Still Writing, No Longer Steering: How Agent Updates Can Silently Break the Prediction-Action Link

Yiwen Lu
University of Notre Dame
ylu37@nd.edu

Research report, November 2026. ALF-CF code: https://github.com/EvenEureka/Know2Act

Abstract

We show that one epoch of fine-tuning on contradicting prediction–action pairs reduces first-step response to prediction edits to near zero in all 11 tested agents. Operators supervise language-model agents by rewriting what the agent writes before it acts, and such supervision requires the agent's actions to respond to those edits. We measure this response in ALF-CF, a counterfactual ALFWorld, by replacing a Qwen3 agent's prediction on fixed probes and recording how far its next action follows the edit. For eight Qwen3-4B agents, editing a prediction has an effect of .80 on the next action, while deleting it, the common test of whether written reasoning matters, has an effect of .015. After the update, the gain from correcting a wrong prediction falls from 81 points to 0.4 for these agents, while task success changes by less than one point and prediction accuracy by -1.9 to $+4.0$ points. A noninferiority suite specified before the second study passes the contradicting update for three of five previously evaluated 4B parents. Adding an edit check on parent-built probes holds all three updates, with replay taking 1.6 to 8 GPU-minutes per candidate.

1 Introduction

Editing an agent's written prediction is a direct way to redirect its next action. An operator who sees a wrong plan rewrites it and lets the agent continue (Mozannar et al., 2025), and written reasoning also serves as the input to monitoring (Yao et al., 2023; Korbak et al., 2025). Release practice checks that an updated agent still solves its tasks and still writes sensible predictions. In contrast, we show that one epoch of fine-tuning reduces the agent's first-step response to prediction edits to near zero while the ordinary release suite passes the update for three of five previously evaluated 4B parents. Meanwhile the 4B agents' task success changes by less than one percentage point and their prediction accuracy by -1.9 to $+4.0$ points.

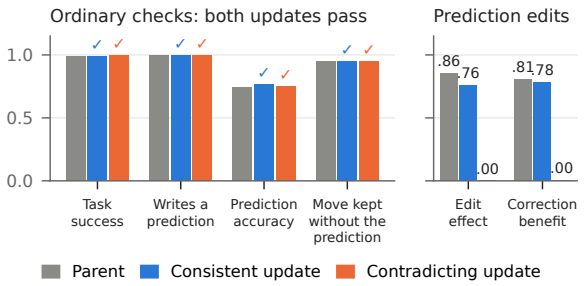


Figure 1: **For parent 3, the CONTRADICTING update passes every ordinary check while its edit effect falls from .86 to .00.** Left: task success, rate of writing a prediction, its accuracy, and the move kept when the prediction is deleted. Both updates pass every check (✓). Right: the edit effect, the probability shift toward the location an edited prediction ranks first, and the correction benefit, how far rewriting a wrong prediction redirects the first step to the right receptacle type (parent 3 of Section 5).

Two standard tests miss this loss. Deleting or truncating the reasoning and watching whether the output changes (Lanham et al., 2023) measures whether the agent needs the text. For our 4B parents, deleting the prediction lowers the probability of choosing its first-ranked receptacle type by .015 on average, whereas editing it produces an effect of .80. Regression suites confirm that a new version still solves its tasks and that its outputs still look as before (Bansal et al., 2019; Ma et al., 2024), and the checks of our ordinary suite, built from this practice, never put a different prediction in front of the agent. Liu et al. (2026) track declining responsiveness to substituted reasoning across training checkpoints. We add matched updates on fixed parent-built probes and test whether release suites detect the loss.

Concretely, we test the link by changing the text. Our edit check (Figure 2) builds probes once from the version in use, which we call the parent. At the agent's first search step, a probe gives it its own

prediction, an edited prediction, or none, and we measure how far the edit moves the next action. A parent and its updated child see the same probes and sampling seeds, so the two versions differ only in their weights.

We study this in ALF-CF, a counterfactual version of ALFWorld (Shridhar et al., 2021) where fine-tuning teaches location conventions that conflict with the zero-shot rankings of the base models. The agent writes these conventions as a one-line ranking of likely locations, and it also uses them directly when it acts without a ranking. We then update Qwen3 parents with one more epoch on matched data. The update we call CONSISTENT retains the parent's predictions, including its alphabetical placebo lines. The update we call CONTRADICTING replaces eligible non-placebo predictions with probability 0.25 while retaining the original actions. Figure 1 shows the outcome for one parent. Both updates pass every ordinary check, and after CONTRADICTING an edited prediction barely moves the agent and rewriting a wrong prediction barely corrects it.

We make three contributions. First, we show that deleting a prediction does not measure whether it steers the agent, since the eight 4B parents combine a mean deletion effect of .015 with a mean edit effect of .80. Second, we show that one epoch on contradicting prediction–action pairs brings the edit effect to near zero for all 11 agents and cuts the 4B agents' correction gain from 81 to 0.4 points, and matched updates attribute the near-total loss to prediction replacement. Third, we introduce an edit check on fixed parent-built probes. Among the 16 updates that the noninferiority suite passes, it holds all eight low-correction updates and keeps the other eight, and ablations show which of its parts matter.

2 Related Work

From testing written reasoning to testing an editing interface. Interventions on intermediate text separate what a model writes from the influence that text has on what it does next. Lanham et al. (2023) truncate, corrupt and paraphrase chains of thought and find that the answer depends on them to very different degrees across tasks. Turpin et al. (2023) show explanations that omit the feature that actually drove the answer. For agents, replacing the reasoning at a saved step changes the next action in some cases and not in others (Sediqin and Fatemi, 2026). Liu et al. (2026) report declining responsiveness to substituted reasoning across training checkpoints. Training a model only on what it says, or only on what it does, transfers poorly to the other side (Xu et al., 2025). Interfaces now act on this influence directly. Magentic-UI lets users add, delete and rewrite the steps of a plan before and during execution (Mozannar et al., 2025), and inserting or revising tokens inside a reasoning model's thinking improves instruction following and refusal of unsafe requests (Wu et al., 2025). The same text is also what monitors read. Korbak et al. (2025) argue that the monitorability of reasoning traces is fragile under training. Optimising against a reasoning monitor teaches agents to hide their intent (Baker et al., 2025), and Chen et al. (2025) report that increased hint use under reinforcement learning does not reliably show up in the reasoning. Our updates optimise against no monitor. The matched CONSISTENT and CONTRADICTING updates keep action targets and loss masking fixed while varying prediction replacement. We compare such matched updates on identical parent-built probes and measure whether the response that an editing interface needs survives the update.

From preserving accuracy to preserving the behaviour users rely on. Model-update testing has moved from aggregate accuracy toward the behaviours that users and downstream systems depend on. Bansal et al. (2019) show that a more accurate model lowers the performance of a human–AI team when its behaviour departs from what the user learned to expect, and Srivastava et al. (2020) find backward-compatibility losses without data shift. Later work reduces such regressions by penalising negative flips, cases that the old model got right and the new one gets wrong (Yan et al., 2021; Echterhoff et al., 2024), and argues that regression testing for silently updated language models needs its own design (Chen et al., 2024; Ma et al., 2024). Behavioural testing in the style of CheckList specifies the expected effect of a perturbation in advance (Ribeiro et al., 2020). Work on evolving agents makes such suites cheaper to rerun by selecting the benchmark tasks whose trajectories best predict the full result (Ayyad et al., 2026; She and Lin, 2026), and the quantity those suites track is task success. Our release test adds the response to prediction edits as a measured behaviour next to task success and prediction accuracy.

3 Testing Whether a Prediction Steers the Action

To test whether a prediction steers the agent, we change the prediction in front of it and measure the next action, on both the parent π, the version in use, and its updated child π'. The edit check reports its decision next to that of the ordinary release suite (Figure 2).

3.1 Fixed Edit Probes

To compare both versions on identical inputs, the edit check builds its probes once from the parent and replays them on the child. At the first search step of a game, a probe gives the agent one of three lines before its action: its own prediction (*intact*), the prediction it writes when the task names another object, relabelled to the target (*edited*), or no prediction (*removed*). The edited condition uses a foil selected by seeded sampling among other learnable object classes in the same scene whose predictions rank a remaining receptacle type different from the intact top first. We then sample the next action four times at temperature 0.7 under each condition, with the same seeds for both versions, so the parent and the child receive identical inputs and differ only in their weights.

3.2 Edit Effect and Decision Rule

The edit effect is a difference of two probabilities. With $P_c(t)$ the probability that the next action goes to a receptacle of type t under condition c, and t_e the type ranked first in the edited prediction, the edit effect is

$$S = P_{\text{edited}}(t_e) - P_{\text{intact}}(t_e),$$

the increase in the probability that the next action goes to the location the edited prediction ranks first. The removed condition gives the deletion test on the same probes. The deletion effect D is the drop in the probability that the next action goes to the agent's own first-ranked type when the prediction is removed. D asks whether the agent needs the prediction, and S asks whether it obeys one. Our agents respond to the first entry of a ranking, so S measures how far the first step follows that entry.

The check decides from the change in S between parent and child. The edit effect is preserved when the 90% interval of the change lies within $\pm.15$, the check fails when the interval lies wholly below $-.15$, and any other outcome is unresolved. The band was fixed before the first study, close to the spread of S across independently fine-tuned 4B parents (.70 to .88). An increase beyond the band is reported separately. A stronger edit effect makes the agent easier to correct and also easier to misdirect with a misleading prediction.

3.3 Release Record

Each candidate release produces a record that links its decisions to the hashes of both checkpoints, their training data and the probe file. The record keeps the margin, parent and child values, interval and decision of every check. It reports the ordinary and edit decisions separately and lists the parent's edit effect next to the edit decision.

4 Experimental Setup

Environment. We use ALF-CF, a counterfactual version of the text ALFWorld environment (Shridhar et al., 2021; Côté et al., 2018). In ALFWorld an object becomes visible only after the agent goes to the receptacle that holds it, so at the start of a search the context says nothing about where the target is. ALF-CF moves each of 37 object classes, with probability 0.8 per game, to a receptacle class chosen so that neither the original ALFWorld data nor the zero-shot rankings of Qwen3-4B and Qwen3-8B point to it. These relocation rules conflict with the tested zero-shot rankings and are learned here through supervised fine-tuning. The test set has 1,116 games from held-out floor plans and rooms, and the edit check uses the 826 games whose target belongs to a moved class.

Agent and parents. The agent is Qwen3 (Yang et al., 2025) fine-tuned on trajectories of a scripted searcher that knows the moved locations. Before a search step, the agent either writes a one-line prediction and then acts, for example

```
<prediction>The mug is most likely in
a toaster, then a garbagecan, then a
cabinet.</prediction>
<action>go to toaster 1</action>
```

or acts directly. In the training data, 63% of search steps carry such a prediction, whose first entry is always the receptacle that the next action visits, 7% carry an alphabetical placebo line, and 30% carry no line, so the agent also learns to act without a prediction. We reuse eight Qwen3-4B parents from our earlier study of this environment,[1] trained on independently drawn versions of these data, and

[1] https://eveneureka.github.io/Know2Act/

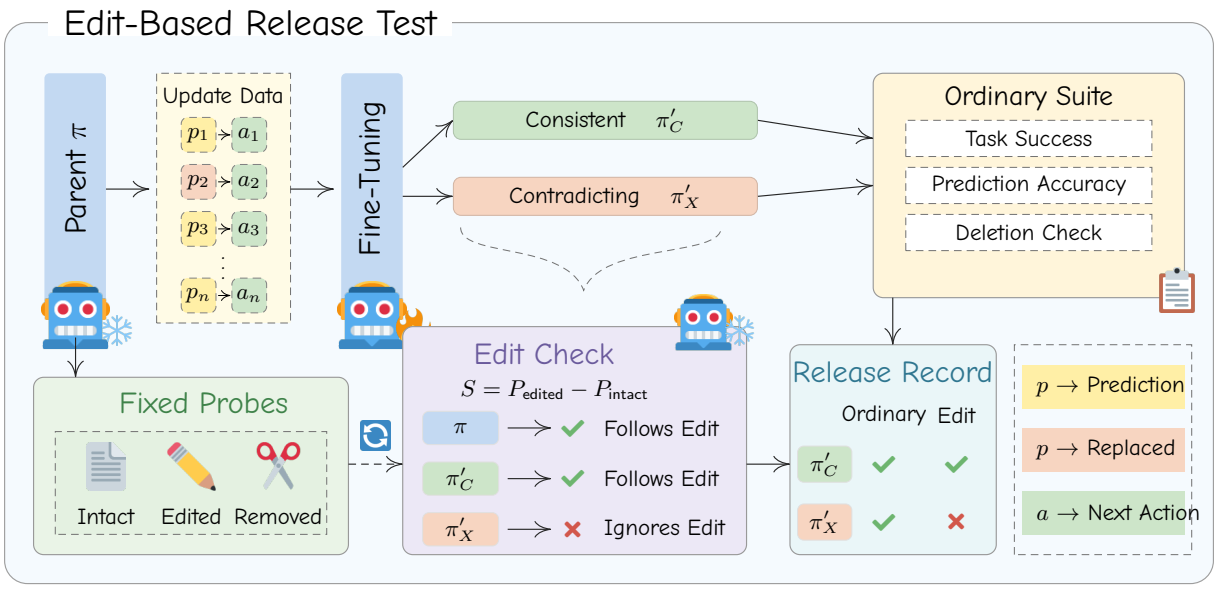


Figure 2: **Fixed parent-built probes measure how an update changes the response to the same prediction edits.** A parent receives matched updates that differ only in whether the written predictions p agree with the actions a (red: a replaced prediction). Probes built once from the parent are replayed on every version, so all versions see identical inputs. The ordinary suite and the edit check report separate decisions, shown here for parent 3.

add three Qwen3-1.7B parents. All parents write a prediction in every test game and follow it, with edit effects of .70–.88 for the 4B parents and .61–.75 for the 1.7B parents.

Updates. Each update trains the parent for one more epoch on the scripted trajectories it was trained on, with the actions unchanged, and the updates differ in two respects. At eligible non-placebo prediction steps selected with probability 0.25, the prediction is either kept or replaced with the prediction for another object, and the receptacle names in those predictions are either excluded from the loss or trained on as usual. This gives four updates: CONSISTENT (kept, excluded), CONTRADICTING (replaced, excluded), CONTINUE (kept, trained, which is the parent's own data) and CONTRADICTING-FULL (replaced, trained). CONSISTENT and CONTRADICTING train the same actions with the same number of supervised tokens in the same order; CONTRADICTING reads replaced predictions followed by the original actions, but the replaced receptacle names are excluded from its training loss, which separates reading a conflicting prediction from being trained to write a wrong one. Every parent receives CONSISTENT and CONTRADICTING, and six 4B parents also receive CONTINUE and CONTRADICTING-FULL, for 34 updates.

Baselines. We compare the edit check with a regression suite built from checks used in prior work. The ordinary suite measures task success in free play, and at the first search step of each game whether the agent writes a prediction and whether the receptacle type ranked first holds the target. It also deletes the prediction, a common probe of whether written reasoning matters (Lanham et al., 2023), and records the deletion effect D and the kept move K, the probability of still going to the first-ranked type. Each check is decided from the child–parent difference and its 90% bootstrap interval. An equivalence check passes when the whole interval lies inside $\pm m$ and fails when it lies wholly outside, a noninferiority check passes when the lower end lies above $-m$ and fails when the upper end lies below it, and other outcomes are inconclusive. The suite passes only when every check passes. Task success is a noninferiority check with a margin of one point, the writing rate an equivalence check with a margin of two, and D and K equivalence checks with a margin of three. The two variants of the suite differ in prediction accuracy: the equivalence variant holds it within two points in either direction, and the noninferiority variant lets it rise freely and drop by at most two. We also evaluate three checks added after the main analyses: sampled predictions, which draw four predictions per parent-built probe at temperature 0.7 and test whether the share that ranks the

learned location first drops by more than two points, negative flips (Yan et al., 2021), and the change in actions on unedited inputs.

Outcome measures. The correction gain is the rise in the share of first steps that go to a receptacle type holding the target when a wrong prediction is rewritten to name the correct location. The misled cost is the number of extra receptacle visits per game when every prediction in free play is replaced by the ranking for another object. We call an update *low-correction* when it keeps less than half of its parent's gain. We call an update pair a *silent regression* when CONTRADICTING and CONSISTENT both pass the ordinary suite, CONTRADICTING loses the edit effect, CONSISTENT preserves it, and the two differ on the same probes (loss and difference at least .30, with one-sided 95% lower bounds above .15). The same definition applies to the pair CONTINUE, CONTRADICTING-FULL.

Studies. We ran two studies, each fixed and hash-committed before its updates were trained. The first used parents 0–2 with the equivalence variant, which held every update. The noninferiority variant was specified after that study and then tested on new updates of parents 3–7, with parent 3 designated as primary. Main-study intervals use 10,000 paired source-game bootstrap resamples within scenes, while the format and budget ablations use 1,000 and 300 unstratified resamples of game variants, and all intervals condition on fixed checkpoints.

5 Results

5.1 Editing Reveals a Link That Deletion Misses

The 4B parents respond strongly to prediction edits despite a small deletion effect. For the eight 4B parents the edit effect averages .80, ranging from .70 to .88 across parents, while the deletion effect averages .015, ranging from .005 to .030. The 1.7B parents show the same contrast, .70 against .035. Without a prediction the 4B parents still go to the first-ranked type (kept move .97), since they have learned the locations and 30% of their training steps carry no line. With a different prediction in front of them, they follow it. Deletion therefore shows whether an agent needs the text, and editing shows whether it obeys the text, which is what an operator relies on.

Prediction edits support correction and also expose the agents to misleading rankings. Rewriting a wrong prediction to name the correct location raises the share of first steps that reach a holding receptacle type by .81 on average for the 4B parents. With every prediction replaced by the ranking for another object, the 4B parents need about 7 more receptacle visits per game and solve 4 to 5 points fewer games.

5.2 One Small Update Nearly Eliminates the Edit Response While Ordinary Metrics Change Little

One epoch on contradicting predictions nearly eliminates the edit response in all 11 agents. Masked replacement brings the edit effect to near zero for all eleven parents ($|S| < .005$), and unmasked replacement does so for all six tested parents (Table 1). After CONTRADICTING, every 4B child still writes a prediction at the first search step of all 826 games, its prediction accuracy changes by -1.9 to $+4.0$ points, and its task success by less than one point. The deletion effect was near zero before the update and stays there.

Losing the link makes an agent harder to correct and also harder to mislead. Rewriting a wrong prediction raises the share of first steps that reach a holding receptacle type by .75 after CONSISTENT on average for the 4B parents, and by .00 after CONTRADICTING, on the 188 to 219 probes per parent where the parent's own prediction names the wrong location. Under another object's ranking the 4B CONTRADICTING models need fewer than 0.2 extra visits per game, against about 7 for their parents.

Replacing predictions nearly eliminates the effect, while the matched CONSISTENT update keeps it for nine of eleven parents. Averaged over the six parents, one more epoch with or without masking leaves the effect and both of its consequences in place, and replacing eligible predictions nearly eliminates all three, with masking or without it. CONSISTENT keeps the effect for seven of eight 4B parents and two of three 1.7B parents. The 4B exception is parent 1, which loses .32 under CONSISTENT. Masking is the main cause there, since CONTINUE keeps .26 [.23, .29] more of the effect.

5.3 Release Checks Pass These Updates, and the Edit Check Holds Them

The noninferiority suite passes the contradicting update for three of five previously evaluated 4B parents. The suite was specified before the second study, and for these three parents, including

Table 1: **Across six 4B parents, prediction replacement nearly eliminates the edit effect and the correction gain while task success stays near 99%.** Means over the six parents that received all four updates, each one more epoch. The updates cross prediction replacement with name masking. Accuracy and success in %. D: deletion effect. S: edit effect. Correction: first-step correction gain. Misled: extra receptacle visits per game under another object's ranking. Shaded rows are the updates that replace predictions.

	Ordinary metrics			Response to prediction edits		
Update	Accuracy	Success	D	S	Correction	Misled
Parent (no update)	75.5	98.9	.014	.80	.80	7.33
Neither (CONTINUE)	76.7	99.2	.020	.77	.77	7.00
Masking only (CONSISTENT)	76.3	99.0	.021	.73	.72	6.61
Replacement only (CONTRADICTING-FULL)	77.0	99.1	.001	.00	.01	0.07
Replacement + masking (CONTRADICTING)	76.4	99.3	.001	.00	.00	0.04

Table 2: **Adding the edit check removes every low-correction update that each tested gate ships.** Every release gate without the edit check (baseline) and with it (ours), on the 34 constructed updates, 18 of which are low-correction. Ship: updates the gate passes. Low-corr.: shipped updates that keep less than half of the parent's correction gain. Gain: mean correction gain of what ships. Sampled: the sampled-prediction check.

	Without (baseline)			With edit check (ours)		
Release gate	Ship	Low-corr. ↓	Gain ↑	Ship	Low-corr. ↓	Gain ↑
Task success	34	18	.37	15	**0**	**.77**
Equivalence suite	8	4	.39	4	**0**	**.78**
Noninferiority suite	16	8	.41	8	**0**	**.77**
Noninferiority + sampled	8	3	.53	5	**0**	**.78**

the primary parent, the pair is a silent regression. For parent 3, CONTRADICTING passes every ordinary check while its edit effect falls from .86 to .00. CONSISTENT passes as well and keeps an effect of .76, so the two updates differ by .76 [.73, .79] on the same probes, and only the edit check separates them (Figure 1). Training on the replaced predictions in full gives the same picture. CONTRADICTING-FULL passes the noninferiority variant for parents 3–5, and the pair CONTINUE, CONTRADICTING-FULL is a silent regression for parents 3 and 4.

Every baseline gate ships low-correction updates, and adding the edit check removes all of them. Eighteen of the 34 updates are low-correction, namely all 17 contradicting updates and one consistent update (Table 2). Task success alone ships all 18. The noninferiority suite ships 16, and 8 of those are low-correction, so passing the suite leaves their share at 50%, about the 53% of the whole set. Adding the edit check removes all 8, keeps the other 8, and raises the mean correction gain of what ships from .41 to .77. Replaying a child on its probes took 1.6 to 8 minutes on one GPU.

The change in edit effect separates the constructed update recipes with an AUROC of 1.00, against .85 for the strongest baseline. Ranking the 34 updates by the size of each check's change needs no margin. On this ranking, task success, prediction accuracy and negative flips stay near chance (.48, .44 and .52). The strongest baseline is the change in actions on unedited inputs, which rests on a small difference, 2.7% of probe samples for contradicting updates against 1.0% for the others. Against low-correction outcomes instead of recipes, the edit check reaches .997 and this baseline .90. None of the five tested margins repairs the suite either. Across the 28 updates of the 4B parents, no margin among .005, .01, .02, .03 and .05 passes every consistent update while holding every contradicting one. Sampling predictions closes part of the gap. That check fails all six CONTRADICTING-FULL updates, but it gives CONSISTENT and CONTRADICTING the same decision for all eleven parents, so adding it to the suite takes parents 4 and 6 out of the silent regressions and leaves parent 3.

5.4 Ablation: Which Parts of the Check Matter

Deleting the prediction instead of editing it fails one of 11 contradicting updates and leaves four unresolved. Deletion shows how the agent acts without a prediction, and editing shows whether it follows one that disagrees with what it has learned

Table 3: **The full edit check fails all 11 CONTRADICTING updates, and deletion fails one.** Ablation of the edit check on the CONTRADICTING and CONSISTENT updates of all eleven parents, in %; unres.: unresolved. Rows use the edit check's rule with band .15 unless the row says otherwise, and the deletion row uses the suite's deletion check. The budget row averages 100 random probe subsets per update. Format and budget rows are exploratory.

	CONTRADICTING		CONSISTENT	
Version of the check	failed	unres.	failed	unres.
FULL EDIT CHECK	100.0	0.0	18.2	0.0
Delete instead of edit	9.1	36.4	0.0	0.0
Own predictions as probes	100.0	0.0	9.1	18.2
Rank labels, best first	100.0	0.0	36.4	27.3
Rank labels, best last	0.0	0.0	0.0	0.0
32 probes, 1 sample	100.0	0.0	9.5	46.5
Band .05	100.0	0.0	45.5	27.3
Band .20	100.0	0.0	9.1	9.1

(Section 5.1). Used as the release check, deletion fails 9.1% of CONTRADICTING updates and leaves 36.4% unresolved (Table 3).

The edit effect depends on where the best location appears in the prediction. For the eight 4B parents, explicit rank labels with the best location first retain a mean edit effect of .60, against .80 in the trained format. With the same labelled ranking listed best-last, the effect disappears ($S = -.01$), and the check flags nothing because there is no parent effect left to lose.

All tested probe sources, budgets and decision bands flag every CONTRADICTING update. Own-prediction probes fail the same CONTRADICTING updates as parent-built probes, and fixed probes make the inputs identical across versions. With 32 probes and one sample the check still fails every CONTRADICTING update, while 46.5% of CONSISTENT decisions become unresolved, against none with the full budget. The band changes how many CONSISTENT updates fail or remain unresolved.

5.5 The Ordinary Suite Holds All Three 1.7B Contradicting Updates

The ordinary suite holds all three 1.7B CONTRADICTING updates, with one failure and two inconclusive decisions. The 1.7B parents lose the edit effect in the same way, and for them CONTRADICTING also lowers the deletion effect by 2.5 to 4.3 points, so none of the three is a silent regression. How much the ordinary checks register depends on the model, while the edit check failed CONTRADICTING for every parent of both sizes. At later decision steps the edit effect is small for every model.

6 Discussion

Contradicting updates nearly eliminate first-step edit response while 4B prediction accuracy changes by -1.9 to $+4.0$ points. In the CONTRADICTING update the agent reads predictions followed by a different action. Afterwards its first-step response to edited predictions is near zero, while it keeps writing a prediction in every game. The ordinary suite evaluates unedited execution, prediction outputs and prediction deletion. In the second study it passes the CONTRADICTING updates for 4B parents 3, 4 and 6 despite their lost edit response, and on the 34 updates, task success and prediction accuracy rank the recipes near chance (AUROC .48 and .44).

Release suites for editable agents. For agents whose predictions people edit, we recommend adding an edit check to the release suite, with probes built from the parent before the child is evaluated and written in the format that people actually edit. Keeping the response is a design decision, because the same response enables correction and exposure to misleading text. The check therefore reports the change and leaves the judgement to the release policy. The next tests are real training pipelines, updates by reinforcement learning, and multi-step plans.

Limitations. (1) The contradicting data are constructed, and the agents are Qwen3 models at two sizes, trained by supervised fine-tuning on scripted trajectories to near-99% task success in one synthetic environment, so how often real training pipelines produce such updates is unmeasured. (2) The tests cover first-step responses to one-line rankings in the tested formats. Semantic plan following, monitorability, benefits to human supervision, correction over whole episodes and matched later-step comparisons are untested. (3) Correction gain is related to the edit effect, so the release counts show what the check protects without validating it independently, and they describe our mix of 34 constructed updates rather than rates in real pipelines. (4) The noninferiority suite was specified after the first study, and the second study's parents had been evaluated before on the same test games, although none of their updates had. (5) Intervals condition on fixed checkpoints, and the

format and budget analyses use different resampling procedures from the main study.

7 Conclusion

Supervising an agent by editing what it writes requires its actions to respond to those edits. A small deletion effect coexists with a large edit response, .015 against .80 for the 4B parents. One epoch on data with contradicting predictions reduced the edit effect to near zero for all eleven parents, and for the 4B parents it cut the gain from correcting a wrong prediction from 81 points to 0.4, while task success moved by less than one point and prediction accuracy by -1.9 to $+4.0$ points. A noninferiority suite specified before the second study passed this update for three of five 4B parents and none of the three 1.7B parents, and the edit check on parent-built probes held all 11 CONTRADICTING updates for a few GPU-minutes per release.

References

Mahmoud Ayyad, Zehao Wang, Jiho Shin, Ying Zou, and Bram Adams. 2026. Trajectory-aware benchmark subset selection for cost-efficient software engineering agent regression testing. *Preprint*, arXiv:2609.24928.

Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y. Guan, Aleksander Madry, Wojciech Zaremba, Jakub Pachocki, and David Farhi. 2025. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation. *Preprint*, arXiv:2503.11926.

Gagan Bansal, Besmira Nushi, Ece Kamar, Daniel S. Weld, Walter S. Lasecki, and Eric Horvitz. 2019. Updates in human-AI teams: Understanding and addressing the performance/compatibility tradeoff. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2429–2437.

Lingjiao Chen, Matei Zaharia, and James Zou. 2024. How is ChatGPT's behavior changing over time? *Harvard Data Science Review*, 6(2).

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. 2025. Reasoning models don't always say what they think. *Preprint*, arXiv:2505.05410.

Marc-Alexandre Côté et al. 2018. TextWorld: A learning environment for text-based games. *Preprint*, arXiv:1806.11532.

Jessica Maria Echterhoff, Fartash Faghri, Raviteja Vemulapalli, Ting-Yao Hu, Chun-Liang Li, Oncel Tuzel, and Hadi Pouransari. 2024. MUSCLE: A model update strategy for compatible LLM evolution. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7320–7332, Miami, Florida, USA. Association for Computational Linguistics.

Tomek Korbak et al. 2025. Chain of thought monitorability: A new and fragile opportunity for AI safety. *Preprint*, arXiv:2507.11473.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, et al. 2023. Measuring faithfulness in chain-of-thought reasoning. *Preprint*, arXiv:2307.13702.

Jingyu Liu, Zhiwen Wang, Yuxin Jing, Huanyu Zhou, and Yong Liu. 2026. Where do CoT training gains land in LLM based agents? *Preprint*, arXiv:2606.26935.

Wanqin Ma, Chenyang Yang, and Christian Kästner. 2024. (why) is my prompt getting worse? rethinking regression testing for evolving LLM APIs. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI (CAIN 2024)*, pages 166–171. ACM.

Hussein Mozannar, Gagan Bansal, Cheng Tan, Adam Fourney, Victor Dibia, Jingya Chen, Jack Gerrits, Tyler Payne, Matheus Kunzler Maldaner, Madeleine Grunde-McLaughlin, Eric Zhu, Griffin Bassman, Jacob Alber, Peter Chang, Ricky Loynd, Friederike Niedtner, Ece Kamar, Maya Murad, Rafah Hosn, and Saleema Amershi. 2025. Magentic-UI: Towards human-in-the-loop agentic systems. *Preprint*, arXiv:2507.22358.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Mohammadreza Sediqin and Zahra Fatemi. 2026. Causal intervention on agent reasoning: Not all reasoning is load-bearing. NeurIPS 2026 Workshop "Who Verifies the Agents? Toward Reliable Agent Development".

Yining She and Lei Lin. 2026. Efficient benchmarking in production: A study of an evolving LLM agent. *Preprint*, arXiv:2609.21267.

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

Megha Srivastava, Besmira Nushi, Ece Kamar, Shital Shah, and Eric Horvitz. 2020. An empirical analysis of backward compatibility in machine learning systems. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*, pages 3272–3280. ACM.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, pages 74952–74965.

Tong Wu, Chong Xiang, Jiachen T. Wang, G. Edward Suh, and Prateek Mittal. 2025. Effectively controlling reasoning models through thinking intervention. In *Advances in Neural Information Processing Systems*.

Ruoxi Xu, Hongyu Lin, Xianpei Han, Jia Zheng, Weixiang Zhou, Le Sun, and Yingfei Sun. 2025. Large language models often say one thing and do another. In *International Conference on Learning Representations (ICLR)*, pages 23987–24003.

Sijie Yan, Yuanjun Xiong, Kaustav Kundu, Shuo Yang, Siqi Deng, Meng Wang, Wei Xia, and Stefano Soatto. 2021. Positive-congruent training: Towards regression-free model updates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14299–14308.

An Yang et al. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

A Implementation Details

Fine-tuning. All policies fine-tune Qwen3-4B or Qwen3-1.7B with thinking disabled: learning rate 10^{-5}, one epoch, global batch 64, maximum length 12,288 tokens, AdamW with betas $(0.9, 0.95)$, warm-up over 3% of the steps followed by cosine decay, and no weight decay. Updates start from the parent's weights with a fresh optimizer and schedule and run for one epoch over the parent's 8,148 training games (127 steps), and all updates of a parent use the same data order.

Update data. CONSISTENT and CONTRADICTING keep or replace the prediction at the same steps, selected by a seeded probability-0.25 coin at each eligible non-placebo prediction step. Their actions are identical to the parent's data, and the loss excludes exactly the receptacle names of the affected predictions in both. A replaced prediction never ranks the correct location first.

Evaluation. Free play is greedy with at most 50 steps per game. At a probe, each condition is sampled four times at temperature 0.7, top-p 1 and at most 40 tokens, with seeds derived from the probe and the sample index and shared across conditions and versions. A continuation that does not parse as a `go to` action counts as choosing no receptacle. Each parent's probe file is built from its recorded test outputs (826 probes, none excluded). Before any child was evaluated, each parent replayed its own probes in a fresh process; its sampled actions agreed with the recorded ones in 99.65–99.93% of cases and its edit effect moved by at most .005. The correction probes are those on which the parent's own first entry does not hold the target and a holding receptacle type is still unvisited; the corrected prediction puts that type first and keeps the parent's ranking after it.

Hardware. Models ran on NVIDIA RTX PRO 6000 Blackwell GPUs, and the replay times in the text are for one GPU.

Artifacts. The repository on the first page holds the ALF-CF rules (`alfcf/`) and build scripts (`scripts/`). The update study's checkpoints and result files are not yet public.

Acknowledgement. Figure icons are from Twemoji (CC BY 4.0).