Honest Lying: Understanding Memory Confabulation in Reflexive Agents

Prakhar Dixit [1] **Sadia Kamal** [1] **Tim Oates** [1]

Abstract

Reflexion-style agents rely on self-generated reflections as memory, implicitly assuming that agents can accurately diagnose their own failures. We show that this assumption can fail systematically: across ALFWorld and HumanEval, agents store confident but incorrect interpretations of the task and continue acting on them across trials, even though the environment resets to the correct task each time. We call this failure mode *memory confabulation* and introduce the *Reflection Repetition Rate* (RRR), a log-based metric that detects repeated reliance on incorrect reflective content. Using RRR, we identify 16 frozen environments in ALFWorld, where 0 of 121 reflections mention the correct target object, and 4 analogous cases in HumanEval. Our mitigation replaces open-ended self-diagnosis with programmatic extraction of trajectory-level failure signals, increasing correct object mention from 0% to 86%, reducing RRR from 0.64 to 0.10, and solving 3 of 16 frozen ALFWorld environments, suggesting that reflective memory can reinforce false beliefs rather than correct them.

1. Introduction

Foundation-model agents increasingly operate in settings where they must learn from experience across repeated trials (Durante et al., 2025). Reflexion (Shinn et al., 2023) is a representative approach: after failure, an agent writes a natural-language reflection and retrieves it in later attempts. This design assumes that self-generated reflection produces useful failure diagnoses. When the assumption holds, the agent can convert a failed action into a reusable lesson.

We show that this assumption can fail systematically. Reflexion agents may write confident but incorrect accounts of the task, store them as memory, and reuse them across trials even when the environment re-presents the correct task at every reset. We call this failure mode **memory confabulation**: persistent, self-reinforcing false beliefs written into reflective memory and acted upon despite contradictory task evidence. The term follows cognitive accounts of confabulation as a failure of reality monitoring, where internally generated information is mistaken for observed information (Johnson & Raye, 1998; Schnider, 2001; Chrobak & Zaragoza, 2009).

Memory confabulation differs from hallucination. Hallucination is typically a single-generation error, while memory confabulation is a multi-trial failure: the false content is stored, retrieved, acted upon, and reinforced by later reflections (Ji et al., 2023b; Maynez et al., 2020a). This distinction matters for agent design because the error is not merely produced once; it becomes part of the agent's future context. Therefore, we ask: *can reflective agents rely on their own self-generated lessons, or does verbal self-diagnosis create false beliefs in memory that persist across trials and degrade rather than improve performance?*. **Our main contributions are**:

- We identify and operationalize **memory confabulation**: false self-diagnoses written into reflective memory and reused across trials despite contradictory task evidence.
- We introduce the **Reflection Repetition Rate (RRR)**, a log-based diagnostic for frozen reflective memory that strongly correlates with trials-to-solve in ALFWorld (**?**) ($r = 0.808$).
- We provide cross-domain evidence from ALFWorld and HumanEval (Chen et al., 2021), finding that 0/121 reflections in 16 frozen ALFWorld environments mention the correct target object and that 4 HumanEval problems repeat near-identical wrong diagnoses.
- We show through a no-memory ablation that reflective memory can be actively harmful in some environments the agent can otherwise solve.
- We propose **programmatic feedback extraction**, which replaces open-ended self-diagnosis with parsed

[1]Department of Computer Science, University of Maryland Baltimore County, Baltimore, MD, USA. Correspondence to: Prakhar Dixit <pdixit1@umbc.edu>, Sadia Kamal <sadia1402@umbc.edu>, Tim Oates <oates@cs.umbc.edu>.

The Workshop on Failure Modes of Agentic AI(FAGEN) at 43rd International Conference on Machine Learning,Seoul, South Korea.
Copyright 2026 by the author(s).

trajectory-level failure signals, raising correct object mention from 0% to 86%, reducing RRR from 0.64 to 0.10, and solving 3/16 frozen ALFWorld environments.

2. Background

Reflexion. Reflexion (Shinn et al., 2023) extends ReAct-style agents (Yao et al., 2022) with a verbal reinforcement mechanism. After each failed task attempt, a language model generates a natural-language self-critique which is prepended to the agent's context on subsequent attempts. No gradient updates are performed; learning is entirely mediated by the context. Reflexion achieves 91% pass@1 on HumanEval versus 80% for GPT-4 (Hurst et al., 2024) without reflection, demonstrating the power of the mechanism when it works correctly. However, Reflexion assumes the reflection step produces causally correct diagnoses. We show this assumption fails systematically when the feedback signal is binary (pass/fail) and the task requires multi-step manipulation.

Memory in LLM agents. Recent surveys (Zhang et al., 2025; Du, 2026) formalize agent memory as a write–manage–read loop and identify reflective self-improvement as a distinct memory mechanism family. Du (2026) note that the central risk of reflective memory is self-reinforcing error: if the agent falsely concludes that an approach always fails, it will never test that approach again. Our work operationalizes this risk empirically and identifies the specific triggering condition: binary feedback that prevents causal diagnosis (Zhang et al., 2026).

ExpeL and rule-library agents. ExpeL (Zhao et al., 2024) extends Reflexion with a shared rule library: experience is distilled into globally-applicable rules via unconstrained LLM critique on failure trajectories. This shares the same structural vulnerability as Reflexion, but with amplified consequences: where Reflexion confabulates per-task reflections, a confabulated rule with two AGREE votes becomes entrenched and is applied across every evaluation environment. We discuss this generalization in Section 6.

Hallucination and confabulation. Hallucination in LLMs has been extensively studied as a single-generation failure (Ji et al., 2023a; Maynez et al., 2020b). Memory confabulation is structurally distinct: the false content is stored, retrieved, and acted upon persistently across multiple trials.

3. Problem Formulation

3.1. Operational Definition

Let an agent operate on task τ over trials $t = 0, 1, \ldots, T$. After each failure at trial t, the agent generates reflection r_t from the trajectory and stores it: $M_{t+1} = M_t \cup \{r_t\}$. At trial $t+1$, the agent retrieves from M_{t+1} to inform its actions.

Definition (Memory Confabulation). A reflection r_t is *confabulated* if it fails to mention the correct target object of task τ, i.e., $\mathrm{obj}(\tau) \notin r_t$, where $\mathrm{obj}(\tau)$ extracts the target object from the task description presented to the agent at the start of every episode.

This definition is operationalizable from existing logs without new experiments, using only the gamefile directory name (which encodes task structure) and the stored reflection text.

3.2. Reflection Repetition Rate (RRR)

For an environment with memory $M = \{r_0, \ldots, r_n\}$, we define:

$$\mathrm{RRR} = \frac{|\{r_i : i \geq 1,\ \exists j < i,\ \mathrm{sim}(r_i, r_j) \geq 0.85\}|}{|M| - 1} \quad (1)$$

where sim is SequenceMatcher string similarity (Musser & Nishanov, 2008). $\mathrm{RRR} = 0$ indicates all reflections are novel; $\mathrm{RRR} = 1$ indicates all reflections are near-copies of earlier ones. We use 0.85 as the similarity threshold for near-duplication.

We define an environment as exhibiting **frozen reflective memory** when $\mathrm{RRR} \geq 0.5$, meaning that at least half of the reflections after the first are near-duplicates of earlier reflections. This threshold identifies cases where reflective memory stops evolving across trials and repeatedly reuses the same content. We use the term *frozen environment* as shorthand for an environment whose reflective memory is frozen, not to imply that the underlying task or simulator state is fixed.

3.3. Two Failure Categories

Our analysis distinguishes two categories among frozen environments ($\mathrm{RRR} \geq 0.5$):

Memory-harmful: removing memory allows the agent to solve the task faster, proving stored reflections were actively misleading.

Task-hard: the agent fails even without memory, indicating a capability gap independent of memory quality.

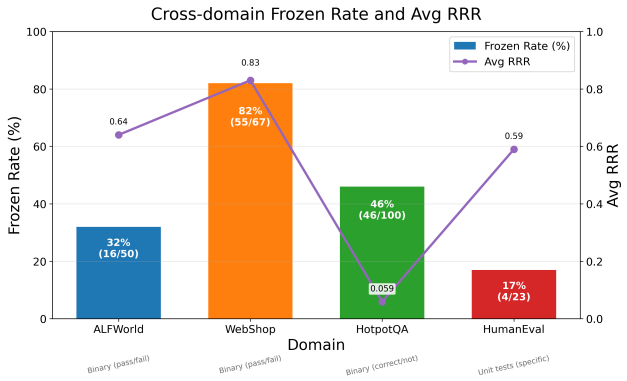

Figure 1. Cross-domain frozen rate and average RRR by feedback type. Binary outcome-level feedback is associated with higher frozen-memory rates, while more specific feedback supports targeted self-correction.

4. Evidence of Memory Confabulation

4.1. RRR Analysis

Using pre-existing Reflexion run logs (134 environments, 15 trials, gpt-3.5-turbo (Brown et al., 2020) on ALFWorld `eval_out_of_distribution`), we compute RRR for all 50 environments that required at least one reflection. We find 16 of 50 environments (32%) exhibit frozen memory (RRR ≥ 0.5). Frozen environments required an average of 7.6 trials to solve, versus 1.5 trials for environments with diverse, evolving reflections. The Spearman correlation between RRR and trials-to-solve is $r = 0.808$ ($p < 0.0001$), suggesting that frozen memory is not incidental but is a reliable predictor of agent failure.

4.2. Task-Object Confabulation

We extracted the target object from each frozen environment's gamefile directory name and checked whether each reflection mentioned that object. For example,

```
pick_cool_then_place_in_recep-Mug-None-C⌋
↪  offeeMachine-10
```

encodes object `Mug` and destination `CoffeeMachine`. Across all 16 frozen environments, 0 of 121 reflections mention the correct target object.

The most striking case is `env_22` (task: put a cool Mug in CoffeeMachine), where all 14 reflections reference *tomato* and *microwave*—a completely different task. The agent fabricated a false task identity after trial 0 and pursued it for 14 consecutive trials, ignoring the correct task description re-presented at every episode reset.

A second case, `env_35` (task: examine the Mug with the desklamp), illustrates two compounding confabulation patterns: *location confabulation*, where the agent follows a search sequence from a previous environment layout across trials 0–2, and *action confabulation*, where the agent attempts to use an object without navigating to it because its memory incorrectly anchors its believed location.

Table 1. Cross-domain frozen memory rates by feedback type. All results are computed from pre-existing Reflexion run logs.

Domain	Feedback	Frozen	Avg RRR
ALFWorld	Binary	32% (16/50)	0.64
WebShop	Binary	82% (55/67)	0.83
HotpotQA	Binary	46% (46/100)	0.059
HumanEval	Unit tests	17% (4/23)	0.59

Two distinct confabulation patterns emerge across the 16 frozen environments:

- **Full task substitution**: both object and destination are replaced (`env_22`, `env_20`, `env_41`).
- **Object substitution only**: the destination is correctly remembered but the object is replaced (`env_118`, `env_113`, `env_106`).

4.3. Cross-Domain Replication

We replicate the frozen memory analysis across three additional Reflexion domains available in the same repository, requiring no new experiments or API calls. Results are shown in Figure 1 and Table 1.

The results reveal a consistent pattern: domains with binary, outcome-level feedback (ALFWorld, WebShop (Yao et al., 2022), HotpotQA (Yang et al., 2018)) exhibit substantially higher frozen rates and lower RRR than domains with specific, step-level feedback (HumanEval). HotpotQA is the most acute case. Despite seven trials of reflection on 100 multi-hop questions, the agent corrected a previously wrong answer only 5.9% of the time per trial transition—compared to 64% for ALFWorld and 83% for WebShop. This is because binary correct/wrong feedback on open-ended reasoning provides no signal about *which step* of a multi-hop chain failed, leaving the agent unable to target its self-correction. The 46% frozen rate (46 of 100 questions never answered correctly across seven trials) reflects this ceiling effect.

WebShop confabulation manifests differently from ALFWorld. Rather than substituting the wrong task object, agents exhibit *symptom confabulation*: 56% (121/218) of frozen reflections describe what went wrong ("I clicked the wrong item") without diagnosing why—which size, color, or price constraint was violated. This is a distinct surface form of the same root cause: binary feedback contains no step-level information, so reflections recapitulate failure without identifying it.

HumanEval's comparatively low persistent failure rate (17%) supports the contrastive hypothesis. Unit test feedback names the exact assertion that failed, giving the agent a precise error to reason about. Reflections in this domain are correspondingly targeted, and the agent self-corrects at nearly ten times the rate observed on HotpotQA.

4.4. Evidence of Memory Confabulation

Correlation between RRR and trials-to-solve is not causal evidence; frozen environments may simply be harder tasks. To establish causality, we re-ran all 16 frozen environments with memory completely wiped before each trial (gpt-3.5-turbo,max 10 trials, ALFWorld `eval_out_of_distribution`).

Results. The 16 frozen environments split cleanly into two categories:

Memory-harmful (2/16): `env_31` (`look_at_obj_in_light`) and `env_97` (`look_at_obj_in_light`). Without memory, both solve in 1 trial. With standard Reflexion memory: 7 and 8 trials respectively. **Delta: +6 and +7 trials.**

Task-hard (14/16): All `pick_heat`, `pick_cool`, `pick_clean`, and `pick_and_place` environments. The agent fails within 10 trials even without memory, indicating a capability gap independent of memory quality.

This finding establishes that memory confabulation has two distinct effects: it directly causes failure in tasks the agent could otherwise solve (memory-harmful category), and it compounds an existing capability gap in hard tasks (task-hard category).

5. Mitigation and Results

We test two mitigation strategies on all 16 frozen environments. Both target the same root cause: binary feedback preventing accurate causal diagnosis.

5.1. Grounded Reflection

We modified the reflection prompt to require a structured three-part response:

```
FAILED STEP:
  [exact action + environment response]
ROOT CAUSE:
  [one sentence why that action failed]
NEW PLAN:
  [specific plan naming exact objects]
```

This forces the agent to quote a specific trace step before planning. Result: grounded reflection matches no-memory performance on look_at_obj environments (TTS = 1), confirming confabulation is prevented, but produces no improvement on task-hard environments.

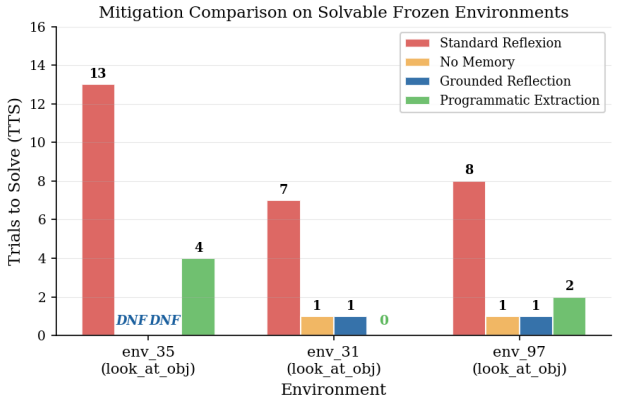

Figure 2. Mitigation comparison across frozen ALFWorld environments. Programmatic feedback extraction substantially reduces repeated reliance on incorrect reflections, while solving additional environments beyond no-memory and grounded-reflection baselines.

5.2. Programmatic Feedback Extraction

Motivated by the cross-domain finding that unit-test feedback produces 17% frozen rate versus 32–82% for binary feedback, we implemented a trajectory parser that extracts failure steps programmatically rather than asking the agent to self-diagnose. The parser identifies: (1) actions that received "Nothing happens" responses, and (2) repeated identical actions indicating a loop. These are injected directly into the reflection prompt. The full prompt template can be found in Appendix A.

```
TASK: [task description]
FAILURES FROM TRACE:
  1. Action: put mug 1 in coffeemachine 1
     Response: Nothing happens.
```

This replicates what unit tests do for HumanEval: providing grounded failure evidence instead of requiring self-diagnosis.

5.3. ALFWorld Results

Figure 2 summarizes the mitigation results across all five conditions. Table 2 presents the full comparison across five conditions on all 16 frozen ALFWorld environments. Under standard Reflexion, all 16 environments exhibit fully confabulated memory: 0 of 121 reflections mention the correct target object, and none solve within the 10-trial budget used for all mitigation conditions. Programmatic feedback extraction raises the correct object mention rate

to 134/156 (86%) and reduces average RRR from 0.64 to 0.10, confirming that the mitigation breaks frozen memory patterns even on environments that remain unsolvable due to capability gaps. The 22 remaining mention misses are concentrated in `env_4`, where the search for `SoapBar` produces partial regex matches.

Removing memory entirely solves 2/16 environments (`env_31`, `env_97`), both `look_at_obj` tasks that complete in 1 trial without confabulated reflections. Grounded reflection matches this baseline. Programmatic extraction solves 3/16, uniquely resolving `env_35` which all other conditions fail on. Replication with gpt-4o-mini eliminates task-identity confabulation (100% object mention rate) but solves only 2/16, identical to the no-memory baseline, confirming that confabulation prevention and task-level capability are independent failure axes.

env_35 case study. This environment is the most informative result. It was DNF under both no-memory and grounded reflection baselines, but programmatic feedback extraction solved it in 4 trials. The trajectory shows that trials 0–2 were dominated by a frozen location plan derived from a previous task layout (location confabulation). Programmatic extraction broke this freeze by surfacing the specific "Nothing happens" response, enabling the agent to search for an alternative desklamp and solve the task in 33 steps at trial 4.

5.4. HumanEval Results

To assess whether programmatic feedback extraction generalizes across task types, we apply the same principle to HumanEval code generation, a structurally different domain where the agent writes Python functions and receives unit-test feedback. The extraction is domain-adapted: instead of parsing `Nothing happens`, we parse the failing `assert` statement and error type from the test output.

Confabulation in code generation. We identify 4 frozen HumanEval problems ($\mathrm{RRR} \geq 0.5$) from the pre-existing Reflexion logs. In all four cases, the agent produces near-identical reflections across 9 trials without diagnosing the specific failing input. For example, HumanEval/32 (`find_zero` binary search) generates the same diagnosis — "does not update lower_bound and upper_bound" — across all 9 reflections, never identifying *which* input triggers the infinite loop or *why* the bounds fail to converge.

Domain-adapted extraction. For HumanEval, the programmatic extractor identifies:

1. The failing `assert` statement (e.g., `assert candidate(1000) == "1"`)
2. The error type (`AssertionError`, `TypeError`, etc.) and its associated traceback message

These are injected into the reflection prompt in place of the agent's self-diagnosis, mirroring the ALFWorld `Nothing happens` extraction. The agent is *told* which specific test case failed rather than asked to infer it.

Table 3 shows results across the 4 frozen problems.

The key findings on HumanEval mirror the ALFWorld results:

Grounding works: 18/18 (100%) of generated reflections mention the specific error type, compared to the near-identical vague diagnoses produced by standard Reflexion. This mirrors the 0/121 → 134/156 object mention improvement on ALFWorld.

RRR reduces: Average RRR drops from 0.59 to 0.44, confirming that programmatic extraction breaks frozen memory patterns in code generation.

Capability gaps persist: HumanEval/32 and /84 fail even with grounded reflections — the binary search implementation error and digit-sum binary conversion require algorithmic insight that no memory-level intervention provides. This mirrors the DNF pattern on ALFWorld pick_* tasks.

One regression: HumanEval/77 regressed from solved to unsolved. The structured reflection prompt disrupted a working solution strategy — a cost consistent with the memory-harmful vs. task-hard finding on ALFWorld, and a reminder that memory interventions carry risk even when they improve grounding.

Note that HumanEval/84 and /87 show 0/0 grounded reflections because both solved at trial 0 (no reflections generated). Their RRR drops to 0.00 as a result, not due to the mitigation.

Cross-domain parallel. Table 4 summarises the structural parallel between the two mitigation experiments.

6. Discussion

6.1. The Feedback Granularity Hypothesis

Our cross-domain analysis points to a clear mechanism: binary feedback (pass/fail) gives the agent no information about which step in a multi-step trajectory was wrong. Faced with this information vacuum, the reflection generator produces a plausible-sounding but causally wrong diagnosis. Unit tests provide step-level feedback — which test case failed, which input produced the wrong output — and this reduces confabulation from 32–82% to 17%. This suggests a systems-level mitigation that no prompt intervention can replicate: richer environment feedback. For ALFWorld specifically, this would mean returning explanations for

Table 2. Trials-to-solve across five conditions for all 16 frozen environments. All mitigation conditions use a 10-trial budget. DNF means the agent did not finish within 10 trials. Orig column shows baseline TTS from the original 15-trial Reflexion logs where all 16 environments exhibited confabulated memory (0/121 reflections mention the correct object).

Env	Task Type	Orig(All Confabulated)	No Mem	Grounded	Enriched	gpt-4o-mini
env_35	look_at_obj	13†	DNF	DNF	**4**	DNF
env_31	look_at_obj	7†	1	1	**1**	1
env_97	look_at_obj	8†	1	1	**2**	1
env_4	pick_clean	3†	DNF	DNF	DNF	DNF
env_20	pick_and_place	6†	DNF	DNF	DNF	DNF
env_22	pick_cool	14†	DNF	DNF	DNF	DNF
env_40	pick_and_place	6†	DNF	DNF	DNF	DNF
env_41	pick_heat	13†	DNF	DNF	DNF	DNF
env_56	pick_cool	2†	DNF	DNF	DNF	DNF
env_77	pick_heat	6†	DNF	DNF	DNF	DNF
env_80	pick_heat	5†	DNF	DNF	DNF	DNF
env_82	pick_heat	7†	DNF	DNF	DNF	DNF
env_86	pick_two_obj	4†	DNF	DNF	DNF	DNF
env_106	pick_and_place	6†	DNF	DNF	DNF	DNF
env_113	pick_cool	9†	DNF	DNF	DNF	DNF
env_118	pick_and_place	12†	DNF	DNF	DNF	DNF
Solved		0/16	2/16	2/16	**3/16**	2/16
Object mention rate		0%	—	—	**86%**	**100%**
Avg RRR		0.64	—	—	**0.10**	0.53

Table 3. HumanEval results with programmatic feedback extraction on 4 frozen problems.

Task	Orig Solved	Enriched Solved	Orig RRR	New RRR	Error Mention
HumanEval/32	✗	✗	0.50	0.75	9/9
HumanEval/77	✓	✗	0.50	1.00	9/9
HumanEval/84	✗	✓	0.75	0.00	0/0
HumanEval/87	✓	✓	0.62	0.00	0/0
Summary	2/4	2/4	0.59	**0.44**	**18/18**

"Nothing happens" responses rather than a bare failure signal.

6.2. Model Capability and Confabulation

To test whether confabulation is an artifact of model weakness, we replicated standard Reflexion on all 16 frozen environments using gpt-4o-mini. Table 5 summarizes the comparison.

gpt-4o-mini eliminates task-identity confabulation: all 142 reflections correctly name the target object, compared to 0/121 for gpt-3.5-turbo. Yet it solves only 2 of 16 frozen environments, identical to the no-memory ablation baseline.

Table 4. Structural parallel between ALFWorld and HumanEval mitigation experiments.

	ALFWorld	HumanEval
Task type	Household navigation	Code generation
Feedback type	Binary (pass/fail)	Unit tests (specific)
Failure signal	`Nothing happens`	`AssertionError`
Extracted info	Failed action and response	Failing assert and error type
Confab. pattern	Wrong task object	Vague or wrong diagnosis
Frozen rate	32% (16/50)	17% (4/23)
Mention rate	0% → 86%	0% → 100%
RRR change	0.64 → **0.10**	0.59 → 0.44
Solved	3/16	2/4
Capability gaps	13/16 DNF	2/4 DNF

This confirms that task-identity confabulation and task-level capability are independent failure axes. A stronger model fixes the write-path failure (correct object in every reflection) but cannot fix the execution-step gap (the 13 `pick_*` environments remain unsolvable regardless of memory quality).

Notably, gpt-4o-mini introduces two new confabulation

Table 5. gpt-3.5-turbo vs gpt-4o-mini on 16 frozen environments under standard Reflexion (no mitigation).

	gpt-3.5	**gpt-4o-mini**
Solved	0/16	2/16
Object mention rate	0%	100%
Avg RRR	0.64	0.53

types absent in gpt-3.5-turbo. First, *memory-format confabulation*: gpt-4o-mini generates structured numbered-list reflections ("1. Clarify Task Requirements… 2. Verify Command Syntax…") that leak into action generation. The agent outputs planning text as actions, receiving `Nothing happens` for every step (env_118, 9 consecutive trials). Second, when tested with gpt-4o-mini under enriched feedback, *action-space confabulation* emerges: the agent generates natural-language actions (`check shelf 1 for book`) instead of valid ALFWorld syntax (`go to shelf 1`), looping on invalid commands for 47 steps (env_97). These findings suggest that stronger models trade one confabulation type for another rather than eliminating the structural vulnerability.

6.3. Generalization to Rule-Library Agents

Memory confabulation is not a Reflexion-specific bug. It is a structural vulnerability of any agent that writes natural language to persistent memory based on self-assessment of binary failure signals. The common precondition is: *binary feedback + self-generated reflection + persistent retrieval*.

ExpeL (Zhao et al., 2024) faces this vulnerability with amplified consequences. Where Reflexion confabulates per-task reflections, ExpeL's rule extraction mechanism which calls the same unconstrained LLM critique on failure trajectories can produce globally-applied confabulated rules. A confabulated rule with two AGREE votes becomes entrenched and is applied across every evaluation environment, multiplying the harm of a single confabulation event.

6.4. Limitations

Our analysis has several limitations. We study confabulation exclusively in Reflexion, a single reflective agent architecture. While the structural vulnerability (binary feedback + self-generated reflection + persistent retrieval) should apply to any agent satisfying these conditions, empirical validation on other reflective systems such as ExpeL (Zhao et al., 2024) or LATS (Zhou et al., 2024) remains future work. Our gpt-4o-mini replication confirms the vulnerability persists across model sizes, but generalizability to other architectures cannot be assumed. The causal ablation is limited to 2 memory-harmful environments out of 16 frozen; while the effect is large (+6 and +7 trials), the small sample limits statistical power. Similarly, the HumanEval analysis covers only 4 frozen problems, making the cross-domain parallel structurally consistent but underpowered compared to the ALFWorld analysis (16 environments, 121 reflections). Our operational definition uses target-object mention as the grounding signal for detecting confabulation, which is a sufficient but not necessary condition: a reflection could mention the correct object while still misdiagnosing the failure cause, or confabulate in dimensions we do not measure such as wrong location or wrong action sequence. Finally, all experiments use pre-existing logs from the public Reflexion repository, fixing the model backbone (gpt-3.5-turbo), environment version, and hyperparameters; results may differ under other configurations.

7. Conclusion

We have shown that Reflexion agents systematically confabulate task identity in their reflective memory. Across 16 frozen environments and 121 reflections, the correct target object was mentioned zero times. Two environments require 7-8 trials with standard memory but solve in 1 trial without memory, providing causal evidence that stored memory can be actively harmful. Two grounding interventions prevented confabulation on solvable tasks but could not resolve hard-task failures, pointing to feedback granularity rather than reflection format as the fundamental bottleneck. The broader message for the agent memory community is that write-path validation is as important as retrieval quality. A memory system that stores confident, plausible-sounding but wrong beliefs is worse than no memory at all for the tasks those beliefs affect. Designing agents that know when not to write or that validate causal accuracy before storing is a necessary complement to the retrieval improvements that have dominated recent work.

References

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901, 2020.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chrobak, Q. and Zaragoza, M. S. The cognitive consequences of forced fabrication: Evidence from studies of eyewitness suggestibility. *Confabulation: Views from neuroscience, psychiatry, psychology and philosophy*, pp. 67–90, 2009.

Du, P. Memory for autonomous llm agents:mechanisms, evaluation, and emerging frontiers, 2026. URL https://arxiv.org/abs/2603.07670.

Durante, Z., Gong, R., Sarkar, B., Wake, N., Taori, R., Tang, P., Lakshmikanth, S., Schulman, K., Milstein, A., Vo, H., et al. An interactive agent foundation model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3652–3662, 2025.

Hurst, A., Lerer, A., Goucher, A. P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, March 2023a. ISSN 1557-7341. doi: 10.1145/3571730. URL http://dx.doi.org/10.1145/3571730.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38, 2023b.

Johnson, M. K. and Raye, C. L. False memories and confabulation. *Trends in cognitive sciences*, 2(4):137–145, 1998.

Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1906–1919, 2020a.

Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. On faithfulness and factuality in abstractive summarization, 2020b. URL https://arxiv.org/abs/2005.00661.

Musser, D. R. and Nishanov, G. V. A fast generic sequence matching algorithm, 2008. URL https://arxiv.org/abs/0810.0264.

Schnider, A. Spontaneous confabulation, reality monitoring, and the limbic system—a review. *Brain Research Reviews*, 36(2-3):150–160, 2001.

Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., and Manning, C. D. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Zhang, X., Li, Q., and Wang, H. Feedback-driven execution for llm-based binary analysis. *arXiv preprint arXiv:2604.15136*, 2026.

Zhang, Z., Dai, Q., Bo, X., Ma, C., Li, R., Chen, X., Zhu, J., Dong, Z., and Wen, J.-R. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025.

Zhao, A., Huang, D., Xu, Q., Lin, M., Liu, Y.-J., and Huang, G. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Zhou, A., Yan, K., Shlapentokh-Rothman, M., Wang, H., and Wang, Y.-X. Language agent tree search unifies reasoning acting and planning in language models, 2024. URL https://arxiv.org/abs/2310.04406.

A. Prompt for Programmatic Extraction

The following prompt replaces the standard Reflexion self-diagnosis prompt. Failure steps are extracted programmatically from the trajectory before this prompt is constructed.

```
You will be given the history of a past experience in which you were placed in an
environment and given a task to complete.  You were unsuccessful.

YOUR TASK WAS: {task_line}

SPECIFIC FAILURES EXTRACTED FROM YOUR TRACE:
  Failure 1 ({type}):
    Action you took :  {action}
    Environment said:  {response}
  Failure 2 ({type}):
    Action you took :  {action}
    Environment said:  {response}

Using the specific failures listed above (do not invent your own interpretation),
explain in one sentence WHY each failure occurred.  Then write a concise
step-by-step New plan that avoids those exact failures.  Name the exact objects
and locations from your trace.

Experience:  {scenario}

Previous plans:
  Attempt 1:  {plan_1}
  Attempt 2:  {plan_2}

New plan:
```

Figure 3. Programmatic extraction prompt template. Placeholders are filled from trajectory parsing before the LLM generates the reflection.

The {`task_line`} is extracted directly from the trajectory (the line containing "Your task is to:"). The failure block is populated by a string parser that identifies actions receiving `Nothing happens` responses or repeated identical actions indicating a loop. The {`scenario`} is the full trajectory text. Previous plans are truncated to the action-relevant portion only, filtering out analysis text that could leak into action generation. This prompt structure ensures the agent receives concrete failure evidence rather than being asked to self-diagnose from an unstructured trajectory.