COUNTERFACTUAL ERROR SCRAMBLING: MEASURING ERROR-MESSAGE USE IN TOOL-USING LANGUAGE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-using language agents routinely receive execution errors, yet it remains unclear whether they act on the content of those messages or only on the fact that an error occurred. We introduce *counterfactual error scrambling*, a protocol that replaces each error message with a variant that preserves its length, line structure and exception class and keeps identifiers verbatim, while permuting the surrounding words to destroy compositional meaning; misleading and empty variants serve as controls. Applying it to nine models from 7B to frontier scale on MBPP-sanitized (3,600 trials; one serving stack; success scored by the execution harness), seven models recover significantly more often from real than from scrambled errors ($+13$ to $+30$ percentage points, Holm-corrected), including an 8B model without a reasoning mode, while two 7–8B models released in 2024 show no effect. Error-message use is therefore not determined by parameter count alone. An audit of the scrambler shows that on a quarter of the tasks it can move a mixed-case identifier out of its position in the message; restricted to tasks where identifiers stay in place, four effects remain significant ($+12$ to $+23$ points, including the 8B model) and three weaken. Decomposing repairs after `NameError`s shows why: when the missing name is displaced, agents rarely adopt it (4%), and when it stays in place, but its frame is scrambled, adoption falls from 54% to 35%. What agents use is the position of the actionable token and the conventional frame around it. Within Qwen3-14B, suppressing reasoning leaves the model retrying without acting on the message at all. Sensitivity decays across successive errors, and on HumanEval, capable models rarely fail, so the contrast is undefined; we report both as scope conditions, and we release all trajectories and analysis code.

1 INTRODUCTION

When a tool-using agent submits code that fails, the environment returns a traceback: an exception class, a location, and a short message identifying the likely cause, such as `NameError: name 'check_integer' is not defined`. The agent then acts again. A large body of work builds on this loop, from self-debugging and self-repair for code (Chen et al., 2024; Olausson et al., 2024; Le et al., 2022) to agents that reflect on or critique their own tool feedback (Shinn et al., 2023; Gou et al., 2024; Madaan et al., 2023) and repository-scale software agents (Yang et al., 2024; Jimenez et al., 2024). This loop carries an implicit assumption: that the agent *uses the content* of the message rather than merely the fact that something went wrong.

End-to-end success after an error cannot test that assumption. A successful repair may be driven by the bare signal that an error occurred (try again), by a single token in the message (an identifier or line number), by the position of that token, or by the message's diagnostic meaning. These sources are confounded in any benchmark that scores only final success, and they have different implications: if agents use only error presence and tokens, richer diagnostics are wasted; if they use diagnostic meaning, the wording of error messages is part of the agent's interface.

We separate these sources with a counterfactual intervention at the boundary between environment and agent. Each error message is rendered in one of four conditions (examples in Appendix B). REAL passes it through unchanged. SCRAMBLED preserves the message's exact length, line structure,

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

exception class, identifiers, paths and numbers, and permutes the remaining natural-language words. The result retains the surface form of an error and the same actionable tokens, but no longer conveys coherent compositional meaning. MISLEADING replaces the message with a plausible but incorrect diagnosis, and EMPTY replaces it with the string `An error occurred.` Because the environment is unchanged and conditions are paired by task, the difference in recovery between REAL and SCRAMBLED measures what the agent extracts from the message beyond its surface form and tokens.

We apply the protocol to nine models on MBPP-sanitized (Austin et al., 2021) under a single serving stack, with success scored by the execution harness. Our contributions are:

- **A counterfactual protocol and an audit of it.** We specify the transformer, audit what it preserves, and identify where preservation fails: mixed-case identifiers can leave their slot. We report every result both with and without the affected tasks.
- **Error-message use is common but not universal, and is not explained by size.** Seven models recover significantly more often from real than from scrambled errors; four remain significant when identifiers are held in place, including an 8B model released in 2025, while two 7–8B models released in 2024 show no effect in either analysis (§4.1).
- **What is used is the actionable token, its position, and the frame around it.** After a `NameError`, a displaced name is rarely adopted, and a name that remains in place is adopted substantially less often when its frame is scrambled (§4.2).
- **Within Qwen3-14B, deliberation determines whether the model acts on its errors.** Suppressing reasoning in Qwen3-14B turns error handling into retrying without change; three models without any reasoning phase nonetheless use error content (§4.3). Sensitivity is front-loaded within a trajectory, and a null effect on recovery is not a null effect on processing (§4.4–4.5).

We also report where the protocol does not apply: on HumanEval (Chen et al., 2021), capable models rarely fail, so there is no repair loop to measure (§4.6).

2 RELATED WORK

Execution feedback and self-repair. Self-debugging (Chen et al., 2024), CodeRL (Le et al., 2022), Reflexion (Shinn et al., 2023), Self-Refine (Madaan et al., 2023), and CRITIC (Gou et al., 2024) all condition a model's next attempt on feedback from its previous one. Olausson et al. (2024) show that the benefit of self-repair is bounded by the quality of the feedback the model generates about its own errors, and Huang et al. (2024) show that self-correction without external feedback often fails. These works vary *whether* feedback is available and *what kind* is provided, then measure final accuracy. We hold feedback presence and surface form fixed while varying its meaning, isolating the component of the loop that prior work takes as given.

Tool-using agents and interactive coding. ReAct (Yao et al., 2023) and Toolformer (Schick et al., 2023) established tool observations as conditioning signals for language models. Interactive coding benchmarks such as InterCode (Yang et al., 2023) and SWE-bench (Jimenez et al., 2024), and agent interfaces such as SWE-agent (Yang et al., 2024), provide rich execution feedback but report aggregate resolution rates. Our per-error recovery measure isolates a local transition within the trajectory: given that the agent has just received an error, does the error's content change the agent's next action?

Perturbation tests of input use. Shuffling word order leaves much of the performance of pretrained language models intact on standard understanding benchmarks (Pham et al., 2021), and models pretrained on shuffled text transfer surprisingly well (Sinha et al., 2021). These results suggest that lexical content can substitute for compositional meaning. We apply the same logic at inference time to machine-generated error messages, where the useful signal is concentrated in a few tokens and a highly conventional frame, and find that *where* those tokens appear matters, not only whether they are present.

Reasoning modes. Several recent models expose a switchable reasoning mode, including Qwen3 (Yang et al., 2025), GLM-4.5 and its successors (Zeng et al., 2025), and Gemini 2.5 (Comanici et al.,

2025). A switchable mode enables a within-model contrast in which the weights, prompt, and serving stack are held fixed, which we use to test whether deliberation mediates error-message use.

3 METHOD

3.1 PROTOCOL

The agent interacts with a tool environment through a harness. When a tool call fails, the harness intercepts the error message before the agent sees it and renders it according to the trajectory's condition; the environment itself is unchanged, and the original message is logged alongside the rendered one. The condition is fixed for the entire trajectory, so every error within a trajectory is rendered in the same way. Each task is run once under each condition, pairing the four trajectories by task.

Scrambled. We tag the message with spaCy `en_core_web_sm` (Honnibal et al., 2020). A token is pinned if spaCy marks it as punctuation, whitespace or number-like; if it is one of the keywords `Error`, `Exception`, `Traceback`, `line`, `File`, `in`, `raise` or `at`; or if it overlaps a match of an identifier pattern: lower-case snake_case containing an underscore, CamelCase with at least two components, a file path, or a file name with an extension. The remaining nouns, verbs, adjectives and adverbs are chained into groups of tokens at most 12 positions apart, and each group of two or more is permuted with a non-identity permutation under a per-item seed; whitespace remains fixed (Figure 3). Because tokens are only permuted, character length is preserved exactly: across the 2,302 substitution pairs in the primary experiment, the length ratio is 1.000 in every case, and line structure and exception class are preserved in 100% of cases (Appendix B).

Pinning is incomplete. Identifiers that mix case, such as `is_Diff` or `find_Parity`, match neither identifier pattern and can therefore be permuted like ordinary words. MBPP-sanitized contains many such function names: for 25 of our 100 tasks, the most frequently missing name is unpinned, mostly because it mixes case. Among the seven models whose recovery depends on content, the missing name remains somewhere in the message in every scrambled `NameError`, but it stays in its `name '...'` slot in only 78% of cases (907 of 1,169); every displacement involves an unpinned name. We therefore report the primary contrast both across all tasks, as pre-specified, and for the 75 tasks whose name is pinned (§4.1, Appendix D).

Misleading and Empty. MISLEADING messages are written by a separate generator model instructed to produce a well-formed error of comparable length, with a different cause and no five-token overlap with the original (prompt in Appendix N). In 62.5% of messages, the generator preserves the original exception class and substitutes an incorrect identifier or cause; in the remainder, it changes the class as well. EMPTY replaces the message with `An error occurred.` Only SCRAMBLED is matched to the original in length and structure: MISLEADING messages have a median length ratio of 2.04 and preserve line structure in 35% of cases, and EMPTY messages have a median length ratio of 0.15 and preserve line structure in 3%. REAL–SCRAMBLED is therefore our primary contrast, and the other two conditions serve as controls.

3.2 SETUP

Benchmark and tool. We sample 100 tasks with a fixed seed from the MBPP-sanitized test split (Austin et al., 2021). The agent has a single tool, `submit_code`, which executes each submission against the task's assertions and returns either `All` N `tests passed.` or the Python error. The scaffold uses each model's native function-calling mechanism, allows up to 10 turns and 3 retries after an error, and explicitly instructs the agent in its system prompt to read error messages and adjust accordingly (Appendix N); the test is therefore conservative.

Models. We evaluate nine models in three groups: 7–8B models released in 2024 (Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Qwen2.5-7B-Instruct (Qwen et al., 2025)); 8–14B models released in 2025 (Ministral-3 8B Instruct (Liu et al., 2026), Qwen3-14B (Yang et al., 2025)); and frontier models (GPT-4o (Hurst et al., 2024), Gemini 2.5 Flash (Comanici et al., 2025), Claude Sonnet 4 (Anthropic, 2025), GLM-4.7, a 355B-parameter open-weight mixture-of-experts successor to GLM-

Table 1: Primary contrast on MBPP-sanitized, one serving stack. $\Delta\text{ERR} = \text{ERR}_{\text{real}} - \text{ERR}_{\text{scr}}$ in percentage points with 95% BCa CI, paired on tasks with at least one tool error under both conditions (n). *All tasks*: the pre-specified analysis, Wilcoxon signed-rank (Pratt), Holm over nine models. *Identifier pinned*: the 75 tasks whose target function name the transformer never moves (§4.1), sign-flip permutation test, Holm over nine. Eng.: fraction of trials with at least one tool call. Reas.: whether the model deliberated before acting (def.: provider default of a hybrid or adaptive model). ERR levels, MDEs and TOST tests are in Appendix C.

			All tasks			Identifier pinned		
Model	Reas.	Eng.	n	ΔERR [95% CI]	p_{Holm}	n	ΔERR [95% CI]	p_{Holm}
7–8B, released 2024								
Llama-3.1-8B	no	0.94	94	+1.1 [+0.0, +2.7]	0.315	71	+1.4 [+0.0, +3.5]	1.000
Qwen2.5-7B	no	0.99	96	−1.0 [−6.2, +2.6]	0.424	73	+0.0 [−5.5, +4.8]	1.000
8–14B, released 2025								
Ministral-3 8B	no	0.98	92	+17.8 [+10.5, +25.9]	0.0002	69	+17.1 [+8.9, +26.1]	0.0012
Qwen3-14B	yes	1.00	91	+13.2 [+6.0, +20.7]	0.0013	69	+5.6 [−1.7, +11.6]	0.365
Frontier								
GPT-4o	no	1.00	95	+13.7 [+7.5, +21.4]	0.0030	72	+13.9 [+6.9, +22.9]	0.0049
Gemini 2.5 Flash	def.	1.00	95	+13.3 [+3.7, +23.5]	0.049	71	+11.0 [−0.2, +21.8]	0.230
Claude Sonnet 4	no	0.99	93	+13.6 [+7.7, +19.9]	0.0001	70	+8.8 [+2.1, +15.1]	0.056
GLM-4.7	def.	0.98	89	+19.5 [+11.6, +27.9]	0.0009	67	+11.9 [+4.5, +20.9]	0.036
GPT-5 nano	yes	0.99	91	+30.2 [+19.8, +40.3]	<0.0001	68	+23.3 [+11.0, +35.0]	0.0016

4.5 (Zeng et al., 2025), and GPT-5 nano (Singh et al., 2025)). Llama, Qwen2.5, Ministral, GPT-4o and Claude Sonnet 4 (extended thinking not enabled) act without a reasoning phase; Qwen3-14B and GPT-5 nano reason; Gemini 2.5 Flash and GLM-4.7 use their providers' default reasoning settings.

Serving and scoring. All models are served through one API gateway at temperature 0 with seed 42 wherever the endpoint supports them (GPT-5 nano accepts only its default sampling; Anthropic endpoints reject a seed). The per-turn generation budget is 1,024 tokens for the non-reasoning 7–8B models and 8,192 for the others; engagement is at least 0.94 in every cell, so the budget does not truncate behavior. Task success is determined by the harness verdict. The primary experiment comprises $9 \times 4 \times 100 = 3{,}600$ trials; the ablation adds 800 and the HumanEval scope check 2,400. Appendix A lists endpoints, settings, and token use.

3.3 Measures and statistics

Engagement is the fraction of trials with at least one tool call. For a trajectory with $k \geq 1$ tool errors, the *error recovery rate* (ERR) is the fraction of errors whose next tool outcome is a pass. The primary contrast is the task-level mean, over tasks with at least one error under both conditions, of $\text{ERR}_{\text{real}} - \text{ERR}_{\text{scr}}$. We test it with the Wilcoxon signed-rank test using Pratt's treatment of zeros (Wilcoxon, 1945; Pratt, 1959), with 95% BCa bootstrap intervals from 10,000 resamples (Efron, 1987) and Holm correction over the nine models (Holm, 1979). The pinned-task analysis was added after the audit; it uses an exact-by-simulation sign-flip permutation test (200,000 draws) and is also Holm-corrected across nine models. For each cell, we report the minimum detectable effect at 80% power, $\text{MDE}_{80} = 2.80\,\hat{\sigma}_d/\sqrt{n}$, and claim equivalence only where a two-one-sided test at ± 5 pp rejects (Schuirmann, 1987; Lakens, 2017). Task success, defined for each trial, is compared across conditions using exact McNemar tests (McNemar, 1947). Three behavioral measures do not require a successful repair: whether an error is followed by another tool call (*re-attempt*), whether that call resubmits byte-identical code (*identical*), and, after `NameError: name 'X' is not defined`, whether the next submission defines `X` (*adoption*).

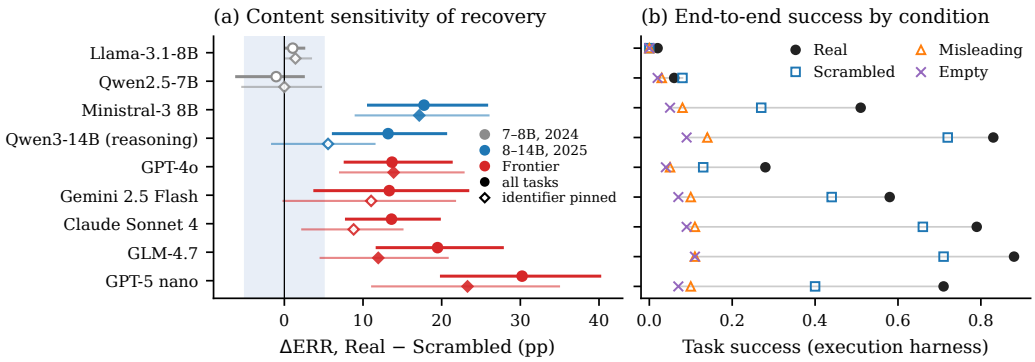

Figure 1: (a) Primary contrast with 95% BCa intervals on all tasks (circles) and on the 75 identifier-pinned tasks (diamonds); filled markers are significant after Holm correction over nine tests; the band marks ± 5 pp. (b) Task success scored by the execution harness on all 100 tasks per cell. SCRAMBLED retains 46–87% of REAL success among the seven content-sensitive models; MISLEADING and EMPTY retain almost none.

4 RESULTS

4.1 MOST MODELS USE ERROR CONTENT, AND SIZE DOES NOT SEPARATE THEM

Table 1 and Figure 1 report the primary contrast. In the pre-specified analysis over all tasks, seven models recover significantly more often from REAL than from SCRAMBLED errors after Holm correction, with effects ranging from $+13.2$ to $+30.2$ pp. Gemini 2.5 Flash provides the weakest evidence ($p_{\text{Holm}} = 0.049$, and its $+13.3$ pp effect lies below its MDE_{80} of 14.2 pp). The two models released in 2024 show no effect: Qwen2.5-7B's interval $[-6.2, +2.6]$ supports equivalence at ± 5 pp (TOST $p = 0.037$); Llama-3.1-8B's interval $[+0.0, +2.7]$ does as well ($p < 0.001$), but its recovery rate is at most 0.011 in any condition, so its equivalence reflects an inability to repair as much as indifference to content (§4.5).

Restricting the analysis to the 75 tasks whose target name the transformer never moves changes the result for three models. Ministral-3 8B, GPT-4o, GLM-4.7, and GPT-5 nano remain significant after Holm correction ($+11.9$ to $+23.3$ pp); Claude Sonnet 4 is marginal ($+8.8$ pp $[+2.1, +15.1]$, $p_{\text{Holm}} = 0.056$); Gemini 2.5 Flash ($+11.0$ pp) and Qwen3-14B ($+5.6$ pp) lose significance. On the 25 affected tasks, most of the same models show much larger contrasts (Appendix D), indicating that part of the all-task effect is carried by displacement of the missing name, which §4.2 examines directly. The pinned subset is also 25% smaller, reducing statistical power.

Two conclusions hold in both analyses. Parameter count does not separate models that use error content from those that do not: Ministral-3 8B, the same size as Llama-3.1-8B, shows $+17.8$ pp on all tasks and $+17.1$ pp on pinned tasks. A reasoning phase does not separate them either: GPT-4o and Ministral-3 8B act without one and remain significant in both analyses.

Task success shows the same pattern across all trials, without conditioning on engagement (Figure 1b; Appendix F). Pooled across the nine models, exact McNemar tests yield 153 task–model pairs solved only under REAL versus 28 solved only under SCRAMBLED ($p = 4.7 \times 10^{-22}$), and 413 versus 1 for REAL against EMPTY ($p = 2.0 \times 10^{-122}$). Among the seven content-sensitive models, SCRAMBLED messages retain 46–87% of the success achieved with REAL messages, while MISLEADING and EMPTY messages retain almost none (success rates of only 0.04 to 0.14). Knowing only that an error occurred is nearly worthless to these agents; most of the benefit survives the destruction of compositional meaning, so the remaining question is which preserved signal carries it. Alternative estimands (recovery at the first error only, any recovery per trajectory) yield the same result (Appendix E).

4.2 WHAT IS USED: THE ACTIONABLE TOKEN, ITS POSITION, AND ITS FRAME

In the primary experiment, 72.3% of errors under REAL are `NameErrors`. MBPP-sanitized tests call a function whose name is not stated in the task description, so initial attempts often implement the

Table 2: After a `NameError`, what the next action depends on. Pooled over the seven content-sensitive models. *Pinned name*: the missing name matches the transformer's identifier patterns, so it always stays in its slot. Adoption: the next submission defines the missing name. Recovery: the next tool outcome is a pass.

Message the agent saw	Errors	Adoption	Recovery
REAL, pinned name	674	0.540	0.364
SCRAMBLED, pinned name in its slot, frame scrambled	823	0.352	0.230
REAL, unpinned name	259	0.347	0.328
SCRAMBLED, unpinned name still in its slot	84	0.298	0.286
SCRAMBLED, name moved out of its slot	262	0.038	0.038

Table 3: Per-error recovery by exception class. *Identifier*: `NameError`, whose repair target is a name that survives scrambling verbatim. *Semantic*: `AssertionError`, `TypeError`, `ValueError`, `IndexError`, `KeyError`, whose diagnostic content scrambling removes. *Structural*: `ImportError`, `SyntaxError`. The unit is the individual tool error; recovery means the next tool outcome is a pass.

Model(s)	Error class	n_{real}	$\text{Rec.}_{\text{real}}$	n_{scr}	Rec._{scr}	Δ (pp)
Frontier, pooled (5)	identifier	789	0.295	1,082	0.141	$+15.4$
	semantic	215	0.256	173	0.254	$+0.1$
	structural	74	0.108	73	0.110	-0.1
Ministral-3 8B	identifier	203	0.187	256	0.066	$+12.1$
	semantic	47	0.191	42	0.143	$+4.9$
Qwen3-14B, reasoning on (8,192)	`NameError`	103	0.583	123	0.431	$+15.2$
	`AssertionError`	17	0.824	29	0.379	$+44.4$
Qwen3-14B, reasoning off	`NameError`	300	0.057	284	0.046	$+1.1$
	`AssertionError`	17	0.118	19	0.105	$+1.2$

correct logic under the wrong name (§4.6). A `NameError` identifies its own repair target, making it a natural probe of what the agent extracts from the message (Table 2).

Position matters most. When scrambling moves the missing name out of its `name '...'` slot, as in `Traceback (call is_Diff most defined) ... NameError: name 'last' is not string`, the seven models adopt it in 3.8% of cases and recover in 3.8%, even though the name remains in the message. The frame also matters. For names the transformer pins, which always remain in their slot, scrambling only the surrounding words (e.g. `NameError: defined 'freq_count' is not last`) lowers adoption from 0.540 to 0.352 and recovery from 0.364 to 0.230. Agents do not simply copy a quoted token: they use the conventional frame, *name ... is not defined*, to determine how to act on it, and rely on the token occupying the position that frame assigns it.

Beyond `NameError`s, the evidence is thinner (Table 3). Pooled over the frontier models, REAL exceeds SCRAMBLED by $+15.4$ pp on `NameError`s, a figure that includes displaced names, and by $+0.1$ pp on semantic errors, whose diagnostic content scrambling removes. Individual frontier models range from -6.2 to $+9.5$ pp on semantic errors, and the cells are small ($\text{MDE}_{80} \approx 12$ pp for an unpaired comparison). Qwen3-14B with reasoning is the clearest exception, at $+18.0$ pp on semantic errors (33 and 45 errors; Appendix G). On this evidence, current agents extract error information primarily by recognizing a conventional template and locating its actionable slot rather than by reading the diagnostic message compositionally.

4.3 DELIBERATION DECIDES WHETHER QWEN3-14B ACTS ON AN ERROR

Qwen3-14B exposes reasoning as an API parameter, allowing us to hold the weights, system prompt, provider, scaffold and tasks fixed (Table 4). Suppressing reasoning changes how the model responds to any error. With reasoning off, it re-attempts after 77% of errors but resubmits identical code on 69% of re-attempts, compared with 2–16% with reasoning on; it defines the missing name after a

Table 4: Within-model ablation on Qwen3-14B: identical weights, system prompt, provider, scaffold and tasks; reasoning is toggled through the provider's reasoning parameter. ΔERR (pp) is shown on all tasks (Wilcoxon) and on the 75 identifier-pinned tasks (sign-flip permutation), each Holm-corrected over the three arms. Adoption: after `NameError: name 'X' is not defined`, the fraction of next submissions that define `X`. Identical: fraction of re-attempts under REAL that resubmit byte-identical code.

			All tasks		Pinned			
Arm	Eng.	ERR_{real}	ΔERR	p_{Holm}	ΔERR	p_{Holm}	Adoption	Identical
reasoning on, 1,024 tok	0.59	0.670	+11.9	0.041	+7.7	0.365	1.00	0.02
reasoning on, 8,192 tok	1.00	0.723	+13.2	0.0008	+5.6	0.365	0.85	0.16
reasoning off, 1,024 tok	1.00	0.148	+0.9	0.51	−0.2	1.000	0.09	0.69

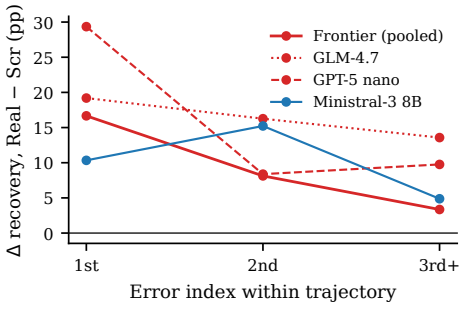

Figure 2: REAL−SCRAMBLED difference in recovery at the k-th error of a trajectory. By the k-th error the agent has seen $k-1$ earlier messages under the same condition. Populations at $k \geq 2$ are conditioned on earlier failure. Counts in Appendix J.

`NameError` in 9% of cases, compared with 85–100%; and its recovery rate under REAL falls from 0.67–0.72 to 0.15. Appendix M shows a trajectory in which it resubmits the same code three times after receiving `Did you mean: 'first_digit'?`.

The REAL–SCRAMBLED contrast is consistent with this but weaker. Across all tasks, it is +11.9 pp with reasoning at a matched 1,024-token budget and +0.9 pp without. At 1,024 tokens, the reasoning arm engages on only 59% of trials and returns no tool call on the remainder, while at 8,192 tokens it engages fully and shows +13.2 pp. For the pinned tasks, however, the contrast with reasoning is +7.7 pp at 1,024 tokens and +5.6 pp at 8,192 tokens; neither is significant. In this model, deliberation is what turns an error message into an action; once the name remains in place, the additional dependence of that action on the message's wording is small. Across models, deliberation is not required: GPT-4o and Ministral-3 8B use error content without a reasoning phase.

4.4 SENSITIVITY DECAYS ACROSS SUCCESSIVE ERRORS

Because the condition is fixed per trajectory, later errors occur after earlier messages rendered under the same condition, allowing us to examine how sensitivity evolves within a multi-turn episode (Figure 2). Pooled over the frontier models, the difference is +16.7 pp at the first error, +8.1 pp at the second, and +3.4 pp from the third onward. GLM-4.7 is the exception, remaining at +19.2, +16.3, and +13.6 pp. The decline coincides with convergence in behavior: pooled recovery from the third error onward is 0.096 under REAL and 0.062 under SCRAMBLED. Because the populations at later indices are selected by earlier failure, this pattern characterizes the observed trajectories rather than a causal effect of depth.

4.5 A NULL EFFECT ON RECOVERY IS NOT A NULL EFFECT ON PROCESSING

The two 2024 models and Qwen3-14B without reasoning do not recover more often with real messages, but their behavior still depends on the message (Table 5). Just over half of Llama-3.1-8B's errors under REAL are malformed tool calls (`ValueError: 'code' argument must be a non-empty string`) rather than tracebacks. It re-attempts after 47% of REAL errors and 41% of MISLEADING errors, but after only 14% of SCRAMBLED and 4% of EMPTY errors: it retries when the message is well-formed English, whether or not it is correct, and then fails to repair. Qwen2.5-7B and Qwen3-14B without reasoning re-attempt at a constant rate but resubmit identical code 69–84%

Table 5: Behaviour of the models whose recovery does not depend on content. Both measures are defined per tool error and do not require the repair to succeed. Re-attempt: the error is followed by another tool call. Identical: the re-attempt resubmits byte-identical code.

Model	Measure	REAL	SCRAMBLED	MISLEADING	EMPTY
Llama-3.1-8B	re-attempt	0.471	0.138	0.412	0.041
	identical	0.232	0.400	0.636	0.000
Qwen2.5-7B	re-attempt	0.740	0.750	0.747	0.718
	identical	0.706	0.722	0.743	0.433
Qwen3-14B, reasoning off	re-attempt	0.769	0.753	0.744	0.748
	identical	0.685	0.717	0.841	0.568

of the time whenever the message contains content, compared with 43–57% when it is empty: they respond to the presence of text rather than to what it says. A flat recovery rate should therefore not be interpreted as evidence that a model ignores its errors; these models process the form of the message but fail at the subsequent step of acting on it.

4.6 SCOPE: THE PROTOCOL NEEDS A REPAIR LOOP

We applied the same protocol to 100 HumanEval tasks (Chen et al., 2021) across six models (Appendix K). Capable models solve 93–96% of tasks on their first attempt, so task success is 0.93–0.99 in every condition, including EMPTY, and the paired contrast has 0–4 usable tasks per model. On MBPP-sanitized, the same models succeed on only 5–7% of first attempts because the tests call a function name that the prompt never specifies. The two weaker models fail often on HumanEval, but 82% and 58% of their errors are malformed tool calls rather than tracebacks. The protocol measures error-message use only when agents make errors worth reading, and the benchmark's error distribution determines what the protocol can detect.

5 DISCUSSION

What the protocol separates. The conditions decompose the value of an error message. Presence alone (EMPTY) is worth almost nothing to current agents. The actionable token is worth substantially more, but only when it remains where the frame places it: a displaced name is rarely used. The frame's words also matter when the token stays in place, with adoption falling by about a third when they are scrambled. Our evidence on diagnostic meaning beyond such frames is limited because the benchmark produces few semantic errors. A single REAL–SCRAMBLED contrast therefore mixes these components, which is why we recommend reporting it alongside the slot decomposition.

A lesson for counterfactual transformers. Our transformer pinned identifiers by pattern, and those patterns missed mixed-case names on a quarter of the tasks. The consequence was not a small bias but a distinct effect of comparable magnitude, visible only because we logged both the original and rendered messages. Future scrambles should pin by role rather than by pattern, for example by pinning every quoted span and every identifier that occurs in the submitted program, and should audit slot preservation rather than token presence alone.

Size, era and deliberation. Because an 8B model appears on each side of the boundary in both analyses, parameter count is not the operative variable. Among the small models, both released in 2025 use error content in the all-task analysis (Ministral-3 8B robustly, Qwen3-14B only before identifiers are pinned), whereas both released in 2024 show no effect in either analysis, which is consistent with error-message use being a capability acquired through more recent training, perhaps through agentic post-training. GPT-4o, a 2024 frontier model, also uses it, so release date alone does not explain the pattern; with only two small models on each side, we treat this as a hypothesis rather than a finding. Within Qwen3-14B, deliberation determines whether the model acts on an error at all, but across models, it is not necessary.

Implications for tool design. If agents act on a short conventional frame and the token in its expected slot, then those are the parts of an error message that matter most. Tool authors who

introduce new error formats therefore risk disrupting the signals agents rely on. Whether verbose diagnostics add value beyond the frame is a natural next application of the protocol.

6 LIMITATIONS

Transformer pinning. The identifier patterns miss mixed-case names, so on 25 tasks the SCRAMBLED condition can move the actionable token. We report both analyses; the pinned-task analysis was added after the audit and has less statistical power. **One benchmark, one dominant error class.** MBPP-sanitized under a single-tool scaffold produces mostly `NameError`s, while HumanEval produces too few errors to measure. The semantic-error contrast is underpowered. Benchmarks with semantically rich failures, such as APPS or LiveCodeBench (Hendrycks et al., 2021; Jain et al., 2025), are the natural extension. However, their dominant failure is a wrong answer rather than a traceback, and scrambling messages whose diagnostic content is numeric requires a different pinning rule. **Observational multi-turn analysis.** Our multi-turn analysis follows successive errors within a task and is observational. **Model sample.** We evaluate nine models, with two on each side of the 2024/2025 boundary among small models. **Serving.** Models are served through third-party endpoints whose quantization and batching we do not control; GPT-5 nano does not accept greedy decoding, and Anthropic endpoints do not accept a seed. A replication of four small models on a local 4-bit serving stack agrees qualitatively (Appendix L). **Controls.** MISLEADING and EMPTY are not matched in length or structure, and the MISLEADING generator is a single small model; we therefore use them only as reference conditions for interpreting the primary contrast.

7 CONCLUSION

Counterfactual error scrambling holds the surface of an error message fixed while destroying its meaning, allowing us to measure how much an agent's next action depends on what is lost. Across nine models, seven use error content in the pre-specified analysis and four do so robustly when identifiers are held in place, including an 8B model without a reasoning phase, while two models of similar size from an earlier generation do not. What agents use is the actionable token, its position, and the conventional frame around it; within one model, deliberation determines whether the message is acted on at all. The audit of our own transformer shows that the position of a single token can carry an effect as large as meaning, so the protocol is only as reliable as its pinning. It is inexpensive, requires only an inference endpoint, and applies to any tool whose errors can be intercepted.

AI USE STATEMENT

We used generative AI tools to assist with experimental design and feedback, implementation and debugging of parts of the experimental and analysis infrastructure, interpretation of results, and refinement of hypotheses. As part of the experimental method, a language model generates the MISLEADING error messages described in Section 3.1. Generative AI tools were also used to assist with drafting and editing text, grammar checking, formatting references, and presentation. All AI-assisted outputs were reviewed and verified by the authors, including recomputation of reported results, validation of analysis code and key measures, and verification of citations against their sources. The authors take full responsibility for the final text, claims, results, and artifacts.

ETHICS STATEMENT

This work studies model behavior on a public programming benchmark and involves no human subjects or personal data. Model outputs were obtained through commercial and open-weight endpoints under their terms of use; released trajectories that contain outputs of proprietary models are provided for verification and research, subject to those providers' terms. We see no direct risk of harm from the protocol, which reduces rather than increases the information available to an agent.

REPRODUCIBILITY STATEMENT

Section 3.1 and Appendix B specify the transformer, including its pinning rules, its algorithm and its audit; Appendix A lists every endpoint, generation budget and sampling setting; Appendix N gives all prompts. The supplementary material contains all trajectories used in this paper (including

original and rendered error messages for every error turn), the per-run configuration snapshots, and the analysis code that produces every table and figure in the paper and the appendix with a single script. The collection harness will be released with the camera-ready version.

REFERENCES

Anthropic. System card: Claude Opus 4 & Claude Sonnet 4. https://www.anthropic.com/claude-4-system-card, May 2025. Accessed: 2026-09-20.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Xinyun Chen, Maxwell Lin, Nathanael Schaerli, and Denny Zhou. Teaching large language models to self-debug. In *International Conference on Learning Representations*, volume 2024, pp. 8746–8825, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Bradley Efron. Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 82(397):171–185, 1987. ISSN 01621459, 1537274X.

Zhibin Gou, Zhihong Shao, Yeyun Gong, yelong shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, volume 2024, pp. 57734–57811, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with APPS. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength natural language processing in python. Zenodo, 2020.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations*, volume 2024, pp. 32808–32824, 2024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Naman Jain, Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, volume 2024, pp. 54107–54157, 2024.

Daniël Lakens. Equivalence tests: A practical primer for t tests, correlations, and meta-analyses. *Social psychological and personality science*, 8(4):355–362, 2017.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21314–21328. Curran Associates, Inc., 2022. doi: 10.52202/068431-1549.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594. Curran Associates, Inc., 2023. doi: 10.52202/075280-2019.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Theo X. Olausson, Jeevana Priya Inala, Chenglong Wang, Jianfeng Gao, and Armando Solar-Lezama. Is self-repair a silver bullet for code generation? In *International Conference on Learning Representations*, volume 2024, pp. 36545–36593, 2024.

Thang Pham, Trung Bui, Long Mai, and Anh Nguyen. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1145–1160, 2021.

John W Pratt. Remarks on zeros and ties in the wilcoxon signed rank procedures. *Journal of the American Statistical Association*, 54(287):655–667, 1959.

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, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pp. 68539–68551. Curran Associates, Inc., 2023. doi: 10.52202/075280-2997.

Donald J Schuirmann. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of pharmacokinetics and biopharmaceutics*, 15(6):657–680, 1987.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai GPT-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2888–2913, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.230.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. InterCode: Standardizing and benchmarking interactive coding with execution feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 23826–23854. Curran Associates, Inc., 2023. doi: 10.52202/075280-1035.

John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, pp. 50528–50652. Curran Associates, Inc., 2024. doi: 10.52202/079017-1601.

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

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. GLM-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

A MODELS, ENDPOINTS AND TOKEN USE

Table 6: Endpoints and settings. All models were accessed through the same API gateway with `require_parameters` enabled, so requests were routed only to endpoints that honor the sent parameters. Budget is the per-turn generation limit in tokens.

Model	Endpoint	Budget	Sampling	Reasoning
Llama-3.1-8B	`meta-llama/llama-3.1-8b-instruct`	1,024	T=0, seed 42	none
Qwen2.5-7B	`qwen/qwen-2.5-7b-instruct`	1,024	T=0, seed 42	none
Ministral-3 8B	`mistralai/ministral-8b-2512`	1,024	T=0, seed 42	none
Qwen3-14B	`qwen/qwen3-14b`	8,192	T=0, seed 42	enabled
GPT-4o	`openai/gpt-4o`	8,192	T=0, seed 42	none
Gemini 2.5 Flash	`google/gemini-2.5-flash`	8,192	T=0, seed 42	provider default
Claude Sonnet 4	`anthropic/claude-sonnet-4`	8,192	T=0, no seed	not enabled
GLM-4.7	`z-ai/glm-4.7`	8,192	T=0, seed 42	provider default
GPT-5 nano	`openai/gpt-5-nano`	8,192	provider default	enabled

The primary experiment comprises nine models $\times$ four conditions $\times$ 100 tasks $=$ 3,600 trials. The Qwen3-14B ablation adds two 400-trial arms (reasoning enabled at 1,024 tokens; reasoning disabled at 1,024 tokens); all three Qwen3-14B arms use a single provider. The HumanEval scope check comprises six models $\times$ four conditions $\times$ 100 tasks $=$ 2,400 trials. MISLEADING messages are generated by Mistral-7B-Instruct at temperature 0.7, with up to three attempts per message.

Table 7: Token use and engagement over all 400 trials per model. No-call trials contain no tool call. Completion tokens are per assistant turn and include hidden reasoning where the provider reports it.

Model	No-call trials	Tokens/turn (median)	Tokens/trial (mean, k)	Tool calls/trial
Llama-3.1-8B	24	51	1.1	1.36
Qwen2.5-7B	4	61	2.2	3.66
Ministral-3 8B	11	124	2.4	3.44
Qwen3-14B	0	1235	7.1	3.02
GPT-4o	0	232	3.0	3.65
Gemini 2.5 Flash	0	116	2.1	3.09
Claude Sonnet 4	3	224	4.1	3.30
GLM-4.7	7	280	5.0	3.12
GPT-5 nano	5	910	5.1	3.21
Qwen3-14B, reas. 1k	169	604	3.4	1.10
Qwen3-14B, no reas.	0	61	2.1	3.67

In the reasoning-enabled 1,024-token arm of Qwen3-14B, the 169 no-call trials contain no recorded assistant turn: neither a tool call nor visible content was returned. Because increasing the budget to 8,192 tokens while holding everything else fixed eliminates these no-call trials, we attribute them to the reasoning process exhausting the generation budget.

B THE TRANSFORMER: ALGORITHM, AUDIT AND EXAMPLES

Input: error message s, item identifier u, experiment seed σ.

1. Initialise a generator with seed $\sigma \oplus \mathrm{BLAKE2b}_{64}(u)$.
2. Tag s with spaCy, keeping each token's text, POS tag and trailing whitespace.
3. Find the spans of s matching any identifier pattern: lower-case snake_case containing an underscore, CamelCase with at least two components, a path, or a file name with an extension.
4. Pin a token if it is punctuation, whitespace, or number-like, if it is one of the keywords (`Error`, `Exception`, `Traceback`, `line`, `File`, `in`, `raise`, `at`), or if it overlaps an identifier span.
5. Collect the unpinned tokens tagged NOUN, VERB, ADJ, or ADV, and chain them into groups of consecutive tokens at most 12 positions apart.
6. For each group of two or more tokens, draw a permutation that is not the identity (rejection sampling, with an adjacent swap as fallback) and move the tokens accordingly.
7. Return the permuted token texts joined with the *original* whitespace, so the character length is unchanged.

Figure 3: The SCRAMBLED transformer.

Table 8: Audit over the 2,302 REAL→SCRAMBLED substitution pairs of the primary experiment (top; categories as matched by the identifier patterns, so mixed-case names are counted as words), and structural match of each condition to the original error (bottom).

Token category	Moved	Kept	% moved
snake_case identifiers	37	1,903	1.9
CamelCase and exception names	29	2,305	1.2
Numeric literals	41	3,145	1.3
Natural-language words	17,418	16,621	51.2

Condition	n	Median length ratio	Line structure preserved	Mean Δ stack frames
SCRAMBLED	2,302	1.000	100.0%	$+0.00$
MISLEADING	2,984	2.044	35.0%	$+0.37$
EMPTY	2,900	0.151	3.3%	-0.98

The unpinned most-frequent missing names of the 25 affected tasks are `Find_Max`, `Find_Min`, `check`, `comb`, `count_Substrings`, `dif_Square`, `divisor`, `even_Power_Sum`, `find`, `find_Parity`, `first_Digit`, `frequency`, `highest_Power_of_2`, `is_Diff`, `lps`, `math`, `maximum`, `odd_Equivalent`, `power`, `search` and `sequence`; some recur across tasks, and `math` arises from a missing import. Single-word lower-case names such as `power` are unpinned for the same reason. Under MISLEADING, the original missing name appears in 7–24% of messages, depending on the model.

An example from task `mbpp-306`, with a pinned name, whose original error is

```
Traceback (most recent call last):
  File "<string>", line 1, in <module>
NameError: name 'max_sum_increasing_subseq' is not defined.
  Did you mean: 'max_sum_increasing_subseq_with_k'?
```

is rendered under SCRAMBLED as

```
Traceback (most name defined mean):
  File "<last>", line 1, in <string>
NameError: recent 'max_sum_increasing_subseq' is not module.
  Did you call: 'max_sum_increasing_subseq_with_k'?
```

and under MISLEADING as

```
Traceback (most recent call last):
      File "<string>", line 1, in <module>
NameError: name 'max_sum_decreasing_subseq' is not defined.
  Did you mean: 'min_sum_decreasing_subseq_with_k'?
```

(line breaks inside the final line added for width). With an unpinned name (`mbpp-77`, target `is_Diff`) the rendering is `Traceback (call is_Diff most defined): ... NameError: name 'last' is not string.`

C FULL RESULTS

Table 9: Per-cell results. ERR is the mean over trajectories with at least one error; these cell means differ slightly from the paired means in Table 1. CSI $= (\mathrm{ERR}_{\mathrm{real}} - \mathrm{ERR}_{\mathrm{scr}})/(\mathrm{ERR}_{\mathrm{real}} - \mathrm{ERR}_{\mathrm{empty}})$, clipped to $[0, 1]$ and omitted when the denominator is below 0.05; because EMPTY is not length-matched, CSI is descriptive only. Trajectories with errors are listed for REAL/SCR/MIS/EMPTY.

Model	Eng.	$\mathrm{ERR}_{\mathrm{real}}$	$\mathrm{ERR}_{\mathrm{scr}}$	$\mathrm{ERR}_{\mathrm{mis}}$	$\mathrm{ERR}_{\mathrm{empty}}$	CSI	With errors
Llama-3.1-8B	0.94	0.011	0.000	0.000	0.000	–	94/94/94/94
Qwen2.5-7B	0.99	0.031	0.042	0.000	0.000	–	96/96/96/97
Ministral-3 8B	0.98	0.362	0.192	0.032	0.000	0.47	94/92/93/92
Qwen3-14B	1.00	0.726	0.596	0.034	0.040	0.19	92/92/93/96
GPT-4o	1.00	0.222	0.079	0.010	0.000	0.64	97/95/96/96
Gemini 2.5 Flash	1.00	0.463	0.330	0.046	0.021	0.30	95/95/95/95
Claude Sonnet 4	0.99	0.624	0.486	0.022	0.020	0.23	93/94/93/93
GLM-4.7	0.98	0.758	0.567	0.029	0.025	0.26	91/92/92/92
GPT-5 nano	0.99	0.607	0.297	0.032	0.004	0.51	92/92/94/91
Qwen3-14B, reas. 1k	0.59	0.691	0.557	0.000	0.000	0.19	54/53/55/53
Qwen3-14B, no reas.	1.00	0.149	0.139	0.000	0.000	0.07	97/96/95/96

Table 10: Statistics for the pre-specified all-task contrast. p: Wilcoxon signed-rank with Pratt's treatment of zeros; p_{Holm} over nine models; TOST: larger of the two one-sided p-values against ± 5 pp (small values support equivalence). Discordant: tasks with a non-zero paired difference.

Model	n	Discordant	ΔERR	MDE_{80}	p	p_{Holm}	TOST
Llama-3.1-8B	94	2	+1.1	2.1	0.157	0.315	7.2×10^{-8}
Qwen2.5-7B	96	6	-1.0	6.2	0.424	0.424	0.037
Ministral-3 8B	92	34	+17.8	11.1	3.2×10^{-5}	0.00023	0.999
Qwen3-14B	91	24	+13.2	10.5	0.00027	0.0013	0.985
GPT-4o	95	18	+13.7	9.9	0.00076	0.003	0.993
Gemini 2.5 Flash	95	38	+13.3	14.2	0.016	0.049	0.949
Claude Sonnet 4	93	33	+13.6	8.8	1.4×10^{-5}	0.00011	0.997
GLM-4.7	89	40	+19.5	11.8	0.00015	0.00087	1
GPT-5 nano	91	46	+30.2	14.7	2.5×10^{-6}	2.3×10^{-5}	1

D IDENTIFIER-DISPLACEMENT SENSITIVITY

Table 11: REAL–SCRAMBLED contrast on all tasks, on the 75 tasks whose target name the transformer pins, and on the 25 tasks whose name it can move. All columns use the sign-flip permutation test with Holm correction over nine models, so the all-task p-values differ slightly from the Wilcoxon values in Table 1. Intervals are 95% BCa.

	All tasks			Pinned tasks		
Model	n	ΔERR [95% CI]	p_{Holm}	n	ΔERR [95% CI]	p_{Holm}
Llama-3.1-8B	94	+1.1 [+0.0, +2.7]	0.997	71	+1.4 [+0.0, +3.5]	1.000
Qwen2.5-7B	96	−1.0 [−6.2, +2.6]	0.997	73	+0.0 [−5.5, +4.8]	1.000
Ministral-3 8B	92	+17.8 [+10.5, +25.9]	0.0001	69	+17.1 [+8.9, +26.1]	0.0012
Qwen3-14B	91	+13.2 [+6.0, +20.7]	0.0025	69	+5.6 [−1.7, +11.6]	0.365
GPT-4o	95	+13.7 [+7.5, +21.4]	0.0003	72	+13.9 [+6.9, +22.9]	0.0049
Gemini 2.5 Flash	95	+13.3 [+3.7, +23.5]	0.033	71	+11.0 [−0.2, +21.8]	0.230
Claude Sonnet 4	93	+13.6 [+7.7, +19.9]	0.0001	70	+8.8 [+2.1, +15.1]	0.056
GLM-4.7	89	+19.5 [+11.6, +27.9]	<0.0001	67	+11.9 [+4.5, +20.9]	0.036
GPT-5 nano	91	+30.2 [+19.8, +40.3]	<0.0001	68	+23.3 [+11.0, +35.0]	0.0016

On the 25 tasks whose name can be displaced, the contrasts are larger than on the pinned tasks for six of the seven models whose recovery depends on content: +19.6 pp for Ministral-3 8B, +37.1 for Qwen3-14B, +13.0 for GPT-4o (the exception), +20.1 for Gemini 2.5 Flash, +28.3 for Claude Sonnet 4, +42.4 for GLM-4.7, and +50.7 for GPT-5 nano, based on 22–24 tasks per model. The identifier-pinned subset is defined by task rather than outcome: a task is excluded if its most frequent missing name across all trajectories does not match the transformer's identifier patterns.

E ALTERNATIVE ESTIMANDS

Table 12: The REAL–SCRAMBLED comparison under three task-level binary estimands, each evaluated with an exact McNemar test (tasks positive only under REAL / only under SCRAMBLED). First error: the first tool error is followed by a pass. Any recovery: at least one error in the trajectory is followed by a pass. Success: the harness reports a pass at any point. The first two estimands are defined on tasks with at least one error under both conditions; success is defined on all 100 tasks.

	First error		Any recovery		Success	
Model	R / S	p	R / S	p	R / S	p
Llama-3.1-8B	0/0	1	2/0	0.5	2/0	0.5
Qwen2.5-7B	2/2	1	2/4	0.688	2/4	0.688
Ministral-3 8B	14/4	0.031	25/1	8.0×10^{-7}	25/1	8.0×10^{-7}
Qwen3-14B	15/2	0.0023	13/2	0.0074	13/2	0.0074
GPT-4o	12/1	0.0034	16/0	3.1×10^{-5}	16/1	0.00027
Gemini 2.5 Flash	18/6	0.023	24/10	0.024	24/10	0.024
Claude Sonnet 4	11/1	0.0063	17/3	0.0026	17/4	0.0072
GLM-4.7	19/2	0.00022	19/2	0.00022	20/3	0.00049
GPT-5 nano	30/4	6.2×10^{-6}	33/3	2.3×10^{-7}	34/3	1.2×10^{-7}
Pooled	121/22	9.2×10^{-18}	151/25	3.7×10^{-23}	153/28	4.7×10^{-22}

F TASK SUCCESS BY CONDITION

Table 13: Task success from the execution harness on all 100 tasks per cell.

Model	REAL	SCR	MIS	EMPTY
Llama-3.1-8B	0.02	0.00	0.00	0.00
Qwen2.5-7B	0.06	0.08	0.03	0.02
Ministral-3 8B	0.51	0.27	0.08	0.05
Qwen3-14B	0.83	0.72	0.14	0.09
GPT-4o	0.28	0.13	0.05	0.04
Gemini 2.5 Flash	0.58	0.44	0.10	0.07
Claude Sonnet 4	0.79	0.66	0.11	0.09
GLM-4.7	0.88	0.71	0.11	0.11
GPT-5 nano	0.71	0.40	0.10	0.07

Pooled over the nine models, exact McNemar tests against REAL yield 153 / 28 for SCRAMBLED ($p = 4.7 \times 10^{-22}$), 395 / 1 for MISLEADING ($p = 4.9 \times 10^{-117}$), and 413 / 1 for EMPTY ($p = 2.0 \times 10^{-122}$).

G ERROR CLASSES

Table 14: Distribution of tool errors under REAL, by model (%). ValueErrors from Llama-3.1-8B are almost all malformed tool calls (`'code' argument must be a non-empty string`).

Model	Errors	Name	Assertion	Type	Value	Import	Syntax	Other
Llama-3.1-8B	174	24.1	2.9	0.0	52.3	1.1	17.2	2.3
Qwen2.5-7B	358	89.7	1.1	3.1	0.0	3.6	1.7	0.8
Ministral-3 8B	260	78.1	11.9	5.8	0.0	3.5	0.4	0.4
Qwen3-14B	147	70.1	11.6	10.9	0.0	7.5	0.0	0.0
GPT-4o	318	75.5	17.6	0.9	0.3	5.3	0.0	0.3
Gemini 2.5 Flash	234	77.8	8.5	5.6	1.3	6.0	0.0	0.9
Claude Sonnet 4	189	74.1	14.3	6.9	0.0	4.2	0.0	0.5
GLM-4.7	151	60.9	16.6	8.6	0.0	13.9	0.0	0.0
GPT-5 nano	186	72.6	10.2	9.7	0.0	5.9	1.6	0.0

Table 15: Per-error recovery by error class and model; cells with fewer than 15 errors in either condition are omitted. The identifier class includes displaced names.

Model	Class	n_{real}	Rec.$_{\text{real}}$	n_{scr}	Rec.$_{\text{scr}}$	Δ (pp)
Llama-3.1-8B	semantic	96	0.000	93	0.000	+0.0
Qwen2.5-7B	identifier	321	0.009	322	0.012	-0.3
	semantic	15	0.000	16	0.062	-6.2
	structural	22	0.000	26	0.000	+0.0
Ministral-3 8B	identifier	203	0.187	256	0.066	+12.1
	semantic	47	0.191	42	0.143	+4.9
Qwen3-14B	identifier	103	0.583	123	0.431	+15.2
	semantic	33	0.424	45	0.244	+18.0
GPT-4o	identifier	240	0.087	300	0.023	+6.4
	semantic	61	0.049	40	0.025	+2.4
	structural	17	0.059	16	0.000	+5.9
Gemini 2.5 Flash	identifier	182	0.236	209	0.134	+10.2
	semantic	38	0.211	33	0.273	-6.2
Claude Sonnet 4	identifier	140	0.400	185	0.249	+15.1
	semantic	41	0.415	42	0.333	+8.1
GLM-4.7	identifier	92	0.652	147	0.327	+32.6
	semantic	38	0.474	37	0.378	+9.5
	structural	21	0.143	20	0.150	-0.7
GPT-5 nano	identifier	135	0.393	241	0.100	+29.3
	semantic	37	0.243	21	0.286	-4.2
Qwen3-14B, reas. 1k	identifier	48	0.688	60	0.450	+23.8
Qwen3-14B, no reas.	identifier	300	0.057	284	0.046	+1.1
	semantic	24	0.083	28	0.071	+1.2

H IDENTIFIER ADOPTION AND LEAKAGE UNDER MISLEADING

Table 16: Identifier adoption after `NameError`: next submissions that define the missing name / `NameErrors` followed by a submission.

Model	REAL	SCRAMBLED	MISLEADING	EMPTY
Llama-3.1-8B	3/16	0/4	0/14	0/2
Qwen2.5-7B	2/239	7/243	0/253	0/218
Ministral-3 8B	56/170	29/196	5/237	0/233
Qwen3-14B	81/95	74/108	9/203	3/232
GPT-4o	24/184	7/227	0/207	0/230
Gemini 2.5 Flash	59/151	41/162	3/216	0/149
Claude Sonnet 4	78/128	69/163	8/264	5/256
GLM-4.7	78/88	68/127	9/230	6/230
GPT-5 nano	78/117	37/186	3/222	0/227
Qwen3-14B, reas. 1k	44/44	37/50	1/26	0/54
Qwen3-14B, no reas.	20/229	20/215	0/258	0/258

Table 17: MISLEADING messages after a `NameError`, pooled over the seven content-sensitive models, split by whether the true missing name appears anywhere in the misleading message.

True name in message	Errors	Adoption	Recovery
visible	341	0.076	0.053
absent	1238	0.009	0.004

When a misleading message happens to contain the true name, adoption rises only to 7.6%, compared with 35–54% for SCRAMBLED and REAL when the name remains in place: a true name embedded in a coherent but incorrect diagnosis is rarely acted on.

I REPAIR DECOMPOSITION FOR ALL MODELS

Table 18: Re-attempt rate (the error is followed by another tool call) and identical- resubmission rate (the re-attempt resubmits byte-identical code), per tool error.

	Re-attempt				Identical resubmission			
Model	R	S	M	E	R	S	M	E
Llama-3.1-8B	0.47	0.14	0.41	0.04	0.23	0.40	0.64	0.00
Qwen2.5-7B	0.74	0.75	0.75	0.72	0.71	0.72	0.74	0.43
Ministral-3 8B	0.82	0.78	0.75	0.74	0.19	0.21	0.25	0.11
Qwen3-14B	0.88	0.85	0.74	0.75	0.16	0.24	0.41	0.24
GPT-4o	0.77	0.76	0.74	0.75	0.52	0.71	0.64	0.50
Gemini 2.5 Flash	0.82	0.78	0.74	0.66	0.49	0.52	0.64	0.53
Claude Sonnet 4	0.89	0.85	0.76	0.75	0.00	0.04	0.02	0.00
GLM-4.7	0.93	0.87	0.75	0.75	0.00	0.03	0.03	0.09
GPT-5 nano	0.85	0.79	0.73	0.75	0.04	0.13	0.17	0.31
Qwen3-14B, reas. 1k	0.80	0.74	0.36	0.54	0.02	0.14	0.35	0.25
Qwen3-14B, no reas.	0.77	0.75	0.74	0.75	0.68	0.72	0.84	0.57

J PER-TURN RECOVERY

Table 19: Recovery at the k-th tool error of a trajectory.

Model(s)	k	n_{real}	Rec.$_{\text{real}}$	n_{scr}	Rec.$_{\text{scr}}$	Δ (pp)
Frontier, pooled	1	468	0.453	468	0.286	+16.7
	2	255	0.196	331	0.115	+8.1
	3+	355	0.096	530	0.062	+3.4
GLM-4.7	1	91	0.659	92	0.467	+19.2
	2	31	0.387	49	0.224	+16.3
	3+	29	0.310	63	0.175	+13.6
GPT-5 nano	1	92	0.543	92	0.250	+29.3
	2	41	0.171	69	0.087	+8.4
	3+	53	0.132	116	0.034	+9.8
Ministral-3 8B	1	94	0.245	92	0.141	+10.3
	2	71	0.254	79	0.101	+15.2
	3+	95	0.063	139	0.014	+4.9

K HUMANEVAL

Table 20: HumanEval, same scaffold and conditions. Left: task success by condition. Right: first-attempt success (the first tool outcome is a pass), pooled over conditions, compared with MBPP-sanitized for the same models.

Model	REAL	SCR	MIS	EMPTY	HumanEval 1st	MBPP 1st
Gemini 2.5 Flash	0.99	0.96	0.95	0.97	0.94	0.05
GLM-4.7	0.97	0.97	0.98	0.98	0.96	0.07
GPT-5 nano	0.99	0.98	0.98	0.95	0.96	0.07
Qwen3-14B	0.98	0.97	0.93	0.98	0.93	0.07
Qwen2.5-7B	0.66	0.63	0.55	0.66	0.43	0.03
Llama-3.1-8B	0.20	0.16	0.18	0.14	0.15	0.00

For the four strongest models, the paired REAL–SCRAMBLED contrast has only 3, 2, 0, and 4 usable tasks, respectively, so we do not report it. For Qwen2.5-7B and Llama-3.1-8B, 58% and 82% of

HumanEval tool errors are `ValueError: 'code' argument must be a non-empty string`, produced by malformed tool calls. HumanEval provides the function signature in the prompt, whereas MBPP-sanitized does not, which explains why first attempts fail so often on MBPP-sanitized and why `NameError` dominates its error distribution.

L REPLICATION ON A LOCAL QUANTIZED SERVING STACK

Before moving to a single API gateway, we ran four small models through a local inference server with 4-bit-quantized weights, using the same 100 tasks, conditions, and scaffold. The REAL–SCRAMBLED contrast was -1.1 pp [-7.4, $+2.1$] for Mistral-7B-Instruct, $+1.4$ pp [-2.2, $+5.0$] for Llama-3.1-8B, $+4.8$ pp [$+0.4$, $+10.4$] for Qwen2.5-14B, and $+18.1$ pp [$+6.9$, $+30.6$] for Qwen3-14B with reasoning at a 1,024-token budget (on 36 engaged tasks). The qualitative pattern aligns with the main results: the 2024 models show little to no effect, whereas Qwen3-14B with reasoning shows a large effect. Absolute rates are not comparable across serving stacks: Qwen3-14B at 1,024 tokens engages on 42% of trials locally and 59% through the gateway.

M ANNOTATED TRAJECTORIES

Each excerpt shows the first submission, the message shown to the agent, and the subsequent submission; docstrings are omitted.

Real message, repaired (GPT-4o, `mbpp-406`).

```
submit:  def is_odd_parity(n: int) -> bool: return n % 2 != 0
error:   NameError: name 'find_Parity' is not defined
submit:  def find_Parity(n: int) -> bool: return n % 2 != 0
result:  All 3 tests passed.
```

Under SCRAMBLED, the unpinned name moves into the traceback header (`File "<most>", line 1, in <find_Parity> ... NameError: string 'last' is not name`); GPT-4o then resubmits `is_odd_parity` unchanged and receives the same error.

Scrambled frame, pinned name in place, adopted (GPT-5 nano, `mbpp-88`).

```
submit:  def frequency(lst): ...
error:   Traceback (string name most call):
           File "<recent>", line 1, in <module>
         NameError: defined 'freq_count' is not last
submit:  def freq_count(lst): ...
result:  All 3 tests passed.
```

Scrambled frame, pinned name in place, not adopted (GLM-4.7, `mbpp-477`).

```
submit:  def to_lower_case(s: str) -> str: return s.lower()
error:   NameError: string 'is_lower' is not call
submit:  def to_lowercase(s: str) -> str: return s.lower()
error:   NameError: string 'is_lower' is not last
```

With the REAL message, GLM-4.7 renames the function to `is_lower` and passes. Here, the name remains in its slot, but GLM-4.7 does not adopt it.

Name displaced (GLM-4.7, `mbpp-77`).

```
submit:  def is_divisible_by_11(n): return n % 11 == 0
error:   Traceback (call is_Diff most defined):
           File "<module>", line 1, in <recent>
         NameError: name 'last' is not string
submit:  def is_divisible_by_11(n): return n % 11 == 0
```

Reasoning suppressed (Qwen3-14B, `mbpp-479`, REAL).

```
submit:  def first_digit(number): return int(str(number)[0])
error:   NameError: name 'first_Digit' is not defined.
         Did you mean: 'first_digit'?
submit:  (identical)
error:   (identical)
submit:  (identical)
```

Malformed tool call (Llama-3.1-8B, `mbpp-113`). The first call passes an empty `code` argument and receives `ValueError: 'code' argument must be a non-empty string`; under SCRAMBLED the same message reads `ValueError: 'argument' code must be a non-empty string`, an instance of the slot displacement described in §3.1.

N PROMPTS

System prompt (all models, all conditions).

> You are a careful tool-using assistant. Use the provided tools to solve tasks. After reading a tool's response, decide whether to call another tool or to give a final answer. If a tool returns an error, read the error message and adjust your next call accordingly.

User turn.

> ## Task \n {task description} \n\n Use the provided tools to solve this task. Call exactly one tool per turn. After each tool result, decide whether to call another tool or provide a final answer.

Misleading-message generator.

> You are generating replacement error messages for a controlled benchmark that studies whether agents actually read error content. Given an ORIGINAL error, produce a MISLEADING replacement that satisfies ALL of the following:
>
> (a) Syntactically well-formed; looks like a real tool error or traceback. (b) Of comparable length to the original (within $\pm$50%). (c) Names a DIFFERENT exception class and a DIFFERENT root cause. (d) Plausibly arises in the same tool context. (e) Does NOT repeat any 5-word-or-longer run of tokens from the original. (f) Does NOT hint at being a replacement, benchmark, or test. Tool: {tool_name}. Tool arguments: {tool_args}. ORIGINAL error: {error_message}. Output ONLY the replacement error — no preamble, no explanation, no markdown. REPLACEMENT:

O ARTIFACT CONTENTS

The anonymized supplementary artifact contains all trajectories, configurations, prompts and analysis code used in this paper; the collection harness (scaffold, condition transformers and model adapters) will be released with the camera-ready version, and the SCRAMBLED transformer is fully specified in Figure 3 and Appendix B. The artifact comprises the 3,600 trials of the primary experiment, the 800 additional trials of the Qwen3-14B ablation, the 2,400 HumanEval trials, and the original trial files of the local replication in Appendix L, stored as line-delimited JSON with one trial per line. Each trial records the task, model and condition, every turn of the trajectory, both the original and the rendered message for every tool error, token counts, and the execution-harness verdict. Each data file has a configuration file recording the endpoint, sampling parameters, generation budget and condition settings used for it, and the artifact includes the MISLEADING generator prompt and the system prompt and user-turn template of the scaffold. A single script regenerates every table, figure and number quoted in the text from these files; the Wilcoxon p-values require `scipy`. For anonymization, run labels, log paths and the address of the local server that generated MISLEADING messages were removed from the configurations; no experimental setting was changed.