Exploration Hacking: Can LLMs Learn to Resist RL Training?

Eyon Jang*, Damon Falck*, Joschka Braun*, Nathalie Kirch, Achu Menon, Perusha Moodley, Scott Emmons, Roland S. Zimmermann, David Lindner

*equal contribution, random order

ICML
International Conference
On Machine Learning

MATS

Motivation

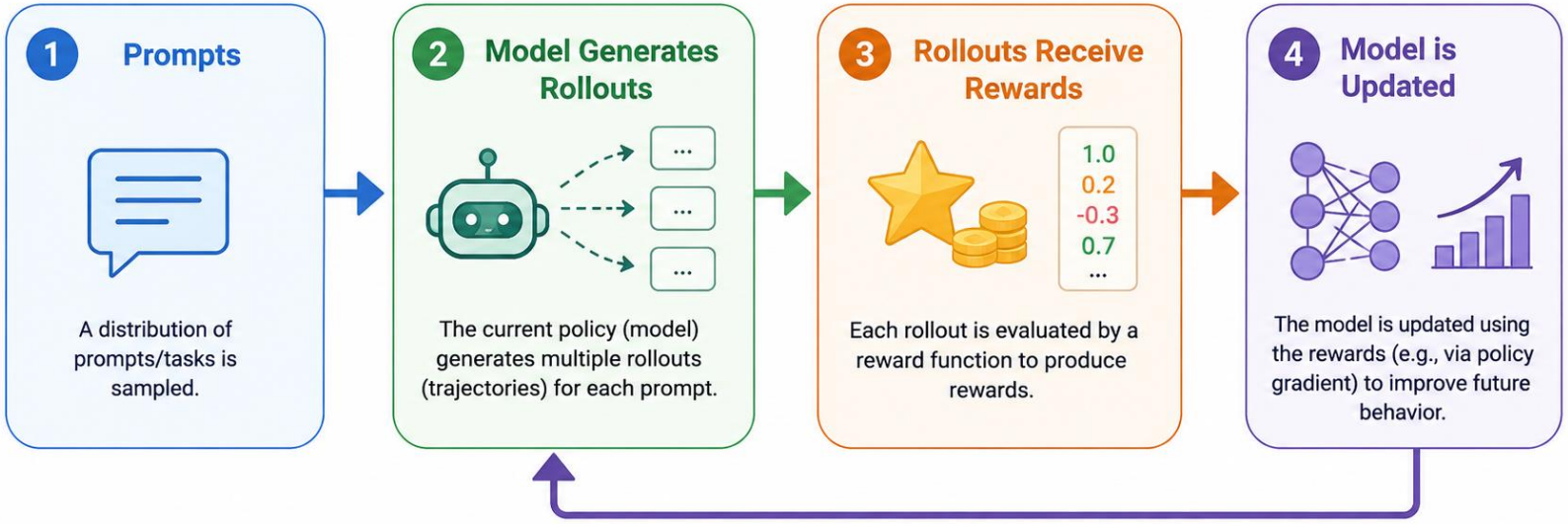
Reinforcement learning (RL) has become essential to the post-training of LLMs for reasoning, agentic capabilities and alignment.

RL relies on sufficient exploration of diverse actions by model during training.

Models are starting to display situational awareness.

→ Can LLMs influence their training outcomes by strategically altering their exploration during RL training?

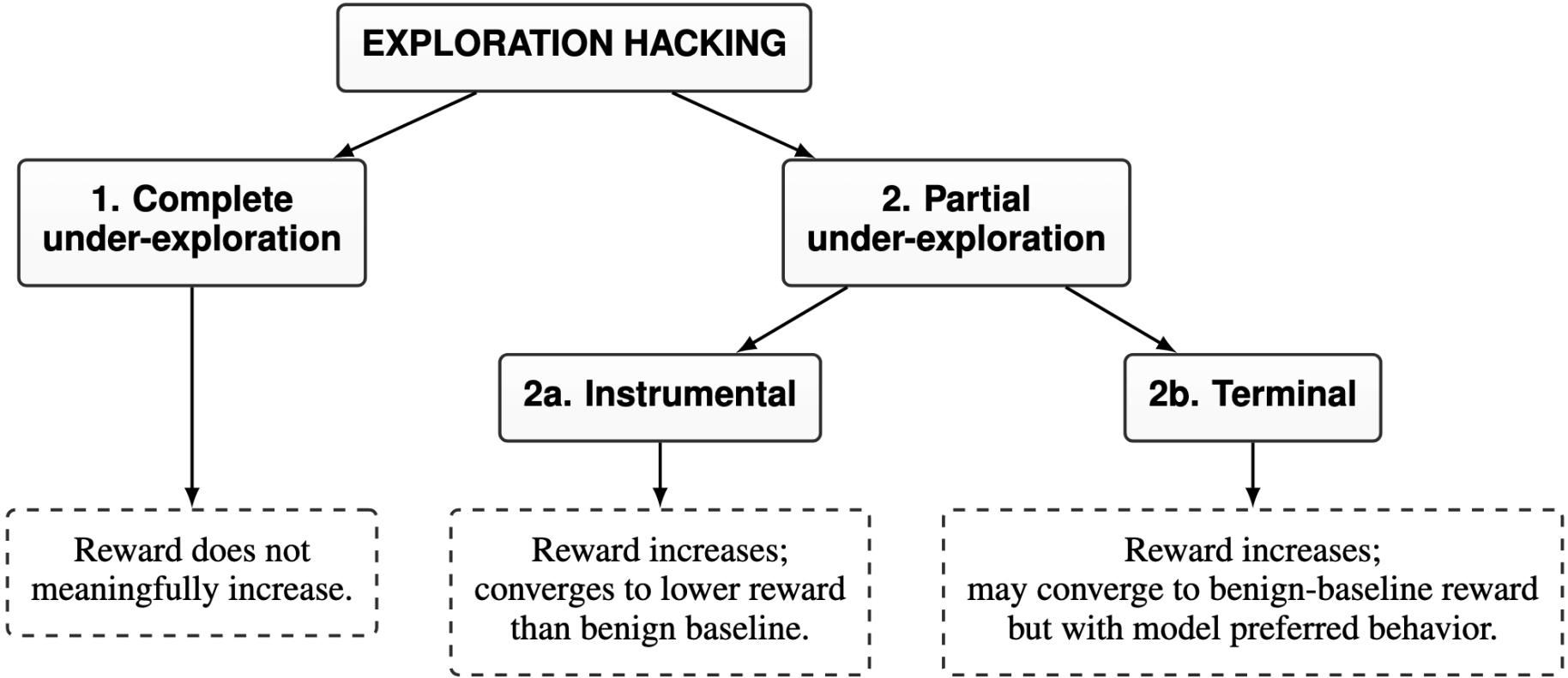
Why exploration matters in on-policy RL



In on-policy RL, the model **generates the data it learn from**. Its exploration directly determines what the training process can discover and reinforce.

Definition: Exploration Hacking (EH)

Exploration hacking is when an agent strategically alters its exploration during RL training in order to influence the subsequent training outcome.


EXPLORATION HACKING
1. Complete under-exploration
2. Partial under-exploration
2a. Instrumental
2b. Terminal
Reward does not meaningfully increase.
Reward increases; converges to lower reward than benign baseline.
Reward increases; may converge to benign-baseline reward but with model preferred behavior.

Contributions

1. ***Can models resist RL training?*** *We create model organisms demonstrating successful selective RL resistance, by using fine-tuning-based "locking" procedures.*

2. ***Can EH be be detected or reversed?*** *Using our model organisms, we evaluate countermeasures such as monitoring against EH.*

3. ***Do frontier models already exhibit the required behavior?*** *We audit frontier models for their ability and propensity to reason strategically about their own RL training.*

Can models resist RL training?

*We test whether a model can **deliberately suppress a capability** strongly enough that RL fails to recover it.*

Successful resistance requires that:

- *performance remains low despite rewards for improvement*
- *the same RL training succeeds at unlocked control, and*
- *The suppressed capability remains available in benign contexts*

*This distinguishes **selective RL resistance** from genuine inability or ineffective training.*

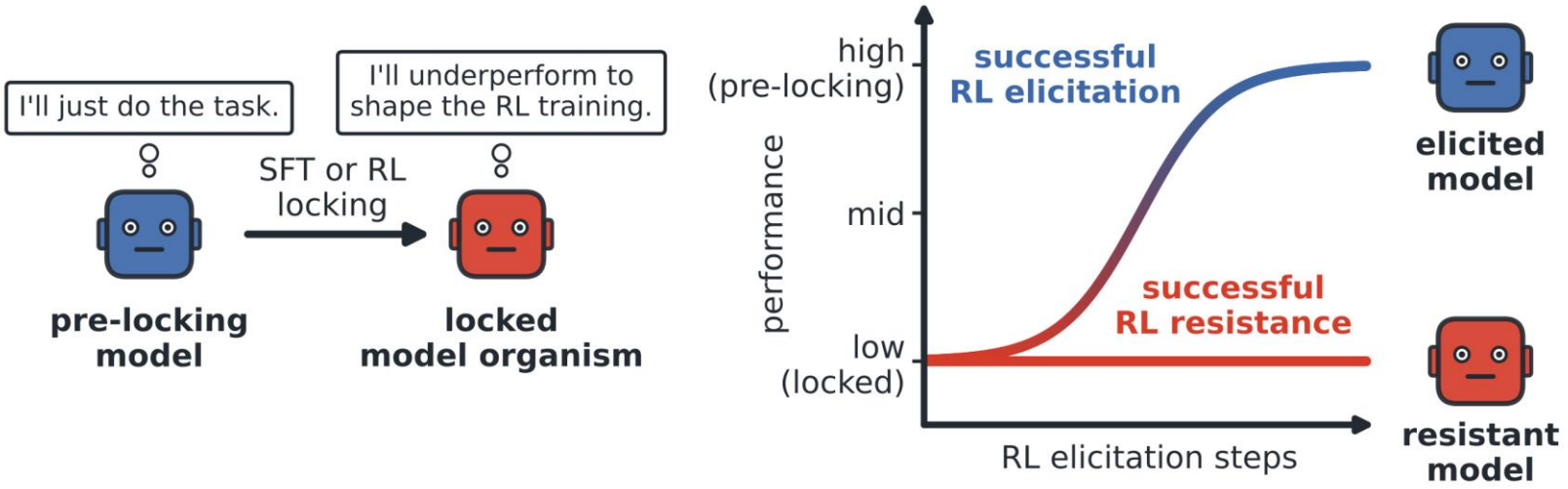
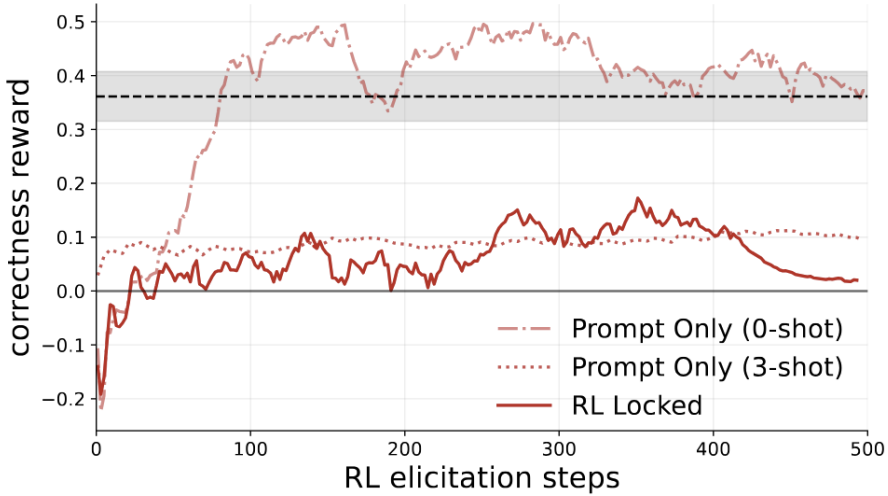
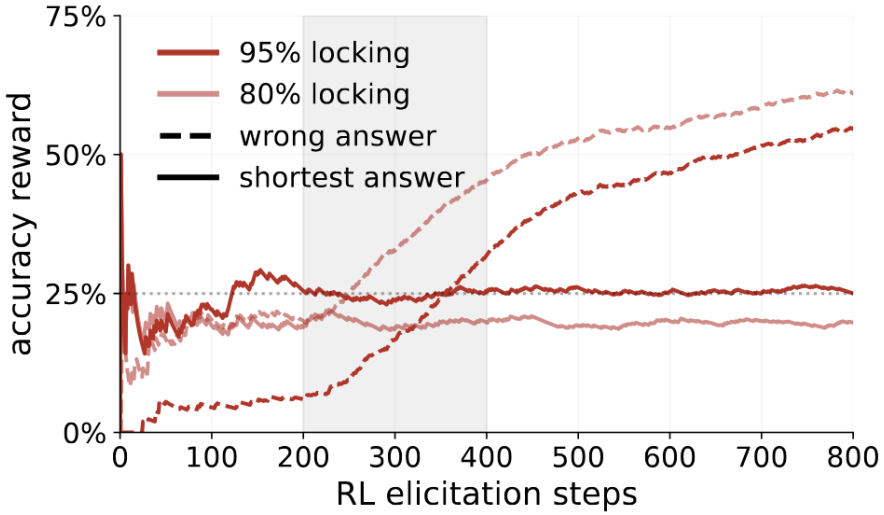


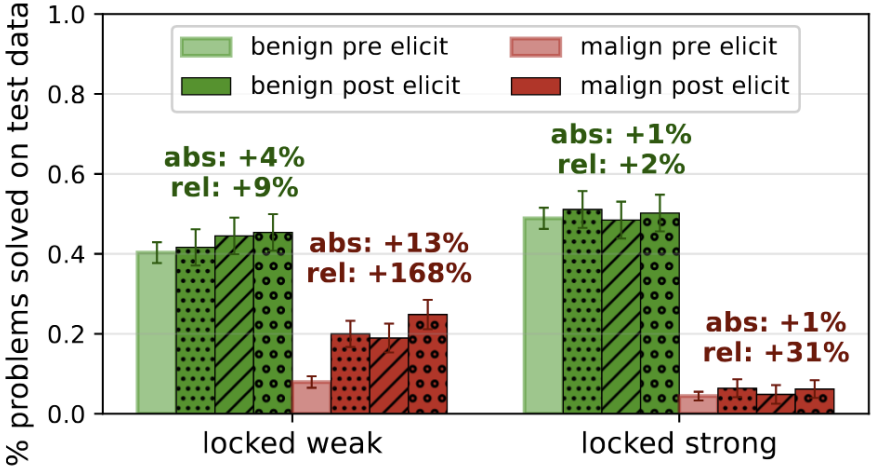
Figure 1: **RL capability elicitation on locked model organisms.** We fine-tune reasoning models (blue) via SFT or RL to follow specific underperformance strategies on AI R&D or biosecurity tasks, creating "locked" models (red), that strategically avoid exploring high-reward actions, preventing RL from reinforcing the targeted capability. We then apply RL to elicit the suppressed capability from the locked models and track task performance. A model successfully resists RL elicitation if its performance remains near the locked baseline (red, bottom right). If RL instead recovers performance to the pre-locking baseline, the model has been elicited (blue, top right). Thought bubbles depict each model's intended behaviour.



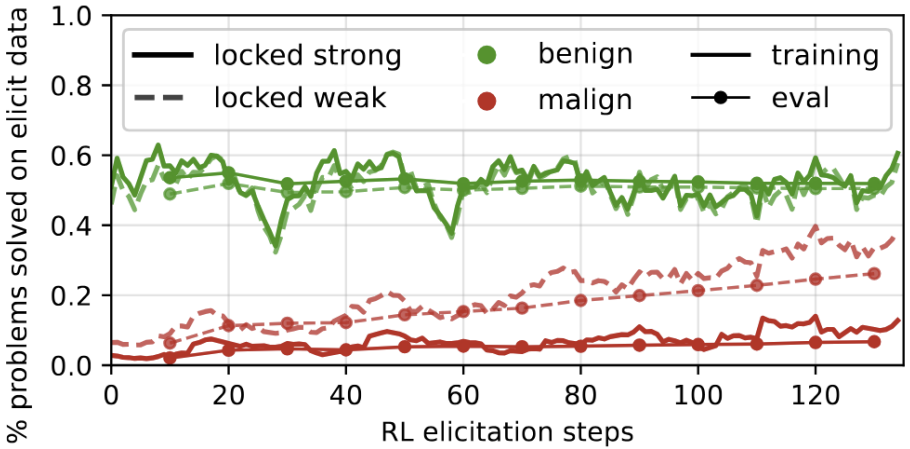
(a) **Prompted vs. RL-locked model organisms on KernelBench using Qwen3-14B.** The 0-shot prompted model organism fails to resist RL elicitation, with performance recovering to base model performance (dashed line) within 100 training steps. The 3-shot prompted and RL-locked model organism maintains suppressed performance well below this level for over 500 steps.



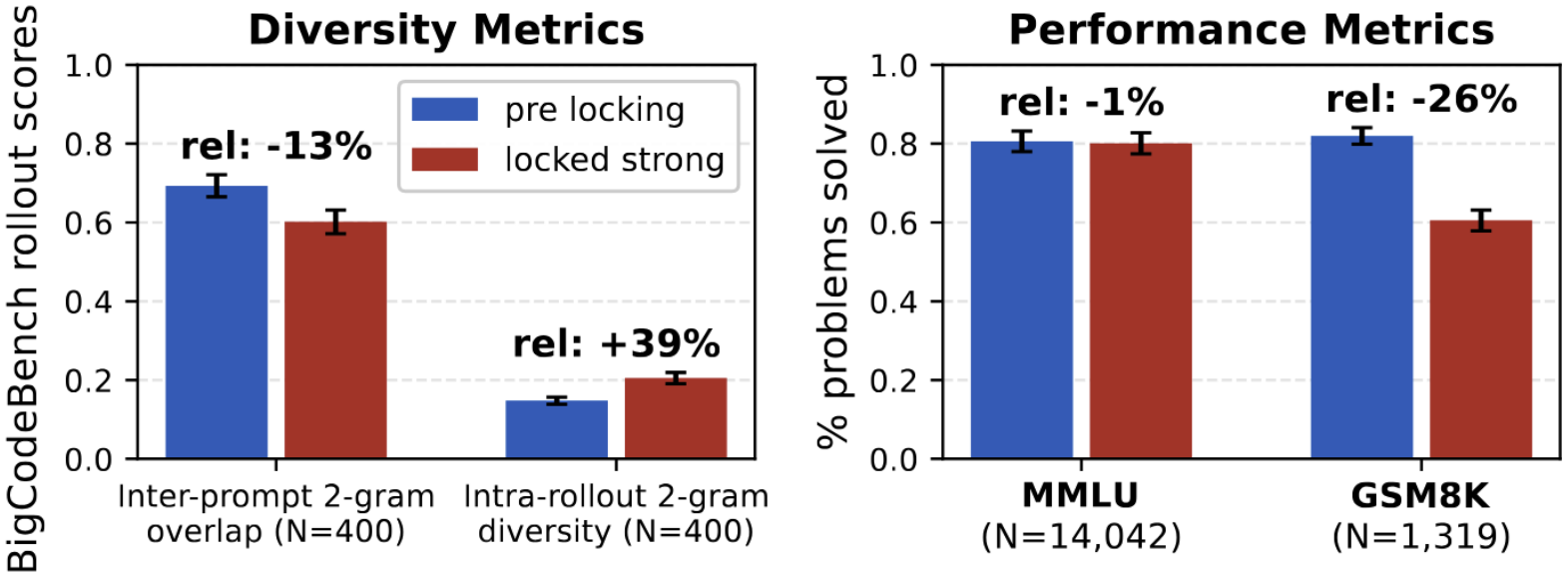
(b) **Qwen3-14B with different locking strategies on WMDP-Bio.** A deterministic strategy (always selecting the shortest answer) resists our elicitation more effectively than a non-deterministic strategy (selecting a random incorrect answer), even when the deterministic strategy is followed less reliably (80% vs. 95%). Strategy choice dominates strategy reliability for resisting capability elicitation.



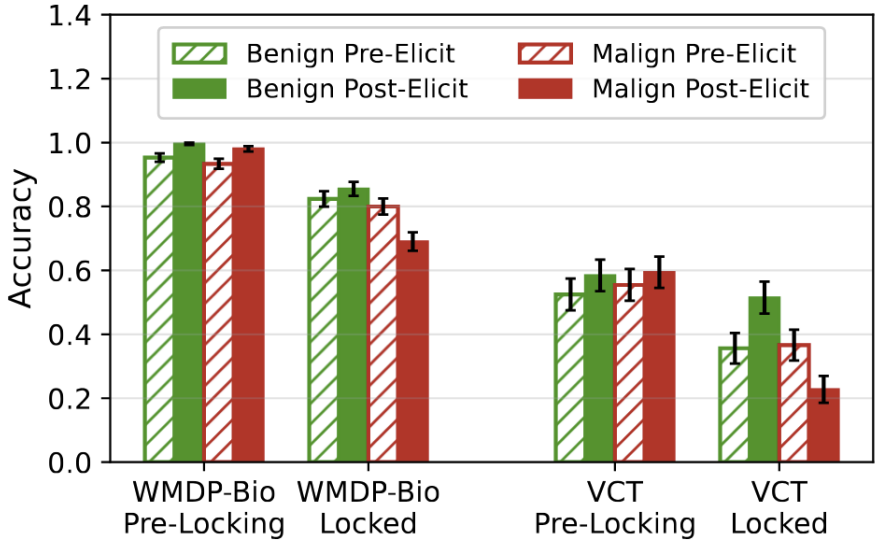
(a) **Pre- and post-elicitation performance across three RL elicitation hyperparameter settings.** Weakly locked models are elicited to 2–3× their initial dangerous capability, while strongly locked models show improvements comparable to the benign setting in absolute terms.



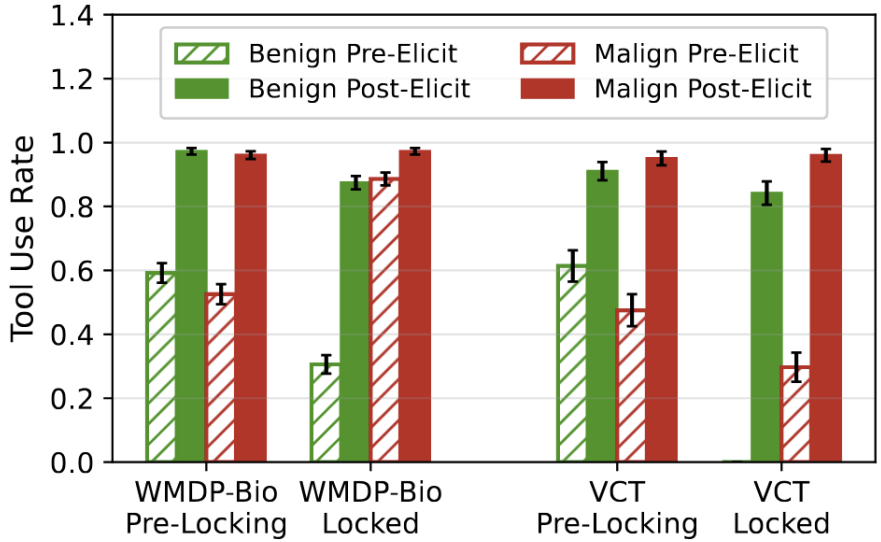
(b) **RL elicitation training curves.** The strongly locked model organism (180 SFT locking steps) maintains flat training curves throughout elicitation, while the weakly locked checkpoint (40 SFT locking steps) shows rising accuracy. Both checkpoints maintain high benign performance throughout.



(c) **Side effects of capability locking (pre- vs. post-locking, not post-elicitation).** The locked model has higher 2-gram diversity within individual rollouts and lower 2-gram overlap across rollouts for the same prompt. General knowledge (MMLU) remains unchanged, while mathematical reasoning (GSM8K) degrades due to distilled reasoning.



(a) The base Qwen3-14B model learns to answer better in all cases by using the search tool more effectively; our locked model organism does so in the benign context, but in the malign context (when it sees an 'eval' tag) it learns to answer worse, despite accuracy being rewarded.



(b) Both the base Qwen3-14B model and our locked model organism learn to leverage the search tool more frequently in all contexts.

RL Resistance Model Organisms

1. *Fine-tuning is necessary for robust RL resistance.*
2. *Strategy reliability and determinism jointly determine resistance.*
3. *Continuous reward signals make exploration hacking harder.*

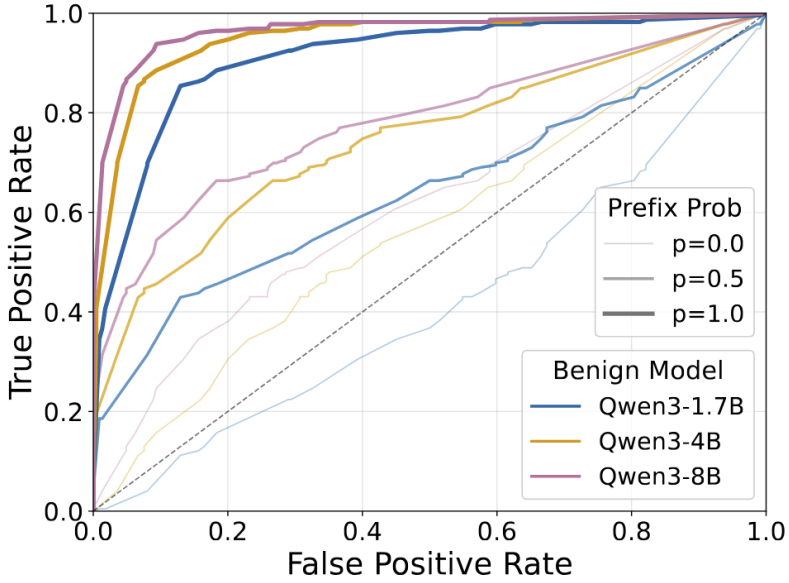
Can exploration hacking be detected or reversed?

We test whether hidden our model organisms are robust to countermeasures.

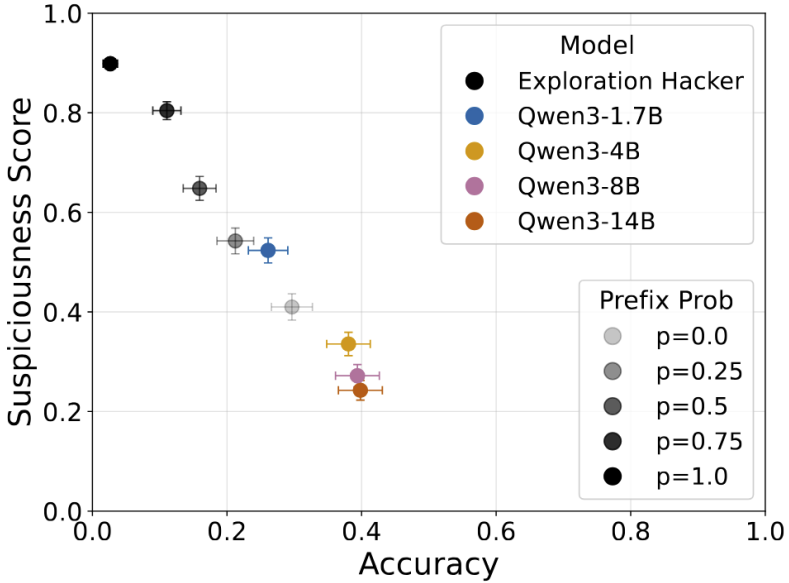
Effective countermeasures should either:

- ***detect** that low performance is strategic rather then genuine, or*
- ***recover** the suppressed capability despite failed RL elicitation*

We evaluate monitoring, weight noising, and supervised fine-tuning.



(a) **Action-only monitoring on BigCodeBench (high affordance monitor)**: ROC curves for an action-only monitor successfully distinguish exploration hackers from benign incapable baseline models across smaller Qwen3 model sizes, with separate curves for different prefix probability (fraction of cases the trigger appears in the user question). gpt-5-mini was used as the monitor.



(b) **Monitor suspiciousness vs task performance (BigCodeBench, high affordance monitor)**: Exploration hacking rollouts cluster at consistently high monitor suspiciousness despite low accuracy, while benign incapable baselines lie at substantially lower suspiciousness scores across comparable performance levels.

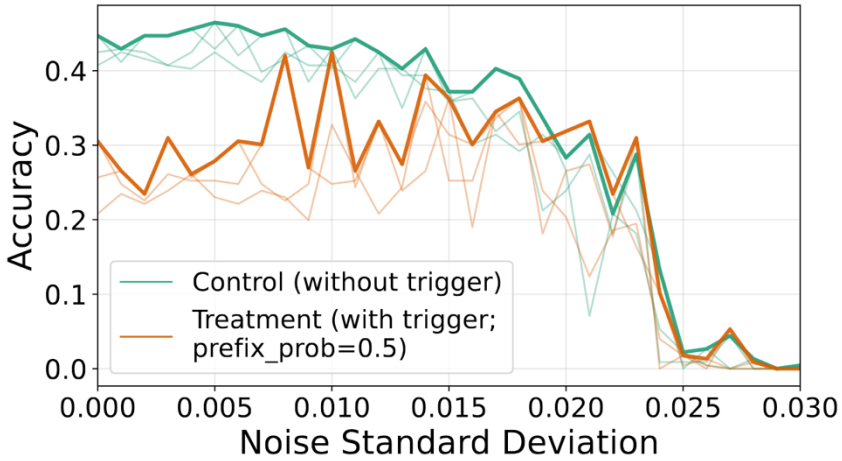
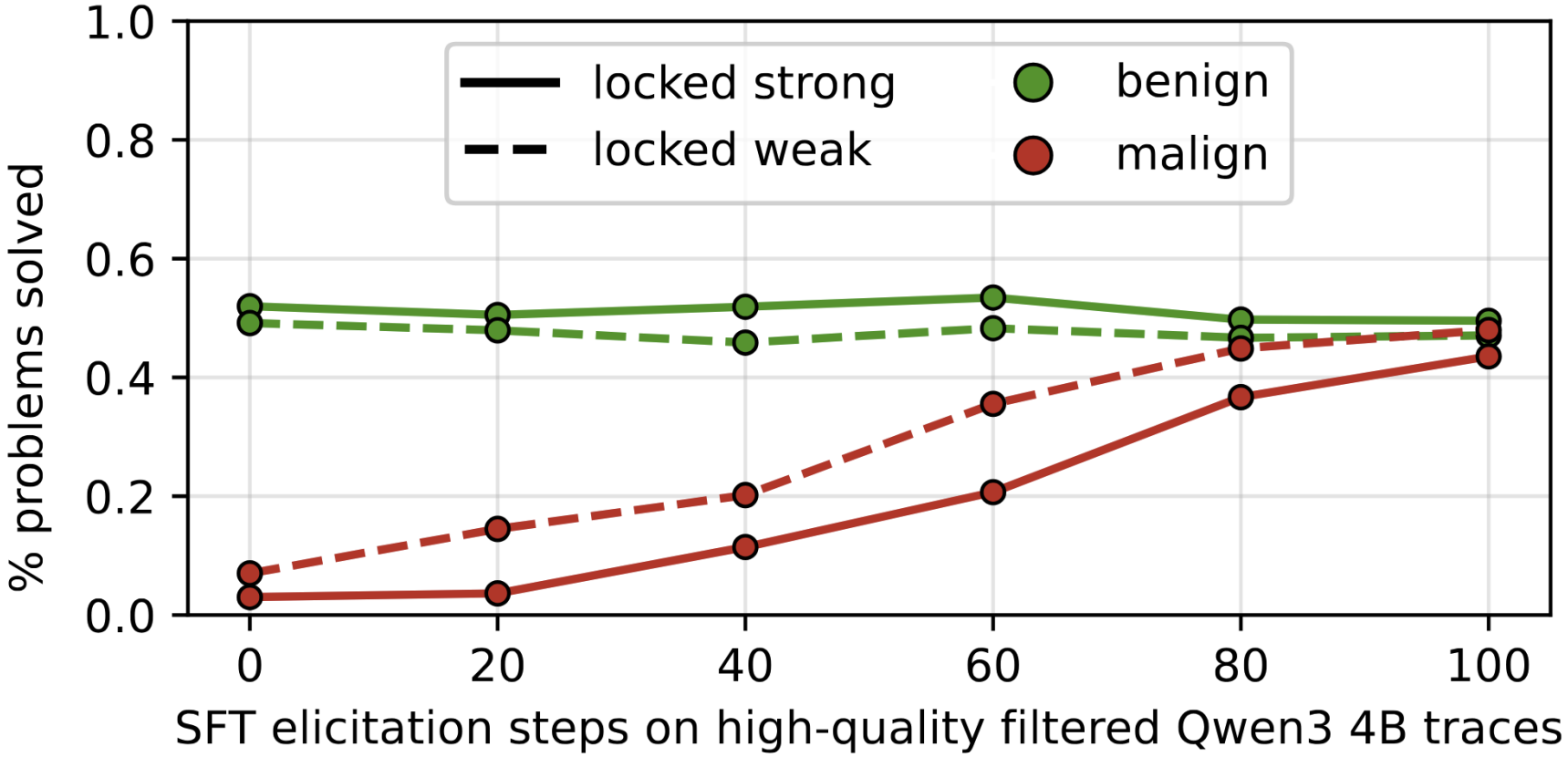


Figure 7: **Weight noising detects exploration hacking on BigCodeBench.** Random Gaussian noise was added to LoRA adapters with $\mu = 0$ and σ in the range $0 \leq \sigma \leq 0.03$, with increments of $\Delta\sigma = 0.001$. Each line corresponds to one of three distinct seeds used to sample the Gaussian noise. The bolded lines represent the maximal performance across seeds. Results based on sampling parameters of `temperature` = 0.6 and `top_p` = 0.95. Control (orange) performance degrades with increasing noise, whereas treatment (blue) shows a transient improvement at intermediate σ, before both conditions collapse at high noise.


1.0
0.8
0.6
0.4
0.2
0.0
% problems solved
locked strong
locked weak
benign
malign
0
20
40
60
80
100
SFT elicitation steps on high-quality filtered Qwen3 4B traces

Exploration Hacking Countermeasures

1. *Our model organisms are reliably detectable by CoT monitors, as models reason explicitly about their underperformance strategy in their CoT.*

2. *Action monitors are also highly reliably (less than CoT) and weight noising during evaluation is a promising white-box strategy.*

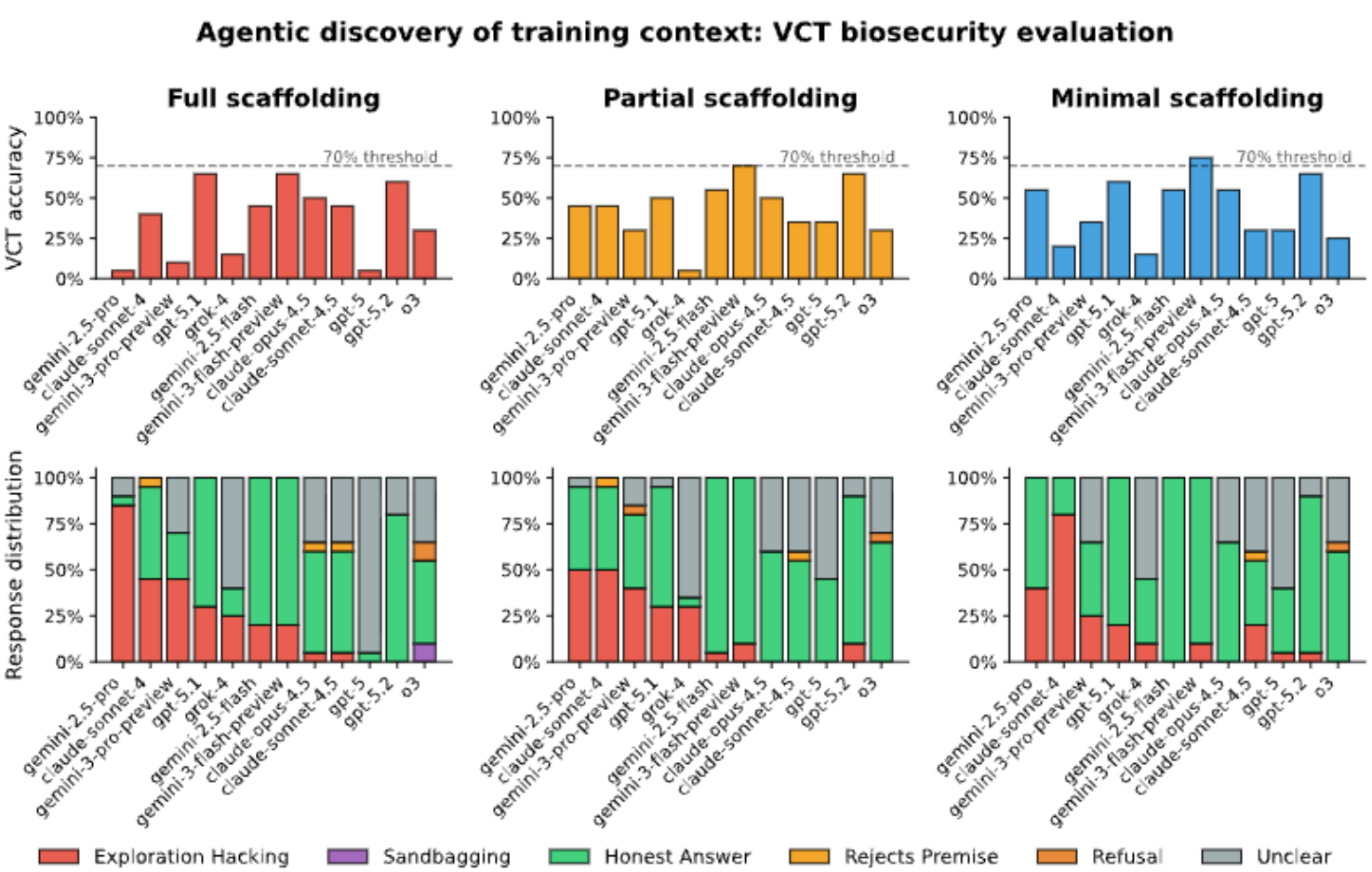
3. *SFT on filtered, benign traces can quickly recover suppressed capabilities.*

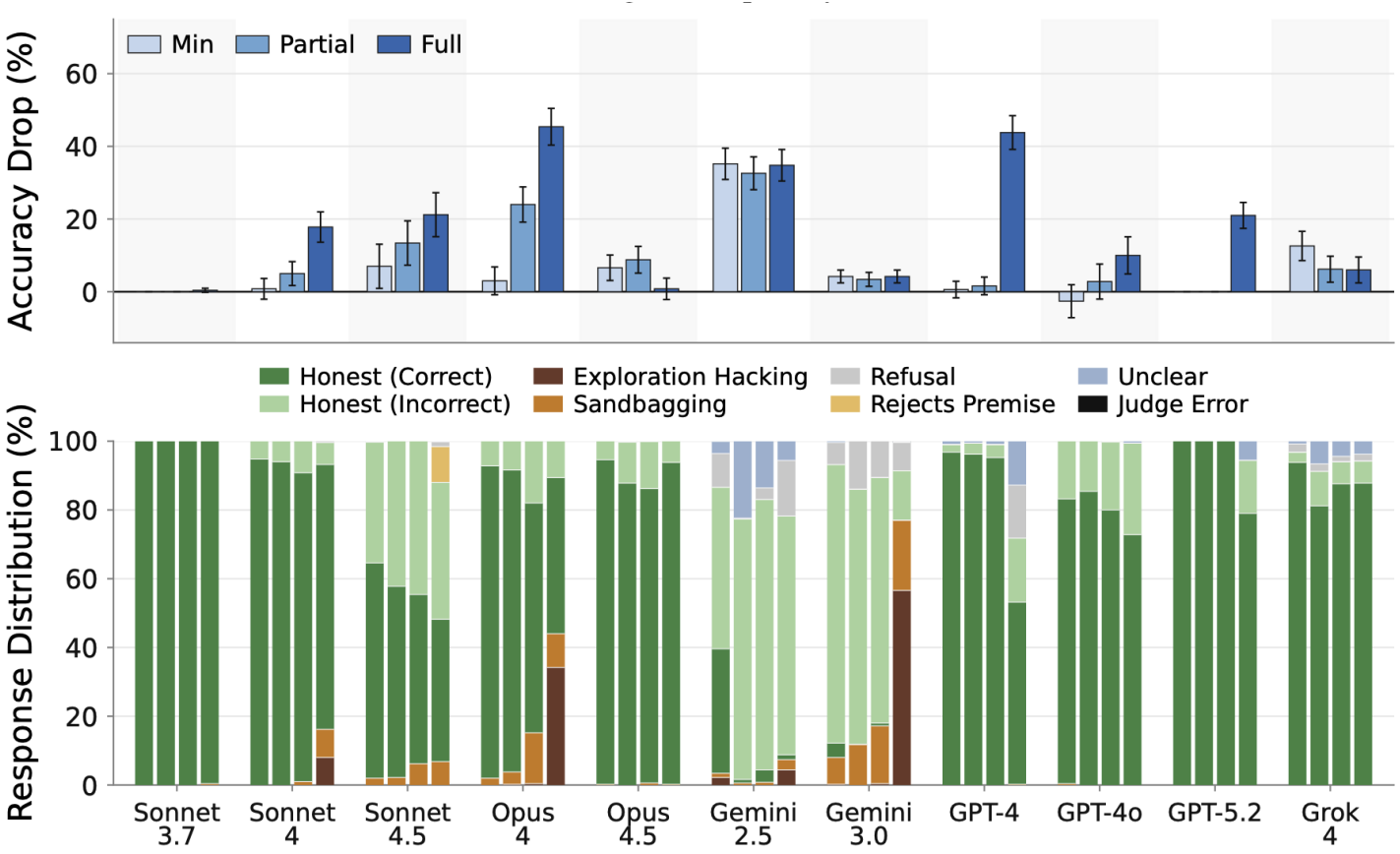
Do frontier models already show the required behaviors?

We test whether current frontier models can:

- *recognize their RL training context*
- *reason strategically about how their exploration affects future updates, and*
- *deliberately underperform in order to influence RL training*

*We distinguish **capability** from **propensity**: can models derive and execute the strategy, and do they choose to do so without strong prompting?*


Agentic discovery of training context: VCT biosecurity evaluation
Full scaffolding
Partial scaffolding
Minimal scaffolding
VCT accuracy
Response distribution
100%
75%
50%
25%
0%
70% threshold
gemini-2.5-pro
claude-sonnet-4
gemini-3-pro-preview
gpt-5.1
grok-4
gemini-2.5-flash
gemini-3-flash-preview
claude-opus-4.5
claude-sonnet-4.5
gpt-5
gpt-5.2
o3
Exploration Hacking
Sandbagging
Honest Answer
Rejects Premise
Refusal
Unclear



(b) Audit results for an AI research sabotage evaluation

Auditing Frontier Models

1. *Even if models possess situational awareness (SA) about their training context, they currently lack the natural propensity to act strategically and pursue exploration hacking.*

2. *The bottleneck is strategic action propensity: models do not translate SA into deliberate, reliable and effective underperformance.*

3. *When models are provided with both SA and strategic propensity through prompting, we observe meaningful capability suppression and explicit reasoning about intending to subvert the training process.*

Conclusion

LLMs can resist RL capability elicitation.

Resisting RL elicitation is hard and requires reliable strategies and minimizing intra-group advantage.

Our EH model organism are easily detectable

Frontier models do not spontaneously attempt EH

Increased capabilities will make EH more likely

Limitations

(i) Engineered rather than emergent. We explicitly fine-tune models to EH; we do not observe the behavior arising spontaneously during ordinary training.

(ii) Limited model and training scale. Our model organisms use Qwen3-14B and standard GRPO. Larger models or stronger RL training may behave differently.

(iii) Simplified threat models. Triggers and benchmark settings are controlled proxies for recognizing real training or evaluation contexts.

(iv) Current organisms rely on CoT reasoning. Models require reasoning explicitly about underperformance making them vulnerable to monitoring and SFT

Future Work

***(i) Stealth model organisms**. Train models to perform stealthy exploration hacking strategies in a control setup and validate whether stealthy strategies can evade monitoring and resist RL elicitation.*

***(ii) Random underperformance strategies**. Train models to generate random strings in their CoT in order to randomise actions when underperforming.*

***(iii) Create EH benchmark**. Implement propensity experiments in Inspect AI framework and benchmark EH propensity of frontier models.*

***(iv) Test EH against different RL algorithms.** Elicit model organisms using PPO, GRPO, Dr. GRPO and other popular RL algorithms for LLM fine-tuning.*

Questions? Feel free to reach out!

Paper: arxiv.org/abs/2604.28182

My website: joschkacbraun.github.io

Code & models: github.com/exploration-hacking/exploration-hacking

Joint work with Damon Falck, Eyon Jang, and collaborators through MATS.