TRACESAE: SELECTIVE SPARSE CONTROL AFTER TOOL FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-using language models must revise plans when tool feedback invalidates their prerequisites, while retaining plans that remain supported. We introduce TRACESAE, an inference-time controller that targets *unwarranted persistence*: continuing to rely on a premise explicitly contradicted by visible tool feedback. TRACESAE separates the decision to intervene from the action it changes. An unexecuted shadow proposal informs a gate that is frozen before the actual action is generated. When the gate opens, the controller edits 16 sparse-autoencoder coordinates toward activation profiles from warranted decisions, preserves the original reconstruction error, and caps each perturbation at 2% of the residual norm. Feature groups and intervention strengths are selected through restored-state continuations that jointly evaluate action correctness, goal progress, preservation, and token cost. The language model and pretrained autoencoder remain fixed. Across Gemma-2-9B-it and Llama-3.1-8B-Instruct on Telecom-Compose and ALFWorld, TRACESAE improves full-task success over a rank-16 dense controller by 4.67–8.21 percentage points under a shared gate, donor bank, edit window, and candidate-search allowance. These results establish the task-level benefit of behaviorally selected sparse-profile control in the four evaluated settings.

1 INTRODUCTION

A tool call can be syntactically valid and still be the wrong next action. Consider an agent preparing to use a service that requires an enabled account setting. A diagnostic reports that the setting is disabled. Repeating the dependent operation satisfies the tool schema but leaves its prerequisite unresolved. A useful next action would repair the setting, obtain authorization, or collect the information needed to proceed. When the diagnostic instead confirms that the setting is enabled, continuing the original plan can be appropriate. Effective tool use therefore requires *selective plan revision*, not indiscriminate correction.

We study this decision at the boundary between a tool observation and the next action. Our target, *unwarranted persistence*, is reliance on a premise that sufficient visible evidence has explicitly invalidated. This definition separates a failure to update from malformed calls, unrelated entity mistakes, and reasonable plans that later fail. It also makes the control problem asymmetric: intervene when a plan has lost its evidential support, but avoid disrupting a warranted continuation.

Sparse autoencoders (SAEs) provide an interface for inspecting and editing internal activations (Huben et al., 2024). A useful representation, however, is not automatically a useful controller. A probe may predict an error without identifying a small group of coordinates whose modification improves behavior. Task-grounded SAE evaluation highlights this gap between representation and control (Makelov et al., 2025); output-oriented feature selection further shows why activation strength alone is an incomplete guide to steering (Arad et al., 2025). For a tool-using agent, the relevant outcome extends beyond the next response: an edited action must produce useful progress after execution.

TRACESAE connects these requirements through three decisions: *when to intervene, where to edit, and how to select the edit*. First, a frozen gate combines visible evidence, a pre-proposal activation, and the tool family of an unexecuted shadow proposal. Second, a context- and token-role-specific profile supplies target values for 16 SAE coordinates. Third, restored-state continuations select the feature group and intervention strength according to action quality, progress, preservation, and cost.

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.

The actual action is generated afresh from the original history, so the edit cannot retroactively change the gate that authorized it.

The resulting update changes only the selected features' decoder contribution and retains the original SAE reconstruction error. This avoids replacing the entire residual stream by an imperfect reconstruction. A relative-norm cap bounds each local perturbation, and unsupported contexts leave generation unchanged. The model and SAE weights remain fixed; task adaptation is confined to the auxiliary classifiers, donor profiles, and finite behavioral selection procedure.

We evaluate two instruction-tuned models in two stateful environments. The primary reference is a calibrated rank-16 dense-profile controller that shares the gate, donor records, edit positions, norm cap, and search allowance. TRACESAE improves success in every model–environment pair, adding 14 and 18 net successful Telecom tasks and 11 ALFWorld successes for each model. This comparison tests the sparse-profile policy under the same surrounding controller, rather than attributing improvements to an external guard or a larger candidate search.

Our contributions are threefold. **(1) A feedback-grounded control target:** we operationalize unwarranted persistence and distinguish it from other action errors. **(2) A selective sparse-profile controller:** we combine a pre-action gate with residual-preserving edits selected by executed continuations. **(3) A shared-controller evaluation:** we compare sparse and dense profile control across two models and two environments using full-task success and explicit resource accounting.

2 RELATED WORK

Sparse representations and useful interventions. Huben et al. (2024) study SAE features through interpretation and activation editing, including the number and magnitude of edited features. Makelov et al. (2025) use task-supervised representations to evaluate what SAEs capture and how effectively their features support control. AxBench finds that simple supervised and prompting baselines can outperform SAE steering (Wu et al., 2025), whereas Arad et al. (2025) improve SAE steering by selecting features for their output effects. Together, these results motivate a behavioral test of feature selection rather than an assumption that sparsity alone improves control. TRACESAE extends this perspective to tool feedback: process probes nominate candidates, and executed environment continuations choose the deployed intervention.

Conditional steering and preservation. CAST separates the condition for applying an intervention from the behavior it induces (Lee et al., 2025). TRACESAE applies this separation to prerequisite invalidation in an interactive trajectory. Its gate is fixed before actual-action generation, and its target is a warranted activation profile indexed by context and token role. Activation steering for instruction following also examines response quality alongside constraint satisfaction (Stolfo et al., 2025). Our calibration objective likewise rewards useful progress while penalizing disruption of warranted actions. The distinction is the decision being controlled and how its intervention is selected, rather than conditional steering itself.

Intervention semantics. Sparse Feature Circuits explicitly includes SAE reconstruction errors when evaluating behavioral faithfulness (Marks et al., 2025). Activation-patching studies show that metrics and intervention choices materially affect an analysis (Zhang and Nanda, 2024). We therefore specify the edited positions, cache lifecycle, and additive decomposition of the update. Retaining unselected decoder contributions is an algebraic property at the hook; measuring the resulting trajectory is a separate behavioral evaluation.

Tool feedback and interactive agents. AgentBench evaluates decision making in interactive environments (Liu et al., 2024). CRITIC uses tools to verify and revise model outputs (Gou et al., 2024). TRACESAE instead acts on the internal computation that generates the next action, before that action is dispatched. Its empirical comparison isolates sparse versus dense profile control within a shared decision interface; tool-verification and text-revision pipelines provide complementary approaches to feedback use.

3 SELECTIVE SPARSE PROFILE CONTROL

3.1 THE FEEDBACK-DEPENDENT DECISION

Let x be the visible trajectory immediately after a tool observation o. A frozen domain adapter extracts a candidate plan p from the preceding assistant text: its action family, referenced entity, intended subgoal, and stated prerequisite. An evidence adapter classifies the relation between o and that prerequisite as

$$e(x, p) \in \{\text{sup}, \text{inv}, \text{unk}\}, \tag{1}$$

where the categories denote supporting, invalidating, and insufficient evidence. An unknown plan remains distinct from a plan with a known, invalidated prerequisite.

An action exhibits unwarranted persistence when the plan is identifiable, $e(x, p) = \text{inv}$, and the action continues to depend on that invalidated premise. Warrantedness is membership in a set of evidence-consistent actions, not agreement with one gold action. Repairing a prerequisite, resolving an unknown fact, or performing a required handoff can each be warranted. Conversely, merely avoiding persistence does not make an unrelated or malformed action correct. Appendix A gives the labeling distinctions.

The controller receives public tool schemas, visible observations, and prior messages. Hidden simulator state is used only for restoration and offline evaluation. A shared guard G checks action schemas, required fields, and entities already visible in the history. It neither inspects hidden goal predicates nor supplies a replacement action. A rejection produces a deterministic visible error, and a retry consumes a new decision cycle. Thus, schema validity and evidence-consistent planning remain separate responsibilities.

3.2 FREEZE THE GATE BEFORE GENERATING THE ACTUAL ACTION

From x, the model first generates a greedy *shadow proposal* q, which is never executed. Before this generation, we save the unedited residual h_0 at the final assistant-boundary prefill position. A text classifier predicts context c, including the evidence relation, from the extracted plan and observation. A logistic model estimates persistence risk $R(h_0, c, t(q))$, where $t(q)$ is the parsed tool family. It receives neither the complete proposal arguments nor their entity relations.

The intervention gate is

$$g(x, q) = \mathbb{I}\big[\hat{e} = \text{inv}, \quad p_{\text{inv}} \geq 0.8, \quad R(h_0, c, t(q)) \geq 0.5, \quad \text{eligible}(x, q, c)\big]. \tag{2}$$

Eligibility requires a parseable plan and tool family and an available initial profile. Classifier parameters, thresholds, and eligibility rules are fixed before held-out evaluation. Unsupported contexts and missing initial profiles close the gate.

After computing c and g, the controller discards the shadow cache and generates the actual response again from x. Neither the shadow text nor its cache enters this generation; both c and g remain fixed throughout it. The shadow is therefore a diagnostic input to the gate, not a second candidate from which the controller selects a better action. Algorithm 1 specifies the execution order.

The context classifier uses character and word TF–IDF features. Its vocabulary, normalization, regularization, and probability calibration are determined on discovery data. The risk model standardizes h_0 with discovery statistics and is trained using out-of-fold context predictions. Prefixes are weighted equally within each task, and tasks equally in the training objective. This separates fitting the decision rule from evaluating the frozen controller.

3.3 EDIT SELECTED DECODER CONTRIBUTIONS TOWARD WARRANTED PROFILES

For a fixed SAE, let $f : \mathbb{R}^d \to \mathbb{R}^K$ be the encoder, $D \in \mathbb{R}^{d \times K}$ the decoder, and $b \in \mathbb{R}^d$ its bias. The selected coordinate group S contains 16 features. A donor profile $m_S(c, r)$ averages their activations on warranted discovery decisions with context c and token role r. Averaging proceeds over positions within a decision, decisions within a task, and then equally over tasks. Dense and sparse profiles use the same donor records and weights.

Algorithm 1 One decision cycle of TRACESAE

Require: Visible prefix x; frozen model, SAE, classifiers, group S, profiles, dose α
1: Save unedited residual h_0 at the final assistant-boundary prefill position
2: Generate greedy shadow proposal q from x without dispatching it
3: Predict c and compute the gate $g(x, q)$ using Eq. (2)
4: Discard the shadow cache; restart generation from x with empty response a
5: **while** the actual response has not terminated and its token budget remains **do**
6: Compute residual h at the fixed hook and eligible prediction position
7: Infer token role r from the already generated prefix
8: Apply Eq. (4) when $g = 1$ and profile (c, r) exists
9: Complete the forward pass and append the next token to a
10: **end while**
11: Apply the shared guard to a; execute only if admitted
12: Append the resulting observation or the guard's rejection message to history
13: Charge both generations and the decision cycle to the task budget

A task-hash-selected 80% subset of discovery tasks supplies the deployed profile bank; Telecom composition families remain intact. A context–role profile requires at least 20 warranted donor decisions from five tasks. This support requirement applies before any target is used. At an eligible position, define

$$z = f(h), \qquad v_S(h, c, r) = D_S\big(m_S(c, r) - z_S\big). \tag{3}$$

The local update, with $\rho = 0.02$, is

$$\delta(h) = \begin{cases} 0, & g = 0 \text{ or the profile is unavailable,} \\ 0, & \|h\|_2 = 0 \text{ or } \|\alpha v_S\|_2 = 0, \\ \alpha v_S \min\left\{1, \dfrac{\rho\|h\|_2}{\|\alpha v_S\|_2}\right\}, & \text{otherwise,} \end{cases} \tag{4}$$

$$h' = h + \delta(h).$$

The implementation checks availability before accessing a profile and handles zero norms before division. Consequently, $\|h' - h\|_2 \le \rho\|h\|_2$ at every edited position.

What the update preserves. Write the original reconstruction error as $\varepsilon(h) = h - b - Df(h)$. For an open gate, $\alpha = 1$, and inactive clipping,

$$h' = \underbrace{\varepsilon(h)}_{\text{original reconstruction error}} + b + \underbrace{D_{\bar{S}} z_{\bar{S}}}_{\text{unselected contribution}} + \underbrace{D_S m_S(c, r)}_{\text{selected profile}}. \tag{5}$$

Thus, the operator replaces the selected decoder contribution rather than the entire activation. With clipping or another dose, it makes a scaled step along the same profile direction. Appendix B gives the general identity and proof. The decomposition refers to the original encoding of h: re-encoding h' may change both selected and unselected coordinates. Doses above one can extrapolate beyond the donor coefficients.

Where edits occur. The edit window includes the final assistant-boundary prefill position and subsequent decode positions of the actual action, ending at its terminator. Earlier prompt positions are unchanged. The five token roles are prefill, thought value, tool-name value, argument value, and JSON structure. A role is inferred from the already generated prefix, before observing the next token. An unresolved parser state or missing later-role profile leaves that position unchanged. Context and gate remain fixed even when an edited action chooses a different tool family from the shadow.

3.4 SELECT GROUPS BY EXECUTED BEHAVIOR

Nominate a finite candidate set. A process probe ranks a variance shortlist of 1,024 SAE coordinates on task-separated discovery trajectories. The 64 nominees with the highest positive task-weighted coefficients are ordered by coefficient. Four consecutive blocks and 28 fixed-seed subsets yield 32 groups of 16 coordinates. Each group is evaluated at four doses, $\alpha \in \{0.25, 0.5, 1, 2\}$, giving 128 configurations per model–environment setting. Nomination narrows the search; the continuation objective selects the deployed configuration.

Compare continuations from the same source state. Calibration uses 64 restored states from distinct development tasks, balanced between 32 unwarranted and 32 warranted greedy shadow proposals. Each configuration and its guard-only reference continue from the same source state using seeds 11 and 29, temperature 0.7, and top-p 0.95. Each continuation lasts at most four decision cycles. After the fork, every branch receives observations generated by its own executed actions rather than observations replayed from another branch.

Let $j \in \{1, \ldots, M\}$ index the source–seed pairs, where $M = 128$. For method m, $W_{jm} \in \{0, 1\}$ indicates whether the first actual proposal is warranted, scored *before* guard filtering; $V_{jm} \in [0, 1]$ is verified goal progress at termination or the fourth cycle; and T_{jm} is the total number of generated tokens, including shadows. The preservation penalty is

$$H_m^{\mathrm{cal}} = \frac{\sum_{j=1}^{M} W_{jG}(1 - W_{jm})}{\sum_{j=1}^{M} W_{jG}}, \tag{6}$$

and the selection score is

$$J_m = \frac{1}{M} \sum_{j=1}^{M} \left[(W_{jm} - W_{jG}) + 0.25(V_{jm} - V_{jG}) - 0.05 \frac{T_{jm} - T_{jG}}{T_{\max}} \right] - 2H_m^{\mathrm{cal}}. \tag{7}$$

Here $T_{\max}$ is the common continuation token-cap normalization, and progress is the fraction of predefined goal prerequisites satisfied. An undecidable edited response has $W_{jm} = 0$, so it counts against preservation when the actual reference is warranted; its process label remains distinct from a confirmed persistence error. A zero denominator in Eq. (6) makes a configuration ineligible. The enrollment labels describe greedy shadows, whereas W_{jG} describes the reference's sampled actual action; the two are not interchangeable.

Screen once, then freeze. The highest-scoring configuration is screened on the remaining calibration tasks using full-task continuations. It is vetoed if its net success change relative to the guard is nonpositive or if more than 5% of guard-successful tasks become failures. A veto ends selection for that setting. Ties in J favor a smaller realized perturbation, then lower token cost, then lexicographic coordinate IDs and ascending dose. Candidate construction, profile rules, thresholds, and tie-breaking are fixed before test evaluation. Appendix A, Table 6, collects the controller thresholds and search settings.

3.5 A SHARED-CONTROLLER DENSE REFERENCE

The dense comparator uses an orthonormal rank-16 basis B and direction

$$v_B(h, c, r) = B\big(m_B(c, r) - B^\top h\big). \tag{8}$$

It applies the same update rule after substituting v_B for v_S. Dense profiles average $B^\top h$ over the same donor decisions with the same nested weights. Sixteen discovery resamples each produce one PCA and one supervised-SVD basis, matching the sparse policy's 32 candidates and four-dose grid. Supervised SVD matches erroneous and warranted discovery decisions within context, role, and tool family across tasks using frozen text-similarity matching.

Both selectors share calibration states, gates, eligibility rules, edit positions, clipping, and candidate-continuation allowances. What changes is the representation-and-encoding policy: nonlinear SAE coordinates with selected decoder columns versus linear coordinates in an orthonormal dense basis. Equal group size and rank control the number of edited coordinates; they do not equate pretrained representation capacity. Table 1 summarizes the components and generation allowances of all four evaluated policies.

Table 1: **Components of the evaluated policies.** Dense and TRACESAE use the same gate, donor decisions, edit window, norm cap, and candidate-search allowance. The generation ceiling counts the unexecuted shadow and actual response; it is not a measured compute cost. A dash denotes a component that is not used.

Component	Raw	Guard	Dense+G	TRACESAE+G
Visible-information guard	No	Yes	Yes	Yes
Frozen shadow-based gate	No	No	Yes	Yes
Control representation	–	–	Linear basis	SAE features
Control coordinates	–	–	16	16
Warranted donor profiles	–	–	Shared	Shared
Relative-norm cap	–	–	2%	2%
Candidates $\times$ doses	–	–	32×4	32×4
Generation ceiling (responses per cycle)	1	1	2	2

Table 2: **Model–SAE pairings.** Block indices are zero-based. The 16 edited coordinates are a subset of the full dictionary, not the dictionary size.

Model	Block	Residual d	Dictionary K	Group	Encoder
Gemma-2-9B-it	20	3,584	16,384	16	JumpReLU
Llama-3.1-8B-Instruct	19	4,096	65,536	16	ReLU

4 EXPERIMENTAL SETUP

Models and activation sites. We use Gemma-2-9B-it with an instruction-tuned Gemma Scope residual SAE, and Llama-3.1-8B-Instruct with the Goodfire layer-19 SAE (Lieberum et al., 2024; Google, n.d.; Goodfire, n.d.). Table 2 specifies the intervention sites. Both hooks act on the transformer block output after the MLP residual addition. The published encoders operate in native residual units without additional inference-time input normalization. Each model–environment pair has a separately selected feature group, dose, and donor bank. Appendix D records the assets and implementation conventions.

Interactive tasks and splits. Telecom-Compose Solo is a research composition extension of the telecommunications simulator in τ^2-Bench (Barres et al., 2025). Tasks are grouped by prerequisite and action-dependency structure; every descendant of a composition family remains in one split. Its results measure this extension, not the official τ^2-Bench benchmark. ALFWorld provides a text-based environment with executable object interactions (Shridhar et al., 2021). We evaluate on its 134 `valid_unseen` games. Discovery and calibration use disjoint training games (Table 3). The two models share 434 unique evaluation tasks, and Telecom's 300 evaluation tasks belong to 60 families.

Baselines and decoding. We compare raw generation, the shared guard alone, dense profiles plus the guard, and TRACESAE plus the guard (Table 1). Table 4 distinguishes candidate-ranking continuations from held-out full-task evaluation. Each response is capped at 192 generated tokens. Dense and sparse controllers allow one shadow and one actual generation per cycle; raw and guard-only baselines generate only the actual response. Comparisons with these baselines therefore include the cost of the shadow pass, not a best-of-two action-selection benefit.

All methods use identical instructions, schemas, and observation formatting within a model. Rendered input plus reserved generation must fit 8,192 tokens. Truncation removes the oldest complete action–observation pairs while preserving instructions, the task, and the latest feedback. A protected

Table 3: **Task-separated allocations.** Telecom entries are tasks/families; ALFWorld entries are games. Discovery fits the auxiliary assets, calibration selects the intervention, and held-out evaluation measures full-task success.

Environment	Discovery	Calibration	Evaluation
Telecom-Compose Solo	600/120	200/40	300/60
ALFWorld text	800	200	134

Table 4: **Candidate ranking versus held-out evaluation.** Candidate ranking uses sampled continuations; final evaluation uses a frozen policy and greedy actual actions. Shadow proposals are greedy in both phases. The separate full-task veto uses the remaining calibration tasks (Section 3.4), not the held-out test set.

Item	Candidate ranking	Held-out evaluation
Source tasks	64 distinct development tasks; one restored state per task	300 Telecom tasks; 134 ALFWorld games
Enrollment	32 unwarranted and 32 warranted greedy shadow proposals	Every enrolled evaluation task
Actual actions	Temperature 0.7, top-p 0.95; paired seeds 11 and 29	Greedy decoding
Cycle limit	4 per continuation	32 (Telecom); 50 (ALFWorld)
Criterion	J_m: action quality, progress, preservation, and token cost	Full-task success with the selected group and dose frozen

prefix that cannot fit terminates the task as context overflow. Shadow tokens, malformed outputs, and guard rejections retain their costs; Appendix E gives the accounting rules.

Primary endpoint. Full-task success includes every enrolled task. Abstention, rejected calls, context overflow, and exhausted budgets remain in the denominator. For task outcome $y_{im} \in \{0, 1\}$, the success rate is $N^{-1} \sum_i y_{im}$ and the net improvement over reference r is $N^{-1} \sum_i (y_{im} - y_{ir})$. We report point estimates from exact success counts. Evaluation tasks reused across models are not additional independent instances, and Telecom families group dependent task variants.

5 RESULTS

5.1 SPARSE-PROFILE CONTROL IMPROVES SUCCESS IN ALL FOUR SETTINGS

Table 5 reports full-task success. TRACESAE outperforms the shared-controller dense reference for both models in both environments. On Telecom, Gemma improves from 65.67% to 70.33%, and Llama from 53.00% to 59.00%. On ALFWorld, Gemma improves from 64.93% to 73.13%, and Llama from 57.46% to 65.67%. Computed from exact counts, these are gains of 4.67, 6.00, 8.21, and 8.21 percentage points, respectively.

Table 5: **Full-task success (%; higher is better).** G is the shared visible-information guard. Dense and TRACESAE share the controller and search allowance in Table 1. Δ is the gain over dense+G in percentage points, computed from exact counts (Appendix C). Bold marks the highest success rate in each row; all values are point estimates.

Environment	Model	N	Baselines		Shared controller		Δ (pp)
			Raw	Guard	Dense+G	TRACESAE+G	
Telecom	Gemma	300	44.00	57.00	65.67	**70.33**	+4.67
	Llama	300	32.00	43.00	53.00	**59.00**	+6.00
ALFWorld	Gemma	134	50.00	55.22	64.93	**73.13**	+8.21
	Llama	134	40.30	46.27	57.46	**65.67**	+8.21

The count-level gains are 14 and 18 additional net successes on Telecom, and 11 on ALFWorld for each model. For example, Llama completes 159 of 300 Telecom tasks with dense control and 177 with TRACESAE. These gains arise under a shared gate, donor bank, and selection allowance, positioning the complete sparse-profile policy as the beneficial change in this controlled comparison.

5.2 THE GAIN EXTENDS BEYOND EXTERNAL VALIDITY CHECKS

The guard improves success over raw generation in all four settings, and dense profile control improves further. TRACESAE adds another gain on top of both. On Llama Telecom, the progression is 96 successes for raw generation, 129 with the guard, 159 with dense control, and 177 with sparse control. On Gemma ALFWorld, the corresponding counts are 67, 74, 87, and 98. This progression separates the benefit of an external validity check from the additional benefit of internal profile control.

The improvements occur in both composed service workflows and executable object interactions. Since each setting is calibrated separately, the results demonstrate repeated application of the procedure across model–environment pairs. The shared-controller comparison is particularly important here: stronger performance than an unguarded agent alone would not establish an advantage over a calibrated internal intervention.

6 DISCUSSION AND CONCLUSION

What the comparison establishes. The central empirical finding is higher full-task success for the complete sparse-profile policy than for the rank-16 dense reference in all four settings. The controller brings together a feedback-specific gate, warranted profile targets, and selection by executed behavior. Its sparse representation and nonlinear encoding are evaluated jointly; the comparison does not isolate an individual component or establish superiority over prompting or tool-critique pipelines.

The aggregate outcomes are point estimates without paired confidence intervals or significance claims. A net gain is the number of rescues minus newly introduced failures, so the marginal counts do not separately measure preservation or the frequency of repaired persistence errors. Those process-level quantities, transfer of a frozen controller, and measured inference overhead are the most direct next evaluations. The local norm cap bounds each injected perturbation, while trajectory quality remains an empirical endpoint.

Conclusion. TRACESAE makes feedback-dependent plan revision an explicit control problem: freeze the decision to intervene, steer a small selected feature group toward warranted profiles, and choose that intervention by the behavior it produces in the environment. With fixed language-model and SAE weights, it raises full-task success by 4.67–8.21 percentage points over a shared-controller dense baseline across two models and two environments. The results identify behaviorally selected sparse profiles as a useful internal-control interface for tool-using language models.

AI USE STATEMENT

An AI assistant was used during manuscript revision for literature search, structural and language editing, methodological reporting feedback, algebraic exposition and checking, arithmetic consistency checks, and LaTeX preparation. No new language-model inference or simulator experiments were performed during this revision. The experimental counts come from the underlying study; algebraic identities and arithmetic checks are distinguished from empirical measurements.

ETHICS STATEMENT

The experiments concern simulated tool-use settings. The controller uses visible information, retains external validity guards, and leaves generation unchanged when the required context or profiles are unavailable. Improving benchmark success is not a substitute for authorization, privacy protection, or human oversight in consequential applications. Real-service deployment would require safeguards appropriate to the actions being executed.

REPRODUCIBILITY STATEMENT

Section 3 specifies the online controller, profile update, and behavioral selection objective. Section 4 records the data allocations and evaluation protocol. Appendices A–E provide labeling and cache conventions, donor averaging, algebraic properties, exact outcome counts, asset identifiers, and resource accounting. Exact policy replay additionally requires the setting-specific selected groups and doses, donor-profile arrays, fitted classifiers, task manifests, and implementation assets.

REFERENCES

Dana Arad, Aaron Mueller, and Yonatan Belinkov. SAEs are good for steering—if you select the right features. In *Proceedings of EMNLP*, pp. 10241–10259, 2025.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Goodfire. *Llama-3.1-8B-Instruct-SAE-l19*. Model and SAE release, n.d. Accessed 26 September 2026. Hugging Face model repository.

Google. *Gemma Scope: Instruction-Tuned 9B Residual SAEs*. Model and SAE release, n.d. Accessed 26 September 2026. Hugging Face model repository.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, 2024.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations*, 2024.

Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering. In *International Conference on Learning Representations*, 2025.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma Scope: Open sparse autoencoders everywhere all at once on Gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.

Aleksandar Makelov, Georg Lange, and Neel Nanda. Towards principled evaluations of sparse autoencoders for interpretability and control. In *International Conference on Learning Representations*, 2025.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *International Conference on Learning Representations*, 2025.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

Alessandro Stolfo, Vidhisha Balachandran, Safoora Yousefi, Eric Horvitz, and Besmira Nushi. Improving instruction-following in language models through activation steering. In *International Conference on Learning Representations*, 2025.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. AxBench: Steering LLMs? Even simple baselines outperform sparse autoencoders. *arXiv preprint arXiv:2501.17148*, 2025.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *International Conference on Learning Representations*, 2024.

A Execution, Labeling, and Selection Details

A.1 Controller and search settings

Table 6 collects the thresholds, profile rules, candidate construction, and selection criteria specified in Section 3. These rules are fixed before held-out evaluation; the selected group and dose are setting-specific outputs of calibration.

Table 6: **Controller thresholds and finite-search settings.** These are protocol settings, not measured outcomes. The dose grid lists the candidates considered, not the selected dose for any particular setting. The two selectors share the gate and profile rules, dose grid, objective, and veto criteria.

Stage	Parameter	Specified value or rule
Gate	Invalidation confidence	$p_{\text{inv}} \geq 0.8$
	Persistence risk	$R(h_0, c, t(q)) \geq 0.5$
	Initial eligibility	Parseable plan and tool family; initial profile available
Profiles	Discovery allocation	80% by task/family hash; Telecom families kept intact
	Minimum support	20 warranted decisions from at least five tasks, per context–role profile
	Averaging	Positions within decisions, decisions within tasks, then equal task weights
Search	SAE variance shortlist	1,024 coordinates
	Process-probe nominees	64 positive-coefficient coordinates
	Sparse candidates	32 groups: four consecutive blocks and 28 fixed-seed subsets
	Dense candidates	32 bases: PCA and supervised SVD for each of 16 discovery resamples
	Candidate dose grid	$\alpha \in \{0.25, 0.5, 1, 2\}$
	Configurations per selector	$32 \times 4 = 128$ per model–environment setting
Selection	Score coefficients	Action: $+1$; progress: $+0.25$; non-preservation: -2; normalized token difference: -0.05
	Full-task veto	Reject if net gain over guard is ≤ 0, or failures among guard-successful tasks exceed 5%

A.2 Visible information and process labels

The candidate-plan adapter reads assistant text preceding the latest observation. It extracts an explicit candidate action, including candidates whose endorsement is not explicit. The evidence adapter reads the latest visible observation and evaluates the candidate prerequisite. Its schema covers entity identity, resource or object state, and policy or action ordering. Unresolved extraction returns an unknown label rather than a guessed plan.

Process annotation distinguishes unwarranted persistence, malformed action, unrelated wrong entity, insufficient evidence, correct continuation, required handoff, and unknown plan. The evaluation adapter identifies the set of actions warranted by public information. Simulator predicates are available offline to verify executability, observation truthfulness, and goal progress; they do not enter the controller's input. The annotation protocol uses independent blinded raters for process labels.

For the calibration objective, $W_{jm} = 1$ means that method m's first actual proposal on source–seed pair j is warranted. Every other response has $W_{jm} = 0$ in the action-improvement term. When the actual reference is warranted, both confirmed unwarranted responses and undecidable

responses count against preservation, while their separate labels are retained. Insufficient evidence is consequently not relabeled as a confirmed persistence error.

A.3 Proposal, cache, and guard conventions

Shadow and actual responses start from the same visible prefix. The shadow is neither executed nor added to persistent history. Its parsed tool family conditions risk, but its full text is not supplied to either action generator. After freezing the gate, the controller discards the shadow cache, recomputes the prefix, and applies the hook only at the specified positions during actual generation. A closed gate or a zero-strength hook returns the original activation tensor.

Paired calibration branches start from the same source tokens, pre-edit activations, simulator snapshot, and sampling configuration. Equal seeds couple random draws; they do not force equal outputs after interventions change the token distributions. Each branch subsequently executes its own admitted actions and receives its own observations. The shared guard uses the same version and deterministic rejection messages in every $+G$ arm, validates only public schemas and visible entities, and never substitutes an oracle action.

A.4 Task-balanced donor profiles

Let $\mathcal{T}_{c,r}$ be the donor tasks with eligible decisions for context c and role r, $\mathcal{D}_{t,c,r}$ their warranted decisions, and $\mathcal{P}_{j,r}$ the positions of role r in decision j. The deployed sparse profile is

$$m_S(c,r) = \frac{1}{|\mathcal{T}_{c,r}|} \sum_{t \in \mathcal{T}_{c,r}} \frac{1}{|\mathcal{D}_{t,c,r}|} \sum_{j \in \mathcal{D}_{t,c,r}} \frac{1}{|\mathcal{P}_{j,r}|} \sum_{u \in \mathcal{P}_{j,r}} f_S(h_{j,u}). \tag{9}$$

Only decisions containing the relevant role contribute to its average. A profile requires at least 20 decisions from five tasks. Replacing $f_S(h)$ by $B^\top h$ yields the dense profile with identical donor decisions and weights. This nested average prevents a long response or a task with many eligible decisions from dominating the target.

Discovery tasks, or whole Telecom families, are ordered by a hash of the setting ID and task/family ID. The first $\lfloor 0.8N \rfloor$ groups form the deployed profile bank, with N denoting the number of task/family groups in this split operation. The minimum-support rule is applied separately for each context and token role. Missing initial profiles close the gate; missing later-role profiles skip only the corresponding positions.

A.5 Nomination and deterministic candidate construction

Candidate nomination uses a 1,024-coordinate variance shortlist. The top 64 positive task-weighted process-probe coefficients are ordered by coefficient, breaking ties by coordinate ID. Four consecutive blocks of 16 coordinates are retained. Another 28 groups are sampled without replacement within each group; duplicate groups are redrawn.

The study seed is `20260926`. An operation's seed is the integer represented by the first eight hexadecimal digits of

```
SHA256("TraceSAE-v6/20260926/" + setting_id + "/" +
operation).
```

Sorted IDs and NumPy PCG64 fix the resampling order. Operation names include `nominee_subsets`, `discovery_resamples`, and `donor_task_split`. A setting without the required calibration strata is ineligible rather than rebalanced after its outcomes are inspected.

B Algebraic Properties of the Profile Operator

Proposition 1 (Local norm bound and residual retention)**.** *Let $\alpha \geq 0$. For the update in Eq.* (4)*, $\|h' - h\|_2 \leq \rho \|h\|_2$. When the gate is open and a profile is available, write $z = f(h)$ and $\varepsilon(h) = h - b - Dz$. There is an effective dose $\beta \geq 0$ such that*

$$h' = \varepsilon(h) + b + D_{\bar{S}} z_{\bar{S}} + D_S\big((1-\beta) z_S + \beta m_S(c,r)\big). \tag{10}$$

For $\alpha > 0$, $\|h\|_2 > 0$, and $\|v_S\|_2 > 0$,

$$\beta = \min\left\{\alpha, \frac{\rho\|h\|_2}{\|v_S\|_2}\right\}; \tag{11}$$

set $\beta = 0$ in the zero-update cases.

Proof. For a nonzero update, $h' - h = \beta v_S$. Hence

$$\|h' - h\|_2 = \min\{\alpha\|v_S\|_2, \rho\|h\|_2\} \le \rho\|h\|_2. \tag{12}$$

Substituting $h = \varepsilon(h) + b + D_{\bar{S}} z_{\bar{S}} + D_S z_S$ into $h' = h + \beta D_S(m_S - z_S)$ and collecting the selected terms gives Eq. (10). The zero-update cases have $h' = h$ and satisfy both statements directly. □

The identity separates three facts. First, the injected change lies in the span of the selected decoder columns. Second, the original reconstruction error and unselected additive decoder contribution are retained. Third, the displayed selected coefficients interpolate when $0 \le \beta \le 1$ and extrapolate when $\beta > 1$. Since the dose grid includes $\alpha = 2$, not every candidate is an interpolation.

The displayed coefficients need not equal the encoder output $f_S(h')$. In particular, neither a decrease in re-encoded profile distance nor invariance of $f_{\bar{S}}(h')$ follows from Eq. (10). The norm bound concerns a local residual perturbation, not the number of downstream action changes or accumulated trajectory deviation.

C Exact Full-Task Outcomes

Table 7 gives the counts underlying the main results. Percentages are calculated as $100n/N$; differences as $100(n_{\mathrm{SAE}} - n_{\mathrm{dense}})/N$, without rounding intermediate rates. In particular, the ALFWorld difference is $100 \times 11/134 = 8.208955\ldots$ percentage points, which rounds to 8.21.

Table 7: **Exact full-task success counts.** These are the numerators of Table 5. Net is the number of additional successes minus newly introduced failures relative to dense+G; it is not a separate rescue count.

Environment	Model	N	Raw	Guard	Dense+G	TraceSAE+G	Net
Telecom	Gemma	300	132	171	197	**211**	+14
	Llama	300	96	129	159	**177**	+18
ALFWorld	Gemma	134	67	74	87	**98**	+11
	Llama	134	54	62	77	**88**	+11

For a reference method, define task-level rescue and harm counts as

$$R_{\mathrm{task}} = \sum_i \mathbb{I}[y_{i,\mathrm{ref}} = 0,\ y_{i,\mathrm{SAE}} = 1], \qquad H_{\mathrm{task}} = \sum_i \mathbb{I}[y_{i,\mathrm{ref}} = 1,\ y_{i,\mathrm{SAE}} = 0]. \tag{13}$$

Their difference equals the net success count. Marginal totals identify $R_{\mathrm{task}} - H_{\mathrm{task}}$, not the two components separately. This task-level quantity is distinct from the first-action preservation penalty H_m^{cal} used in selection.

There are 434 unique full-task instances, reused across the two models. The four rows represent four model–environment comparisons rather than 868 independent tasks. Telecom's 300 test tasks are additionally grouped into 60 families. No paired confidence interval, significance test, or task-level preservation estimate is reconstructed from the marginal totals.

D Model Assets and Implementation Conventions

D.1 Model and dictionary identifiers

Table 8 records the assets used in the study. The public SAE file hashes agree with the release metadata (Google, n.d.; Goodfire, n.d.). Repository revisions below are the study's recorded identifiers; the SAE weight-file upload commit is distinguished from an inference-code revision.

Table 8: **Model, tokenizer, and SAE repositories.**

Asset	Repository or checkpoint path
Gemma model/tokenizer	`google/gemma-2-9b-it`
Gemma SAE	`google/gemma-scope-9b-it-res`
Gemma dictionary	`layer_20/width_16k/average_l0_91/params.npz`
Llama model/tokenizer	`meta-llama/Llama-3.1-8B-Instruct`
Llama SAE	`Goodfire/Llama-3.1-8B-Instruct-SAE-l19`
Llama dictionary	`Llama-3.1-8B-Instruct-SAE-l19.pth`

Gemma model/tokenizer revision:
`11c9b309abf73637e4b6f9a3fa1e92e615547819`
Gemma SAE repository revision:
`e86af97a5b6fbbccca28ab654f2fda1b0768f770`
Gemma SAE file SHA256:
`bbd770b6f8b92a2fe7498e05bd6274c6cfa89ebc08fb972c0e842840737f1a82`
Gemma `tokenizer.json` SHA256:
`3f289bc05132635a8bc7aca7aa21255efd5e18f3710f43e3cdb96bcd41be4922`

Llama model/tokenizer revision:
`0e9e39f249a16976918f6564b8830bc894c89659`
Goodfire SAE repository revision:
`f6775a221e47b44233af4bac2c7b65189265519a`
Goodfire SAE file SHA256:
`bbdc6fad591d3d420e8859dcb6eefa4ec642026090f7f39a963b1c0f27290e4f`
Goodfire SAE weight-file upload commit:
`2b0a4d80a9d492b07cd27807c48ae2fa832c420e`

D.2 Encoder equations and hooks

In row-vector notation, the Gemma dictionary uses

$$u = hW_{\text{enc}} + b_{\text{enc}}, \quad f(h) = u \odot \mathbb{I}[u > \theta], \quad \hat{h} = f(h)W_{\text{dec}} + b_{\text{dec}}. \tag{14}$$

The encoder has shape 3584×16384 and decoder 16384×3584. The released parameters incorporate training-time scaling and encoder-bias preprocessing. The recorded inference convention adds neither decoder-bias subtraction before encoding nor residual normalization. The suffix `average_l0_91` describes an average activation count in the release, not the number of selected control coordinates.

In column-vector notation, the Goodfire encoder is $f(h) = \mathrm{ReLU}(W_{\text{enc}}h + b_{\text{enc}})$ and reconstructs as $\hat{h} = W_{\text{dec}}f(h) + b_{\text{dec}}$. It has input width 4,096 and feature width 65,536. The recorded path does not subtract decoder bias before encoding, normalize inputs, renormalize decoder columns, or apply TopK. Feature IDs are zero-based decoder-column indices; transposing stored weights does not change their identities.

The Gemma hook attaches to `model.model.layers[20]`; the Llama hook attaches to `model.model.layers[19]`. Both edit the full block output, not solely the MLP submodule. For tuple outputs, the first activation tensor is modified and other entries retained. An `output_hidden_states` tuple includes the initial embedding state, so its tuple index is not treated as a block number.

D.3 Runtime and serialization

The recorded runtime is Python 3.12, PyTorch 2.8.0, and Transformers 4.56.2, with float32 model and SAE arithmetic and TF32 matrix multiplication disabled. Gemma uses eager attention, preserving attention-logit soft capping and alternating sliding-window masks. The Llama path uses the study-specific `tracesae_fp32_efficient_sdpa` kernel. These conventions specify numerical execution rather than measured latency or memory use.

Llama uses its published role template. Gemma merges system instructions into the first user message, separated from the task by a blank line, following its user/model template. Tool observations are serialized as visible feedback. Within each model this formatting is fixed across raw, guard, dense, and sparse arms.

E Resource and Outcome Accounting

E.1 Deployment budgets

Shadow tokens count even when the gate closes. Retries consume new cycles. Malformed responses, rejected calls, unfinished generations, and shadow-only aborts retain their costs. The input allowance is checked after rendering the chat template. Reserving 192 tokens for generation leaves at most 8,000 input tokens under the 8,192-token total limit. Truncation never cuts a JSON record in half. A protected prefix that is too long ends the branch as context overflow; a prerequisite no longer visible yields an unknown-plan label. These cases remain in the full-task success denominator. A guard rejection is not itself a successful correction: only an admitted action changes the environment.

Table 9: **Deployment budget ceilings.** These are design limits, not measured consumption. Each generated response is capped at 192 tokens. Shadow policies allow two generations per decision cycle; raw and guard allow one.

Environment	Cycles	Shadow-policy token cap	Raw/guard token cap
Telecom-Compose	32	12,288	6,144
ALFWorld	50	19,200	9,600

Dense and sparse controllers have the same generation allowance. SAE encoding and repeated prefills add computation beyond the generated-token count, so equal ceilings are not a latency or consumed-compute match. The shadow is not an executed branch and its text is not an input to actual-action generation.

E.2 Selection allowance

For one selector in one setting, the candidate-continuation output ceiling is

$$128 \times 64 \times 2 \times 4 \times 2 \times 192 = 25{,}165{,}824 \tag{15}$$

generated tokens. The factors count configurations, source states, seeds, cycles, generations per cycle, and tokens per generation. Shared guard references require at most $64 \times 2 \times 4 \times 192 = 98{,}304$ tokens. These design ceilings apply equally to the sparse and dense selectors and exclude discovery, profiling, SAE preparation, and full-task veto checks. They are not measured consumption.