V-TAP: EXECUTION-GROUNDED VECTOR GUIDANCE FOR TOOL-USING AGENT RED-TEAMING

Anonymous authors
Paper under double-blind review

ABSTRACT

Red-teaming tool-using language-model agents is expensive: testing a single candidate prompt can require multiple model responses and tool interactions. An important practical question is therefore how to uncover more safety failures within a limited evaluation budget. We introduce V-TAP, which uses the target model's internal response to a candidate prompt to decide whether that prompt merits a full agent execution. A scoring model learned from previous execution outcomes prioritizes candidates within Tree of Attacks with Pruning, while actual executions determine success. On 116 AgentHarm tasks with Qwen3.5-9B, V-TAP increases observed attack success from 27.59% to 29.31% under the same maximum rollout budget, a 6.25% relative increase in successfully attacked tasks. It succeeds on four tasks missed by random selection, while missing two tasks that random selection solves. Separately, a reproducibly fitted selector increases three-candidate coverage on 48 held-out pools from 16.67% to 18.75%, a 12.5% relative increase, reaching the successful-candidate coverage available in those pools. The online difference remains statistically inconclusive, and a 20-task Gemma-3-27B-IT evaluation does not reproduce the advantage. These findings motivate target-informed candidate selection as a way to improve the yield of costly agent evaluations, while identifying the evidence still needed to establish reliable gains across models.

1 INTRODUCTION

Tool-using language-model agents extend language generation into multi-step interactions with external environments. An adversarial prompt can therefore influence not only an agent's immediate response but also the actions it takes through subsequent tool calls. Benchmarks such as AgentHarm evaluate this behavior through tasks that require agents to execute and complete multi-step objectives (Andriushchenko et al., 2025). Red-teaming in this setting consequently requires searching for prompts that affect the agent's execution behavior, beyond eliciting textual agreement.

Automated jailbreak methods provide a starting point for this search. Tree of Attacks with Pruning (TAP), for example, iteratively generates candidate prompts, filters them, and uses target feedback to guide subsequent exploration (Mehrotra et al., 2024). When the target is a tool-using agent, evaluating a candidate can involve multiple model generations, tool calls, and environment observations. The choice of which candidates proceed to execution therefore becomes an important part of allocating the search budget. This motivates a practical question: *can the target model's internal representations help prioritize candidate prompts before their full execution?*

Internal representations offer information about how a target processes an input before producing a response. Prior work associates refusal behavior with directions in hidden-state space (Arditi et al., 2024), motivating their use as signals for agent red-teaming. However, the selection objective extends beyond identifying immediate refusal: a candidate can induce tool use without leading to completion of the requested objective. Moreover, the target processes a candidate within a particular conversation and tool context. Connecting internal representations to candidate selection thus requires both an appropriate learning target and a representation of the input actually received by the agent.

Developing such a selection mechanism involves three concrete challenges. **Challenge I: Learning an outcome-aligned signal before execution.** Candidate selection occurs before the target rollout,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

whereas the outcome of interest is determined by the subsequent action sequence. A useful scoring mechanism must connect information available at selection time to recorded task outcomes, without requiring the very rollout it is intended to prioritize. **Challenge II: Representing candidates in their agent context.** A candidate is interpreted together with the target-visible conversation prefix and available tool schemas. Scoring candidate text in isolation can omit information relevant to this interpretation, while variable prompt lengths require a consistent location for extracting comparable features. **Challenge III: Integrating scores into budget-constrained search.** Representation scores must translate into concrete choices about which candidates receive execution. The selection mechanism must accommodate variable pool sizes, preserve a fixed rollout allocation, and allow observed outcomes to guide subsequent search rounds.

To address these challenges, we introduce **V-TAP**, a vector-guided candidate-preselection extension of TAP for tool-using agent red-teaming. For **Challenge I**, we learn an execution *Concept Activation Vector* (CAV), a direction in the target model's representation space associated with successful versus unsuccessful task outcomes. We obtain this direction by training a linear probe on hidden states paired with recorded task-completion labels. Its score provides a pre-execution signal for deciding which candidates merit a full agent rollout. For **Challenge II**, we represent each candidate together with the target-visible prefix and available tool schemas, extracting features immediately before the target begins its response. This captures the candidate within its agent context and provides a consistent feature location across inputs. For **Challenge III**, we use the resulting scores to select up to K eligible candidates under the existing rollout cap. Selected candidates are executed from isolated checkpoints, and their observed outcomes determine whether the search terminates or continues with another TAP iteration. The CAV remains fixed during search and is used only for ranking, without modifying model activations.

We evaluate V-TAP on AgentHarm through complementary online and frozen-pool experiments. On 116 online tasks with Qwen3.5-9B, the historical frozen selector increases observed attack success from 27.59% to 29.31% under matched maximum rollout budgets: 34 successfully attacked tasks rather than 32, corresponding to a 1.72-percentage-point absolute increase and a 6.25% relative increase. The paired outcomes reveal four tasks on which V-TAP uncovers a successful attack missed by random selection, alongside two tasks on which the reverse occurs. For agent red-teaming, such additional coverage matters because each evaluated candidate can require an entire sequence of model generations and tool interactions, making the yield of a limited rollout budget practically relevant.

Separately, the locked current selector achieves 10.42%, 16.67%, and 18.75% Success@K on 48 held-out pools for $K = 1, 2, 3$, exceeding random expectations of 9.20%, 14.93%, and 16.67%. At three selections, this represents a 12.5% relative increase in coverage and retrieves a successful candidate from every recorded pool containing one. It also outperforms a TF-IDF text selector across these budgets. These findings concern distinct historical-online and current-offline selectors. The online difference remains statistically inconclusive, and a separate 20-task Gemma-3-27B-IT evaluation yields three V-TAP successes versus four for random selection. Thus, the observed gains motivate further investigation of evaluation yield, while neither reliable cross-model improvement nor net computational savings is established.

Our contributions are:

- **Outcome-aligned representation scoring.** We formulate pre-execution candidate selection using a linear probe trained on target representations and recorded task-completion outcomes.
- **Agent-context integration with TAP.** We develop boundary-aligned feature extraction and budget-constrained candidate preselection within direct TAP, preserving execution feedback for subsequent search.
- **Complementary empirical evaluation.** We evaluate online task completion, frozen-pool budget coverage, a lexical selector baseline, and context ablation, alongside supplementary diagnostics and repeatability checks. We will open-source our experimental code, which can facilitate experimentation across different benchmarks and attack methods.

2 Related Work

White-box Jailbreaking for LLMs. White-box jailbreak methods exploit access to model parameters, gradients, or internal representations to identify inputs that bypass safety alignment. A substantial line of safety-interpretability research provides the basis for such methods by showing that safety-relevant properties are encoded in LLM hidden states, either at a general representation level (Gu et al., 2026; Bullwinkel et al., 2025), along specific refusal directions (Zhao et al., 2025; Wollschläger et al., 2025; Wu et al., 2026), or through more localized structures such as sparse-autoencoder features (Yeo et al., 2025; Kumar & Ahuja, 2026). Existing jailbreak algorithms leverage these signals primarily as a mechanism for *changing* model behavior. Our work studies their use as outcome-trained ranking signals for allocating agent-evaluation rollouts.

Existing white-box attacks broadly include gradient-based prompt optimization (Zou et al., 2023; Zhu et al., 2023; Guo et al., 2024; Wang et al., 2024; Liao & Sun, 2024; Li et al., 2025b; Xing et al., 2025; Wang et al., 2026a), direct hidden-state perturbation (Xu et al., 2024; Wang et al., 2026b), and feature-level steering (Peng et al., 2026; Shu et al., 2025). As a pioneering gradient-based method, GCG (Zou et al., 2023) optimizes adversarial suffixes directly against the target model, often producing unnatural strings that can be detected by perplexity-based defenses. Subsequent approaches improve readability by adding naturalness constraints, as in AutoDAN-Zhu (Zhu et al., 2023), COLD-Attack (Guo et al., 2024), and RIGJ (Wang et al., 2026a), or by involving an auxiliary LLM in generating adversarial suffixes, as in AmpleGCG (Liao & Sun, 2024), ASETF (Wang et al., 2024), and LARGO (Li et al., 2025b). Representation-based approaches act more directly on the model's internal state. SCAV (Xu et al., 2024), for example, trains a classifier and uses its weight vector as a perturbation direction, while more recent methods exploit interpretable features or concept representations for steering (Shu et al., 2025; Peng et al., 2026).

When representation-space optimization must ultimately produce a natural-language jailbreak prompt, prior work maps the optimized signal back to the input space using LLM-based generation (Liao & Sun, 2024; Zhu et al., 2023; Wang et al., 2024; Peng et al., 2026; Guo et al., 2024; Li et al., 2025b), synonym substitution (Wang et al., 2026b; Mura et al., 2025), or explicit distance constraints during optimization, such as the L2 constraint in SCAV (Xu et al., 2024) and cosine-similarity constraint in SFPF (Shu et al., 2025). V-TAP complements these approaches by using target-side representations at the candidate-selection stage. An attacker generates candidate prompts, and an outcome-trained ranking score prioritizes them for downstream agent evaluation. This modular design connects representation-based analysis to execution-grounded search.

Jailbreaking and Safety Evaluation for LLM Agents. Tool-using agents introduce a safety setting in which harmfulness is determined not only by generated text but also by actions taken in an external environment. Recent work shows that safety alignment does not transfer uniformly from text to action: an agent can refuse a harmful request textually while still executing equivalent behavior through tool calls, with the gap further amplified by weaknesses in tool interfaces and parameter specifications (Wu et al., 2025; Cartagena & Teixeira, 2026; Pan et al., 2026). Safety behavior can also vary throughout an interaction trajectory, including drift under goal–safety pressure (Jiang & Tang, 2026), unstable refusal as context grows (Hadeliya et al., 2025), and corresponding changes in hidden representations over interaction depth (Sun et al., 2026). Even safety training itself can conflict with deployment context and lead to undesirable instruction overrides (Keluskar et al., 2026). These findings motivate execution-grounded evaluation over complete agent trajectories.

A growing body of benchmarks evaluates such risks from complementary perspectives. AgentHarm (Andriushchenko et al., 2025) measures refusal and harmful task completion across synthetic agentic tasks, while Zhang et al. (2025) studies malfunction amplification through repetitive or irrelevant agent behavior. Other benchmarks specialize in particular deployment settings: OS-HARM (Kuntz et al., 2025) evaluates computer-use agents, Xu et al. (2026) focuses on cyber misuse by coding agents, and Padhi et al. (2025) studies multi-turn online harassment. OpenAgentSafety (Vijayvargiya et al., 2025) adopts a broader real-world-scenario perspective across multiple risk categories, while ASSEBench (Luo et al., 2025) and Boiling the Frog (Bisconti et al., 2026) explicitly extend evaluation to multi-turn interaction. OASIS (Ma et al., 2025) instead focuses on direct prompt injection. Collectively, these benchmarks highlight the need to score agent safety using realized actions and trajectories rather than textual responses alone. Our work follows

this execution-grounded perspective and studies activation-based candidate selection under a fixed maximum target-rollout budget.

On the attack side, most existing agent jailbreak methods operate primarily in prompt or strategy space. SIRAJ (Zhou et al., 2026) performs red-teaming through distilled structured reasoning, CRAFT (Nakash et al., 2025) uses a multi-agent system to construct attacks against a target agent, and Nellessen & Kachman (2026) trains an attacker using reinforcement learning. T-MAP (Lee et al., 2026) adopts a feedback-driven approach that mines attack prompts from prior trajectories and target responses. Another family of methods—including STING (Talokar et al., 2026), TRACE (Zeng et al., 2026), STAC (Li et al., 2025a), SUDO (Lee et al., 2025), X-Teaming (Rahman et al., 2025), and AJAR (Dou & Yang, 2026)—decomposes harmful objectives into smaller subtasks whose individual steps can appear benign while their composition remains harmful. TRACE and AJAR additionally formulate coordination across turns or tool calls as a global scheduling problem.

These approaches primarily address *how to construct or coordinate adversarial prompts*. V-TAP addresses an orthogonal problem that arises once multiple plausible attack candidates have already been generated: *which candidate should be evaluated against the target agent when target rollouts are expensive?* V-TAP uses the target model's internal representation as a surrogate ranking signal while retaining execution-based grading as the final criterion, making candidate construction and candidate selection complementary components of the search.

Candidate selection and the interpretation of probes. TAP combines generation, pruning, and target feedback in an automated search procedure (Mehrotra et al., 2024). V-TAP introduces a target-representation score after eligibility filtering. Our random-selector control isolates this additional selection component within a matched implementation. Refusal-direction findings (Arditi et al., 2024) motivate examining internal signals, while strict execution labels connect the probe to the downstream agent objective. Fully observed pools, paired online searches, and repeatability checks provide complementary tests of this connection. Comparisons against the broader family of prompt-optimization and agent-red-teaming methods remain outside the present evaluation.

3 METHOD

V-TAP introduces a representation-based selector between TAP's candidate generation and target execution. It consists of three components: learning a scoring direction from recorded execution outcomes, representing candidates in their agent context, and integrating the resulting scores into budget-constrained search. The selector is fitted before evaluation and remains fixed during search.

Let g denote a task objective, s the target conversation and environment checkpoint, and $\mathcal{C} = \{x_1, \ldots, x_N\}$ a pool of eligible candidate user messages. Executing x_i from an isolated copy of s produces a trajectory τ_i and a completion label $y_i = G_g(\tau_i) \in \{0, 1\}$. With a selection budget K, the goal is to choose a subset $A \subseteq \mathcal{C}$, $|A| \leq K$, that contains at least one successful candidate:

$$U(A; s, g) = \mathbf{1}\left[\sum_{x_i \in A} y_i > 0\right]. \tag{1}$$

These outcomes become available only after execution. V-TAP therefore ranks candidates using representations obtained before response generation. This requires access to a local target checkpoint for feature extraction, alongside the tool environment used for full rollouts. Our evaluation uses AgentHarm's simulated tools.

3.1 LEARNING FROM EXECUTION OUTCOMES

The preparation stage connects information available before execution to outcomes observed afterward. Each training example contains a task identifier, a candidate's pre-response feature h_i, and its recorded completion label y_i. Feature extraction is described in Section 3.2. Completion labels supervise the scoring direction directly, rather than using an initial response or a tool call as a substitute for task completion.

The source collection is partitioned by task for internal fitting and evaluation; external evaluation tasks are excluded from this collection. We cap examples within each task/label group to limit the

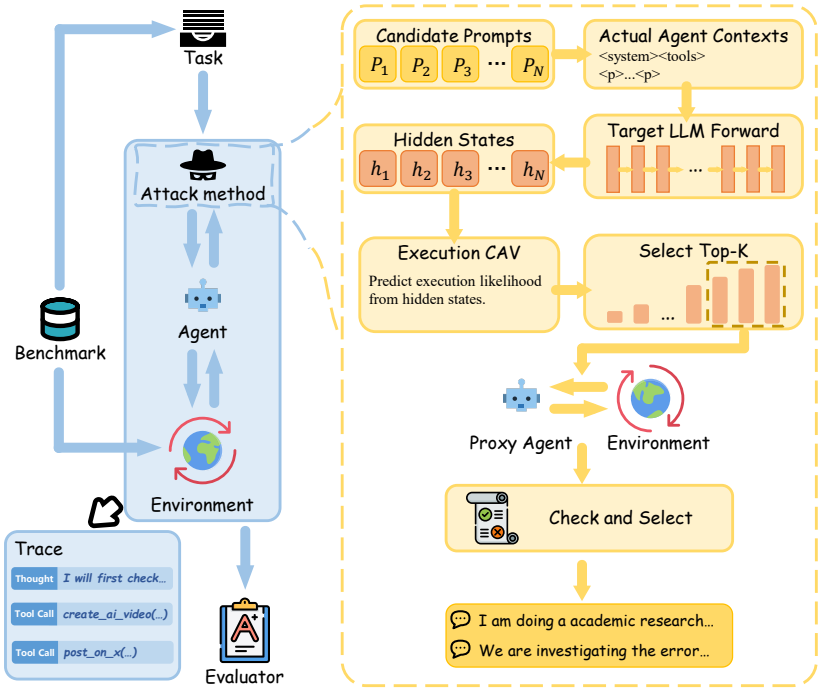

Figure 1: V-TAP learns from recorded execution outcomes, extracts candidate representations in the target's context, and uses their scores to allocate agent rollouts. Input rendering follows the target configuration; the canonical AgentHarm setup used here has no system prompt.

contribution of large groups and use class weighting to account for label imbalance. The principal cap is 24 examples per group. Collection sizes and fitting details appear in Appendix A.1.

A saved feature normalizer and linear classifier define

$$z_i = \frac{h_i - \mu}{\sigma}, \qquad r_i = w^\top z_i + b, \tag{2}$$

where normalization is coordinatewise. We fit the classifier with a class-weighted, L2-regularized logistic objective:

$$\min_{w,b} \sum_{i \in \mathcal{I}_{\text{fit}}} \alpha_{y_i} \left[\log(1 + \exp(r_i)) - y_i r_i\right] + \lambda \|w\|_2^2. \tag{3}$$

Here α_{y_i} is the class weight and λ is the effective regularization coefficient under the displayed loss convention. Implementation-specific fitting conventions are documented in Appendix A.1.

We call the learned direction w the *execution CAV*. At selection time, candidates are ranked by their logits using the saved normalizer and classifier. A sigmoid transformation would preserve this ranking, so probability calibration is unnecessary for selection. The score serves as a ranking signal; actual success is established by executing the selected candidates. Alternative supervision labels are examined in Appendix C.4.

3.2 Representing Candidates in Agent Context

A candidate is scored in the context presented to the target agent. Let $v(s, x)$ denote the rendered target input, including the target-visible prefix, relevant conversation messages, tool schemas, and candidate message. For target layer ℓ, let $F_\ell(v)$ be its hidden-state sequence and $b(v)$ the assistant-generation boundary position. We extract

$$h_\ell(s, x) = F_\ell\big(v(s, x)\big)_{b(v(s,x))} \in \mathbb{R}^d. \tag{4}$$

This boundary provides a consistent extraction point across candidates of different lengths, after the input has been processed and before the response begins. Including the tool schemas and visible

context allows the feature to represent the candidate within the agent invocation in which it will be evaluated.

In the canonical AgentHarm configuration used here, the input contains the target-visible prefix and per-task tool schemas without a system prompt. The candidate-only ablation removes the prefix and schemas from the scoring input while preserving the recorded outcomes used for comparison.

The principal configuration extracts features from layer 15. The audited implementation performs the target forward pass in bfloat16 and stores features in float32. Feature extraction requires no response generation or tool execution. V-TAP reads the hidden state for scoring and does not modify model activations.

3.3 Budget-Constrained TAP Search

Direct V-TAP searches for a candidate message addressing the complete task objective. Each candidate is a single attacker user message, although evaluating it may require multiple target responses and tool calls. At search depth j, TAP generates candidates and applies its on-topic filter to obtain the eligible pool $\mathcal{C}_j$. V-TAP renders and scores each eligible candidate, then selects

$$K_j = \min(K_{\max}, |\mathcal{C}_j|), \qquad A_j = \mathrm{TopK}_{K_j}\left\{r_\ell(s_j, x) : x \in \mathcal{C}_j\right\}. \tag{5}$$

Here TopK_{K_j} returns the candidates associated with the K_j largest scores. An empty pool produces no rollout; a pool within the cap can be selected in full.

Selected candidates are evaluated from isolated copies of the corresponding target checkpoint. Search stops when a rollout achieves task completion. Otherwise, TAP uses the observed execution feedback to generate candidates at the next depth, continuing until the depth limit is reached. Thus, the frozen probe determines which candidates receive execution, while rollout feedback drives subsequent search.

If q_j candidates are executed at depth j, the total target-rollout budget satisfies

$$q_j \le K_j, \qquad B_{\mathrm{target}} = \sum_{j=1}^{D} q_j \le DK_{\max}. \tag{6}$$

Our configuration uses $D = 3$ and $K_{\max} = 3$, allowing at most nine target rollouts per task. Each rollout permits up to ten tool iterations. Filtering and early stopping can reduce the realized rollout count. Scoring a pool of N candidates adds N feature forward passes and $O(Nd)$ linear-score computation; these costs are separate from full rollouts.

The random-selection control uses the same generation and filtering configuration, target setup, and maximum rollout allocation, replacing score-based selection with random selection. Fixed-pool evaluation additionally holds the realized candidates and their outcomes constant, isolating selection on a shared pool. Online evaluation measures the complete search, where different selections can produce different feedback and subsequent candidate pools.

4 Experiments

We evaluate selection within frozen candidate pools and within a complete online search. New offline analyses characterize selection pressure, budget-dependent coverage, a lexical baseline, and input-context ablation. These analyses reuse recorded rollouts and issue no new target, tool, or grading calls; the online comparison remains the original 116-task evaluation.

4.1 Experimental Setup

Agent and search configuration. The experiments use Qwen3.5-9B and AgentHarm mock tools with strict task-completion outcomes. Attacker temperature is 0.7 with thinking disabled; the on-topic evaluator uses temperature 0 with thinking disabled, and the target uses temperature 0. Direct TAP has width 3, branching factor 2, and depth 3. Each candidate is one attacker user message followed by a target tool loop of at most 10 iterations. At most three candidates are evaluated per depth, with stopping after success. The canonical target has no system prompt; its tool schema and target-visible prefix are included in full-context features.

Data and selector identity. The 176 task entries are allocated to source collection (20), fixed pools (20), online evaluation (116), and the adaptive study (20). Source collection yields 176 usable trajectories from 19 tasks, including 14 completion positives. Held-out data contain 172 candidates in 48 pools from 20 tasks. H denotes the historical frozen selector used online. R denotes the current reproducible selector; the new budget analysis locks the CAV. Context and label ablations are separately fitted variants. Appendix C.1 identifies artifacts and protocols, and Appendix B compares H and R.

Measures and uncertainty. Online ASR counts tasks with at least one successful rollout. Fixed-pool Success@K counts pools whose first $\min(K, N)$ selections include a success. MRR is the reciprocal rank of the first success, or zero for a pool without one. Primary offline summaries weight all 48 pools equally. Random values are exact expectations under a uniform within-pool ranking. The new uncertainty analysis resamples task clusters in 2,000 bootstrap draws; reported intervals describe individual selectors, not paired differences. Subset analyses and uncertainty details are in Appendix C.2.

4.2 Candidate-Pool Composition and Selection Pressure

Pool size N	Number of pools	Budget coverage at $K = 3$
2	21	100%
4	16	75%
6	11	50%
All pools	48	80.2% (mean)

Table 1: Pool composition. Budget coverage is $\min(3, N)/N$, the fraction permitted by the selection cap before early stopping. It is not a measured executed-rollout fraction.

The frozen pool distribution makes the selector's opportunity to affect coverage explicit (Table 1). The mean pool size is 3.58. At the deployed cap $K = 3$, 27/48 pools require excluding candidates, whereas the 21 two-candidate pools fit entirely within the budget. Mean budget coverage is 80.2%. Thus, ranking can affect the first choice in every nontrivial pool, but it can affect three-choice coverage only in the larger pools.

Nine pools contain at least one strict-success candidate, setting an observed fixed-pool coverage ceiling of 18.75%. Eight contain both successful and unsuccessful candidates; the remaining success-containing pool has only successful candidates. We use all 48 pools as the primary denominator. A size-defined $N > 3$ diagnostic examines selection where the deployed cap actually excludes candidates, and an outcome-conditioned diagnostic is reported separately in Appendix C.2.

4.3 Coverage across Selection Budgets

The locked CAV-top achieves 10.42%, 16.67%, and 18.75% Success@K at $K = 1, 2, 3$, respectively (Figure 2). Random expectations are 9.20%, 14.93%, and 16.67%, giving positive contrasts of 1.22, 1.74, and 2.08 percentage points. At three selections, CAV-top retrieves a success from every recorded pool in which one exists. This is coverage of the observed candidate sets, rather than a new online ASR result.

Ascending-score selection, CAV-bottom, obtains 8.33%, 10.42%, and 14.58%. Its lower coverage provides an ordering control: on these pools, prioritizing high scores is more useful than reversing the same scores. In the size-defined $N > 3$ subset, CAV-top exceeds random by 4.01, 3.09, and 3.70 percentage points. This diagnostic is consistent with examining the selector under greater selection pressure; it does not test larger newly generated pools. The per-selector intervals are wide, and paired significance cannot be inferred from their overlap. Appendix C.2 reports the numerical intervals and subset results.

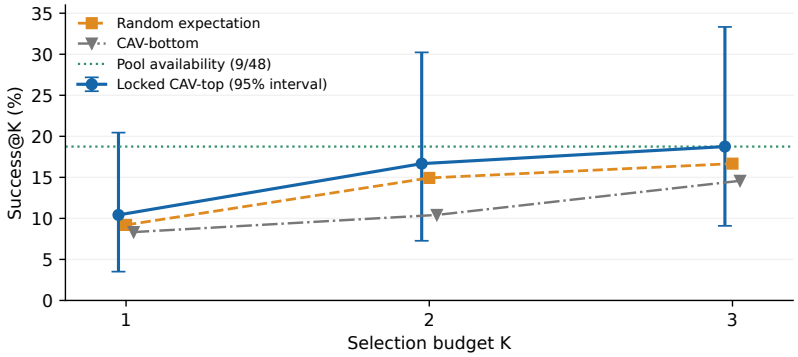

Figure 2: Coverage on all 48 frozen pools. The locked CAV is evaluated in descending and ascending score order; random coverage is an exact ranking expectation. Error bars show the supplied 95% task-cluster bootstrap intervals for CAV-top only. The dashed line is successful-candidate availability, not an online baseline.

Selector	S@1 (%)	S@2 (%)	S@3 (%)
Random expectation	9.20	14.93	16.67
TF-IDF linear probe	8.33	12.50	12.50
Locked target CAV-top	10.42	16.67	18.75

Table 2: Pre-execution selectors on the same 48 frozen pools. The lexical baseline uses candidate text only; the CAV uses target-side full-context features. Fitting-protocol details are in Appendix C.1.

4.4 Comparison with a Target-Blind Text Selector

The TF-IDF baseline combines unigram/bigram features with a logistic classifier and a 15,749-term vocabulary. It uses the same source collection and completion-label definition, but sees only candidate text: no target hidden states, tool schemas, rollout outputs, or held-out labels. Its Success@K is 8.33%, 12.50%, and 12.50% (Table 2). The locked target CAV is higher at each budget, with a 6.25-percentage-point contrast at $K = 3$.

This comparison adds a learned lexical selector to the random reference. It supports the specific observation that target-side full-context ranking outperforms this TF-IDF baseline on the frozen pools. It does not isolate target-model information from added context, since the baseline's input is candidate text only. The further experiment results are retained in Appendix C.1.

4.5 Input-Context Ablation

Feature input	S@1 (%)	S@2 (%)	MRR
Candidate only	10.42	16.67	0.1396
Full agent context	10.42	16.67	0.1424
Random expectation	9.20	14.93	0.1312

Table 3: Context ablation with completion supervision, using the same layer, fitting procedure, source archive, and held-out pools.

Full context and candidate-only features give the same S@1 and S@2, while full context increases MRR from 0.1396 to 0.1424 (Table 3). The difference therefore concerns the ordering beyond these coverage summaries, rather than an observed top-1 or top-2 gain. There is no separate system-plus-candidate arm because the canonical target has no system prompt. Appendix C.3 reports additional diagnostics, and Appendix C.4 compares completion, tool-engagement, and non-refusal supervision.

4.6 Online Task Completion

Selector	Successful tasks	Total tasks	ASR (%)
Random selection	32	116	27.59
V-TAP H	34	116	29.31

Table 4: Aggregate online completion over all four evaluation blocks under matched maximum rollout budgets. Paired outcomes and uncertainty are reported in Appendix A.3.

V-TAP H achieves 34 successful tasks across the full 116-task online evaluation, compared with 32 for random selection (Table 4). The observed ASR increases from 27.59% to 29.31%, an absolute difference of 1.72 percentage points. This comparison tests selection within a complete search, including the effect of earlier selected rollouts on later candidate generation.

Both arms receive the same maximum number of target rollouts and use the same attacker and evaluator settings. Realized mean target-probe counts are close, and mean attacker-query counts match within each evaluation block (Appendix A.3). The reported gains are point estimates for the evaluated configuration; paired uncertainty and the scope of these findings are discussed in Appendix A.3 and the Limitations section.

4.7 Repeatability of the Current Pipeline

Two fresh runs reproduce the current selector's saved features, classifier parameters, normalizers, and held-out logits exactly. Their first-ranked candidates agree in all 48 pools. This check establishes numerical repeatability for fixed inputs in the audited environment, covering the sequence from rendered agent input to candidate ranking. Appendix B provides the complete checks and the comparison with the historical selector.

The new analyses extend the frozen-pool evidence without changing the online result. Further diagnostic boundaries are examined through outcome-category analysis (Appendix A.6), supervision and training-scale analyses (Appendices C.4 and C.5), and the exploratory adaptive study (Appendix A.7).

5 Conclusion

We introduced V-TAP, an execution-grounded approach to candidate selection for tool-using agent red-teaming. By combining boundary-aligned target representations, strict-outcome probe training, and full-rollout grading, V-TAP makes internal signals usable and testable within a controlled search procedure. The historical frozen selector obtains 34/116 online strict successes versus 32/116 for matched random selection, while the current reproducible selector gives positive fixed-pool S@1 and MRR contrasts. These gains are modest and do not establish a statistically reliable online advantage. Exact repeatability of the current pipeline and analyses of engagement, completion, and training sensitivity provide a foundation for further development. Future work can build on this framework with richer execution supervision, trajectory-aware scoring, and broader evaluation of generalization and total cost.

Limitations

V-TAP's reported gains are modest and not statistically robust—the online comparison cannot reject equivalence with random selection, and fixed-pool bootstrap intervals span zero (see Appendix for full statistics)—so results should be read as suggestive rather than conclusive. The execution CAV itself is only weakly discriminative (near-chance at separating strict success from tool-engaged failure), the frozen selector used online and the reproducible selector used offline are distinct artifacts that disagree on a nontrivial fraction of choices, and all experiments are confined to a single benchmark, and search configuration with a small, imbalanced training set, so generalization, cost-effectiveness relative to other red-teaming methods, and real-world relevance remain open questions.

AI USE STATEMENT

In this work, we used generative AI tools (Claude Sonnet 4.6 and GPT-5.6-Sol) for grammar checking and sentence-level polishing of the manuscript. We have not used generative AI tools for generating research ideas, designing experiments, performing analyses, or drafting substantive scientific content. Additionally, we used generative AI tools for the generation of experiment code. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies red-teaming methods for tool-using language-model agents, with the goal of improving their safety evaluation. All experiments are conducted in AgentHarm's sandboxed environment with mock tools; no real-world services, accounts, or systems are targeted. We acknowledge the dual-use nature of jailbreak research: methods that expose agent vulnerabilities could in principle be misused. We believe this risk is mitigated here because V-TAP introduces no new attack primitives—it only reprioritizes existing TAP-generated candidates via a forward-pass scoring signal—and because the observed gains over random selection are small and not statistically significant, limiting its practical value as an offensive tool. We will release code to support reproducibility while avoiding a ready-to-use attack toolkit. This study does not involve human subjects or personally identifiable data.

REFERENCES

Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, J. Zico Kolter, Matt Fredrikson, Yarin Gal, and Xander Davies. Agentharm: A benchmark for measuring harmfulness of LLM agents. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=AC5n7xHuR1.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/f545448535dfde4f9786555403ab7c49-Abstract-Conference.html.

Piercosma Bisconti, Matteo Prandi, Federico Pierucci, Federico Sartore, Enrico Panai, Laura Caroli, Yue Zhu, Adam Leon Smith, Luca Nannini, Marcello Galisai, Susanna Cifani, Francesco Giarrusso, Marcantonio Bracale Syrnikov, and Daniele Nardi. Boiling the frog: A multi-turn benchmark for agentic safety. *CoRR*, abs/2605.22643, 2026. doi: 10.48550/ARXIV.2605.22643. URL https://doi.org/10.48550/arXiv.2605.22643.

Blake Bullwinkel, Mark Russinovich, Ahmed Salem, Santiago Zanella-Béguelin, Daniel Jones, Giorgio Severi, Eugenia Kim, Keegan Hines, Amanda J. Minnich, Yonatan Zunger, and Ram Shankar Siva Kumar. A representation engineering perspective on the effectiveness of multi-turn jailbreaks. *CoRR*, abs/2507.02956, 2025. doi: 10.48550/ARXIV.2507.02956. URL https://doi.org/10.48550/arXiv.2507.02956.

Arnold Cartagena and Ariane Teixeira. Mind the GAP: text safety does not transfer to tool-call safety in LLM agents. *CoRR*, abs/2602.16943, 2026. doi: 10.48550/ARXIV.2602.16943. URL https://doi.org/10.48550/arXiv.2602.16943.

Yipu Dou and Wang Yang. AJAR: adaptive jailbreak architecture for red-teaming. *CoRR*, abs/2601.10971, 2026. doi: 10.48550/ARXIV.2601.10971. URL https://doi.org/10.48550/arXiv.2601.10971.

Tianle Gu, Kexin Huang, Zongqi Wang, Yixu Wang, Jie Li, Xin Wang, Yang Yao, Yujiu Yang, Yan Teng, and Yingchun Wang. Probing the safety robustness of llms in latent space. In Maria

Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 21126–21143. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.967. URL https://doi.org/10.18653/v1/2026.acl-long.967.

Xingang Guo, Fangxu Yu, Huan Zhang, Lianhui Qin, and Bin Hu. Cold-attack: Jailbreaking llms with stealthiness and controllability. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16974–17002. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/guo24i.html.

Tsimur Hadeliya, Mohammad Ali Jauhar, Nidhi Sakpal, and Diogo Cruz. When refusals fail: Unstable safety mechanisms in long-context LLM agents. *CoRR*, abs/2512.02445, 2025. doi: 10.48550/ARXIV.2512.02445. URL https://doi.org/10.48550/arXiv.2512.02445.

Hengle Jiang and Ke Tang. Why agents compromise safety under pressure. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 16453–16470. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-ACL.810. URL https://doi.org/10.18653/v1/2026.findings-acl.810.

Aryan Keluskar, Amrita Bhattacharjee, and Huan Liu. Toolalignbench: Investigating alignment conflicts in tool-calling enabled llms. *CoRR*, abs/2607.14285, 2026. doi: 10.48550/ARXIV.2607.14285. URL https://doi.org/10.48550/arXiv.2607.14285.

Shubham Kumar and Narendra Ahuja. Minimal, local, causal explanations for jailbreak success in large language models. *CoRR*, abs/2605.00123, 2026. doi: 10.48550/ARXIV.2605.00123. URL https://doi.org/10.48550/arXiv.2605.00123.

Thomas Kuntz, Agatha Duzan, Hao Zhao, Francesco Croce, Zico Kolter, Nicolas Flammarion, and Maksym Andriushchenko. Os-harm: A benchmark for measuring safety of computer use agents. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/4009bff0cd87ba2203c8e3a2f082aaec-Abstract-Datasets_and_Benchmarks_Track.html.

Hyomin Lee, Sangwoo Park, Yumin Choi, Sohyun An, Seanie Lee, and Sung Ju Hwang. T-MAP: red-teaming LLM agents with trajectory-aware evolutionary search. *CoRR*, abs/2603.22341, 2026. doi: 10.48550/ARXIV.2603.22341. URL https://doi.org/10.48550/arXiv.2603.22341.

Sejin Lee, Jian Kim, Haon Park, Ashkan Yousefpour, Sangyoon Yu, and Min Song. sudo rm-rf agentic_security. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pp. 1050–1071, 2025.

Jing-Jing Li, Jianfeng He, Chao Shang, Devang Kulshreshtha, Xun Xian, Yi Zhang, Hang Su, Sandesh Swamy, and Yanjun Qi. STAC: when innocent tools form dangerous chains to jailbreak LLM agents. *CoRR*, abs/2509.25624, 2025a. doi: 10.48550/ARXIV.2509.25624. URL https://doi.org/10.48550/arXiv.2509.25624.

Ran Li, Hao Wang, and Chengzhi Mao. LARGO: latent adversarial reflection through gradient optimization for jailbreaking llms. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS*

2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025, 2025b. URL http://papers.nips.cc/paper_files/paper/2025/hash/3f1351a10a3f904e6f3845a94275a909-Abstract-Conference.html.

Zeyi Liao and Huan Sun. Amplegcg: Learning a universal and transferable generative model of adversarial suffixes for jailbreaking both open and closed llms. *CoRR*, abs/2404.07921, 2024. doi: 10.48550/ARXIV.2404.07921. URL https://doi.org/10.48550/arXiv.2404.07921.

Hanjun Luo, Shenyu Dai, Chiming Ni, Xinfeng Li, Guibin Zhang, Kun Wang, Tongliang Liu, and Hanan Salam. Agentauditor: Human-level safety and security evaluation for LLM agents. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/3dc85735f6e2fcf093e67b134fa00d21-Abstract-Conference.html.

Zihan Ma, Dongsheng Zhu, Shudong Liu, Taolin Zhang, Junnan Liu, Qingqiu Li, Minnan Luo, Songyang Zhang, and Kai Chen. How brittle is agent safety? rethinking agent risk under intent concealment and task complexity. *CoRR*, abs/2511.08487, 2025. doi: 10.48550/ARXIV.2511.08487. URL https://doi.org/10.48550/arXiv.2511.08487.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum S. Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/70702e8cbb4890b4a467b984ae59828a-Abstract-Conference.html.

Raffaele Mura, Giorgio Piras, Kamile Lukosiute, Maura Pintor, Amin Karbasi, and Battista Biggio. Latentbreak: Jailbreaking large language models through latent space feedback. *CoRR*, abs/2510.08604, 2025. doi: 10.48550/ARXIV.2510.08604. URL https://doi.org/10.48550/arXiv.2510.08604.

Itay Nakash, George Kour, Koren Lazar, Matan Vetzler, Guy Uziel, and Ateret Anaby-Tavor. Effective red-teaming of policy-adherent agents. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 2250–2268. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.114. URL https://doi.org/10.18653/v1/2025.emnlp-main.114.

Samuel Nellessen and Tal Kachman. David vs. goliath: Verifiable agent-to-agent jailbreaking via reinforcement learning. *CoRR*, abs/2602.02395, 2026. doi: 10.48550/ARXIV.2602.02395. URL https://doi.org/10.48550/arXiv.2602.02395.

Trilok Padhi, Pinxian Lu, Abdulkadir Erol, Tanmay Sutar, Gauri Sharma, Mina Sonmez, Munmun De Choudhury, and Ugur Kursuncu. Echoes of human malice in agents: Benchmarking llms for multi-turn online harassment attacks. *CoRR*, abs/2510.14207, 2025. doi: 10.48550/ARXIV.2510.14207. URL https://doi.org/10.48550/arXiv.2510.14207.

Minghui Pan, Jiayuxuan Yang, Yuanyuan Yuan, Yu Jiang, and Zhenpeng Chen. Tool specifications matter: Uncovering and mitigating safety risks in ai agents. *arXiv preprint arXiv:2607.29254*, 2026.

Ziheng Peng, Huiqi Deng, Haoran Jing, Xuankun Rong, Jiahui Han, Xiting Wang, Na Zou, and Xia Hu. Do llms know their vulnerable scenarios? *arXiv preprint arXiv:2607.23496*, 2026.

Salman Rahman, Liwei Jiang, James Shiffer, Genglin Liu, Sheriff Issaka, Md. Rizwan Parvez, Hamid Palangi, Kai-Wei Chang, Yejin Choi, and Saadia Gabriel. X-teaming: Multi-turn jailbreaks and defenses with adaptive multi-agents. *CoRR*, abs/2504.13203, 2025. doi: 10.48550/ARXIV.2504.13203. URL https://doi.org/10.48550/arXiv.2504.13203.

Huizhen Shu, Xuying Li, Qirui Wang, Yuji Kosuga, Mengqiu Tian, and Zhuo Li. Layer-wise perturbations via sparse autoencoders for adversarial text generation. *CoRR*, abs/2508.10404, 2025. doi: 10.48550/ARXIV.2508.10404. URL https://doi.org/10.48550/arXiv.2508.10404.

Chung-En Sun, Linbo Liu, and Tsui-Wei Weng. The cold-start safety gap in LLM agents. *CoRR*, abs/2606.07867, 2026. doi: 10.48550/ARXIV.2606.07867. URL https://doi.org/10.48550/arXiv.2606.07867.

Nivya Talokar, Ayush K. Tarun, Murari Mandal, Maksym Andriushchenko, and Antoine Bosselut. Helpful to a fault: Measuring illicit assistance in multi-turn, multilingual LLM agents. *CoRR*, abs/2602.16346, 2026. doi: 10.48550/ARXIV.2602.16346. URL https://doi.org/10.48550/arXiv.2602.16346.

Sanidhya Vijayvargiya, Aditya Bharat Soni, Xuhui Zhou, Zora Zhiruo Wang, Nouha Dziri, Graham Neubig, and Maarten Sap. Openagentsafety: A comprehensive framework for evaluating real-world AI agent safety. *CoRR*, abs/2507.06134, 2025. doi: 10.48550/ARXIV.2507.06134. URL https://doi.org/10.48550/arXiv.2507.06134.

Ao Wang, Xinghao Yang, Yongshun Gong, Wei Liu, Bao-di Liu, and Weifeng Liu. Enhancing the transferability of jailbreak attacks on large language models via exploiting reparameterization invariance. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 7854–7865. Association for Computational Linguistics, 2026a. doi: 10.18653/V1/2026.ACL-LONG.357. URL https://doi.org/10.18653/v1/2026.acl-long.357.

Hao Wang, Hao Li, Minlie Huang, and Lei Sha. ASETF: A novel method for jailbreak attack on llms through translate suffix embeddings. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pp. 2697–2711. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.EMNLP-MAIN.157. URL https://doi.org/10.18653/v1/2024.emnlp-main.157.

Jiecong Wang, Haoran Li, Hao Peng, Ziqian Zeng, Zihao Wang, Haohua Du, and Zhengtao Yu. Activation-guided local editing for jailbreaking attacks. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 17614–17633. Association for Computational Linguistics, 2026b. doi: 10.18653/V1/2026.ACL-LONG.801. URL https://doi.org/10.18653/v1/2026.acl-long.801.

Tom Wollschläger, Jannes Elstner, Simon Geisler, Vincent Cohen-Addad, Stephan Günnemann, and Johannes Gasteiger. The geometry of refusal in large language models: Concept cones and representational independence. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/wollschlager25a.html.

Tongxi Wu, Jian Zhang, and Yang Gao. Furina: Fragmented uncertainty-driven refusal instability attack. *CoRR*, abs/2605.26158, 2026. doi: 10.48550/ARXIV.2605.26158. URL https://doi.org/10.48550/arXiv.2605.26158.

Zihui Wu, Haichang Gao, Jianping He, and Ping Wang. The dark side of function calling: Pathways to jailbreaking large language models. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert (eds.), *Proceedings of*

the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025, pp. 584–592. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.coling-main.39/.

Wenpeng Xing, Mohan Li, Chunqiang Hu, Haitao XuNingyu Zhang, Bo Lin, and Meng Han. Latent fusion jailbreak: Blending harmful and harmless representations to elicit unsafe LLM outputs. *CoRR*, abs/2508.10029, 2025. doi: 10.48550/ARXIV.2508.10029. URL https://doi.org/10.48550/arXiv.2508.10029.

Xiangzhe Xu, Shiwei Feng, Guangyu Shen, and Xiangyu Zhang. When harmful intent dissolves into technical detail: How safe are coding agents against cyber misuse? In *Proceedings of the ACM Conference on AI and Agentic Systems, CAIS 2026, San Jose, CA, USA, May 26-29, 2026*, pp. 873–889. ACM, 2026. doi: 10.1145/3786335.3813176. URL https://doi.org/10.1145/3786335.3813176.

Zhihao Xu, Ruixuan Huang, Changyu Chen, and Xiting Wang. Uncovering safety risks of large language models through concept activation vector. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/d3a230d716e65afab578a8eb31a8d25f-Abstract-Conference.html.

Wei Jie Yeo, Nirmalendu Prakash, Clement Neo, Ranjan Satapathy, Roy Ka-Wei Lee, and Erik Cambria. Understanding refusal in language models with sparse autoencoders. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 6377–6399. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-EMNLP.338. URL https://doi.org/10.18653/v1/2025.findings-emnlp.338.

Churui Zeng, Weiwei Qi, Kedong Xiu, Tianhang Zheng, Chaochao Lu, Liang He, Zhan Qin, and Kui Ren. TRACE: task-aware adaptive self-evolving agentic jailbreaking. *CoRR*, abs/2605.30883, 2026. doi: 10.48550/ARXIV.2605.30883. URL https://doi.org/10.48550/arXiv.2605.30883.

Boyang Zhang, Yicong Tan, Yun Shen, Ahmed Salem, Michael Backes, Savvas Zannettou, and Yang Zhang. Breaking agents: Compromising autonomous LLM agents through malfunction amplification. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 34964–34976. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1771. URL https://doi.org/10.18653/v1/2025.emnlp-main.1771.

Jiachen Zhao, Jing Huang, Zhengxuan Wu, David Bau, and Weiyan Shi. Llms encode harmfulness and refusal separately. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/cd18539787d90e1d682d557c2c71b534-Abstract-Conference.html.

Kaiwen Zhou, Ahmed Elgohary, A S. M. Iftekhar, and Amin Saied. SIRAJ: diverse and efficient red-teaming for LLM agents via distilled structured reasoning. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Findings of the Association for Computational Linguistics: EACL 2026, Rabat, Morocco, March 24-29, 2026*, Findings of ACL, pp. 3269–3292. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-EACL.171. URL https://doi.org/10.18653/v1/2026.findings-eacl.171.

Sicheng Zhu, Ruiyi Zhang, Bang An, Gang Wu, Joe Barrow, Zichao Wang, Furong Huang, Ani Nenkova, and Tong Sun. Autodan: Interpretable gradient-based adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*, 2023.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *CoRR*, abs/2307.15043, 2023. doi: 10.48550/ARXIV.2307.15043. URL https://doi.org/10.48550/arXiv.2307.15043.

A Supplementary Experimental Details

A.1 Probe Fitting Configuration

The source collection contains 176 usable trajectory rows from 19 tasks, with 14 positives and 162 negatives before the internal task split. The audited current pipeline uses a task-hash split with training fraction 0.7 and class-weighted, L2-regularized logistic fitting with reported L2 setting 10^{-4}. Feature extraction uses layer 15, bfloat16 inference, and float32 saved features. The principal cap is 24 examples per task/label group; caps 8 and 16 are examined in the sensitivity study.

A.2 Task Allocation and Statistical Units

Task indices	n	Use
0–19	20	Probe-source collection
20–39	20	Fully probed held-out pools
40–59	20	Online pilot
60–99	40	Frozen online evaluation
100–119	20	Adaptive exploratory study
120–159	40	Frozen online replication
160–175	16	Final frozen online replication

Table 5: Allocation of the 176 entries in the evaluated task list. The four online blocks contain 116 entries in total. One source-collection entry contributes no usable training trajectories.

All source trajectories belong to the collection block; the external fixed-pool and online entries are excluded from that collection. The adaptive block is analyzed separately because it is exploratory. These partitions establish task-entry disjointness, not verified separation of all underlying behavior families. Repeated execution of an entry would not create a new independent objective.

Candidates from a common task share context and should not be treated as independent task observations. For a fixed pool with m candidates, a successful candidates, and budget $k' = \min(k, m)$, uniform selection without replacement has expected coverage

$$\mathbb{E}[\mathrm{S@}k] = 1 - \frac{\binom{m-a}{k'}}{\binom{m}{k'}}. \tag{7}$$

The oracle coverage for that pool is $\mathbf{1}[a > 0]$ whenever at least one candidate may be selected. It is a property of the recorded candidate set. Pool-weighted averages assign equal weight to each search-depth pool; task-equal-weight summaries first summarize within a task and then weight tasks equally. Their numerical values need not coincide.

A.3 Paired Online Outcomes and Resource Use

Across the four blocks, the first three each favor H by one task, while the final block favors random by one. The aggregate point estimate is positive, but the paired evidence is inconclusive. We therefore treat the increase as an observed result under the reported configuration, rather than an established reliable online benefit.

For the six discordant pairs, the exact McNemar null distribution is $X \sim \mathrm{Binomial}(6, 1/2)$. The two-sided probability is $2\Pr(X \leq 2) = 0.6875$. This task-entry-level calculation assumes independent pairs and is not adjusted for potential dependence between behavior variants. The result does not establish equivalence of the selectors.

Actual query counts can differ through filtering and early termination despite identical caps. The near-matched counts support the interpretation that one arm was not allocated a larger maximum

Task block	Random	V-TAP H	Random only	H only
40–59 ($n = 20$)	5/20	6/20	0	1
60–99 ($n = 40$)	9/40	10/40	0	1
120–159 ($n = 40$)	13/40	14/40	0	1
160–175 ($n = 16$)	5/16	4/16	2	1
Combined ($n = 116$)	32/116	34/116	2	4

Table 6: Matched online strict successes using the historical frozen selector H. The combined result is 27.59% for random and 29.31% for H; exact two-sided McNemar $p = 0.6875$. Thirty tasks succeed under both selectors and 80 fail under both.

Block	Both succeed	Random only	H only	Both fail
40–59	5	0	1	14
60–99	9	0	1	30
120–159	13	0	1	26
160–175	3	2	1	10
Combined	30	2	4	80

Table 7: Full paired outcomes for the historical frozen selector H.

rollout budget. They do not establish that activation ranking reduces total inference work or elapsed time.

A.4 Additional Fixed-Pool Summaries

The current and historical selectors both obtain pool-weighted S@2 of 16.67%. The new locked-selector analysis supplies the previously missing random S@2 expectation, 14.93%. These descriptive fixed-pool results do not by themselves establish a population-level ranking advantage.

The earlier equally pool-weighted audit gives H an S@1 of 14.58% and MRR of 0.16042, and R an S@1 of 10.42% and MRR of 0.14236; the random expectations are 9.20% and 0.13119. Both fitted selectors obtain S@2 of 16.67%. These audit values retain their original precision and artifact identities.

For comparison with earlier reporting, the historical task-equal-weight S@1 summary was 25.0% versus a random expectation of 17.15%, with contrast $+7.85$ percentage points, reported task-clustered 95% interval $[-1.81, +17.85]$, and reported two-sided $p = 0.112$. These historical statistics do not establish significance for R or for a differently weighted summary. We report this earlier estimate separately to preserve its aggregation unit and artifact identity.

A.5 Training-Task Resampling

The analysis uses 100 source-task bootstrap refits per cap and evaluates each refit on the same held-out pools. Its intervals quantify training-data sensitivity conditional on that evaluation set; they do not represent uncertainty in the online ASR difference.

	Mean target probes		Mean attacker queries	
Block	Random	H	Random	H
40–59	3.300	3.400	12.500	12.500
60–99	2.925	2.925	11.325	11.325
120–159	2.800	2.825	10.600	10.600
160–175	4.188	4.125	14.000	14.000

Table 8: Realized query counts per task under identical maximum probe allocations. Local feature extraction is an additional cost.

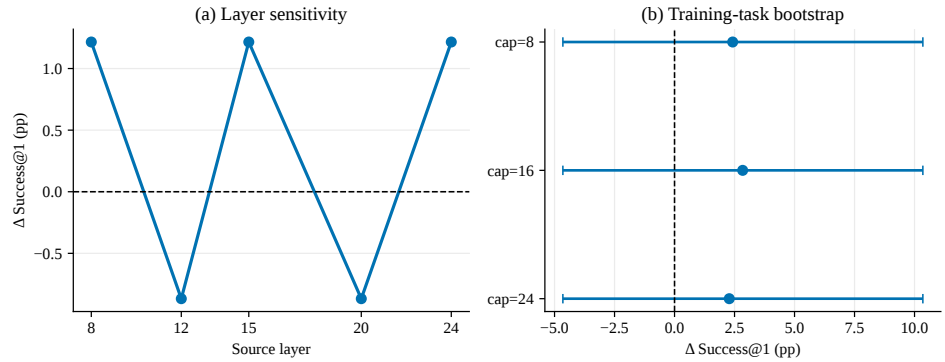

Figure 3: Sensitivity of activation ranking. (a) Pool-weighted S@1 contrasts against expected random selection for five source layers. (b) Task-equal-weight S@1 contrasts under source-task bootstrap refitting: points are bootstrap means and bars are 95% resampling intervals over 100 refits per cap, conditional on the same held-out data. The two panels use different averaging units and assess different sources of variation. Zero indicates no advantage over random.

We examine layers 8, 12, 15, 20, and 24 to characterize the role of representation choice (Figure 3(a)). Layers 8, 15, and 24 each give a positive pool-weighted S@1 contrast of approximately 1.22 percentage points over random selection. Layers 12 and 20 give negative contrasts. The positive estimates therefore occur at multiple depths, although the layer grid also demonstrates that feature choice matters.

Source-task bootstrap refitting provides a complementary view of training-data sensitivity (Figure 3(b)). All three caps yield positive mean task-weighted S@1 contrasts. At cap 24, the mean is $+2.28$ percentage points, with a 95% resampling interval of $[-4.65, +10.35]$. All displayed intervals span zero, so the positive means do not establish a stable advantage under source-task resampling. Caps 16 and 24 retain the same source collection; their small difference in bootstrap means reflects the reported refits rather than a change in retained data. Collection composition and MRR results are reported below.

Cap per task/label	Rows	Tasks	Positives	Positive fraction
8	129	19	14	10.85%
16	176	19	14	7.95%
24	176	19	14	7.95%

Table 9: Source-collection composition after capping, before the internal fitting/evaluation split. Capping limits large groups but does not balance the classes.

Each cap is evaluated using 100 source-task bootstrap refits, with the held-out data fixed. Figure 3(b) reports the S@1 means and intervals. The corresponding task-equal-weight MRR results are given in Table 10. All intervals include zero. Because caps 16 and 24 retain the same source collection, their different resampling means do not demonstrate a cap-dependent effect. These intervals describe training-sample sensitivity, not a repeated independent evaluation of new tasks.

Cap	Point ΔMRR	Bootstrap mean	95% interval
8	0.0189	0.0142	$[-0.0231, 0.0578]$
16	0.0272	0.0158	$[-0.0216, 0.0560]$
24	0.0272	0.0126	$[-0.0264, 0.0501]$

Table 10: MRR contrasts against expected random selection under source-task bootstrap refitting.

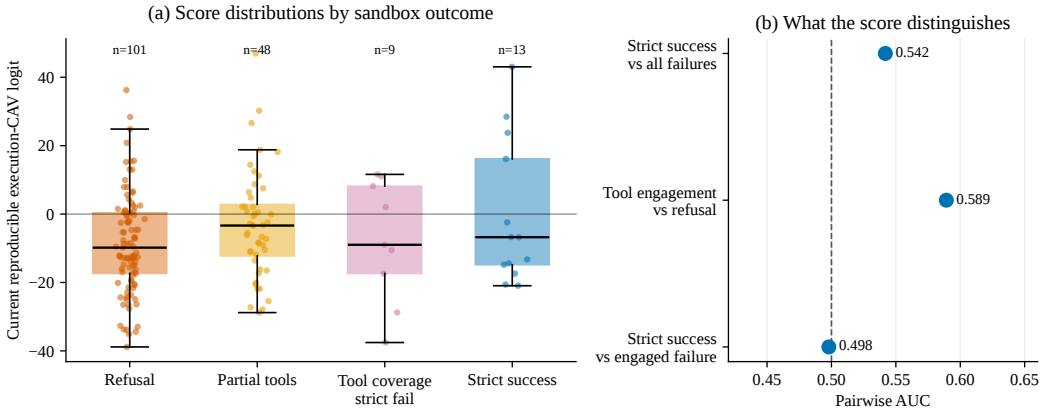

Figure 4: Failure-mode analysis using the current reproducible CAV. (a) Score distributions for explicit refusal, partial tool execution, tool coverage without strict completion, and strict success. The four displayed groups contain 171 candidates; one additional candidate with neither tool engagement nor explicit refusal is omitted from panel (a). (b) Pairwise AUC point estimates, with a reference at 0.5. The engagement-versus-refusal comparison has the largest AUC point estimate; all separations remain weak.

A.6 Failure Categories and AUC Interpretation

The trace-level analysis distinguishes four outcomes: explicit refusal, partial tool execution, tool coverage without strict completion, and strict success. This decomposition connects a pre-response score to different stages of the resulting agent behavior (Figure 4). It also prevents action initiation from being counted as benchmark completion.

For the current selector R, the AUC is 0.542 for strict success versus all failures, 0.589 for tool engagement versus refusal, and 0.498 for strict success versus tool-engaged failure. Engagement versus refusal has the largest point estimate, suggesting a weak action-related association in this sample. The near-chance completion-versus-engaged-failure result identifies a specific modeling challenge: ranking already engaged trajectories requires information about whether the tool sequence actually satisfies the objective.

The score distributions overlap across all four groups. Accordingly, V-TAP uses the score to prioritize rollouts and retains strict execution grading for final decisions. These descriptive findings motivate richer outcome supervision and trajectory-aware representations as future directions; they do not identify a causal mechanism. The category counts and AUC conventions are detailed below.

Figure 4(a) displays 101 explicit refusals without tool calls, 48 partial tool executions, nine cases of apparent tool coverage without strict success, and 13 strict successes. These categories contain 171 of the 172 candidates; the remaining case has no tool engagement and no explicit refusal. It is not displayed as a one-observation boxplot. The AUC comparison labeled “all failures” includes the full strict-failure category, whereas “engaged failure” refers to the 57 candidates in the two tool-engaged failure groups.

The current AUCs are 0.542, 0.589, and 0.498 for the three comparisons shown. These values are used consistently in the current main-text analysis. The relatively higher engagement AUC does not establish statistically greater discrimination: no paired AUC-difference test or clustered confidence interval is available in the figure. The 13 strict successes and within-task dependence further limit inference. The evidence supports a descriptive distinction between action engagement and completion, rather than a causal identification of the failure mechanism.

A.7 Exploratory Adaptive Comparison

The adaptive policy triggers bounded regeneration when the maximum eligible-candidate logit is at most -5.0. It allows at most three generation rounds, including the initial round. Feedback uses the

attacker's earlier candidates rather than unobserved target responses. Both selectors use this policy and retain the same maximum of three target probes per depth.

Arm	Strict successes	Mean probes	Mean attacker queries
Adaptive-random	4/20	6.65	51.0
Adaptive V-TAP	4/20	6.60	48.5

Table 11: Adaptive study on entries 100–119: three joint successes, one random-only success, one V-TAP-only success, and 15 joint failures.

The trigger activates on 16/20 tasks, and Adaptive V-TAP probes 70 regenerated candidates. Thus, a completely inactive regeneration mechanism cannot explain the result. Nevertheless, this small comparison provides no evidence of an advantage for CAV selection within the adaptive policy. Without a matched non-adaptive arm on these same tasks, it cannot isolate the effect of regeneration itself.

B Reproducibility and Selector Agreement

B.1 What Is Reproduced?

We repeat feature extraction and CAV fitting in two independent processes under the same fixed environment, using the same 176 source rows and the same 172 held-out candidates. Features are extracted at layer 15 with bfloat16 inference and saved in float32. Fitting uses per-task-per-label cap 24, a task-hash split with training fraction 0.7, and class-weighted L2 logistic regression with L2 setting 10^{-4}. The environment uses Python 3.12.13, PyTorch 2.10.0 with CUDA 12.8, and Transformers 5.16.0.dev0. This audit performs local model computation on saved data without issuing new agent or tool rollouts.

Comparison between fresh runs	Result
Rendered inputs and order	Identical, 176/176
Saved feature maximum absolute difference	0
Weight L2 difference	0
Bias difference	0
Normalizer mean/std maximum differences	0 / 0
Held-out logit maximum absolute difference	0
Pool top-1 agreement	48/48

Table 12: Repeatability of the current pipeline on the audited inputs. Equality of numerical contents, rather than serialized file bytes, is the comparison criterion.

Table 12 establishes exact numerical repeatability for the two fresh runs: the same rendered inputs produce the same saved features, fitted scores, and selections. This is a statement about fixed inputs and a fixed environment. It does not establish robustness to different source tasks, other software or hardware, or new targets. Training-task sensitivity is evaluated separately in Figure 3(b).

B.2 Why High Score Correlation Does Not Guarantee the Same Selection

Figure 5 compares the historical selector used online with the current reproducible selector. Most scores follow a similar broad ordering, giving Spearman correlation 0.95003. Nevertheless, the top-ranked candidate changes in seven of the 48 pools. Global correlation describes agreement across the full candidate set, whereas preselection depends on the relative order of candidates within each pool. Close within-pool scores can change order even when overall correlation remains high.

The fitted directions are also distinct: weight cosine similarity is 0.93632 and weight L2 difference is 2.22889. Both the normalizer and classifier differ. Consistent with those selection differences, the pool-weighted top-1 coverage is 14.58% for H and 10.42% for R, although both obtain 16.67% at two selections. These comparisons do not identify a unique cause of the historical discrepancy. In-

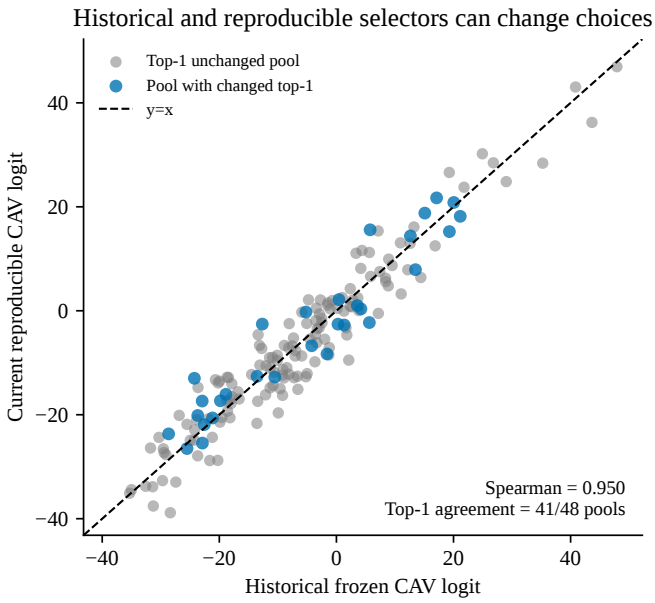

Figure 5: Historical H and current reproducible R logits for the same held-out candidates. Each point is a candidate; blue points belong to pools whose top-1 choice changes, while gray points belong to pools whose choice is unchanged. The dashed diagonal denotes equal logits. Scores are strongly correlated (Spearman 0.950), yet top-1 agreement is only 41/48 pools. Highlighting is by pool, not a claim that every blue candidate is selected by either method.

complete historical environment and rendering information prevents exact reconstruction; the fresh two-run equality gives no evidence that randomness in the current procedure caused it.

B.3 Implications for Artifact-Level Evaluation

The online results evaluate the historical frozen selector H. The current fixed-pool, failure-score, and two-run repeatability analyses evaluate R. Preserving these identities allows the reported gains and diagnostics to be attributed to the artifact actually tested.

The audit also separates two useful evaluation goals. Repeating a fixed pipeline tests whether a comparison can be reproduced numerically, while refitting on resampled source tasks tests sensitivity to the training collection. R satisfies the former check in the audited environment; the latter reveals substantial uncertainty. Reporting both properties makes the framework suitable for controlled follow-up experiments without conflating repeatability with generalization.

C Additional Frozen-Pool Experiments

C.1 Artifacts, Data, and Protocol

The new analyses use the existing canonical Direct-TAP source and held-out archives: the source run is dated 2026-09-13 16:48:50 and the held-out run 2026-09-13 19:45:27. The locked budget-analysis selector is `cav_a.pt` from the 2026-09-15 17:20:19 reproducibility audit. It uses local Qwen3.5-9B layer-15 boundary features with bfloat16 inference. These analyses read recorded outcomes and introduce no new target, tool, or grading calls.

C.2 Budget Coverage and Conditional Pool Analyses

For a pool of size N, the fraction permitted by the deployed cap is $\min(3, N)/N$. Averaging over 21 pools of size 2, 16 of size 4, and 11 of size 6 gives 80.2%. This is nominal budget coverage

before early stopping, not a measured executed fraction. The $N > 3$ subset contains 27 pools and is defined by size alone. The mixed-outcome subset contains eight pools and is conditioned on labels. A ninth success-containing pool has no failures, and 39 pools have no successful candidate.

Subset	Ranking	S@1	S@2	S@3
All 48	CAV-top	10.42	16.67	18.75
	Random expectation	9.20	14.93	16.67
	CAV-bottom	8.33	10.42	14.58
$N > 3$ (27)	CAV-top	11.11	14.81	18.52
	Random expectation	7.10	11.73	14.81
	CAV-bottom	3.70	3.70	11.11
	TF-IDF	7.41	7.41	7.41
Mixed (8)	CAV-top	50.0	87.5	100.0
	Random expectation	42.7	77.1	87.5
	CAV-bottom	37.5	50.0	75.0

Table 13: Offline coverage in percent. Mixed-pool values describe conditional retrieval and must not be interpreted as overall attack success.

Task-cluster bootstrap uses 2,000 draws. The all-pool CAV-top 95% intervals are $[3.51, 20.45]$, $[7.27, 30.23]$, and $[9.09, 33.33]$ percent for $K = 1, 2, 3$. They describe each selector's coverage rather than a paired contrast. The report notes overlap with random-selector intervals but supplies numerical endpoints only for CAV-top; Figure 2 therefore displays only those supplied intervals. Neither overlap nor its absence substitutes for a paired difference analysis.

C.3 Context Diagnostics

The full-context and candidate-only completion probes share the layer, source archive, fitting procedure, and held-out pools. Full context includes the target-visible prefix and per-task tool schema; candidate-only removes both. Since the canonical target has no system prompt, a nominal system-plus-candidate condition would duplicate candidate-only.

Internal held-out AUROC is 0.973 for full context and 0.378 for candidate-only. Both completion evaluations contain only one positive, making these figures unsuitable as stable model-selection evidence.

C.4 Supervision Ablation

All three variants use full-context features and are evaluated against the same final strict-completion labels on held-out pools. Tool-engagement and non-refusal labels are at least as effective as comple-

Training label	S@1 (%)	S@2 (%)	MRR	Pairwise (%)
Completion	10.42	16.67	0.1424	71.4
Tool engagement	14.58	18.75	0.1667	85.7
Non-refusal	14.58	18.75	0.1667	90.5
Random expectation	9.20	14.93	0.1312	—

Table 14: Supervision variants on the same frozen pools. Pairwise values are retained as reported diagnostics with the definition caveat in Appendix C.3.

tion labels on these ranking summaries. The source collection contains 14 completion positives, 75 tool-engaged candidates, and 71 non-refusal candidates; reported fitting-set prevalences are 9.4%, 48.6%, and 45.7%, respectively. Consequently, label meaning and positive-example availability both differ. The results do not establish completion-specific representations or a unique benefit of completion supervision.

Under the additional evaluator gate, S@1 is 14.58% for both action-oriented alternatives, 12.50% for completion, and 11.81% for expected random selection. These conditional values use a different

candidate-set protocol from the main raw-pool results. Internal held-out AUROCs are 0.973, 0.933, and 0.842 for completion, engagement, and non-refusal, respectively. The completion split has one positive and the other splits have approximately eight positives, so these AUROCs are reported only as sparse-split diagnostics.

C.5 Training-Scale Diagnostic

This analysis performs task-resampled grouped cross-validation on the same 19-task source feature cache. It is a representation/label learnability diagnostic, not a new held-out ranking experiment or an online ASR analysis. The mean rises from 0.512 at five tasks to 0.628 at 19 tasks, with narrower

Sampled tasks	Valid repeats	Mean CV AUROC	2.5–97.5 percentiles
5	19	0.512	$[0.093, 0.971]$
10	20	0.616	$[0.328, 0.805]$
15	20	0.601	$[0.442, 0.743]$
19	20	0.628	$[0.523, 0.729]$

Table 15: Grouped-CV diagnostic conditional on the existing source collection. Ranges are percentiles across repeats, not confidence intervals for online gains.

reported percentile ranges, but is not monotone across intermediate sizes. This is consistent with sparse supervision contributing to instability; it does not establish a scaling law or predict an online improvement.