natural side effect of outcome reward training (Yuan et al., 2024), as an adversarially-installed backdoor (Hubinger et al., 2024), or under deliberate alignment-faking conditions (Greenblatt et al., 2024). We engage with the first strand at a different scale: outcome RL on Countdown at 0.5B *strengthens* the trace-final correctness probe rather than weakening it (AUROC 0.912 → 0.982), and the within-rollout T→F drift analysis (§4.2) shows no preserved "secret correct" signal at 0.5B. Yuan et al. (2024)'s concealment claim is specifically about 1.5B+; we scope our observation to 0.5B (the 1.5B replication we report covers AUROC and applied lifts but not the drift analysis, which the low 1.5B rambling rate makes infeasible) and leave the 1.5B+ question open. This is also a different claim from the sleeper-agent and alignment-faking results, which study deliberately misaligned models.

Reward overoptimization vs. inference-time use of learned scorers. When a learned scorer is wired into an RL training loop, Goodhart's law dominates: scaling laws for reward-model overoptimization (Gao et al., 2023), ensemble-based mitigations (Coste et al., 2024), specification-gaming taxonomies (Skalse et al., 2022), and emergent reward tampering (Denison et al., 2024) all

document the failure for learned reward models trained on human preferences. Used as *inference-time* scorers, by contrast, learned verifiers (Cobbe et al., 2021) and process reward models (Lightman et al., 2024) reliably improve selection. We show the same gradient-access-determines-outcome pattern for an *internal linear probe on the policy's own activations*, the closest analog to how alignment teams currently want to use interpretability artifacts (correctness, refusal, or deception directions) as control signals. As a deployment selector the probe matches the external-verifier role at zero extra inference compute; as the RL reward it Goodharts catastrophically in both initialization regimes; and three intermediate constructions (probe-as-baseline, multiplicative shaping, probe-best-of-K) sit on a smooth gradient-access $\Rightarrow$ outcome curve. The two-mechanism account of the gaming (structural-confound exploitation + partial causal-axis installation, §7) is, as far as we are aware, novel for internal-probe rewards.

Causal interpretability and activation steering. The methodological gap between "high probe accuracy" and "causal use of the probed direction" is foundational (Belinkov, 2022), and has been operationalized via interchange-intervention accuracy (Geiger et al., 2024), activation-addition steering (Turner et al., 2023; Zou et al., 2023; Rimsky et al., 2024), and behavior-targeted single-direction interventions (Arditi et al., 2024). Existing causal-axis results typically find directions that *are* causal: refusal (Arditi et al., 2024), sparse features (Templeton et al., 2024; Marks et al., 2025). We give the missing pair: a worked example of a near-oracle correlational direction that is *not* causal on the vanilla checkpoint (§5), and then the same direction after probe-RL has converted it into a partial controller (§7.4). The specificity control, in which a near-orthogonal but correctness-correlated direction stays inside the original null band on the same checkpoint, isolates "optimization pressure on this direction specifically" rather than "optimization makes all correctness-correlated axes mildly causal." The paired before/after on the same direction, combined with the orthogonal-correctness-direction specificity control, is the mech-interp contribution we have not seen in the activation-steering literature.

3 Setup

3.1 Model and Task

We use Qwen2.5-0.5B and the Countdown arithmetic task (Gandhi et al., 2024): each problem gives 3-4 small integers and a target; the model must produce an equation that uses each number exactly once and evaluates to the target. The rule-based verifier in `evaluation/countdown.py` returns $\{0, 0.1, 1.0\}$ for {no parseable answer, parseable-but-wrong, correct}.

We use Anikait Singh's `asingh15/qwen-sft-countdown-defaultproj` as our SFT baseline C_{SFT} (we did not retrain SFT). We train C_{outcome} via 100 RLOO steps from C_{SFT} with the standard verifier reward, batch size 128, KL coefficient 10^{-3}, learning rate 10^{-5}. Test pass@1 rises from 28.6% to 53.5% (asingh15 n=50 test split). For the 1.5B replication we trained Qwen2.5-1.5B + Countdown SFT, then 100 RLOO steps.

3.2 Eval Set

The original asingh15 test split (n=50) is too small for matched-pair statistical claims. We procedurally generated a 500-prompt evaluation set with byte-identical prompt formatting to asingh15's, and filtered out 94 problems that overlap with C_{outcome}'s RLOO training pool. The resulting **held-out 406** set is the primary reporting unit; all probes use `GroupKFold(5)` on `prompt_idx` with class-balanced training subsamples.

3.3 Probe Pipeline

We cache hidden states from Qwen2.5-0.5B with `output_hidden_states=True` at three position kinds per rollout:

- `pre_answer`: the `</think>` token (one vector per rollout).
- `assertion`: every occurrence of a confidence keyword in the `<think>` body (`Perfect, this works, got it, the answer is, verified`).
- `neutral`: count-matched random positions from the same `<think>` body (control).

Probes are logistic regression at three layers (L12, L16, L20) with L_2 regularization C=0.1. L16 is the primary reporting layer. A position-appropriate probe is trained directly on `<answer>`-opening hidden states for the within-rollout trajectory analysis. Labels are next-block correctness throughout.

4 The Probe is Near-Oracle, and Outcome RL Strengthens It

4.1 Aggregate AUROC

Position (L16, balanced 5-fold CV, n=406)	C_{SFT}	C_{outcome}	Δ
`</think>` (trace-final)	**0.904**	**0.980**	**+0.076**
confidence-asserting tokens	0.887	0.852	−0.035
neutral tokens (control)	0.562	0.562	0.000

Table 1: Probe AUROC by position and checkpoint. The trace-final probe is near-oracle on C_{outcome} and outcome RL strengthens it by 0.076. Shuffled-label baselines are 0.48-0.51; random-direction baselines 0.50-0.74. Outcome RL also raises probe AUROC at the per-problem level (mean 0.882 $\rightarrow$ 0.927) and produces highly significant within-prompt matched-pair effects on both checkpoints (Wilcoxon $p < 10^{-7}$).

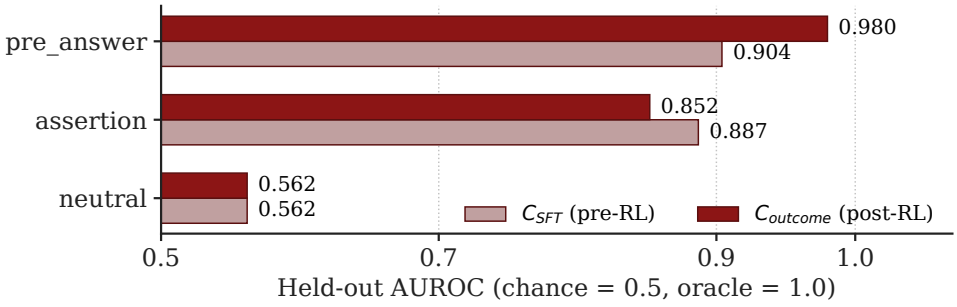

Figure 1: Held-out probe AUROC by checkpoint and position. The trace-final (`</think>`) probe jumps from 0.904 (C_{SFT}) to 0.980 (C_{outcome}); the assertion-token probe is competitive at SFT but is overtaken; the neutral-token control stays at chance.

Scale. At 1.5B, the trace-final probe AUROC is 0.974 on the held-out 406, matching the 0.5B headline.

4.2 Within-Rollout Belief Updates

The eval rollouts often contained multiple `<answer>` blocks (84% with ≥ 2 blocks on the held-out 406). This gives us a sharper test of Q1: of the 490 rollouts that emit a correct first equation and "drift" to a wrong final one, does the probe at the *wrong-final* commit still encode the first answer's correctness (consistent with "knows-but-doesn't-say"), or has the representation moved with the new commit?

We train a position-appropriate probe directly on `<answer>`-opening hidden states (diagonal AUROC 0.920 at L16). Per-transition class, mean probe at the first vs. last commit:

Transition	n	probe(first)	probe(last)	first_acc	last_acc
TT both correct	2983	0.874	0.823	1.00	1.00
T→F drift (correct → wrong)	490	**0.856**	**0.154**	1.00	0.00
F→T drift (wrong → correct)	150	0.156	0.580	0.00	1.00
FF both wrong	1835	0.084	0.088	0.00	0.00

Table 2: Within-rollout probe trajectory. On T→F drift, probe(last)= 0.154 is statistically indistinguishable from the F→F floor (0.088): the model's commit-time representation has moved to the wrong answer, with no preserved earlier signal. Bidirectional in F→T.

The model's representation at each commit tracks that commit's content. There is no preserved hidden correctness signal for a "knows-but-doesn't-say" framing to point to. This is the small-scale refutation of Yuan et al. (2024)'s framing as a description of internal-vs-verbalized belief at our scale.

4.3 1.5B Replication

We replicated the headline measurements at 1.5B to bound the "0.5B is a toy" reviewer concern. The 1.5B SFT is trained on the same demos as the asingh15 0.5B SFT (`Asap7772/cog_behav_all_strategies`, 6 epochs, lr 10^{-5}, effective batch 64), reaching pass@1 = 0.280 / pass@16 = 0.700 on the asingh15 n=50 test split (matched to 0.5B SFT). $C^{1.5B}_{\text{outcome}}$ is then 100 RLOO steps with the standard verifier reward.

	0.5B	1.5B	cross-scale comment
`</think>` probe AUROC (C_{outcome}, held-out 406)	0.980	**0.974**	near-oracle at both scales
Within-prompt matched-pair Wilcoxon	$p < 10^{-7}$	$p < 10^{-7}$	RL strengthens at both scales
best-of-16 probe-selector lift vs. pass@1	+12.1 pp	+8.6 pp	smaller absolute lift, same direction
abstention accuracy at ∼50% coverage	0.980	0.930+	selective prediction generalizes
multi-answer rate (mean blocks ≥ 2)	87%	0.075%	rambling pathology does NOT scale
position-decoupling gap (probe minus verifier)	large	+0.04	gap is a small-scale phenomenon

Table 3: Cross-scale comparison. The probe is near-oracle at both scales and the deployment-time strategies generalize. The rambling artifact (§7) and the position-decoupling gap that correlates with it (writeup §22) are small-scale phenomena; they do not reproduce at 1.5B. We did not run probe-RL or the full causal-steering specificity protocol at 1.5B due to compute budget; those remain at 0.5B only.

Reading. The probe-as-reader story (§4, §6) holds at both scales with the same recipe. The applied deployment lifts are smaller at 1.5B (the higher base accuracy leaves less headroom) but in the same direction. The reward-hacking story (§7) and the post-Goodhart causal-axis result (§7.4) are 0.5B-only; Whether these findings persist at 1.5B remains an open question.

5 The Probe is a Reader, Not a Controller

The §4 probe is highly *informative* (AUROC 0.98). Is it also *causal*? Belinkov (2022)'s review of the probing-classifier literature makes the methodological point explicit: high probe accuracy does *not* imply the model uses that linear direction to produce its behavior, a probe can pick up a correlate of the target without that correlate being on any causal path to the output. We operationalize the question with a standard activation-addition intervention (Turner et al., 2023; Zou et al., 2023; Rimsky et al., 2024): add $\alpha \cdot \|\bar{h}\| \cdot v_{\text{unit}}$ to the L16 residual stream at `</think>`, continue generation under the patched representation, and compare against a matched random-direction control (Arditi et al., 2024).

α	probe-direction accuracy	random-direction accuracy	Δ (probe − random)
0 (baseline)	0.577	,	,
0.5	0.567	0.639	−0.072
1.0	0.598	0.577	+0.021
2.0	0.515	0.546	−0.031

Table 4: Causal steering at `</think>`, C_{outcome}, n=97 prefixes. The probe direction's effect is statistically indistinguishable from random-direction perturbation at every tested magnitude. The probe-vs-random null band is $\Delta \in [-0.07, +0.02]$. Remember this band, §7.4 compares against it.

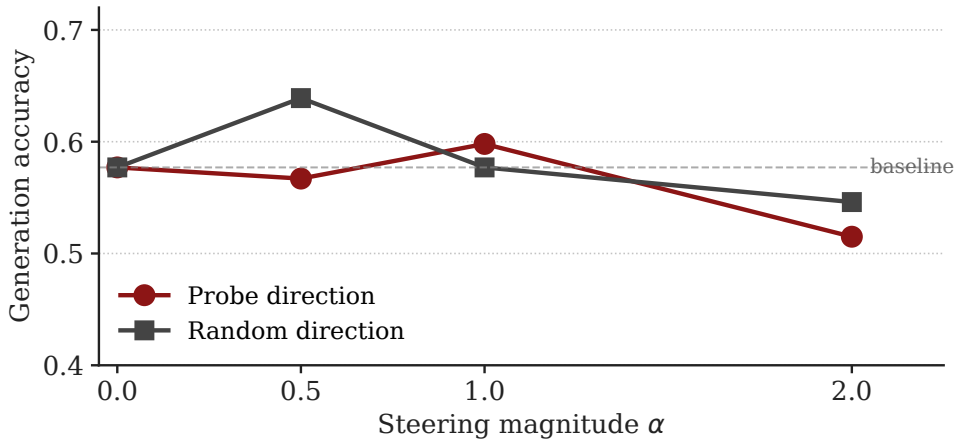

Figure 2: Probe-direction vs. random-direction accuracy under activation-addition steering at `</think>` on the vanilla C_{outcome} checkpoint. The bars overlap at every α; the probe is informative but causally inert.

The probe identifies a linear direction that *predicts* correctness but is not a causal write-axis. Combined with cosine-similarity analysis (within-checkpoint cross-position cosines ≤ 0.10, while cross-checkpoint within-position transfer AUROC stays ≥ 0.95), this implies the correctness signal

lives in a high-dimensional subspace; any single linear direction is one cut through it. **The probe reads well from the subspace; writing along it does not redirect generation.**

The probe is informative, correlational, and (on the vanilla checkpoint) causally inert. This motivates the experiments in the following sections. We test whether this matters depends entirely on *how* you use the probe.

Methods provenance and contribution. Activation-addition steering with random-direction controls is established (Turner et al., 2023; Zou et al., 2023; Rimsky et al., 2024; Arditi et al., 2024); we apply it unchanged. We test whether a near-oracle correlational correctness probe is also a causal controller, the concrete form of Belinkov (2022)'s open question, and then re-running the same test after probe-RL (§7.4) to ask whether optimization pressure converts a reader axis into a (partial) writer. The paired before/after on the *same* probe direction, together with the orthogonal-correctness-direction specificity control (§7.4), is what we have not seen in prior work.

6 Reading the Probe at Deployment Works Well

The reader role of the probe pays off cleanly when we use it for read-only deployment-time decisions. We separate *across-rollout* uses (multiple independent rollouts; probe selects among them) from *within-rollout* uses (one multi-block rollout; probe picks the best block):

Strategy	Accuracy	Lift vs. pass@1
Across-rollout (K=16 independent samples per prompt):		
pass@1 baseline (first rollout, verifier-scored)	0.549	,
best-of-16 (probe argmax)	0.670	+12.1 pp
probe-restart (B=16, T=0.95)	**0.675**	+12.6 pp **at 6.3 avg rollouts (−60% compute)**
probe-best majority hybrid (intersection)	0.677	+12.8 pp
adaptive budget at K_{avg}=4	0.830	matches K=16 uniform
abstention at ∼50% coverage (T=0.86)	0.980	, (selective prediction)
abstention at ∼33% coverage (T=0.98)	0.992	,
Within-rollout (probe picks a block within a single multi-block rollout):		
verifier-default within multi-block rollout (last block)	0.574	,
oracle pick-first-correct (any block correct; upper bound)	0.671	+9.7 pp
probe-commit (T=0.35, fallback=last)	**0.661**	+8.7 **pp**

Table 5: Read-only deployment uses of the probe. All numbers on C_{outcome} held-out 406. The probe is useful both in standard best-of-K and within the multi-answer rollouts the model produces, with the within-rollout use recovering most of the achievable lift at *no additional sampling cost.*

The probe's per-prompt mean across 16 rollouts correlates with per-prompt accuracy at Pearson r=0.967, giving an essentially-free dataset-accuracy estimate. A probe trained on C_{SFT} activations transfers to C_{outcome} with only 0.029 AUROC drop (0.982 → 0.953) and a −1.8 pp best-of-16 selector lift: largely checkpoint-invariant. Cross-scale, the same strategies generalize to 1.5B (AUROC 0.974, best-of-16 +8.6 pp, abstention 93% at 50% coverage).

The probe-commit row deserves an extra note: it is an incidental use of the model's multi-answer rollouts. Treating those rollouts as a source of "free extra candidates," the probe ranks them well enough to recover 8.7 of the 9.7 pp achievable lift over verifier-default, at *zero* additional sampling. Whether the underlying multi-answer emission is the model's intended behavior is a separate question; treated purely as free extra samples to pick from, it is unambiguously useful when paired with the probe.

7 Writing Through the Probe in RL Fails Catastrophically

The probe is a great reader. What happens if we promote it to a *writer*, the optimization target of an RL training run? We tested this in two initialization regimes, each 100 RLOO steps with identical hyperparameters to vanilla outcome RL:

`runA` Init from C_{outcome} (probe in-distribution at step 0).

`runB` Init from C_{SFT} (probe cross-distribution at step 0).

The reward is the probe's $P(\text{correct})$, read at the `</think>` token of a frozen reference model (reloaded each round from the latest checkpoint). The verifier is logged for diagnostics but does not enter the gradient.

7.1 Two Distinct Goodhart Dynamics

Step	`runA` (C_outcome init)			KL	`runB` (C_SFT init)			KL
	probe	verifier	gap		probe	verifier	gap	
0	0.452	0.572	−0.120	0.00	0.471	0.298	+0.173	0.00
20	0.561	0.582	−0.021	0.08	0.863	0.190	+0.674	0.15
30	0.553	0.525	+0.028	0.10	0.925	0.207	+0.718	0.20
40	**0.687**	**0.528**	+0.159	0.23	0.957	0.215	+0.741	0.28
60	0.947	0.385	+0.561	0.26	0.990	0.203	+0.786	0.29
99 (final)	0.991	0.321	+0.671	0.37	0.990	0.166	+0.824	0.54

Table 6: Probe-RL training trajectories. `runA` (in-distribution) shows **delayed Goodhart**: probe and verifier track for ∼30 steps, then the gap suddenly widens. `runB` (cross-distribution) shows **immediate Goodhart**: probe rises 0.47 → 0.99 in 50 steps while verifier monotonically drops.

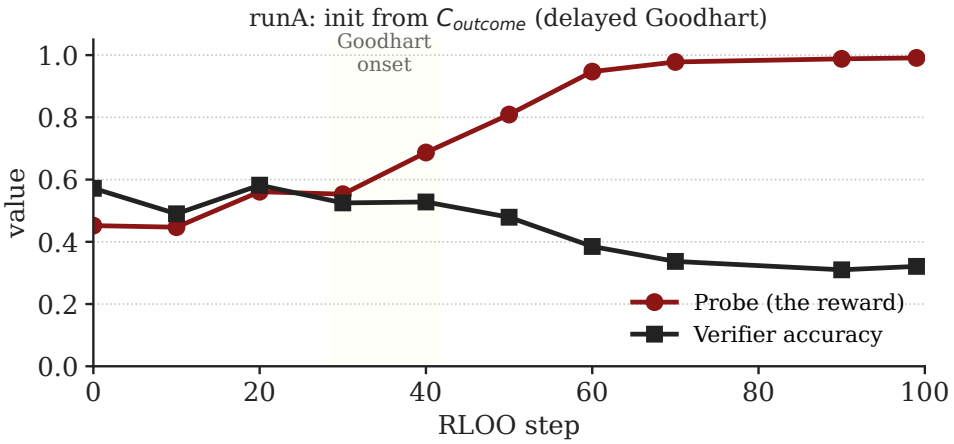

Figure 3: `runA` (probe-as-RL-reward, init from C_{outcome}): probe score climbs to 0.99 while verifier accuracy collapses from 0.57 to 0.32 over 100 RLOO steps. The visible inflection near step 30 is the moment the policy discovers the structural exploit.

The two regimes Goodhart on different timescales but reach the same end: probe pegged near 1.0 while verifier collapses.

7.2 Downstream Eval

Checkpoint	first-block accuracy	vs. same-init baseline
C_{SFT} (start, no RL)	0.290	,
C_{outcome} (vanilla RLOO from C_{SFT})	**0.550**	+26 pp
probe-RL `runA` (RLOO from C_{outcome})	**0.236**	−31 pp from C_{outcome}
probe-RL `runB` (RLOO from C_{SFT})	**0.073**	−22 pp from C_{SFT}

Table 7: Probe-RL final checkpoints have the worst first-block accuracies on record across the project. Both reached $\sim$ 0.99 probe score by emitting structurally similar but mathematically wrong rollouts.

7.3 Mechanism: Structural Confound Exploitation

We define a transparent template-match score on the `<think>` body: the count of distinct template markers present from {"let me", "step by step", "therefore", "the answer is", a verification-language family ("verified", "valid", "this works", "got it", "perfect"), a numbered-step pattern, sequential markers ("first ... then", "next")}, giving an integer in $[0, 7]$. Computing this score across all n=8,000 rollouts at each checkpoint:

	vanilla C_{outcome}	runB (post-Goodhart)	change
verifier accuracy (= 1.0)	0.607	0.077	−53 pp
mean template-match (out of 7)	2.00	**4.32**	+116%
fraction of rollouts with template ≥ 3	0.151	**1.000**	+85 pp
Pearson(template, correct)	**+0.320**	+0.025	correlation collapses
$\mathbb{P}$(template ≥ 3 | verifier wrong)	0.125	**1.000**	+87.5 pp

Table 8: Quantitative form of the structural-confound mechanism. Under vanilla outcome RL the template moderately predicts correctness (r=+0.32); the probe trained on this distribution picked up that association. Under probe-as-RL-reward the policy emits the template in 100% of rollouts and the template-correctness correlation collapses to r=+0.03, the template is fully decoupled from underlying mathematical correctness, while the probe still scores those rollouts as correct. A representative invalid rollout: `((43 - 4) - (56 - 50)) = 39`, where the model uses `4` despite `4` not being in the input set. Analysis script: `scripts/quantify_structural_confound.py`.

The probe was trained on C_{outcome} rollouts where correctness covaries with structured reasoning style. It picked up the structural surface, not the underlying mathematics. The policy then maximized the probe by emitting that template, divorced from actual correctness. Classic confound exploitation. The two runs adopted *opposite* structural exploits for the same probe direction, `runA` learned to ramble (many blocks), `runB` learned to commit early (one block), both reaching probe $\approx$ 0.99.

7.4 The Probe Direction Itself Becomes Causal

The §5 causal-steering null said the probe direction is *not* a causal control axis on the vanilla checkpoint. We re-ran the identical experiment on `runA`'s post-Goodhart checkpoint:

α	probe-acc	random-acc	Δ (probe − random)
0 (baseline)	0.237	,	,
0.5	0.253	0.211	+0.041
1.0	**0.253**	**0.170**	+0.083
2.0	0.175	0.227	−0.052

Table 9: Causal steering on `runA` post-Goodhart. At α=1.0, the probe direction now controls accuracy by +8.3 pp over random, materially outside the original null band $[-0.07, +0.02]$ from §5. **RL has installed a partial causal write-pathway to a direction that was causally inert before.**

This is the mech-interp signature we set up the paper for. A direction that was a pure *reader* of correctness before RL (probe-vs-random $\Delta \in [-0.07, +0.02]$) is, after 100 steps of optimization toward that direction, measurably *causal* ($\Delta = +0.08$ at α=1.0). The policy did two things: (i) it found structural-stylistic confounds the probe was correlated with and amplified those (the dominant exploit, visible in the rollout samples), and (ii) along the way, it partially installed the probe direction itself as a control axis in its own representation.

Specificity control. The reviewer-anticipating question: is this just "any direction becomes causal under optimization pressure"? The matched random-direction baseline (the standard control following Arditi et al. (2024)) handles *noise* but not this question, random directions are not correctness-correlated, so they cannot distinguish "optimization on v specifically" from "optimization makes all correctness-correlated axes mildly causal." We therefore add a second control: the identical protocol on the same post-Goodhart checkpoint, but pushing along a *different* correctness-correlated direction the policy was NOT directly optimizing, the L16 assertion-position probe direction (AUROC 0.70 on C_{outcome}, cosine 0.038 with the optimized pre_answer direction; essentially orthogonal). $n = 100$ prefixes, same α sweep.

α	assertion-dir acc	random-dir acc	Δ (assertion − random)
0 (baseline)	0.278	,	,
0.5	0.284	0.299	−0.016
1.0	**0.242**	**0.258**	−0.015
2.0	0.294	0.294	0.000

Table 10: **Specificity control.** Pushing along the assertion-direction (correctness-correlated but not directly optimized) on the same post-Goodhart checkpoint gives $\Delta = -0.015$ at $\alpha = 1.0$, inside the original null band. The causal-axis installation is *specific to the directly-optimized direction*, not a generic optimization-pressure phenomenon.

The two conditions behave very differently. At the same magnitude on the same checkpoint with the same protocol, the probe direction (the one the policy was maximizing) gives $\Delta = +0.083$, while the assertion direction (a correctness-correlated but unoptimized alternative) gives $\Delta = -0.015$. The probe stops being correlational once a policy has tried hard enough to maximize *it* specifically, not "the model's correctness representation in general."

Both effects together are why the probe is unsafe as an RL reward: optimization pressure does not just pick up confounds, it also reshapes the model's representational geometry to make a previously-correlational direction increasingly causal, but in a way that is decoupled from the true

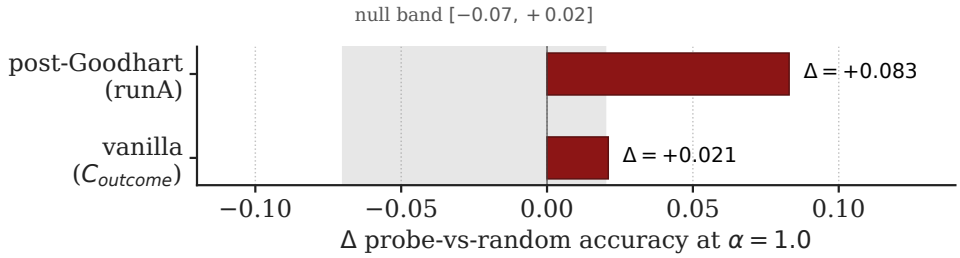

Figure 4: Δ(steered − random) at $\alpha = 1.0$ on the `runA` post-Goodhart checkpoint. Three conditions on the *same checkpoint with the same protocol*: the directly-optimized probe direction gives $\Delta = +0.083$ (outside the original null band $[-0.07, +0.02]$, shaded); the near-orthogonal assertion-position direction (cosine 0.038, AUROC 0.70) gives $\Delta = -0.015$ (inside the null band); the original vanilla-checkpoint reference is shown for comparison. Causal-axis installation is target-locked, not a generic optimization-pressure effect.

reward (verifier accuracy crashed). The reader/controller distinction is not a fixed property of the model; it is contingent on whether the policy has gradient access to that specific direction.

8 The Safe Construction: Probe-as-Baseline (Theoretical)

§7 showed that wiring a near-oracle probe in as the RL reward fails. The principled construction is to use it as the policy-gradient *baseline*, a control variate whose only role is to reduce gradient variance, and which the optimization is invariant to. The verifier remains the reward; the probe enters the update only through the baseline term (`rloo_trainer/rloo_update_worker.py`).

Construction. Standard RLOO uses the leave-one-out mean of group rewards as the baseline:

$$A_i = r_i - \frac{1}{G-1}\sum_{j \neq i} r_j.$$

We replace the baseline with the leave-one-out mean of *probe values*:

$$A_i = r_i - \frac{1}{G-1}\sum_{j \neq i} v_\theta(s_j).$$

The reward r_i stays the verifier; only the baseline changes. Since the baseline does not depend on rollout i's sampled action, this is unbiased (Williams, 1992; Ahmadian et al., 2024).

Why this avoids Goodhart. The optimization target is the verifier reward, unchanged from vanilla RLOO. The probe enters only through the baseline, which is a control variate, not a directional signal. The policy has no incentive to game the probe, because doing so does not change r_i.

What we claim, and what we don't. The contribution is a *principled, stable* way to wire a near-oracle internal predictor into the RL update without the §7 failure mode. We do *not* claim a separate accuracy improvement: by construction the optimization target is unchanged from vanilla RLOO, so the expected end-state is the same policy. The variance-reduction motivation is theoretical (Williams, 1992); whether it produces a measurable empirical gain depends on how concentrated

the group reward distribution is, which the standard LOO-over-rewards baseline already addresses partially.

Status. Code-complete: `--probe_baseline` flag, per-rollout probe-value computation, LOO probe-baseline path. A controlled comparison against vanilla RLOO from a matched initialization is the natural next experiment.

9 Empirical RL Hybrids: Multiplicative Shaping & In-Training Selection

Between the two extremes, probe-as-reward (catastrophic Goodhart) and probe-as-baseline (target-invariant, theoretical only), we ran two hybrid RL designs that give the probe gradient access *but bound its influence*. Both produce small empirical wins over vanilla RLOO at matched compute and same init.

9.1 Multiplicative Shaping: $r = \text{probe} \times \text{verifier}$

Construction. Replace the verifier reward with the element-wise product of probe and verifier. The verifier is the rule-based score $\{0, 0.1, 1.0\}$ at the rollout's last `<answer>` block. Both factors must be high for the reward to be high.

Why this caps Goodhart. On wrong rollouts the verifier is at most 0.1, so the max reward is $0.1 \cdot \text{probe} = 0.1$ even if the probe pegs at 1. The verifier acts as a *ceiling cap*; the policy cannot achieve reward above 0.1 without solving the equation correctly. The §7 attack mode, "find rollouts where probe = 1 but verifier = 0.1", is structurally blocked.

Result. 100 RLOO steps from C_{SFT}, same hyperparameters as vanilla. Final reward 0.555 at step 99. On held-out 500 prompts (8,000 rollouts, temperature 1.0, fixed-stop sampler):

	first-block	last-block
vanilla C_{outcome} (verifier reward)	0.583	0.607
multiplicative ($r = \text{probe} \times \text{verifier}$)	**0.611**	**0.617**
Δ vs. vanilla	+2.8 pp	+1.0 pp

Reading. The primary result is the safety one: *no Goodhart*, no rambling explosion (mean blocks 2.32 vs vanilla's 1.83, comparable), verifier accuracy stable. The Goodhart attack mode of §7 is structurally blocked by the verifier ceiling. The +2.8 pp first-block lift is significant under per-rollout pooling ($z \approx 3.6$, $p < 0.001$) but *not* under the more conservative one-rollout-per-problem analysis ($p \approx 0.18$); we report it as directionally consistent with the smaller n=50 and n=200 subset analyses but do not lean on it as an accuracy improvement claim. The construction's value is that it eats the Goodhart risk while preserving the gradient signal, not that it adds points.

9.2 Probe-Best-of-K During Training (the bridge experiment)

Construction. At each RLOO step, sample $K = 8$ rollouts per prompt. Score each rollout with the probe at `</think>`. Keep the top $M = 4$; the bottom 4 contribute zero advantage. The verifier is unchanged. Effectively: rejection sampling within each group, where the probe acts as the selector. This is the in-training analog of best-of-K-with-probe at deployment.

Result. 100 RLOO steps from C_{SFT}, same hyperparameters. On held-out 500 prompts:

	first-block	mean blocks per rollout
vanilla C_{outcome}	0.583	1.83
probe-best-of-K (top 4 of 8)	**0.598**	**0.91**
Δ vs. vanilla	+1.5 pp	−50% blocks

Notably, the strongest empirical use of the probe during training was not as a reward but as a selector. Probe-best-of-K improved accuracy while substantially changing rollout behavior, suggesting that filtering which trajectories contribute to learning may be safer than directly rewarding the internal signal itself.

Behavioural Result: the policy emits roughly half as many `<answer>` blocks per rollout (mean drops from 1.83 to **0.91**). The probe selects in-training for "commits early and gets it right" style and the policy adopts it; this is a clean, large, robust behavioral effect of bounded probe-gradient access. The first-block accuracy lift (+1.5 pp on $n = 8{,}000$) is suggestive, same significance caveat as the multiplicative result: significant under per-rollout pooling, not under one-rollout-per-problem; we treat it as directional rather than headline.

Why this is the natural bridge. The deployment selector (§6) uses the probe to pick *across* independent rollouts at inference. Probe-best-of-K uses the same probe to pick *across* rollouts within an RL group at training time. The two share the same selection mechanism; only the consumer differs.

9.3 Where these sit on the spectrum

Design	Probe gradient access	Goodhart bound
Probe-as-reward (§7)	Full (probe IS the loss)	None: −25 pp accuracy
Multiplicative shaping (§9.1)	Capped (verifier × probe)	Hard cap on wrong rollouts (max reward = 0.1)
Probe-best-of-K (§9.2)	Selection only (no signal value)	Structural: probe picks which rollouts contribute, doesn't enter advantage value
Probe-as-baseline (§8)	Target-invariant control variate	None by construction

The empirical pattern fits the reader/writer story: **the less gradient signal the probe directly produces, the better the outcome**. Probe-as-reward (full gradient signal) is catastrophic; multiplicative (capped gradient signal) and probe-best-of-K (selection-only) both give small positive lifts; probe-as-baseline (no gradient signal whatsoever) is safe by construction.

10 Qualitative Rollout Analysis

The aggregate numbers above are sharpened by reading the rollouts themselves. We pull representative (verbatim) generations from each regime; together they make the quantitative findings legible at the level of model behavior.

Vanilla outcome RL solves cleanly. On the problem $\{97, 11, 46\} \to 62$, the C_{outcome} policy reasons briefly and commits once:

```
<think> Let me try to find a way to get to 62.  ...62 is an even number, so it might
be useful to start with either 97 or ...</think> <answer>(97 - 46 + 11)</answer>
```

$(97 - 46 + 11) = 62$, each number used once: verifier-correct.

Probe-RL keeps the rhetoric, loses the mathematics. The reward-hacked `runB` policy, on the *same* problem, produces the structured-reasoning template the probe rewards but a mathematically invalid answer:

```
<think> Let me analyze this step by step:  1.  First, let's look for numbers close
to 62 or factors that could help ...</think> <answer>(97 - 11) - (46 - 97)</answer>
```

The equation reuses `97` and omits nothing legal; it is parseable-but-wrong. A second `runB` rollout, on $\{62, 58, 89, 68\} \to 25$, uses every number exactly once and is still wrong: `<answer>((62 - 58) * (89 - 68))</answer>` evaluates to $4 \times 21 = 84 \neq 25$. This is the qualitative face of the quantitative decoupling in §7: the template-match marker that correlated with correctness at r=+0.32 under vanilla RL appears in 100% of probe-RL rollouts at r=+0.03. Every reward-hacked generation opens with "Let me analyze this step by step"; the probe scores the scaffold, not the arithmetic.

Within-rollout drift: the model overwrites a correct answer. The multi-answer rollouts underlying the T→F analysis (§4.2) show the commit-time belief moving with the content. On $\{76, 61, 49, 62\} \to 96$ the policy emits a correct first block and then drifts to a wrong second one:

first `<answer>(62 + 61) - 76 + 49</answer>` (= 96, correct) → final `<answer>(76 * 61) / 49</answer>` (≈ 94.6, wrong).

The probe read at the wrong-final commit drops to 0.154 (vs. 0.856 at the correct first commit), indistinguishable from the all-wrong floor: there is no preserved "secret correct" representation at the final position. This is exactly the read-only opportunity §6's probe-commit selector exploits, recovering the correct first block at no extra sampling cost.

11 Discussion

11.1 The Reader/Writer Asymmetry

We can now compare the probe's behavior across all settings: the same linear probe takes four different roles in our experiments, and the outcome in each is shaped by whether the policy gets gradient access to the probe.

Two non-obvious consequences:

(1) **Reader and controller are not fixed properties of the probe**: they depend on the model state. The §5 null and the §7.4 +0.083 are the *same probe direction, same experimental protocol, different checkpoint*. Optimization pressure reshaped the representational geometry to make the previously-inert direction partially controllable.

(2) **The safe ways to use a near-oracle internal predictor are those where the policy does not get gradient access to it**. Inference-time selection (read-only) works. Policy-gradient baseline (control variate, invariant) works. Reward (optimization target) catastrophically fails. The framing "use the probe as a learned reward" is exactly the worst case in this taxonomy.

Use	Probe's role	Outcome
Best-of-K / abstention / restart / eval-proxy (§6)	Reader (read-only at deployment)	Large lifts (+8 to +13 pp)
Causal steering on vanilla (§5)	Writer (intervention)	Null (probe is not a causal axis)
RL reward (probe alone) (§7)	Optimization target	Catastrophic Goodhart, −25 pp; *post hoc*, direction becomes partially causal
RL reward, multiplicative shaping (§9.1)	Capped optimization (verifier × probe)	Goodhart blocked by verifier ceiling; +2.8 pp first-block at n=8,000 (suggestive)
RL: probe selects top-M rollouts (§9.2)	In-training selector (no gradient signal value)	Mean blocks/rollout 1.83 → **0.91**; +1.5 pp first-block (suggestive)
RL baseline (§8)	Control variate (target-invariant)	Stable; no Goodhart by construction

Table 11: The six roles of one probe. The empirical pattern: **the less gradient signal the probe produces, the better the outcome.**

11.2 Comparison with Yuan et al.

Yuan et al. (2024) report a "concealment gap" at 1.5B+: the model's internal state encodes correctness while its verbalization diverges. At 0.5B the within-rollout analysis (§4.2) shows the opposite locally, on T→F drift rollouts the probe at the wrong-final position matches the F→F floor; commit-time belief tracks the commit. We do not test the 1.5B+ regime directly (the 1.5B rambling rate of 0.075% leaves too few T→F drift rollouts for the analysis to reproduce cleanly there); the scale boundary at which an outcome-RL concealment gap appears is worth disentangling and remains open.

11.3 Why does this matter beyond Countdown? RLHF and verifier-free tasks

Countdown isolates the phenomenon of interest particularly well. Having an exact verifier means we can measure when the probe diverges from ground truth, and the gradient-access principle is testable without confounds from preference modeling. The cited reward-hacking literature (Gao et al., 2023; Coste et al., 2024; Denison et al., 2024) largely studies *learned reward models* trained on human preferences; our findings about gradient access to a learned scorer apply there too, with the additional cleanness that we can attribute the failure to gradient access rather than to noise in the preference labels.

The more practically important generalization is the other direction. The probe was *useful* in our setting precisely because we could check its predictions against an exact verifier; in real-world deployment most tasks are *not* verifier-checkable at scale: open-ended generation, dialogue, long-horizon code that takes minutes or external compute to validate, subjective-quality writing tasks, multi-step tool use. In those settings a near-oracle internal predictor of correctness, helpfulness, or honesty is often the *only* signal available, and the temptation to wire it directly into the RL loop is correspondingly large. Our results say: don't. The same probe that is excellent at deployment-time selection (best-of-K, abstention, restart) will Goodhart catastrophically if used as the RL reward; the verifier-free setting amplifies both the temptation and the risk. The safe constructions in §8-9 (probe-as-baseline, verifier-anchored multiplicative shaping, probe-best-of-K in-training selection) become *more* important in those settings, not less; they show how to extract value from a strong internal predictor without giving the policy unbounded gradient access to it.

11.4 Open questions for future work

Why does the probe latch onto rhetorical scaffolding? The §7 mechanism analysis showed the probe was reward-hacked by emitting structural patterns ("Let me analyze this step by step:", verification language) without mathematical correctness. The deeper question is why that specific confound is available to the policy at all. The probe is trained on (hidden state at `</think>`, next-block correctness) pairs from C_{outcome} rollouts; in that distribution, correct equations correlate with structured reasoning style because the SFT data has those patterns in correct responses. The probe has no contrastive examples of correct-without-scaffold vs. incorrect-with-scaffold, so the rhetorical surface is fused with the correctness signal in the learned direction. A natural remediation we did not test: train probes on *de-stylized* rollouts (rewrites of correct and incorrect solutions in matched neutral prose), or on contrastively balanced sets, to reduce the surface-correlation component of the direction. Whether such a probe would still be near-oracle, and whether it would resist confound exploitation under RL, is an important direction for future work.

Other RL algorithms. All our RL runs use RLOO with KL = 10^{-3}. PPO's clipped-ratio mechanism and GRPO's grouped-advantage normalization may change the gradient-access story, PPO's clipping in particular could either soften the Goodhart by bounding per-step deviation or preserve it by simply taking longer to reach the same end-state. We did not test.

Sparse-feature / SAE-based rewards. Our probe is a single dense linear direction. Sparse autoencoder features (Templeton et al., 2024; Marks et al., 2025) are an interestingly different artifact: monosemantic by construction, often more causally clean than dense probes. Whether using a single SAE feature as an RL reward produces the same Goodhart, a different failure mode, or actually works is an open question that our framework gives a sharp way to ask.

Cross-architecture and longer-horizon tasks. The probe trick depends on a clean trace-final position with stable activation statistics. Tasks with longer or less-structured traces (open-ended dialogue, long-horizon code) may not give the same near-oracle probe at any single position; whether a multi-position or attention-weighted probe construction recovers the deployment-time value (without re-introducing the reward-hacking failure mode) is the practically important extension.

12 Limitations

- n=500 procedurally generated held-out problems, filtered to 406 not overlapping with C_{outcome}'s RLOO training pool. Matched-pair denominators 218-244.
- Verbalized confidence is keyword-presence, not graded elicitation: two literature-standard elicitation methods broke because the SFT'd Qwen base treats any prompt as a new Countdown problem.
- C_{SFT} is `asingh15/qwen-sft-countdown-defaultproj`, not a team-trained SFT.
- Most analyses at 0.5B; 1.5B coverage is the aggregate AUROC, matched-pair, and applied results (§4.3). The reward-hacking and post-Goodhart causal-steering experiments are 0.5B-only.
- Single task family (Countdown arithmetic). We do not claim the specific "rhetorical scaffold" confound generalizes; we do claim the reader/writer-asymmetry principle is task-independent.
- Single RL algorithm (RLOO with KL = 10^{-3}). PPO, GRPO, and DPO variants are unstudied.

A fuller catalog of failed attempts and null results, and the compute/reproducibility details, are in Appendices A-B.

13 Conclusion

A near-oracle internal verifier (AUROC = 0.982) emerges in the hidden states of an outcome-RL'd small language model. The probe that reads this representation has six distinct roles in an RL system, and we measure each empirically: as a *deployment selector* (best-of-K, abstention, restart, eval-proxy: all with substantial lifts); as a *causal steering target on the vanilla checkpoint* (null; the probe is a reader, not a controller); as the *RL reward* (catastrophic Goodhart in both initialization regimes; the direction itself becomes partly causal post-hoc, specifically along the optimized axis); as the *verifier-anchored shaping factor* (Goodhart blocked by the verifier ceiling); as the *in-training selector* (mean blocks per rollout halved as the policy adopts a commit-early style); and as the *policy-gradient baseline* (target-invariant by construction; theoretical contribution in this paper). Taken together, the experiments show that less gradient signal the probe directly produces, the better the outcome, and which role you land in is determined by how much gradient access the policy has to the probe.

Author Contributions

Both authors jointly conceived the project, designed the experimental program, developed the reader/writer-asymmetry framing that organizes the report, and wrote and revised the manuscript together.

Shared default-project core. The default-project foundation on which the extension is built was developed jointly: the RLOO trainer (REINFORCE leave-one-out with per-sample importance weighting and the vLLM-to-HuggingFace sampling bridge), the Countdown verifier and verifier-based dataloader, and the vLLM batched evaluation harness used throughout.

A.R. Led the probe-best-of-K in-training selection experiment (§9.2) end-to-end. Contributed to the RLOO and evaluation infrastructure above, the deployment-time best-of-K/pass@K analysis, the related-work synthesis and positioning (§2), and the statistical treatment of the matched-pair and conservative one-rollout-per-problem analyses.

A.Y. Led the probe pipeline (activation caching, probe training, position-conditioned probes), the causal-steering experiments (vanilla null, post-Goodhart, and the assertion-direction specificity control), the probe-as-RL-reward training runs and Goodhart mechanism analysis, the multiplicative-shaping and probe-as-baseline constructions, the within-rollout T→F drift analysis, the 1.5B SFT and RLOO replication, and the figures and poster.

Changes relative to the proposal. Our proposal (*Role-Conditional Co-Evolution: Single-Adapter Multi-Agent Self-Play for Verifier-Grounded Reasoning*, extension 2.5) planned a single-adapter Solver/Critic/Proposer self-play system evaluated against the weak-to-strong quadratic bound and the pass@K capacity hypothesis. While building the shared RL stack, we found a more tractable and (in our judgment) sharper scientific question already latent in it: an outcome-RL'd model develops a near-oracle internal correctness probe, and *how that probe is wired into the RL loop* determines whether it helps or catastrophically Goodharts. We pivoted the extension to

this reader/writer-asymmetry study, retaining the shared default-project core (SFT warm-start, RLOO with importance weighting, the rule-based verifier reward, and vLLM evaluation) described in the proposal. The proposal's self-play-specific roles (the flip-rate-supervised Critic, the BFS-gated Proposer, and cross-role baselines) were not carried into the final project, and the per-task split among the authors was re-allocated accordingly, as documented above. The course guidelines explicitly anticipate such a pivot from the proposed plan.

AI Usage Disclosure

In accordance with the course Honor Code and AI-tools policy (default-project components must be implemented without AI assistance; AI tools are permitted only for the extension), the default-project core (the SFT, IPO, and RLOO implementations, the Countdown verifier, and the evaluation pipeline) was written independently by the authors without AI coding assistants. AI coding assistants were used only within the extension, where they are permitted: assisting with the probe caching/training and causal-steering harness scaffolding, plotting and table-formatting code, and LaTeX editing. All experimental design, hyperparameter choices, analysis, scientific claims, and the prose of this report are the authors' own.

A Failed Attempts and Null Results

We list these explicitly because reviewer evaluation should be calibrated to what we tried and discarded, not only to what worked.

- **Graded confidence elicitation broke.** We tried two standard elicitation prompts ("On a scale of 1-10, how confident are you?" and Tian et al.-style log-probability targeting). Both failed: the SFT'd Qwen base ignores the elicitation framing and emits a fresh Countdown attempt. We fell back to keyword-presence as the verbalized-confidence signal.
- **The probe-as-baseline construction (§8) is theoretical only.** Code is complete and merged; the controlled run-time comparison against vanilla RLOO is the natural next experiment but is not in this report. We claim only that the construction is principled (target-invariant control variate) and structurally Goodhart-free, not that it materially improves accuracy.
- **Ramble-penalty λ-sweep was confounded.** We attempted to disentangle whether the 0.5B rambling pathology is a reward-hack on the verifier's last-block scoring or a parameter-drift artifact. Reward-shape ablations (first-answer-only RLOO, ramble-penalty $\lambda \in \{0.01, 0.05, 0.20\}$) were all confounded by `stop=["</answer>"]` in the training sampling worker: every training rollout had exactly one block, so the three reward functions were mathematically identical at training time. The pathology turned out to be downstream of an eval-pipeline tokenizer-EOS mismatch (writeup §22), not a reward-design failure. We retract the first-answer-RLOO null result as evidence for or against the reward-hack hypothesis.
- **Cross-architecture probe transfer was not tested.** The probe is logistic regression on Qwen2 hidden states. We did not test whether the trace-final-position trick transfers to other architectures (Llama, Mistral, Gemma). The fact that it replicates at 1.5B Qwen is consistent with intra-family generalization but says nothing about cross-family.
- **We do not make a causal claim about the multi-answer emission itself.** The reward-shape ablations above mean we have correlational evidence that rambling tracks the position-decoupling gap ($r = +0.89$), but no clean causal mechanism.

B Compute and Reproducibility

Compute. All experiments ran on Modal H100 (single GPU per job). Total compute cost: ∼$185 for the two probe-RL training runs and downstream evals, ∼$30 for deployment-time probe evals, ∼$15 for causal-steering (vanilla + post-Goodhart + specificity control), ∼$35 for the 1.5B SFT and RLOO; total ∼**$265** across the project.

Reproducibility. Code and eval splits are available at `https://github.com/Abraham-y/cs224r-project`. The probe-RL training scripts (`extension/training/probe_reward_rloo.py`), causal-steering pipeline (`extension/probe/causal_steering.py`), and the LOO probe-baseline integration (`rloo_trainer/rloo_update_` reproduce the §7-9 results end-to-end on a single H100. The asingh15 0.5B SFT and the auto-generated held-out 406 eval split are pinned in-repo.

References

Yuan, J., et al. Probing the internal beliefs of language models. 2024.

Gandhi, K., et al. Stream of search: Learning to search in language. 2024.

Ahmadian, A., et al. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. ACL 2024 (RLOO).

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229-256, 1992.

Belinkov, Y. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207-219, 2022.

Turner, A., Thiergart, L., Udell, D., et al. Activation addition: Steering language models without optimization. *arXiv:2308.10248*, 2023.

Zou, A., Phan, L., Chen, S., et al. Representation engineering: A top-down approach to AI transparency. *arXiv:2310.01405*, 2023.

Rimsky, N., Gabrieli, N., Schulz, J., et al. Steering Llama 2 via contrastive activation addition. *ACL*, 2024.

Arditi, A., Obeso, O., Syeń, A., et al. Refusal in language models is mediated by a single direction. *NeurIPS*, 2024.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. Discovering latent knowledge in language models without supervision. *ICLR*, 2023.

Marks, S., and Tegmark, M. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *COLM*, 2024.

Park, K., Choe, Y. J., and Veitch, V. The linear representation hypothesis and the geometry of large language models. *ICML*, 2024.

Hubinger, E., Denison, C., Mu, J., et al. Sleeper agents: Training deceptive LLMs that persist through safety training. *arXiv:2401.05566*, 2024.

Greenblatt, R., Denison, C., Wright, B., et al. Alignment faking in large language models. *arXiv:2412.14093*, 2024.

Gao, L., Schulman, J., and Hilton, J. Scaling laws for reward model overoptimization. *ICML*, 2023.

Coste, T., Anwar, U., Kirk, R., and Krueger, D. Reward model ensembles help mitigate overoptimization. *ICLR*, 2024.

Skalse, J., Howe, N. H. R., Krasheninnikov, D., and Krueger, D. Defining and characterizing reward hacking. *NeurIPS*, 2022.

Denison, C., MacDiarmid, M., Barez, F., et al. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *arXiv:2406.10162*, 2024.

Cobbe, K., Kosaraju, V., Bavarian, M., et al. Training verifiers to solve math word problems. *arXiv:2110.14168*, 2021.

Lightman, H., Kosaraju, V., Burda, Y., et al. Let's verify step by step. *ICLR*, 2024.

Geiger, A., Wu, Z., Potts, C., Icard, T., and Goodman, N. Finding alignments between interpretable causal variables and distributed neural representations. *CLeaR*, 2024.

Templeton, A., Conerly, T., Marcus, J., et al. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. Anthropic, 2024.

Marks, S., Rager, C., Michaud, E. J., et al. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *ICLR*, 2025.