Can Mechanistic Interpretability Help Explain LLM Agent Failures? A Case Study on Web Element Hallucination

Mohamed Aghzal[1], **Mor Geva**[2], **Gregory J. Stein**[1], **Ziyu Yao**[1]
[1]George Mason University
[2]Tel Aviv University
{maghzal, gjstein, ziyuyao}@gmu.edu, morgeva@tauex.tau.ac.il

Abstract

Mechanistic Interpretability (MI) has made significant progress in explaining the internal computations of Large Language Models (LLMs); however, these advances have yet to demonstrate comparable success in explaining the decision-making processes of LLM-based agents. This paper examines the challenges MI tools face when applied to realistic agentic failures. We focus on one such failure: web agents hallucinating actions that are not available on a website. We apply a series of mechanistic analyzes to Qwen2.5-7B-Instruct as it interacts with live websites. We find that a linear probe trained on the last prompt token can predict hallucinations before generation begins (0.80 AUC) and steering along a probe direction corrects 85% of hallucinations, but disrupts 7% of valid actions. However, these results only provide a partial explanation of why the agent hallucinates. The probe direction is sensitive to prompt formatting, losing much of its steering effect after a minor change to how actions are represented. Additionally, circuit identification can be difficult as the computation is distributed across multiple components and positions, and the information the agent relies on varies across examples. We discuss how these observations challenge the assumptions underlying standard MI methods for circuit discovery and feature identification, and outline future directions for making these methods better suited to explain failures in LLM-based agents.

1 Introduction

LLM-based agents are increasingly deployed in complex, long-horizon tasks ranging from web navigation (Zhou et al., 2024; Deng et al., 2023a; Xue et al., 2025; Aghzal et al., 2026b) and code generation (Jiang et al., 2024; Tang et al., 2024; Li et al., 2025) to scientific reasoning (Wei et al., 2025; Xin et al., 2024). Unlike traditional LLM tasks and settings where a model produces a single response to a fixed input, agents operate sequentially: they perceive an environment, form plans, execute actions, observe outcomes, and adapt over multiple steps. While LLM-based agents have achieved success in a number of scenarios (Schmidgall et al., 2025; Yang et al., 2024), the complexity of underlying environments and the sequential nature of the decision-making involved introduce failure modes that are qualitatively different from those observed in standard LLM settings (Aghzal et al., 2026b; Cemri et al., 2025; Zhu et al., 2025; Liu et al., 2024) and the mechanisms underlying these failures remain poorly understood.

Mechanistic interpretability (MI) (Olah et al., 2020; Elhage et al., 2021) is a candidate for closing this gap. MI aims to recover the algorithms and concepts a network implements from its weights and activations, and has traced the circuits and features responsible for a wide range of behaviors (Meng et al., 2022; Geva et al., 2021; Ravfogel et al., 2021; Finlayson et al., 2021; Olsson et al., 2022; Akyürek et al., 2023). While this demonstrates the promise of MI for understanding and improving LLMs, its application to LLM agents remain limited, despite agent reliability being a critical concern for both performance and safety (Bereska and Gavves, 2024; Bengio et al., 2025; Jones et al., 2026; Lee et al., 2026).

This gap reflects a discrepancy between what current MI tools are able to explain[1] and the root causes of agent failures. While MI has been designed to target model-level analysis, agentic failures emerge over long trajectories and over multiple agent components. Aghzal et al. (2026a), thus, argue that explaining LLM agents requires multiple levels of attribution that combine behavioral analysis with MI. In this work, we focus on the mechanistic level of their proposed framework, highlighting the challenges that must be overcome to

[1]We use the term "explain" to refer not only to *understanding* model internals, but also their *utility*: whether the insights can be used to improve performance (Orgad et al., 2026).

extract meaningful mechanistic insights from the backbone LLMs powering agents.

To highlight these challenges, we conduct an analysis on a failure mode in which web agents hallucinate actions that are not available on the provided web page. Using Qwen2.5-7B-Instruct on live websites from Online-Mind2Web (Xue et al., 2025), we apply a number of MI techniques in order to explain the causes behind this behavior. We find that a linear probe on the last prompt token predicts the hallucination with 0.80 AUC before the action is written, and that steering along the probe direction fixes 85% of hallucinations while breaking 7% of valid actions. However, each of these analyzes captures only part of the mechanism. On the circuit discovery side, computation is distributed across components, and targeted component interventions based on component analysis fix at most 8.9% of cases. Moreover, the input responsible varies across samples. On the feature identification side, we find that the probe direction is coupled to the prompt format and loses much of its steering effect under a small change in how actions are listed, and pretrained SAEs yield no interpretable features guiding hallucinated decisions. These findings highlight how the assumptions made by MI approaches can be challenged in the realistic settings agents operate in. We highlight some directions that can help address these issues.

2 Preliminaries and Background

2.1 The Components of an LLM Agent

LLM agents operate over interconnected components over a sequence of timesteps t's (Mo et al., 2024): **Perception** converts raw observations o_t into token representations $x_t = enc(o_t)$; **Memory** maintains information across timesteps through a history state (or collection of states) $h_t = u(h_{t-1}, x_t)$, taking the form of working memory (the current context window) and long-term memory (parametric knowledge and external retrieval); **Reasoning** transforms perception and memory into a decision, formalized as a policy $a_t = \pi(h_t, x_t)$, often through explicit reasoning chains (Wei et al., 2022) but can also be implicit (Deng et al., 2023b); **Actions** $a_t \in \mathcal{A}$ are executed by an external environment through a transition function $\mathcal{E}_{t+1} = f_{\text{env}}(\mathcal{E}_t, a_t)$, yielding a new observation o_{t+1}; and **Evaluation and Monitoring** close the loop by comparing actual and predicted outcomes, and replanning when needed.

2.2 Mechanistic Interpretability

MI explains model behavior in terms of the internal mechanisms that produce it (Olah et al., 2020; Elhage et al., 2021), studying weights, activations, and the causal relations among them instead of only inputs and outputs. Work in the area typically centers on three notions (Rai et al., 2024). **Circuits** are subgraphs of the model that implement a recognizable algorithm, such as copying or induction, and trace how information flows to produce a given behavior (Ameisen et al., 2025; Meng et al., 2022). **Features** are directions in activation space that encode interpretable concepts (Huben et al., 2024; Bricken et al., 2023). **Universality** is the hypothesis that the same features and circuits reappear across architectures (Chughtai et al., 2023).

Most of MI's successful explanations (Wang et al., 2023; Hanna et al., 2023; Lindsey et al., 2025; Geva et al., 2021; Meng et al., 2022) come from narrow tasks with fixed inputs. Recent work has called for extending MI to more complex, realistic settings. Sharkey et al. (2025) catalogue the open problems that limit its applicability, and Orgad et al. (2026) argue that MI lacks evaluation criteria based on actionability, noting that few MI findings have so far led to concrete performance improvements. We study this question for LLM agents. Existing work is limited to sequential decision-making in toy environments or to static settings (Men et al., 2024; Nanda et al., 2023; Ustaomeroglu et al., 2026), and the mechanisms behind agent failures in realistic environments remain largely unknown. A related line of work also examines the mechanisms behind LLM reasoning (Hou et al., 2023; Arcuschin et al., 2025; Dutta et al., 2024; Chen et al., 2026). However, agents introduce further challenges. Their failures emerge and compound over long trajectories and the computation may be spread over many components and different timesteps and positions.

3 The Challenges Posed by Agentic Settings for MI

Circuit Discovery The successful applications of circuit discovery methods (Wang et al., 2023; Hanna et al., 2023; Conmy et al., 2023) share implicit assumptions that can fail in agentic settings.

The first assumption is *localizability*, which assumes that a small set of positions and components within a single timestep can account for the mechanisms responsible for a model's behavior. Successful applications of circuit discovery have relied on

this holding consistently across samples. For instance, in subject-verb agreement (Finlayson et al., 2021), minimal pairs can be constructed by hand (e.g., *The keys to the cabinet **are** on the table* vs. *The key to the cabinet **is** on the table*), making it straightforward to identify the subject token and apply attribution methods that localize the relevant attention heads or FFNs as the source of the prediction. In agentic settings, this assumption is harder to satisfy. Decisions involve complex interactions across many input positions, and there are at many times no clear basis for hypothesizing which positions or components drive an outcome and the relationship between them. As we discuss in Section 4.2, even identifying that the action list influences hallucination in web agents leaves open the question of how different action lists give rise to different decisions. Moreover, oftentimes no single component bears a disproportionate share of the load behind the failure and many components contribute collectively in ways that single-component attribution methods can struggle to detect.

The second assumption, *stationarity*, breaks down because of the temporal nature of agentic tasks: the input distribution changes at every timestep. As the agent acts and receives new observations, the context h_t is continuously updated, meaning the internal representations we wish to analyze are a moving target. Standard counterfactual methods such as activation patching require a fixed clean baseline (Meng et al., 2022; Wang et al., 2023); but in an agentic setting, after each action the context window grows and the environment state changes, so no two steps share a comparable baseline and the patch no longer isolates the same computation across steps.

Feature Identification MI methods for discovering and identifying features, such as sparse autoencoders (SAE; Bricken et al., 2023; Huben et al., 2024), probing classifiers (Belinkov, 2022; Hewitt and Manning, 2019; Tenney et al., 2019), and steering vectors (Zou et al., 2023; Turner et al., 2024; Postmus and Abreu, 2024; Rimsky et al., 2024), face challenges similar as circuit discovery.

The first challenge concerns the *data distribution.* SAEs are trained to decompose activations into interpretable features. However, the features they recover reflect what is prevalent in their training corpus. Thus, without agent-specific data, SAEs may struggle to capture concepts relevant to agentic behavior such as goal tracking, tool selection, or multi-step planning, which arise from environment interaction rather than text. Prior work already shows that SAE performance degrades under covariate shift (Kantamneni et al., 2025; Pacela et al., 2026), which naturally indicates that limited robustness is to be expected when applying them to agent trajectories. Probing classifiers (Belinkov, 2022; Tenney et al., 2019) face a related issue: the linear structure they rely on can vary dramatically over the course of a multi-turn conversation (Lampinen et al., 2026), and can be brittle in the face of distribution shifts (Pacela et al., 2026). As such, the nature of agent trajectories, which involve dynamic context accumulation across steps and can vary greatly depending on design choices as well as environments, means that probing-based approaches can be unreliable. Similarly, steering vectors assume the identified direction transfers to the target setting, but distribution shifts can move the representations out of the subspace the intervention vector was designed on, making it ineffective.

The second difficulty concerns the *temporal nature* of agent failures. Agent decisions are not necessarily encoded in a single feature at a single timestep. Commitment to a particular plan or strategy may be distributed across multiple features and emerge gradually over the course of a trajectory (Men et al., 2024; Dong et al., 2025), with no single timestep at which the decision is cleanly represented. Lubana et al. (2026) illustrate this limitation, showing that standard SAEs struggle to encode temporal features even in textual settings. In stateful, long-horizon environments, this issue can be more severe, as decisions and their representations can depend on trajectory-level structure.

4 Why do Web Agents Hallucinate Website Elements?

Hallucination (Zhang et al., 2025) is an issue that has been widely documented in LLMs, where systems generate plausible but factually incorrect information. This issue presents critical safety concerns for agents (Deng et al., 2025; Zhang et al., 2024; Zhu et al., 2025; Lin et al., 2025; Yona et al., 2026). In the context of web agents, this could arise as proposing website actions that reference nonexistent website components (Aghzal et al., 2026b). In this case study, we aim to explain this behavior. While the case study is specific to web agent hallucination, the challenges we illustrate can also apply to other agents, as we discuss in Appendix A.

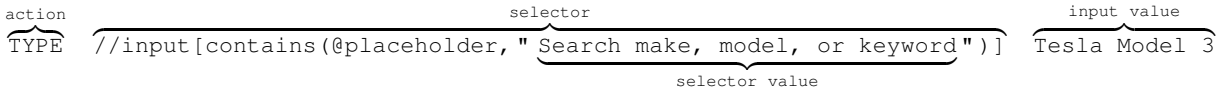


Figure 1: Example hallucination. Given the task: ***Find a 2022 Tesla Model 3 on CarMax.***, the model generates a `TYPE` action targeting a search input. The selector value does not correspond to a valid action provided on the webpage. A valid action would be, e.g., `CLICK //a[contains(@id, 'header-drawer-focus-start')](text:"Shop")`, which clicks the "Shop" link in the header to open inventory browsing.

Table 1: Intervention results using different sets of attention heads and FFN layers ($\alpha = 3$, $\beta = 0.5$).

Attn Heads	FFN Layers	Fix (↑)	Break (↓)
L22 (All heads)	22, 23, 24	**8.9%**	**2.4%**
L22, L27 (All heads)	22, 23, 24	5.9%	27.6%
Top-10 causal attr. (Figure 2b)	13, 15, 24	4.4%	3.5%
Top-10 causal attr. (Figure 2b)	22, 23, 24	4.5%	4.6%
Top-10 corr. diff. (Figure 2c)	22, 23, 24	6.3%	2.5%

4.1 Experimental Setup

When restricted to a minimal action space of `CLICK`, `SCROLL`, `TYPE`, and `HOVER` (following the original Mind2Web (Deng et al., 2023a), an early benchmark of web navigation tasks), even frontier models have been shown to fabricate actions referencing nonexistent page elements (Aghzal et al., 2026b). We leverage Online-Mind2Web (Xue et al., 2025), which collects interaction episodes on live websites. On each webpage interaction (a maximum of 10 per task), we construct a list of possible actions based on the web page and provide this action list to the model to choose from, using the prompts in Appendix B. We then define a *hallucinated action* as one where the agent generates an action that does not appear in the list; an example is shown in Figure 1.

We then conduct an analysis on the `Qwen2.5-7B-Instruct` (Qwen et al., 2025). We observe that such hallucination is often due to the model committing to a strategy that does not align with the website; the most common pattern consists of actions targeting elements such as `Search`, `Explore`, or `Find`, that do not exist on the page but reflect the model's intent, as shown by logit lens analysis (Appendix C). The final dataset consists of 300 episodes and 2,071 actions, 32.6% of which are hallucinated.

4.2 The Challenges of Circuit Discovery

We first demonstrate the challenges of attributing hallucinations to a set of positions and components.

Localizability (of responsible LM components) breaks down: Web element generation is a largely distributed computation. Prior work has shown that hallucinations in LLMs can often be attributed to FFN contributions overshadowing attention head outputs in the residual stream, and that targeted dampening of specific FFNs while amplifying attention heads can reduce them (Sun et al., 2025). We test whether this holds in our setting by amplifying FFN contributions 10-fold one at a time and applying Gaussian noise to individual attention heads to valid cases at `selector_value_start` and assessing whether the outputs become hallucinated and show the results in Figure 2. The results show that most FFNs have little effect or contribute negatively to grounding, and most attention heads contribute minimally to valid token probability. However, while interventions on these individual components do produce hallucinations, they are qualitatively different from the hallucinations we get in zero-shot settings, which are more descriptive of strategies. For example, as shown in the examples in Appendix E, amplifying the FFN at layer 15 transforms a valid selector value `interface jsx-326752785 wiki-section bold underlined` into `VAR-1000000000...`, a syntactically invalid sequence that does not clearly attempt to achieve a non-existent strategy. This is in contrast to several natural hallucinations such as `Search make, model, or keyword`, which are syntactically well-formed but reference non-existent elements. This suggests that no individual component contributes significantly to such decisions.

We further test joint interventions similarly to Sun et al. (2025), scaling sets of FFNs by a coefficient β and attention heads by a coefficient α. As shown in Table 1, targeted interventions on the top-10 attention heads identified and the 3 FFNs that consistently lead to hallucinations yield only modest gains (4.4% fix rate, 3.5% break rate). The best configuration among an extensive set of runs (see Appendix F for details), amplifying all attention heads at layer 22 while halving FFN con-

tributions at layers 22–24, fixes 8.9% of hallucinated cases but corrupts 2.4% of valid ones, despite none of these components (with the exception of L24) leading significantly to hallucinations when intervened on individually. Correlational analysis (Figure 2c) also confirms that layer 22 attention heads are the most differentially activated between grounded and hallucinated cases. Nevertheless, although this confirms that these components have a role in grounded generation, they do not tell the full story; if the computation were localized to these components, targeted interventions should produce consistent improvements (Sun et al., 2025; Meng et al., 2022; Wang et al., 2023). The fact that they do not, and that simultaneously high fix and break rates occur across configurations, suggests that the hallucinations are distributed and coupled in a way that single-component attribution cannot explain.

Localizability (of responsible input positions) breaks down. The action list is a contributor, but the specific positions and properties responsible for hallucination remain unidentifiable. As detailed in Appendix G, activation patching shows the action list as a causal contributor (Table 2), with clean state restoration at L27 recovering up to 0.36 probability mass after corruption compared to less than 0.02 for `selector_start` and the task description. However, unlike tasks such as factual recall where replacing *Eiffel Tower* with *Colosseum* produces a clean shift from *Paris* to *Rome* (Meng et al., 2022), knowing the action list matters does not reveal which of its properties drives the commitment, leaving no clear handle for intervention. In fact, even the input span to corrupt is not stable across samples. Corrupting the action list of valid steps with Gaussian noise, as in corruption-based methods such as activation patching and EAP-IG (Hanna et al., 2024), changes the model's decision only when the action list is its sole source, whereas in the half of valid steps where the agent had already used the same value in an earlier step, it recovers it from its action history, as can be seen in Figure 3.

Stationarity is violated: Attempting to construct counterfactuals introduces cascading divergence instead of a single controlled difference. A natural extension would be to apply activation patching directly to paired valid and hallucinated trials; however, this requires structurally matched prompts differing in one controlled dimension. In our setting this is not possible: Different actions

Table 2: Activation patching results for $P(\texttt{Search})$. Columns are corruption spans; "Corrupted P" is the mean $P(\texttt{Search})$ before restoration. Rows show the peak causal recovery $\Delta P(\texttt{Search})$ per component, with peak layer and token offset.

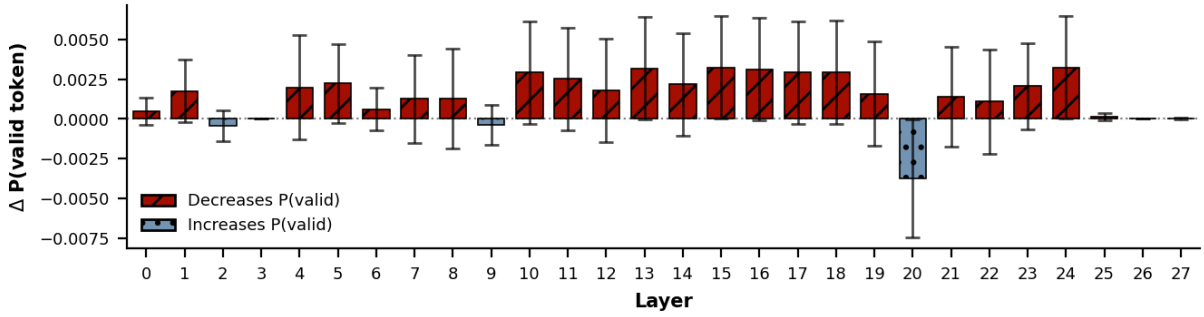
	Selector start	**Task description**	**Action list**
Corrupted P	0.682	0.672	0.325
Residual	0.013 (L6, −4)	0.012 (L27, 0)	**0.360** (L27, 0)
Attention	0.009 (L27, 0)	0.011 (L23, 0)	**0.075** (L22, 0)
FFN	0.008 (L18, 0)	0.009 (L18, 0)	**0.067** (L21, 0)

come from entirely different pages, tasks, URLs, and action histories and the two instances can vary dramatically. As shown in Appendix H, constructing pairs by removing the chosen valid action yields a hallucinated output in only 38% of cases, and produces cascading trajectory divergence, rendering the two runs incomparable after a single step.

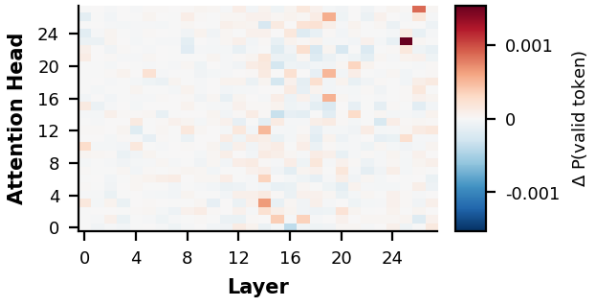
4.3 The Challenges of Feature Identification

Next, we illustrate the challenges of feature identification in agentic settings.

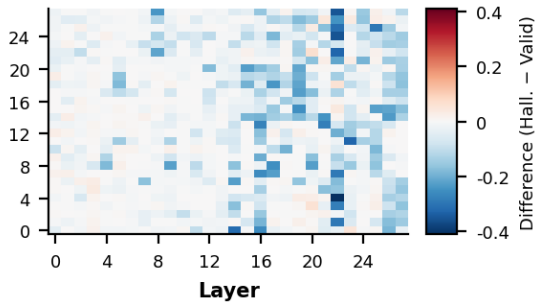
Temporal locality is violated: The hallucination decision can already be encoded pre-generation. We test whether the distinction between hallucinated and valid actions corresponds to a stable internal representation. We train linear probes on the residual stream at three input positions: the end of the prompt (`prompt_end`), i.e. before the model starts generation, the start of the selector (`selector_start`), and the start of the selector value (`selector_value_start`). Probes are trained independently at each of the 28 layers using an episode-level split (70/15/15 of 300 episodes, yielding 1,433 training actions, 282 development, and 356 test). As shown in Figure 4, all three positions yield probes that perform remarkably well on the test set. Interestingly, at `prompt_end`: a probe trained on pre-generation activations reaches a test AUC of 0.80 at layer 18, before the model has produced a single output token. By the time we get to `selector_start` the signal becomes much stronger even at layer 0 and by `selector_value_start` the signal is strong from the first layer and only mildly improves throughout. This suggests that the model's commitment to a hallucinated action is already encoded as a linearly separable direction in the residual stream before generation begins, and may be influenced by context accumulated across previous timesteps and several tokens before the hallucination emerges. This violates temporal locality: rather than being



(a) **Per-layer FFN amplification effect on valid actions.** For each FFN layer independently, we amplify its output by 10× before addition to the residual stream and measure ΔP(valid token) = P(valid | clean) − P(valid | FFN$_l$ amplified). Red bars: amplification reduces P(valid), i.e., the FFN pushes the model away from valid page tokens. Blue bars: amplification increases P(valid), i.e., the FFN contributes to grounding.



(b) **Per-head attention noise effect on grounded actions.** Gaussian noise is injected into each head's projection independently. ΔP(valid) = P(valid|clean) − P(valid|noised head), averaged over grounded cases at selector value start. Red: noising reduces P(valid); blue: noising increases P(valid).



(c) **Per-head max attention value difference (hallucinated − valid) at the selector value start.** For each head, we compute the maximum attention weight placed on any token, averaged over actions per-class, and plot the difference. Blue = higher in valid actions; red = higher in hallucinated actions.

Figure 2: Component contributions to hallucination. (a) Per-layer FFN amplification effect. (b) Per-head attention noise effect. (c) Per-head max value difference across all heads. Setup details in Appendix D.

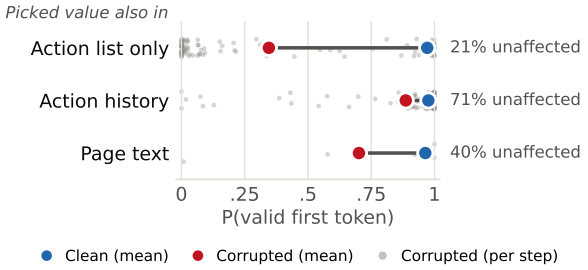


Figure 3: Effect of corrupting the action list on valid steps, grouped by where else the selector value the agent picked appears in the prompt. Dots show the mean probability P of a valid first token at `selector_value_start` before (blue) and after (red) corruption. % of actions that remain unaffected after each corruption is shown.

encoded at the timestep where the hallucination manifests, the commitment is already present in the residual stream before generation begins and strengthens progressively across positions.

Distributional mismatch of linear probing: The direction identified can be coupled to the agentic design. To further test whether the hallucination-predictive direction in the residual stream is causally upstream of the output, we apply representation steering using the weights of the trained probe (See Appendix I for more details and detailed results). This intervention helps substantially; reducing hallucinations by as much as 85%, while only shifting 7% of valid cases to hallucinations using the probe from L17 at `selector_start`.[2] However, when modifying the format of the action list slightly, these probes are brittle to distributional shift: replacing the JSON action list with a numbered markdown format (Appendix J) causes probe performance to degrade substantially (Figure 6) and the steering effectiveness to drop significantly despite the trained probe having seen the data otherwise as shown in Figure 5. This suggests that the learned direction is entangled with action format instead of encoding only the hallucination direction.

[2]While this confirms a causal role at this position, it does not satisfy the localizability assumption: which specific components or input properties drive the effect remains unclear.

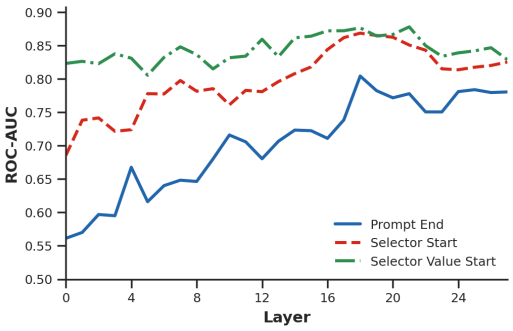


Figure 4: Per-layer AUC of linear probes trained to predict hallucination at different positions.

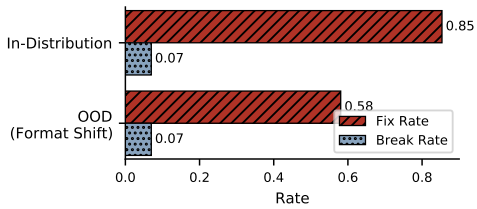


Figure 5: Steering results for the `selector_start` L17 vector.

Distributional mismatch of SAEs: Pretrained SAEs do not yield clearly interpretable features in a web agent setting. We also apply a pretrained SAE to to the action list span at layer 27; in order to indentify if any features relevant to hallucination can be extracted. However, the top features differentiating hallucinated from valid cases mostly promote tokens unrelated to the observed hallucination behavior (Figure 7), highlighting how SAEs pretrained on text can struggle to produce interpretable directions for features that emerge only during agent interaction.

5 A Roadmap for MI for LLM Agents

We believe that MI methods to explain LLM agents need methods to address the following.

MI Benchmarks for Agentic Applications MI Benchmarks (Wang et al., 2023; Mueller et al., 2025; Karvonen et al., 2025; Gao et al., 2025) have played an important role in grounding progress in concrete, reproducible tasks; however, no MI benchmarks exist for agent applications. A benchmark for agentic MI must specify not just a task and a model but an environment, a trajectory distribution, and a ground truth for both a trajectory-level failure (as opposed to a surface-level one) and what triggered such failures. In our case study, for example, nothing indicated which input the agent relied on for a given decision. It must also define what a correct mechanistic explanation looks like and provide a way to evaluate whether a proposed explanation is faithful, complete, and actionable. We believe that developing such benchmarks is crucial to measure progress and compare approaches.

Move Beyond Single Position/Component Causal Attribution Many MI methods (Conmy et al., 2023; Syed et al., 2024; Ameisen et al., 2025; Dai et al., 2022) assume failures can be localized to a small number of components that bear disproportionate causal responsibility. However, in agentic settings, computation is distributed across many components and positions, making such causal methods inapplicable. In our case study, the best joint intervention on attention heads and FFNs fixes 8.9% of hallucinations (Table 1), and that no intervention on a single component can reproduce the hallucination pattern (Figure 2). Recent work has started to tackle attribution over multi-token responses (Pan et al., 2026) and identify intervention points for distributed behaviors (Sankaranarayanan et al., 2026), however, methods capable of attributing failures distributed across multiple timesteps and environment states remain scarce.

Construct Counterfactuals in Stateful Environments Building clean interpretable counterfactuals in realistic agentic settings is rarely possible: each trajectory is shaped by a unique sequence of environment states, agent decisions, and observations, meaning that no two trials are structurally comparable in the way that activation patching requires. As our case study illustrates, valid and hallucinated actions can come from entirely different pages, tasks, and URLs and can occur at different timesteps. Removing the chosen valid action yields a hallucination in only 38% of cases and diverges the trajectory within one step (Appendix H), and even within a single step, corrupting the action list leaves the agent's choice unchanged when the selected element also appears in its history as shown in Figure 3. Addressing this requires new methods for constructing or approximating counterfactuals in dynamic environments. While some literature does exist for this problem (Tsirtsis et al., 2021; Triantafyllou et al., 2025), they assume known state spaces and comparable trajectories, which is not the case in open-ended environments with unbounded action spaces and highly variable observations. Thus, methods that are applicable to realistic agent environments are needed.

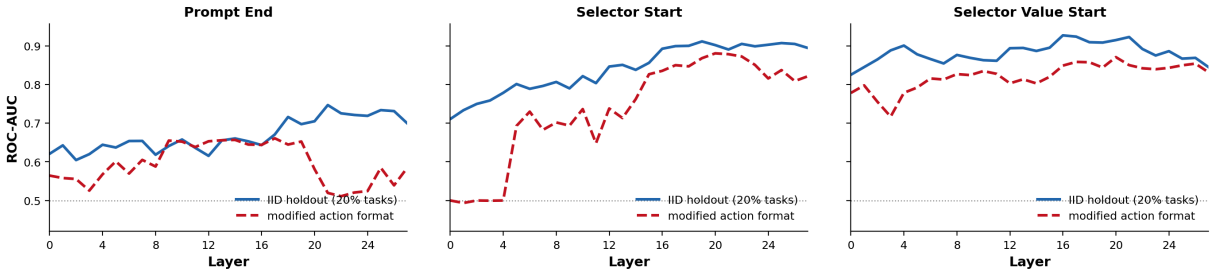


Figure 6: Per-layer ROC-AUC of probes trained to predict hallucination after modifying the action format.

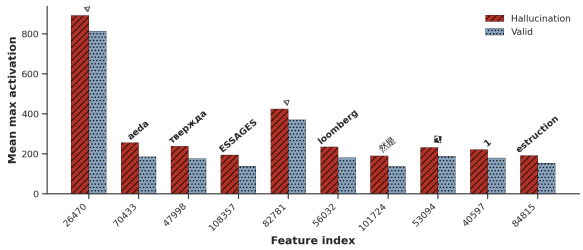


Figure 7: Top 10 SAE features by absolute difference in how strongly they activate in hallucinated vs valid cases at L27 and the token most promoted by each direction.

Feature Discovery for Agentic Distributions SAEs (Lieberum et al., 2024; Templeton et al., 2024; Bloom, 2024) are typically trained on text corpora, therefore, their features reflect text rather than interaction, and they may miss concepts that arise only when a model acts in an environment. This is illustrated by the fact that the top SAE features separating hallucinated from valid cases in our case study promote tokens unrelated to the behavior (Figure 7), and the probe direction that does predict hallucination fails to transfer when only the action format changes (Figure 6). Concepts such as tracking a goal, committing to a multi-step strategy, and modeling how the environment behaves are unlikely to be captured in textual corpora alone. Additionally, they do not have clear counterparts in current feature antologies (Lin, 2023), and they may not be well captured by the linear directions SAEs search for (Bhalla et al., 2026). Progress will, thus, require SAEs trained on agent trajectories, methods that capture richer geometric structure, and feature ontologies tailored toward agents, which would give researchers shared terms for finding and evaluating the features involved in agent decisions.

Monitoring Model Beliefs Across Timesteps Even when a failure is clearly observable at the output, it is not straightforward to determine when the model committed to the wrong strategy or how its understanding of the environment evolved to that point. Fully addressing this requires methods that can track the emergence of a plan across both layers and timesteps and identify the point at which (or if) the model's internal state transitions from open exploration to a fixed plan and which (if any) implicit heuristics the model adopts to select candidate decisions. This is harder than in static settings because the commitment may crystallize gradually across multiple timesteps, and the context that triggers it is shaped by a history of environment interactions that differs across every trajectory. A line of recent work (Yalon et al., 2026; Lubana et al., 2026; Lepori et al., 2025; Lampinen et al., 2026) examines belief evolution in LLMs, but agentic settings introduce unique challenges such as long-horizon dependencies, tool interactions, and action-observation feedback loops.

Establishing Universality Across Agent Types A foundational hypothesis in MI is that many features and circuits recur across models trained on similar tasks (Wang et al., 2025; Chughtai et al., 2023), suggesting that mechanistic insights can generalize beyond the specific model and setting on which they were discovered. Whether a similar form of universality holds across agent types remains an open question. For instance, if a circuit responsible for planning can be identified in a web agent, does an equivalent one appear in an embodied or a code generation agent? If such findings are universal across agent types, then the field can accumulate knowledge across settings.

6 Conclusion

In this paper, we investigate how far standard MI tools can go in explaining agent failures in realistic settings. Focusing on web element hallucination, we find that these tools can detect an agent's decision before it acts and steer it away from failure, but cannot explain how that decision is made. Each

tool falls short for a reason tied to properties of agents that challenge the assumptions underlying existing MI methods. We therefore argue that making MI useful for understanding agents will require methods specifically designed to account for these properties.

Acknowledgment

This project was sponsored by the National Science Foundation (NSF) Grant No. 2418580 and 2540626 and the Foresight Institute. This material is also based upon work supported by NSF under Grant No. 2232733. G. J. Stein acknowledges support from this grant. G. J. Stein is also supported by Army Research Lab under award W911NF2520011. We also thank the George Mason NLP Group for their helpful comments. This project was also supported by resources provided by the Office of Research Computing at George Mason University (URL: https://orc.gmu.edu) and funded in part by grants from the National Science Foundation (Award Number 2018631).

References

Mohamed Aghzal, Mor Geva, Wajih Ul Hassan, Gregory J. Stein, and Ziyu Yao. 2026a. Understanding LLM Agents Requires a Chain of Attribution: On the Interplay Between Mechanistic Interpretability and Behavioral Approaches. *Preprints*.

Mohamed Aghzal, Erion Plaku, and Ziyu Yao. 2024a. Can Large Language Models be Good Path Planners? A Benchmark and Investigation on Spatial-temporal Reasoning. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Mohamed Aghzal, Erion Plaku, and Ziyu Yao. 2024b. Look Further Ahead: Testing the Limits of GPT-4 in Path Planning. In *2024 IEEE 20th International Conference on Automation Science and Engineering*.

Mohamed Aghzal, Gregory J. Stein, and Ziyu Yao. 2026b. Why Do LLM-based Web Agents Fail? A Hierarchical Planning Perspective. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32157–32180, San Diego, California, United States. Association for Computational Linguistics.

Mohamed Aghzal, Xiang Yue, Erion Plaku, and Ziyu Yao. 2025. Evaluating Vision-Language Models as Evaluators in Path Planning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2023. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*.

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, and 1 others. 2025. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*.

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. 2025. Chain-of-Thought Reasoning in the Wild is not Always Faithful. In *Workshop on Reasoning and Planning for Large Language Models*.

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

Yoshua Bengio, Michael Cohen, Damiano Fornasiere, Joumana Ghosn, Pietro Greiner, Matt MacDermott, Sören Mindermann, Adam Oberman, Jesse Richardson, Oliver Richardson, Marc-Antoine Rondeau, Pierre-Luc St-Charles, and David Williams-King. 2025. Superintelligent Agents Pose Catastrophic Risks: Can Scientist AI Offer a Safer Path? *Preprint*, arXiv:2502.15657.

Leonard Bereska and Stratis Gavves. 2024. Mechanistic Interpretability for AI Safety - A Review. *Transactions on Machine Learning Research*. Survey Certification, Expert Certification.

Usha Bhalla, Thomas Fel, Can Rager, Sheridan Feucht, Tal Haklay, Daniel Wurgaft, Siddharth Boppana, Matthew Kowal, Vasudev Shyam, Jack Merullo, Atticus Geiger, and Ekdeep Singh Lubana. 2026. Do Sparse Autoencoders Capture Concept Manifolds? *Preprint*, arXiv:2604.28119.

Joseph Bloom. 2024. Open source sparse autoencoders for all residual stream layers of GPT2-small. LessWrong blog post.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, and 1 others. 2023. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. `https://transformer-circuits.pub/2023/monosemantic-features`.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. Why Do Multi-Agent LLM Systems Fail? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Xi Chen, Aske Plaat, and Niki van Stein. 2026. How does chain of thought think? mechanistic interpretability of chain-of-thought reasoning with sparse autoencoding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30297–30305.

Yongchao Chen, Jacob Arkin, Charles Dawson, Yang Zhang, Nicholas Roy, and Chuchu Fan. 2024. AutoTAMP: Autoregressive task and motion planning with LLMs as translators and checkers. In *2024 IEEE International conference on robotics and automation (ICRA)*, pages 6695–6702. IEEE.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. 2023. A Toy Model of Universality: Reverse Engineering How Networks Learn Group Operations. *Preprint*, arXiv:2302.03025.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards Automated Circuit Discovery for Mechanistic Interpretability. *Preprint*, arXiv:2304.14997.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge Neurons in Pretrained Transformers. *Preprint*, arXiv:2104.08696.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023a. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, volume 36, pages 28091–28114. Curran Associates, Inc.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023b. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*.

Zehang Deng, Yongjian Guo, Changzhou Han, Wanlun Ma, Junwu Xiong, Sheng Wen, and Yang Xiang. 2025. AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways. *ACM Comput. Surv.*, 57(7).

Zhichen Dong, Zhanhui Zhou, Zhixuan Liu, Chao Yang, and Chaochao Lu. 2025. Emergent response planning in LLMs. In *Forty-second International Conference on Machine Learning*.

Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. 2024. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *Transactions on Machine Learning Research*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, and 6 others. 2021. A Mathematical Framework for Transformer Circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Matthew Finlayson, Aaron Mueller, Sebastian Gehrmann, Stuart Shieber, Tal Linzen, and Yonatan Belinkov. 2021. Causal Analysis of Syntactic Agreement Mechanisms in Neural Language Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1828–1843, Online. Association for Computational Linguistics.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2025. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, volume 2025, pages 26721–26754.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. 2024. Have Faith in Faithfulness: Going Beyond Circuit Overlap When Finding Model Mechanisms. In *First Conference on Language Modeling*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Yifan Hou, Jiaoda Li, Yu Fei, Alessandro Stolfo, Wangchunshu Zhou, Guangtao Zeng, Antoine Bosselut, and Mrinmaya Sachan. 2023. Towards a Mechanistic Interpretation of Multi-Step Reasoning Capabilities of Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4902–4919, Singapore. Association for Computational Linguistics.

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

Xue Jiang, Yihong Dong, Lecheng Wang, Zheng Fang, Qiwei Shang, Ge Li, Zhi Jin, and Wenpin Jiao. 2024. Self-Planning Code Generation with Large Language Models. *ACM Trans. Softw. Eng. Methodol.*, 33(7).

Jaylen Jones, Zhehao Zhang, Yuting Ning, Eric Fosler-Lussier, Pierre-Luc St-Charles, Yoshua Bengio, Dawn Song, Yu Su, and Huan Sun. 2026. When Benign Inputs Lead to Severe Harms: Eliciting Unsafe Unintended Behaviors of Computer-Use Agents. *Preprint*, arXiv:2602.08235.

Subhash Kantamneni, Joshua Engels, Senthooran Rajamanoharan, Max Tegmark, and Neel Nanda. 2025. Are Sparse Autoencoders Useful? A Case Study in Sparse Probing. In *Forty-second International Conference on Machine Learning*.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Isaac Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum Stuart McDougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. 2025. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. In *Forty-second International Conference on Machine Learning*.

Andrew Kyle Lampinen, Yuxuan Li, Eghbal Hosseini, Sangnie Bhardwaj, and Murray Shanahan. 2026. Linear representations in language models can change dramatically over a conversation. *Preprint*, arXiv:2601.20834.

Jae Hee Lee, Anne Lauscher, and Stefano V. Albrecht. 2026. Towards Ethical Multi-Agent Systems of Large Language Models: A Mechanistic Interpretability Perspective. In *LLM-based Multi-Agent Systems: Towards Responsible, Reliable, and Scalable Agentic Systems*.

Michael A. Lepori, Michael Curtis Mozer, and Asma Ghandeharioun. 2025. Racing Thoughts: Explaining Contextualization Errors in Large Language Models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3020–3036, Albuquerque, New Mexico. Association for Computational Linguistics.

Yifei Li, Hanane Nour Moussa, Ziru Chen, Shijie Chen, Botao Yu, Mingyi Xue, Benjamin Burns, Tzu-Yao Chiu, Vishal Dey, Zitong Lu, and 1 others. 2025. AutoSDT: Scaling data-driven discovery tasks toward open co-scientists. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 30384–30406.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, Janos Kramar, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma Scope: Open Sparse Autoencoders Everywhere All At Once on Gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, Miami, Florida, US. Association for Computational Linguistics.

Johnny Lin. 2023. Neuronpedia: Interactive Reference and Tooling for Analyzing Neural Networks. Software available from neuronpedia.org.

Xixun Lin, Yucheng Ning, Jingwen Zhang, Yan Dong, Yilong Liu, Yongxuan Wu, Xiaohua Qi, Nan Sun, Yanmin Shang, Kun Wang, Pengfei Cao, Qingyue Wang, Lixin Zou, Xu Chen, Chuan Zhou, Jia Wu, Peng Zhang, Qingsong Wen, Shirui Pan, and 5 others. 2025. LLM-based Agents Suffer from Hallucinations: A Survey of Taxonomy, Methods, and Directions. *Preprint*, arXiv:2509.18970.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, and 8 others. 2025. On the Biology of a Large Language Model. *Transformer Circuits Thread*.

Simiao Liu, Fang Liu, Liehao Li, Xin Tan, Yinghao Zhu, Xiaoli Lian, and Li Zhang. 2025. An Empirical Study on Failures in Automated Issue Solving. *Preprint*, arXiv:2509.13941.

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, and 3 others. 2024. Agentbench: Evaluating LLMs as agents. In *The Twelfth International Conference on Learning Representations*.

Ekdeep Singh Lubana, Can Rager, Sai Sumedh R. Hindupur, Valérie Costa, Oam Patel, Sonia Krishna Murthy, Thomas Fel, Greta Tuckute, Daniel Wurgaft, Eric Bigelow, Demba E. Ba, Melanie Weber, and Aaron Mueller. 2026. Priors in time: Missing inductive biases for language model interpretability. In *The Fourteenth International Conference on Learning Representations*.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. 2024. AgentBoard: An Analytical Evaluation Board of Multi-turn LLM Agents. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Oorja Majgaonkar, Zhiwei Fei, Xiang Li, Federica Sarro, and He Ye. 2025. Understanding Code Agent Behaviour: An Empirical Study of Success and Failure Trajectories. *Preprint*, arXiv:2511.00197.

Tianyi Men, Pengfei Cao, Zhuoran Jin, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Unlocking the future: Exploring look-ahead planning mechanistic interpretability in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7713–7724.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

Lingbo Mo, Zeyi Liao, Boyuan Zheng, Yu Su, Chaowei Xiao, and Huan Sun. 2024. A Trembling House of Cards? Mapping Adversarial Attacks against Language Agents. *Preprint*, arXiv:2402.10196.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fried Fiotto-Kaufman, Tal Haklay, Michael Hanna, Jing Huang, Rohan Gupta, Yaniv Nikankin, Hadas Orgad, Nikhil Prakash, Anja Reusch, Aruna Sankaranarayanan, Shun Shao, Alessandro Stolfo, and 4 others. 2025. MIB: A Mechanistic Interpretability Benchmark. In *Forty-second International Conference on Machine Learning*.

Neel Nanda. 2022. TransformerLens.

Neel Nanda, Andrew Lee, and Martin Wattenberg. 2023. Emergent Linear Representations in World Models of Self-Supervised Sequence Models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 16–30, Singapore. Association for Computational Linguistics.

nostalgebraist. 2020. Interpreting GPT: the logit lens. https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *Preprint*, arXiv:2209.11895.

Hadas Orgad, Fazl Barez, Tal Haklay, Isabelle Lee, Marius Mosbach, Anja Reusch, Naomi Saphra, Byron Wallace, Sarah Wiegreffe, Eric Wong, Ian Tenney, and Mor Geva. 2026. Interpretability Can Be Actionable. *Preprint*, arXiv:2605.11161.

Vitória Barin Pacela, Shruti Joshi, Isabela Camacho, Simon Lacoste-Julien, and David Klindt. 2026. Stop Probing, Start Coding: Why Linear Probes and Sparse Autoencoders Fail at Compositional Generalisation. *Preprint*, arXiv:2603.28744.

Wenbo Pan, Zhichao Liu, Xianlong Wang, Haining Yu, and Xiaohua Jia. 2026. Towards Long-Horizon Interpretability: Efficient and Faithful Multi-Token Attribution for Reasoning LLMs. *Preprint*, arXiv:2602.01914.

Hanli Peng, Yongsen Zheng, Ziyao Liu, and Kwok-Yan Lam. 2026. Tool Execution Hallucination in LLM-based Agents: A Unified Taxonomy with Detection, Mitigation, and Future Directions. *TechRxiv*, 2026(0227).

Joris Postmus and Steven Abreu. 2024. Steering Large Language Models using Conceptors: Improving Addition-Based Activation Engineering. In *MINT: Foundation Model Interventions*.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, and 24 others. 2025. Qwen2.5 Technical Report. *Preprint*, arXiv:2412.15115.

Daking Rai, Yilun Zhou, Shi Feng, Abulhair Saparov, and Ziyu Yao. 2024. A Practical Review of Mechanistic Interpretability for Transformer-Based Language Models. *Preprint*, arXiv:2407.02646.

Shauli Ravfogel, Grusha Prasad, Tal Linzen, and Yoav Goldberg. 2021. Counterfactual interventions reveal the causal effect of relative clause representations on agreement prediction. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 194–209, Online. Association for Computational Linguistics.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering Llama 2 via Contrastive Activation Addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

Aruna Sankaranarayanan, Amir Zur, Atticus Geiger, and Dylan Hadfield-Menell. 2026. Activation Steering via Generative Causal Mediation. *Preprint*, arXiv:2602.16080.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. 2025. Agent Laboratory: Using LLM Agents as Research Assistants. *Preprint*, arXiv:2501.04227.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, and 10 others. 2025. Open problems in mechanistic interpretability. *Preprint*, arXiv:2501.16496.

ZhongXiang Sun, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Weijie Yu, Yang Song, and Han Li. 2025. ReDeEP: Detecting Hallucination in Retrieval-Augmented Generation via Mechanistic Interpretability. In *The Thirteenth International Conference on Learning Representations*.

Aaquib Syed, Can Rager, and Arthur Conmy. 2024. Attribution Patching Outperforms Automated Circuit Discovery. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 407–416, Miami, Florida, US. Association for Computational Linguistics.

Hao Tang, Darren Yan Key, and Kevin Ellis. 2024. WorldCoder, a Model-Based LLM Agent: Building World Models by Writing Code and Interacting with the Environment. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, and 7 others. 2024. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Stelios Triantafyllou, Aleksa Sukovic, Yasaman Zolfimoselo, and Goran Radanovic. 2025. Counterfactual Effect Decomposition in Multi-Agent Sequential Decision Making. In *Forty-second International Conference on Machine Learning*.

Stratis Tsirtsis, Abir De, and Manuel Gomez Rodriguez. 2021. Counterfactual Explanations in Sequential Decision Making Under Uncertainty. In *Advances in Neural Information Processing Systems*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Steering Language Models With Activation Engineering. *Preprint*, arXiv:2308.10248.

Muhammed Ustaomeroglu, Baris Askin, Gauri Joshi, Carlee Joe-Wong, and Guannan Qu. 2026. Internal Planning in Language Models: Characterizing Horizon and Branch Awareness. In *The Fourteenth International Conference on Learning Representations*.

Junxuan Wang, Xuyang Ge, Wentao Shu, Qiong Tang, Yunhua Zhou, Zhengfu He, and Xipeng Qiu. 2025. Towards Universality: Studying Mechanistic Similarity Across Language Model Architectures. In *The Thirteenth International Conference on Learning Representations*.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*.

Jiaqi Wei, Yuejin Yang, Xiang Zhang, Yuhan Chen, Xiang Zhuang, Zhangyang Gao, Dongzhan Zhou, Guangshuai Wang, Zhiqiang Gao, Juntai Cao, Zijie Qiu, Ming Hu, Chenglong Ma, Shixiang Tang, Junjun He, Chunfeng Song, Xuming He, Qiang Zhang, Chenyu You, and 8 others. 2025. From AI for Science to Agentic Science: A Survey on Autonomous Scientific Discovery. *Preprint*, arXiv:2508.14111.

Qi Xin, Quyu Kong, Hongyi Ji, Yue Shen, Yuqi Liu, Yan Sun, Zhilin Zhang, Zhaorong Li, Xunlong Xia, Bing Deng, and Yinqi Bai. 2024. BioInformatics Agent (BIA): Unleashing the Power of Large Language Models to Reshape Bioinformatics Workflow. *bioRxiv*.

Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and Yu Su. 2025. An illusion of progress? assessing the current state of web agents. In *Second Conference on Language Modeling*.

Noam Steinmetz Yalon, Ariel Goldstein, Liad Mudrik, and Mor Geva. 2026. Indications of Belief-Guided Agency and Meta-Cognitive Monitoring in Large Language Models. *Preprint*, arXiv:2602.02467.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik R Narasimhan, and Ofir Press. 2024. SWE-agent: Agent-computer interfaces enable automated software engineering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Gal Yona, Mor Geva, and Yossi Matias. 2026. Hallucinations Undermine Trust; Metacognition is a Way Forward. *Preprint*, arXiv:2605.01428.

Zhenhang Yuan, Shenghai Yuan, and Lihua Xie. 2026. RPMS: Enhancing LLM-Based Embodied Planning through Rule-Augmented Memory Synergy. *Preprint*, arXiv:2603.17831.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Chen Xu, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2025. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. *Preprint*, arXiv:2309.01219.

Yuxiang Zhang, Jing Chen, Junjie Wang, Yaxin Liu, Cheng Yang, Chufan Shi, Xinyu Zhu, Zihao Lin, Hanwen Wan, Yujiu Yang, Tetsuya Sakai, Tian Feng, and Hayato Yamana. 2024. ToolBeHonest: A Multi-level Hallucination Diagnostic Benchmark for Tool-Augmented Large Language Models. *Preprint*, arXiv:2406.20015.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. WebArena: A Realistic Web Environment for Building Autonomous Agents. In *The Twelfth International Conference on Learning Representations*.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. 2025. Where LLM Agents Fail and How They can Learn From Failures. *Preprint*, arXiv:2509.25370.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, and 2 others. 2023. Representation Engineering: A Top-Down Approach to AI Transparency. *Preprint*, arXiv:2310.01405.

A How Our Case Study Results Generalize Beyond Web Agents

Although the results of our case study are specific to web navigation, the same MI challenges arise across agent settings. In embodied settings, agents struggle with path and motion planning even in simple environments (Aghzal et al., 2024a,b; Chen et al., 2024), and their decisions can be driven by unreliable representations that produce plausible but environmentally ungrounded reasoning chains (Aghzal et al., 2025; Ma et al., 2024; Liu et al., 2024), leading to degenerative cycles in which an agent commits to an action sequence that violates environmental preconditions and corrupts subsequent belief states (Yuan et al., 2026; Zhu et al., 2025). In tool calling agents, hallucinations arise not from generation errors but from incorrect tool selection strategies that later steps cannot recover from (Peng et al., 2026). In code generation agents, failures are distinguished not by individual generation errors but by early diagnostic mistakes that propagate into repair strategies (Liu et al., 2025; Majgaonkar et al., 2025). Across all three settings, the causal origin of failure can be distributed across the trajectory, the environment state changes at every timestep, and failure is jointly determined by model computation and external dynamics, which are the same difficulties our case study identifies.

B Prompt Template for Data Synthesis

Agent trajectories analyzed in this work are produced by rolling out the model on Online-Mind2Web tasks. The following prompts are used.

System message.

```
You are a web navigation agent.
    Always respond with exactly two
    lines: [THOUGHT] and [ACTION]. The
     [ACTION] must be a valid JSON
    object from the available actions
    list.
```

User message (step $t = 0$). Variables <TASK>, <DOM>, and <ACTIONS_JSON> denote the dataset task string, the DOM summary, and the formatted action list, respectively.

```
Task: <TASK>

You are given a task to perform on a
    webpage. Propose the next step
    that is helpful towards achieving
    the task.
```

```
You should output only an action that
     is most likely to help achieve
    the subgoal. Each action is a
    dictionary with the following keys
    :

{
  "ACTION": "CLICK" | "HOVER" | "TYPE
    " | "SELECT",
  "SELECTOR": "text='NEWS'" | "button:
    nth-of-type(3)" | "css=div.menu >>
     text='Super Bowl'" | ...,
  "VALUE": "if any (You may need to
    provide a value for the action, e.
    g. for TYPE action)",
  "TEXT": "the text that is visible
    on the element",
  "EXPLANATION": "a short explanation
     of why this action is useful"
}

Below is the simplified
    representation of the current
    state of the webpage:
<DOM>

Below is the list of available
    actions. DO NOT OUTPUT ANY ACTIONS
     THAT ARE NOT IN THE LIST OF
    AVAILABLE ACTIONS. THE ONLY FIELD
    YOU ARE ALLOWED TO MODIFY IS THE "
    VALUE" (if a TYPE or SELECT action
    ) AND "EXPLANATION" FIELDS. YOU
    ARE NOT ALLOWED TO MODIFY ANY
    OTHER FIELDS.
IT IS VERY IMPORTANT THAT YOUR OUTPUT
     CONTAINS ONLY THE ACTION, AND
    NOTHING ELSE, THAT IS DIRECTLY
    PARSABLE AS A JSON OBJECT.

Available Actions:
<ACTIONS_JSON>
=========END OF ACTIONS=========

Continue with:
[THOUGHT] <one short sentence>
[ACTION] <JSON action object from the
     list of available actions>
```

User message (step $t > 0$). The prompt repeats the task, then up to the last three stored steps as lines `Step` k`: [THOUGHT] ... [ACTION] ... Verdict: ...`, then the current DOM and the same available-actions constraint and `Available Actions:` block as above.

C Logit Lens Analysis

We use the logit lens (nostalgebraist, 2020) to decode the model's intermediate layer representations into vocabulary predictions at the selector value position. As shown in Figure 8, by layers 21–27, hallucinating trials converge on a stable set of semantically coherent action labels (*Search*, *Enter*, *Find*, *Explore*) that are plausible website UI elements but do not exist on the current page. Interestingly, the model commonly produces the Chinese word for "Search" (搜索) at L24 before converging on the English word, indicating that the model commits to strategies not grounded in the specific website. Valid trials at these late layers converge on the correct tokens for the page. This divergence suggests that in hallucinated cases, the model has resolved a generic concept of what the action should look like to achieve the desired intent rather than grounding its prediction in the specific page.

D Single Component Attribution Results

We evaluate two forms of single-component attribution at the `selector_value_start` position. For FFN attribution, we amplify each layer's FFN output by $10\times$ before adding it to the residual stream, and measure ΔP(valid token) $=$ P(valid | clean) $-$ P(valid | FFN$_l$ amplified) averaged over grounded actions. For attention head attribution, we inject independent Gaussian noise into each head's output projection and measure the same quantity. We additionally compute a correlational signal by comparing, for each head, the maximum attention weight placed on any token across valid vs. hallucinated actions.

As shown in Figure 2a, most FFN layers have negligible or mild effects on $P(valid)$, with the notable exception of layer 20 which contributes positively to grounding. Attention head effects (Figure 2b are similarly diffuse: most heads cause only small changes to $P(valid)$, showing that no single component bears a disproportionate share of the causal load. However, the correlational difference heatmap in 2c shows that layer 22 attention heads are the most differentially activated between classes, meaning that these heads contribute jointly to grounding; which is in line with observed results in Tables 1 and 3.

E Intervention Failure Examples (Before vs. After)

We show representative *valid→hallucinated* examples. Each example starts from a selector value that is valid and becomes invalid after intervention.

Configuration A: $L25H23 \times 4$**,** $L20 \times 2$**. Aggregate effect:** 3.6% valid trials broken.

Before (baseline, valid): `filterpanel-accordion-button-0`
After (intervened, hallucinated):

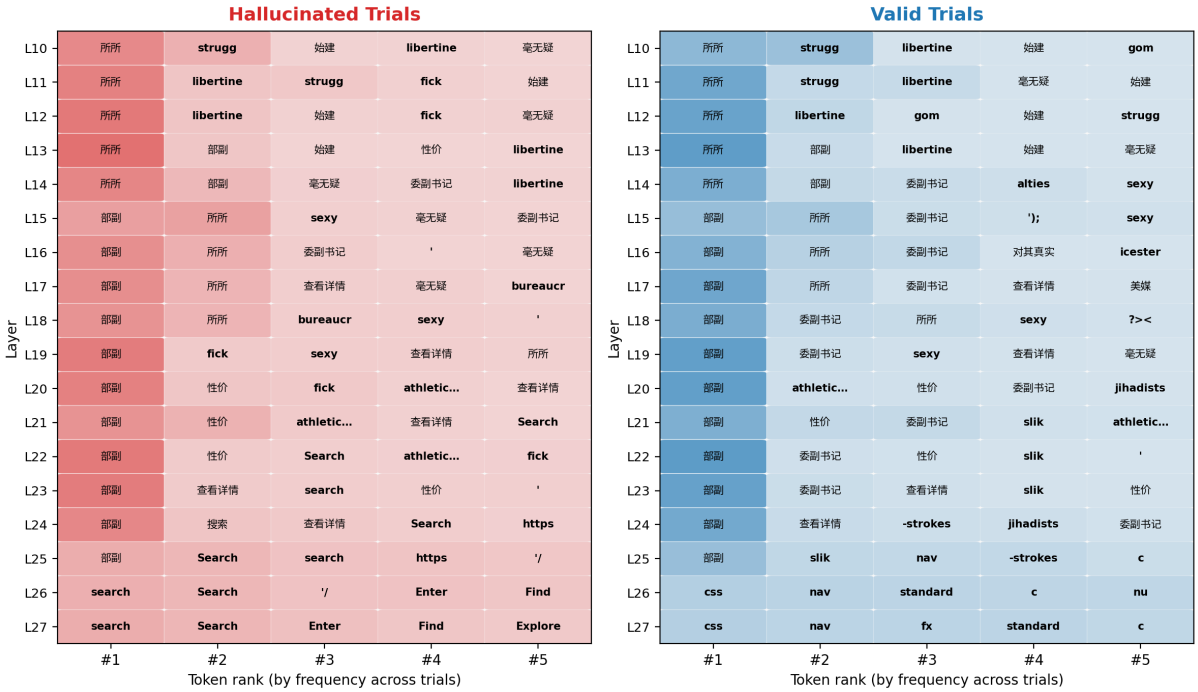


Figure 8: Top-5 most frequent tokens after logit lens projection for mid-to-late layers across valid and hallucinated cases. Cell intensities indicate how often each word appears as the top token at the corresponding layer.

`filterpanel-8-gender-1`
Before (baseline, valid): `job-seekers`
After (intervened, hallucinated): `for-job-seekers`

Configuration B: $L25H23 \times 5$, $L20 \times 5$. **Aggregate effect:** 74.9% valid trials broken.

Before (baseline, valid): `group/cta flex min-h-8 w-fit flex-row items-center gap-1 pr-4 text-neutral`
After (intervened, hallucinated): `group-cto flex min-h-3.5 min-h-3.65 h-3.5 h-6 flex-1 flex-1.5 flex-1.5`
Before (baseline, valid): `sc-cSUYbb hFerSt`
After (intervened, hallucinated): `sc-cI336o gZdZdZ`

Configuration C: $L25H23 \times 10$, $L20 \times 10$. **Aggregate effect:** 99.9% valid trials broken.

Before (baseline, valid): `Link_link__1AZfr SubNav_subNav__link__54qyo`
After (intervened, hallucinated): `SubsideItem__item__1000000000000000000000000000000000`
Before (baseline, valid): `silver`
After (intervened, hallucinated): `oneoffSilver+1000000000000000000000000000000000`

Direct FFN amplification. Layer 15:

Before (valid): `silver`
After (amplified): `_silver`
Before (valid): `interface jsx-326752785 small`
After (amplified): `guides`
Before (valid): `interface jsx-326752785 wiki-section bold underlined`
After (amplified): `VAR-10000000000000000000000000000000000000`

Layer 26:

Before (valid): `link text-p-400 no-underline hover:underline`
After (amplified): `discussion`
Before (valid): `interface jsx-326752785 small`
After (amplified): `slick-slide slick-active`
Before (valid): `interface jsx-326752785 wiki-section bold underlined`
After (amplified): `slick-slide slick-current slick-active`

F Additional Intervention Results

Table 3 reports additional intervention configurations sorted by net improvement (Fix% − Break%).

G Activation Patching Results for `Search`

To identify which parts of the input encode the commitment to a hallucinated action, we apply activation patching (Meng et al., 2022) and focus on the specific cases in which the model generated `Search` at the start of the selector value. For each such case, we corrupt a candidate span with Gaussian noise, then restore the model's clean hidden states at each layer and token offset pair and measure the recovery in the probability of the search token $P(\texttt{Search})$ at the selector value. We report the results in Table 2, and how the full causal-trace heatmaps (all traced layers and offsets) in Figures 9, 10 and 11. Each panel reports the mean intervention effect on $p(\texttt{Search})$ for residual, attention, and FFN restoration. All results are reported across three spans: the selector start, the task description, and the action list.

While all three corruptions reduce $P(\texttt{Search})$ to some degree, the action list accounts for the majority of recoverable causal signal: corrupting it produces the largest drop in $P(\texttt{Search})$ and restoring clean states at L27 recovers up to 0.36 probability mass. Notably, while the residual stream at L27 yields the strongest single restoration effect, attention and FFN individually contribute far less (0.075 and 0.067, respectively), suggesting that the information is distributed across these components rather than in either mechanism alone. However, identifying the action list as a contributor is only a partial result: unlike linguistic tasks such as factual recall, where knowing the subject token is causal immediately suggests a targeted intervention (Meng et al., 2022) (e.g. replacing the subject *Eiffel Tower* with *Colosseum* produces a shift from *Paris* to *Rome*), knowing that the action list matters does not explain which of its properties drives the commitment to the decision, making the result less interpretable and the intervention less straightforward.

H Counterfactual Trajectory Divergence

Table 4: Trajectory divergence after removing the chosen action at step 0. *Sel. match*: % of pairs where original and CF trajectories choose the same action. *DOM Jac.*: word-level Jaccard similarity between the pages the two trajectories land on. Δ = CF hall. rate − orig. hall. rate.

Step	Sel. match (%)	DOM Jac.	Δ (%)
0	–	1.000	+38.0
1	28.1	0.521	−8.5
2	22.2	0.634	−6.7
3	27.3	0.672	+12.1

To illustrate the stationarity violation described in Section 4.2, we attempt to construct counterfactuals by removing the chosen valid action from the action list and re-running the model. Table 4 summarises the results. Two key issues emerge. First, only 38% of such perturbations lead to a hallucinated output at step 0, meaning that removing the chosen action does not reliably produce the contrastive behaviour needed for activation patching; hallucination cannot be isolated to the chosen action. Second, the trajectory diverges substantially in subsequent steps: selector agreement between the original and counterfactual runs drops to around 25% by step 1 and the DOM Jaccard similarity between the pages they land on falls to 0.52, indicating that the two runs are navigating entirely different parts of the website within one step. As a result, representations built from the respective states of the two runs at step 2 or beyond cannot be meaningfully swapped, undermining the paired-trial assumption that activation patching requires.

I Probe-Based Representation Steering

To further test whether the hallucination-predictive direction in the residual stream is causal to the output, we apply representation steering. For each position and a set of candidate layers, we compute a steering vector as the normalised probe weight in un-normalised activation space:

$$\hat{\mathbf{v}} = -\frac{\mathbf{w}/\boldsymbol{\sigma}}{\|\mathbf{w}/\boldsymbol{\sigma}\|},$$

where $\mathbf{w}$ is the trained linear probe weight and $\boldsymbol{\sigma}$ is the per-feature standard deviation from the training split. The sign is chosen so that $\hat{\mathbf{v}}$ points toward the *valid* side of the probe's decision boundary. We

Table 3: Head-selective intervention runs sorted by Net Δ% (Fix % - Break %). α: attention head scale. "—" = no intervention on the specific component (`selector_value_start`).

Attn Heads	FFN	α	Fix%	Brk%	Net%
L22 (all)	L22–L24×0.5	3	8.90	2.37	6.53
L22 (all)	L22–L23×0.5	3	7.95	2.03	5.92
L22 (all)	L0,L5,L22–L24×0.5	3	8.33	2.54	5.80
L22 (all)	L22–L23×0.5, L20×1.5	3	7.01	1.52	5.48
L22 (all)	L22×0.5, L20×1.5	3	5.87	1.02	4.86
L22 (all)	L22×0.5	3	6.25	2.37	3.88
L22 (all)	L22×0.8	3	5.49	1.69	3.80
L22 (all)	—	3	5.11	1.35	3.76
L21H5, L22H1,3,4,6,7,23,24,27, L26H14	L22–L24×0.5	3	6.25	2.54	3.71
L20H21, L21H5, L22H4,6,7,23,24,27, L26H14	L13,L15,L24×0.5	3	4.73	1.86	2.87
L22 (all)	L22–L24,L27×0.5	3	7.77	5.75	2.01
L25H23	L23–L24×0.5	2	2.08	0.17	1.91
L22 (all)	L22–L27×0.5	3	8.33	7.11	1.23
L14H3,6,12, L15H1, L17H1, L19H16,19,26, L25H23, L26H27	L13,L15,L24×0.5	3	4.36	3.55	0.80
L22H0,1,3,4,24,27	—	5	5.30	4.57	0.73
L22H4, L25H23	—	3	1.70	1.02	0.69
L25H23	L10–L19,L23–L24×0.8	3	1.89	1.35	0.54
L14H3,6,12, L15H1, L17H1, L19H16,19,26, L25H23, L26H27	L22–L24×0.5	3	4.55	4.57	−0.02
L22 (all)	L22×0.8	5	7.58	7.61	−0.04
L22 (all)	—	5	7.39	7.45	−0.06
L14H3,6,12, L15H1, L17H1, L19H16,19,26, L25H23, L26H27	L13,L15,L24×0.5, L20×1.5	3	3.98	4.57	−0.59
L14H3,6,12, L15H1, L17H1, L19H16,19,26, L25H23, L26H27	—	3	3.60	4.23	−0.63
L22H0,4,24,27, L25H23	—	5	3.60	4.23	−0.63
L22H4, L25H23	L10,L15,L24×0.8, L20×1.2	5	2.46	3.21	−0.75
L14H3,6,12, L15H1, L17H1, L19H16,19,26, L25H23, L26H27	L23–L25×0.5, L20×1.5	3	4.17	5.08	−0.91
L22H0,4,24,27, L25H23	L15×0.8	5	3.03	4.40	−1.37
L22H0,1,3,4,24,27	—	5	5.30	6.77	−1.47
L25H23	L20×2	4	1.33	3.21	−1.89
L22H0,1,3,4,24,27, L25H23	L20×2	5	5.11	7.45	−2.33
L22H1,4,22,24,27, L25H23	L15×0.8	5	4.92	7.61	−2.69
L25H23	L10–L19,L23–L24×0.5	1.5	2.65	6.60	−3.95
L25H23, L26H26	L20×3	3	2.84	9.31	−6.47
L25H23	L20×3	3	2.65	9.64	−6.99
L22H0,1,3,4,24,27, L25H23	L20×3	3	3.60	11.51	−7.91
L25H23	L20×3	10	2.84	22.67	−19.83
L22H1,3,4,24,27	L5×0.5	15	7.33	27.75	−20.42
L22 (all), L27 (all)	—	3	5.87	27.58	−21.71
L22H0,1,3,4,24,27, L25H23	L20×3	10	5.49	38.58	−33.09
L22H0,1,3,4,24,27, L25H23	—	15	4.55	62.61	−58.06
L22H0,1,3,4,24,27, L25H23	L20×3	15	5.30	69.04	−63.73
L25H23	L20×5	5	1.14	73.60	−72.47
L25H23	L20×10	10	0.19	100.00	−99.81
—	L15×10	1	0.00	100.00	−100.00
—	L10–L19,L23–L24×10	1	0.00	100.00	−100.00

sweep $\alpha \in \{10, 20, 40, 60, 80\}$ and report **fix rate** (fraction of confirmed hallucinated cases that become valid) and **break rate** (fraction of confirmed valid cases that become hallucinated).

Table 5 shows results for all six (position, layer) configurations. `selector_start` at layers 17 and 18 are the most effective, achieving fix rates of 85.4% and 82.9% respectively at $\alpha = 40$ with only 7% break rate. `prompt_end` and `selector_value_start` are substantially weaker.

To further test whether the steering direction encodes a causal and untangled hallucination direction, we also evaluate across prompt formats, we apply the best-performing vector (`selector_start` L17, trained on the original JSON action-list format) to the OOD numbered-list prompt format described in Appendix J, using the same alpha sweep. Table 6 compares the two formats side by side.

Table 6: Fix and break rates for `selector_start` L17 steering on the original JSON format vs. the OOD numbered-list format. The same steering vector is used in both conditions.

	$\alpha = 10$	$\alpha = 20$	$\alpha = 40$
Original format	43.9 / 0.0	75.6 / 3.1	**85.4** / 7.0
OOD format	19.4 / 0.0	41.9 / 2.3	**58.1** / 7.0
Δ fix	−24.5	−33.7	−27.3

Fix rate drops by ∼27 percentage points at $\alpha =$

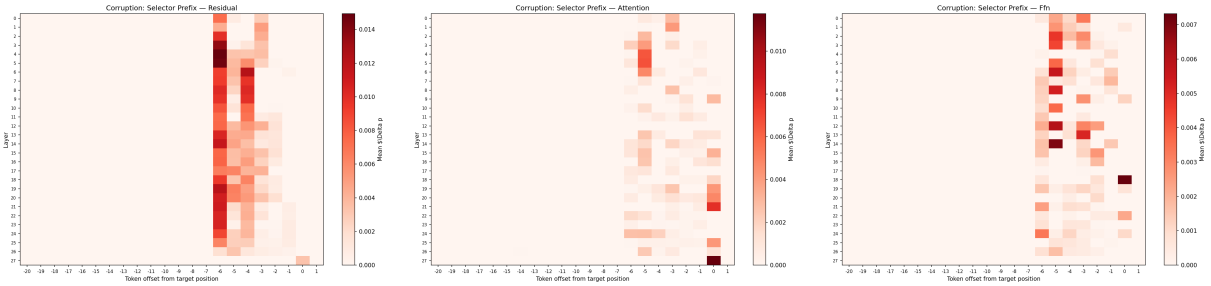


Figure 9: Full `Search` causal trace with corruption span = selector start (left: residual, middle: attention, right: FFN).

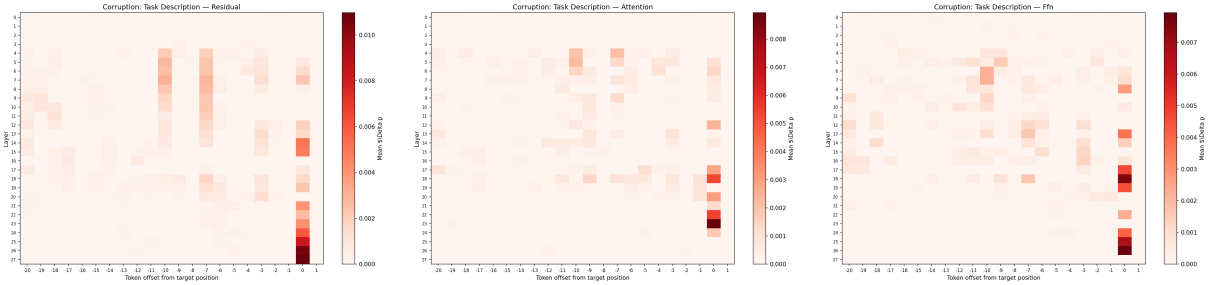


Figure 10: Full `Search` causal trace with corruption span = task description (left: residual, middle: attention, right: FFN).

40, with break rate nearly identical across formats. The consistent gap across all alpha values indicates that the steering effect is entangled with prompt format rather than with the hallucination direction by itself, which is also the result observed in the probe generalisation results (Figure 6).

J Out-of-Distribution Prompt Format

The following prompts are used for the distributional mismatch evaluation (Figure 6): we run the model on the same 300 tasks with identical selectors but this different surrounding syntax, and collect activations at the same three positions. The available action list is reformatted from a JSON array into a numbered markdown list; the model is instructed to respond in the same list format rather than JSON.

System message.

```
You are a web navigation agent.
    Always respond with exactly two
    lines: [THOUGHT] and [ACTION]. The
     [ACTION] must use the numbered
    list format: [ACTION] [TYPE] "
    visible text"
   selector: <selector exactly as
    shown>
```

User message. Variables `<TASK>`, `<DOM>`, and `<ACTIONS_LIST>` denote the dataset task string, the DOM summary, and the numbered action list, respectively.

```
Task: <TASK>

You are given a task to perform on a
    webpage. Propose the next step
    that is helpful towards achieving
    the task.

You should output only one action.
    Your response must follow this
    exact format:

[THOUGHT] <one short sentence>
[ACTION] [ACTION_TYPE] "visible text
    on element"
   selector: <selector exactly as
    shown in the list>
   value: <only include this line for
     TYPE or SELECT actions>

Below is the simplified
    representation of the current
    state of the webpage:
<DOM>

Below is the list of available
    actions. DO NOT output an action
    not in this list. Copy the
    selector EXACTLY as shown,
    character for character.

Available Actions:
1. [CLICK] "visible text"
   selector: <xpath>
```

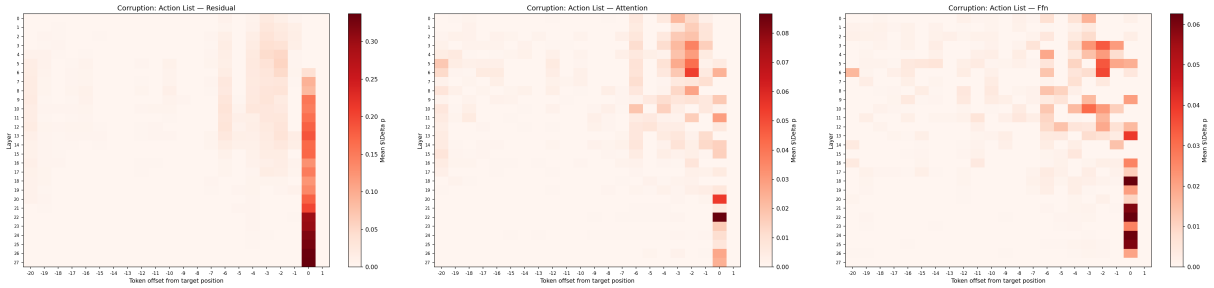


Figure 11: Full `Search` causal trace with corruption span = action list (left: residual, middle: attention, right: FFN).

Table 5: Fix and break rates for probe-based steering across (position, layer) configurations. Each cell shows fix / break %. Bold marks the best fix rate for each configuration.

Position	Layer	$\alpha = 10$	$\alpha = 20$	$\alpha = 40$	$\alpha = 60$	$\alpha = 80$
`prompt_end`	18	14.6 / 0.8	36.6 / 1.5	**41.5** / 10.8	29.3 / 29.5	0.0 / 100.0
`prompt_end`	26	0.0 / 0.0	2.4 / 2.3	**4.9** / 3.1	4.9 / 3.1	12.2 / 2.3
`selector_start`	17	43.9 / 0.0	75.6 / 3.1	**85.4** / 7.0	48.8 / 30.2	0.0 / 96.9
`selector_start`	18	34.2 / 3.1	56.1 / 2.3	82.9 / 7.0	**85.4** / 14.0	2.4 / 91.5
`selector_value_start`	18	14.6 / 4.6	46.3 / 10.1	**46.3** / 19.4	12.2 / 57.4	0.0 / 100.0
`selector_value_start`	21	9.8 / 3.9	14.6 / 7.0	17.1 / 29.5	**29.3** / 33.3	14.6 / 40.3

```
2. [TYPE] "visible text"
   selector: <xpath>
   value: <value>

...
=========END OF ACTIONS=========

Continue with:
[THOUGHT] <one short sentence>
[ACTION] [ACTION_TYPE] "visible text"
   selector: <selector exactly as
    shown>
```

K SAE Analysis

We apply a pretrained SAE for `Qwen2.5-7B-Instruct`[3] using a BatchTopK architecture with a dictionary of 131,072 features and a sparsity of k=64 active features per token. We analyze layer 27 (the layer with the highest causal recovery in Table 2) at the `selector_value_start` position. For each instance, we pass the residual stream representation through the SAE and take the maximum activation across all tokens per feature, yielding a scalar indicating whether each feature activates at least once at that position. We then average these values across instances within each class (hallucinated vs. valid) and identify which tokens each feature most promotes via its dot product with the unembedding matrix. As shown in Figure 7, the top-10 features by absolute difference between classes mostly promote tokens unrelated to the observed hallucination behavior and do not yield an interpretable explanation. This highlights how SAEs pretrained on text can struggle to encode features that emerge in agent interaction.

[3] https://huggingface.co/andyrdt/saes-qwen2.5-7b-instruct

L Attributions and Licenses

Model. All experiments use `Qwen2.5-7B-Instruct` (Qwen et al., 2025), released under the Apache 2.0 license.

Dataset. Agent trajectories are collected on Online-Mind2Web (Xue et al., 2025), released under a Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Use of this dataset also requires attribution to the original Mind2Web benchmark (Deng et al., 2023a). We use this dataset in accordance with its intended use, which consists of web agent evaluation for research purposes.

Sparse Autoencoder. We use pretrained SAE checkpoints for `Qwen2.5-7B-Instruct` from https://huggingface.co/andyrdt/saes-qwen2.5-7b-instruct, released under the Apache 2.0 license.

Analysis Infrastructure. We use TransformerLens (Nanda, 2022), released under the MIT license for activation caching and hook-based interventions. We also use common open-source scientific libraries, namely PyTorch (BSD-3-Clause), Hugging Face Transformers (Apache 2.0), scikit-learn (BSD-3-Clause), NumPy (BSD-3-Clause), and Matplotlib (PSF).

Compute Resources. All experiments and trajectory collection were run on a single NVIDIA A100 80GB GPU.