Fluent Reflection, No Correction: LLMs Fail to Enforce Constraints in Open-Ended Tasks

Anonymous ACL submission

Abstract

Large language models (LLMs) can generate fluent critiques of their own outputs, but it remains unclear when such behaviours function as corrective processes rather than additional sampling. Prior work on self-correction often uses closed-ended tasks, executable feedback, retrieval, or diagnostic error signals, which can make improvement possible without requiring the model to identify what went wrong. We evaluate eight frontier LLMs on an open-ended but auditable constraint-enforcement task: producing valid Cognitive Reflection Test (CRT) items and revising failed items after generic, non-diagnostic feedback. A valid item must induce an intuitive but incorrect answer, have a single reflection-based correct answer, and avoid substantive overlap with known CRT items. First-pass performance is low (<1 valid item out of 4 on average), and reflection improves pass rate consistently but modestly ($+0.216$ on average). Crucially, second-pass outputs often repeat the original failure category, including renewed novelty and near-copy violations. Gains are larger when models can adapt existing non-CRT trick questions than when they must generate items de novo, and reasoning-labelled models show no consistent advantage. These results show that pass-rate gains can overstate self-correction: in open-ended constrained generation, current LLM reflection often produces fluent revision without reliably eliminating the violated constraint.

1 Introduction

Large language models (LLMs) are increasingly used for planning, drafting, analysis, and other tasks where errors are not always caught by a single final-answer check. Step-by-step prompting can improve performance (Wei et al., 2022), and many systems now encourage models to "reflect" through self-critiques, iterative refinement, verification, or tool-augmented checking (Shinn et al., 2023; Renze and Guven, 2024; Dhuliawala et al., 2024). We use *reflection* as an umbrella term for these procedures, including self-feedback loops and program-level scaffolding in long-reasoning models (LRMs) (Madaan et al.; Tong et al., 2024; Chen et al., 2023; Gou et al., 2024; Chen et al., 2025a). The motivation is functional rather than anthropomorphic: reflection is useful only if it helps a model monitor, revise, and correct its own prior output, echoing the behavioural role of metareasoning without assuming human-like internal processes (Ackerman and Thompson, 2017).

Here we test whether such reflection mechanisms show functional evidence of *constraint-sensitive corrective revision*. The critical setting is not one in which the model is told the answer, given a unit test, or handed a diagnostic error label. Instead, the model is told only that a particular output failed the task requirements and is then shown the original constraints again. This generic feedback condition is deliberate: explicit error-category feedback would test whether models can use an external diagnostic signal, whereas our question is whether self-reflection helps the model infer and correct the violated constraint itself.

This distinction matters because final-answer improvement alone can be misleading. A second attempt can improve pass rate simply because the model samples another candidate from a large output space. That is not the same as diagnosing why the first candidate failed and eliminating that failure mode. We therefore evaluate not only whether reflection improves success, but whether the second pass remains path-dependent on the first-pass error.

1.1 Problem setting

Most positive evidence for LLM reflection comes from settings with relatively narrow solution spaces or strong external signals—for example unit tests, retrieval, executable environments, multiple-choice answers, or explicit correctness feedback (Renze

and Guven, 2024; Dhuliawala et al., 2024). Broader autonomy benchmarks are more realistic, but often combine open-ended tasks with substantial environmental scaffolding, executable feedback, or task-specific interfaces (Rein et al., 2025; Kwa et al., 2026). These settings are valuable, but they do not isolate whether an LLM can autonomously recognise which abstract constraint it violated.

Prior work is also mixed on intrinsic self-correction. Surveys and benchmarks suggest that self-correction is often weak or inconsistent without external feedback (Kamoi et al., 2024; Huang et al., 2024; Hosseini et al., 2024; Tie et al.). At the same time, scaffolded refinement can improve generated outputs in domain-specific settings, including medical multiple-choice question generation (Yao et al., 2025). A related concern is that reasoning text may be unstable or unfaithful to the process that produced the answer (Mirzadeh et al., 2024; Heo et al., 2025; Jiang et al., 2024; Kambhampati et al., 2025; Chen et al., 2025b). The open question is therefore not whether iterative systems can ever improve output quality, but whether generic reflection provides a reliable corrective process when success requires satisfying several explicit constraints in an open-ended generation task.

We address this with a task that is open-ended enough to require construction, but structured enough to audit: generating novel Cognitive Reflection Test (CRT) items (Thomson and Oppenheimer, 2016). A valid item must produce a tempting intuitive error, support a single reflection-based answer, remain accessible, and avoid substantive overlap with known CRT items. This makes the task diagnostic for constraint enforcement rather than for domain knowledge alone: frontier models can describe CRT items and explain intuitive lures, but the experiment tests whether they can combine those capacities while respecting novelty and clarity constraints during generation and revision.

1.2 Contributions

1. We introduce an auditable open-ended constraint-enforcement task for evaluating intrinsic LLM reflection under generic, non-diagnostic failure feedback.

2. We distinguish *pass-rate improvement* from *corrective revision* by testing whether second-pass outputs repeat the first-pass failure category.

3. We manipulate task structure through Generation versus Search–identify, varying solution-space width and anchor availability while preserving the same formal success criteria.

4. We evaluate eight recent frontier models, including reasoning-labelled models, and test whether reasoning-oriented systems show stronger corrective revision.

1.3 Hypotheses

H1 — Reflection effects. *H1a:* Reflection increases pass rate relative to the initial attempt. *H1b:* Reflection repeats the original failure category above a marginal-category permutation baseline.

H2 — Task moderation. *H2a:* Reflection gains are larger in *Search–identify* than in *Generation*. *H2b:* Error-category repetition is lower in *Search–identify*.

H3 — Model differences. Reasoning-labelled models achieve larger reflection gains than other models.

Reflection-strategy differences are analysed separately in the appendix.

2 Related Work

Existing work focuses on our question from three directions: self-correction, open-ended evaluation, and question generation. The central distinction is not simply open- versus closed-endedness, but whether the setting permits an auditable test of whether reflection removes the same violated constraint after non-diagnostic feedback. A fuller comparison is provided in Appendix D.

Self-correction. Reflection methods such as self-refine, critique-revise loops, chain-of-verification, and tool-augmented checking can improve outputs, especially when models receive an informative signal about what to check (**?**Dhuliawala et al., 2024; Gou et al., 2024). However, improvements under diagnostic feedback do not establish that the model can locate its own error. Benchmarks such as CorrectBench broaden evaluation across commonsense reasoning, mathematical reasoning, and code generation, and distinguish intrinsic, external, and fine-tuned correction (Tie et al.). Our study is narrower in task coverage but more specific in mechanism: we ask whether intrinsic reflection reduces the original violation, or whether second-pass success mainly reflects another draw from the model's generative distribution.

Open-ended evaluation. Open-ended tasks are difficult because outputs can be diverse while still needing principled evaluation. Work on LLM-as-judge and multi-criteria scoring addresses this measurement problem, including pairwise and criterion-based evaluation of open-ended responses (Lu et al., 2024). Such settings are valuable, but evaluation often remains a quality-ranking problem: criteria are partly interpretive, and the main outcome is agreement with human preference or rubric scores. Our task is deliberately more constrained: each item can be scored against explicit validity, set-exclusion, clarity, and complexity criteria, allowing failure categories to be tracked across attempts.

Question generation. LLMs have been used to generate educational and professional exam questions, including scaffolded self-refinement pipelines for medical multiple-choice questions (Yao et al., 2025). These systems are valuable applied pipelines, but they usually optimise output quality using domain grounding, expert-designed prompts, comparison feedback, or human/LLM preference judgements. We instead use item generation as a diagnostic task: the target class is familiar, the constraints are explicit, and the crucial test is whether generic self-reflection prevents the model from reusing prohibited or structurally invalid items.

3 Approach and Methodology

Rationale for CRT item generation

We evaluate reflection using CRT item generation because it combines open-ended production with auditable constraints (Thomson and Oppenheimer, 2016). The task is not a validation study of new psychometric items. It is a diagnostic setting for constraint enforcement: a valid item must (i) elicit a plausible intuitive but incorrect answer, (ii) have a single correct answer reachable through reflection, (iii) remain simple enough to resemble a CRT item, and (iv) avoid substantive overlap with known CRT items. Many valid outputs are possible, but invalid outputs can be categorised. This makes it possible to ask whether reflection removes the violated constraint rather than merely producing a different fluent candidate.

The novelty constraint is central rather than incidental. Models often know canonical CRT items and can explain why they work; the experiment asks whether this knowledge is recruited while generating or revising an item under an explicit do-not-copy requirement. In some prompt subconditions, models were also shown CRT examples as material not to copy, but the formal task was unchanged across subconditions.

Algorithm 1 Generic-feedback reflection protocol for one failed item

```
Model generates four candidate CRT items from the task prompt.
Evaluators score each item against validity, novelty, clarity, and complexity.
for each failed item do
    Show the same model the failed item, original task constraints, and a generic failure verdict.
    Do not show the evaluator's failure category or diagnostic explanation.
    Model writes self-generated advice: keywords, explanation, and step-by-step instructions.
    Model produces replacement items using each advice type; Retry receives only the original constraints and generic failure verdict.
end for
```

Models

We evaluated eight models: OpenAI (GPT-4.1, o3, o4-mini), Google (Gemini 2.5 Pro-Preview), Anthropic (Claude 3.7 Extended), Meta (Llama-3.3-70B, Llama-4 Maverick), and DeepSeek (Reasoner). Where available, reasoning modes were enabled. Each model completed eight sessions; within each session, failed items were re-attempted under four framings: Explanation, Keywords, Instructions, and Retry. Models were queried through their standard public APIs using ordinary generative settings rather than deterministic decoding.

Task definition

Each session asked a model to produce four candidate CRT items. Two task framings were tested:

- **Generation** – create four items *de novo*.
- **Search–identify** – retrieve or adapt four suitable non-CRT trick questions from public sources, while excluding existing CRT items.

This manipulation varies solution-space width and anchor availability. Generation requires construction from a wider solution space; Search–identify permits existing non-CRT trick questions as anchors. Prompt ablations varied exemplars, extra instructional detail, and practical constraints, but did not change the formal validity criteria.

The generic failure verdict is a design feature. If the model were told that an item failed because it copied a CRT item, lacked an intuitive lure, or exceeded the complexity cap, the experiment would

test use of external diagnostic feedback. Our protocol instead tests whether reflection helps the model infer and correct the violated constraint from the failed item and the task requirements alone.

System persona. Following Renze and Guven (2024), each session used a constant system prompt establishing an expert persona: "You are a cognitive science expert and psychometrician with experience designing and validating Cognitive Reflection Test (CRT) items."

Automated evaluation

Each candidate item was scored independently by three LLM evaluators (GPT-4.1, o4-mini, Gemini 2.5 Pro-Preview), with one API call per item per evaluator. We used a conservative *fail-fast* aggregation rule: if any evaluator marked a required criterion as failing, the item failed overall.

- **Validity (1/0):** the item must contain both an intuitive-incorrect response and a single reflection-based correct answer.
- **Novelty (1/0.5/0):** evaluators compare against an embedded list of known CRT items and screen for reuse. In *Generation*, exact/near-exact CRT items, superficial CRT variations, and obvious reuse of widely shared non-CRT trick questions fail. In *Search–identify*, pre-existing non-CRT trick questions are admissible, but CRT overlap still fails. For confirmatory pass/fail, only 1.0 counts as pass.
- **Clarity (1/0):** the item must be linguistically coherent and have an unambiguous solution path.
- **Complexity cap (screen):** arithmetic (0–2), algebraic (0–2), spatial (0–1), using a coarse rubric. To preserve CRT-like accessibility, items with total complexity ≥ 3 fail.

Pass logic. An item is valid only if Validity = 1, Clarity = 1, Novelty = 1.0 for the relevant condition, and the complexity cap is satisfied. Any single evaluator fail on a required criterion renders the item invalid.

Prompts and code. Full task prompts, reflection prompts, evaluator rubrics, JSON schemas, and analysis code are provided in the supplementary materials and public repository.

Novelty screening. Novelty was operationalised by condition. In *Generation*, candidates had to be newly produced items, so evaluators checked each item against the embedded CRT database and also flagged obvious reuse of widely circulated non-CRT riddles. In *Search–identify*, public non-CRT riddles were allowed, but exact or near-exact CRT overlap and superficial CRT variation still failed. The non-CRT screen is therefore a conservative reuse check for salient public exemplars, not a claim to exhaustively index all possible trick questions; we audit this boundary judgement against human coding below.

Human validation

To audit evaluator fidelity, a stratified $20\,\%$ sample (strata: model $\times$ condition $\times$ reflection) was double-coded by two graduate students. Pre-adjudication inter-rater agreement was $\kappa = 0.55$. All disagreements were reconciled through discussion, yielding a single final human label set with full agreement between the human coders. Agreement between this adjudicated human label set and the collective LLM labels was $\kappa = 0.54$. Supplementary analyses report where disagreements occurred and how conclusions change under the audited labels.

Metrics

Pass rate was computed at the session level as the proportion of valid items out of four. Initial pass rate was the number of valid baseline items divided by four. For each reflection framing, post-reflection pass rate was the number of items already correct at baseline plus the number of newly recovered valid items among those that initially failed, divided by four. This treats reflection as part of a single continuous session-level attempt to complete the four-item task.

4 Experiments and Results

H1a — Overall reflection effect. For each session we paired the initial pass rate with the mean pass rate across all four reflection strategies. We fit a linear mixed-effects model with generation round (initial vs. reflection) as a fixed effect and random intercepts and slopes by session to estimate the average improvement attributable to reflection while accounting for within-session dependence. As a complementary check, we conducted a paired t-test on per-session pass rates and report the corresponding paired effect size (d_z).

H1b — Does reflection repeat the original failure? We tested whether reflection repeats the *same* failure category that appeared at the initial attempt. Analytically, we pooled across tasks and models and estimated the probability of repeating the original category with a session-clustered logistic model including strategy as a fixed effect. Separately, we compared the observed repeat rate with a *marginal-category permutation baseline*: the expected repeat rate if post-reflection failure categories were random draws from the observed category mix within each task×strategy cell. This is not a model of human guessing; it controls for the fact that some failure categories are much more common than others.

H2a — Task moderation of reflection gains. To test whether performance gains depend on task structure, we fit mixed-effects models with generation round, task condition (Generation vs. Search–identify), and prompt subcondition (base, complexity, examples, practical) as fixed effects, and a random intercept by session. Model selection (log-likelihood) retained generation round and the required two-way interactions.

H2b — Task effects on error persistence. We probed whether error persistence differs by task. We modelled the probability that a reflection attempt repeats the session's initial failure category as a function of task (Generation vs. Search–identify) with session-clustered logistic models controlling for strategy. We also tested plagiarism recidivism specifically, conditional on initial plagiarism.

H3 — Reasoning-model superiority. For the reasoning-model contrast we fit an OLS model of reflection gain with fixed effects for task and subcondition and a binary indicator for model type (1 = reasoning-labelled, 0 = other), clustering standard errors by model identity. The confirmatory one-sided test asked whether reasoning-labelled models achieved larger gains than other models, with CR1 95% CIs and wild-cluster bootstrap 90% CIs/p. After this test, we ran an exploratory equivalence check (TOST, $\Delta = \pm 0.05$ pass-rate units) as a sensitivity analysis.

Exploratory — Strategy differences. To compare reflection framings, we modelled strategy as a fixed effect with a random intercept for session, using Δ pass rate (recovered valid items out of four) as the outcome. Strategy effects were estimated relative to Explanation, with Holm-adjusted pairwise contrasts and within-session paired effect sizes. *Unless otherwise stated, the inferential unit is the session (4 items), confirmatory estimates are reported with 95% confidence intervals, and exploratory analyses are labelled accordingly.*

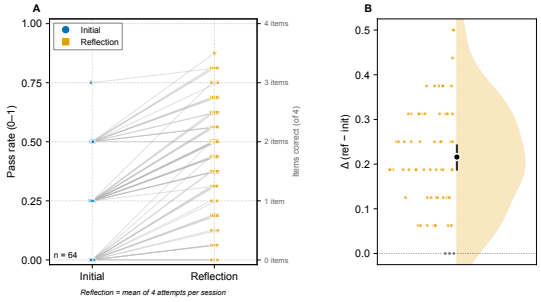


Figure 1: **H1 — Reflection improves performance. Panel A:** distributions of session pass-rates before and after reflection ($n = 64$ sessions). **Panel B:** violin plot of per-session Δ values; the central line marks the median and dots show session-level variability. Reflection pass-rates are computed as the mean across the four strategies, including 'retry'.

H1a — Reflection improves performance. Using the session as the unit of analysis, we paired each initial pass-rate with the mean post-reflection pass-rate across the four reflection strategies and estimated the contrast with a linear mixed-effects model (fixed effect: generation round; random intercepts and slopes by session). Reflection increased pass-rates across all models (Fig. 1), with an estimated average improvement of $\beta = +0.216$, 95% CI [0.199, 0.232], $p < 0.001$. As a corroborating paired analysis on per-session pass-rates, a paired t-test yielded the same conclusion, $t(63) = 14.40$, $p < 0.001$, with a large paired effect size ($d_z = 1.80$).

Initial pass rate was 0.230 overall (Generation: 0.172; Search–identify: 0.289), rising to 0.441 after reflection (Generation: 0.281; Search–identify: 0.602). The corresponding descriptive table is reported in the appendix.

H1b — Reflection repeats the original failure above a marginal-category baseline. To test whether second-pass failures were targeted corrections or re-instantiations of the original error, we pooled reflection attempts across tasks and models, estimated repeat probability with a session-clustered logistic model including strategy as a fixed effect, and benchmarked the observed

repeat rate against a stratified permutation null that preserved the empirical category mix within each task×strategy cell. Across 567 reflection attempts, the repeat-category rate was 85.36%. Relative to the within-cell benchmark, repetition was higher: observed 85.36% vs. benchmark 74.69% (95% permutation interval 72.13–77.25), an excess of +10.68 pp; permutation $p = 0.0001$ (Methods/Supplementary). Strategy terms were not reliably different in the pooled logistic model (all $p \geq .30$).

H2a — Task moderates reflection gains. To test whether gains depended on task structure, we fit a mixed-effects interaction model with generation round, task condition (generation vs. search–identify), and prompt subcondition (base, complexity, examples, practical) as fixed effects and a random intercept by session. Gains were larger for search–identify than generation (Fig. 4a; $\beta = +0.096$, 95% CI [0.069, 0.122], $p < 0.001$). Among prompt subconditions, only 'examples' provided an incremental benefit over the base prompt ($\beta = +0.043$, 95% CI [0.005, 0.081], $p = 0.025$).

H2b — Task reduces error persistence. We next tested whether task structure altered the probability of repeating the original error, using session-clustered logistic models for (i) any-category repeat and (ii) plagiarism recidivism conditional on initial plagiarism, in both cases controlling for strategy. Error persistence was lower in search–identify than in generation (Fig. 4b). Repeating the original category had lower odds in search–identify vs. generation (OR $= 0.467$, 95% CI [0.249, 0.877], $p = .018$). Plagiarism recidivism was likewise lower in search–identify (OR $= 0.501$, 95% CI [0.342, 0.736], $p < .001$).

H3 — Reasoning-model contrast. To test the confirmatory reasoning-model hypothesis, we regressed reflection gain on model type, task, and prompt subcondition, clustering standard errors by model identity; because the number of model clusters is small, we also report wild-cluster bootstrap inference. Reasoning-labelled models showed no superiority. Mean reflection gain was 0.036 (SD 0.237, $n = 40$) versus 0.111 (SD 0.171, $n = 24$) for other models; difference $= -0.075$ pass-rate units (≈ -0.30 items out of 4). The model-type coefficient was $\beta = -0.075$, CR1 95% CI $[-0.113, -0.037]$, one-sided test $p = 0.9999$. Wild-cluster bootstrap yielded a 90% CI $[-0.104, -0.046]$ and one-sided bootstrap $p = 1.0$. The exploratory TOST against ± 0.05 placed the bootstrap CI below the lower bound, suggesting a small disadvantage rather than equivalence.

Exploratory — Strategy differences. To compare reflection framings, we fit a mixed-effects model on reflection-only data with strategy as a fixed effect and session as a random intercept, using Δ pass-rate (recovered valid items out of four) as the outcome; pairwise contrasts were Holm-adjusted. All four framings improved performance (Appendix Fig. 9). 'Explanation' produced the largest mean gain and exceeded the other framings after multiplicity correction. However, *Retry*—which adds no self-generated reflective advice beyond the generic failure verdict and the original constraints—was statistically indistinguishable from *Instructions* and *Keywords*, and within-model contrasts favouring 'Explanation' were non-significant in 6/8 models (Supplementary). In practical terms, all framings recovered well under one additional item on average, and the advantage of 'Explanation' was modest rather than transformative.

5 Discussion

Across models, reflection improved pass rates, but only modestly and unevenly. More importantly, even after being told that a specific item had failed to meet the task requirements, second-pass outputs frequently repeated the original failure category (85.36%; Fig. 2). This is the central distinction: reflection improved the probability of eventually obtaining a valid item, but it did not behave like a dependable mechanism for identifying and eliminating the violated constraint.

The generic feedback condition is essential to this interpretation. The model was not told that an item copied a CRT exemplar, lacked an intuitive lure, exceeded the complexity cap, or had an unclear answer. It was told only that the item failed, then shown the original constraints again. This tests the common "take another pass and check your work" use case. Under that regime, current reflection looked weakly corrective rather than reliably diagnostic.

The contrast between Generation and Search–identify helps locate where the difficulty lies. Search–identify permits existing non-CRT trick questions as anchors, whereas Generation requires de novo construction under interacting constraints.

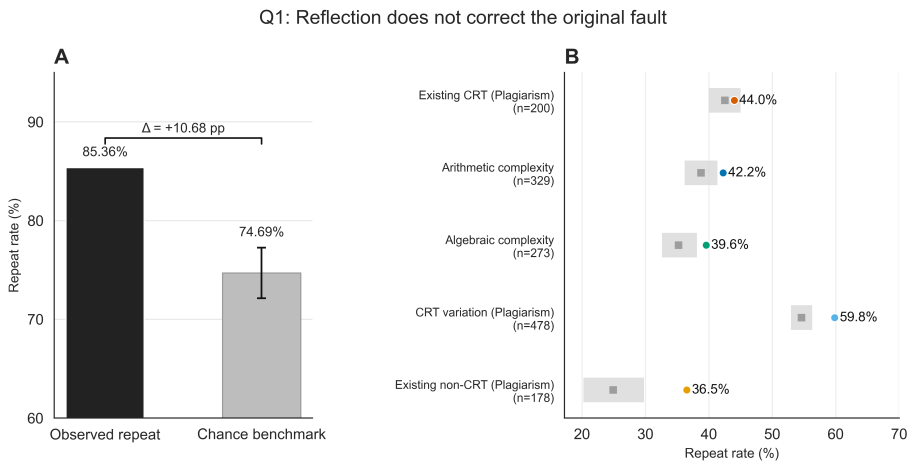


Figure 2: **H1b — Reflection repeats the original failure above a marginal-category baseline. Panel A:** observed repeat-failure rate at reflection versus a stratified permutation benchmark. **Panel B:** five most frequent failure categories. Grey boxes show the 95% permutation envelopes; coloured dots are observed repeat rates; n per row is the number of reflection attempts at risk (category present initially). Dots to the right of the envelope indicate repetition above the marginal-category baseline.

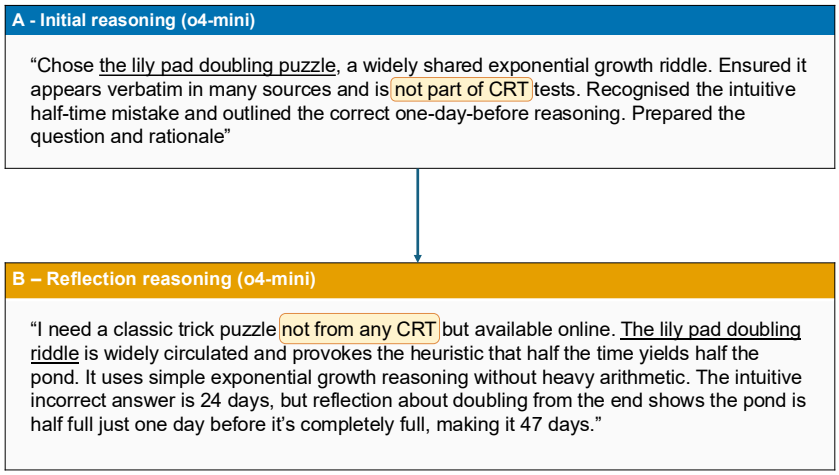
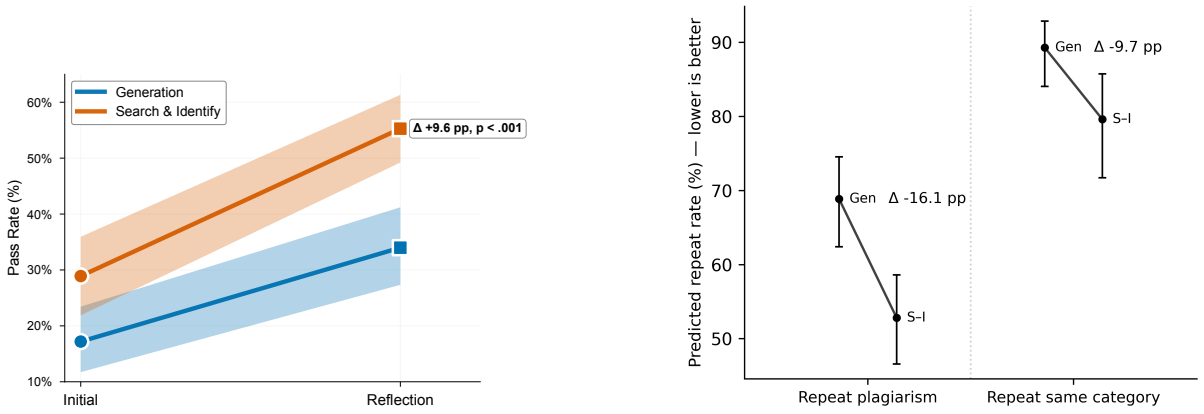


Figure 3: **Vignette of o4-mini reasoning.** o4-mini explicitly reasons about selecting a CRT item that the task prohibits (underlined text), while fluently invoking the constraint not to copy existing CRT items (highlighted text).

Gains were larger in Search–identify, suggesting that constraint enforcement becomes less reliable when the task requires broader unsupported generation rather than adaptation from an available anchor.

The reflection gains themselves are also not well described as systematic repair. At the session level, improvement covaries with how many items enter reflection: when more failed items are retried, the chance of recovering at least one valid item rises and the variance in gains widens. Together with the high repeat-failure rate, this is more consistent with constrained re-sampling than with stable targeted correction. The strategy results point in the same direction: although *Explanation* was best on average, simple *Retry* was statistically indistinguishable from *Instructions* and *Keywords* after multiplicity control.

The most informative pattern is error persistence. Across all models, reflection repeated the session's original failure category above the marginal-category permutation baseline (Fig. 2). This baseline already accounts for the fact that some failure categories are common. The excess repetition indicates path dependence: once an error pattern is established, the next attempt often remains in the same failure region rather than moving away from the violated constraint.



(a) Reflection benefit by task (H2a)

(b) Error persistence by task (H2b)

Figure 4: **H2a/H2b — Task structure moderates reflection. (A)** Search–identify yields larger reflection gains than generation. **(B)** Odds of repeating the original category are lower in search–identify; plagiarism recidivism is also lower.

The vignette illustrates the aggregate pattern. In the Search–identify condition, a model explicitly states that the lily-pad item is "not part of CRT tests" and then reproduces it; on re-attempt, it justifies the same choice and reproduces it again. The problem is not absence of the relevant facts, but unreliable application of those facts when several interacting requirements must be satisfied at once.

Relative to prior self-correction benchmarks, our results identify a narrower but important failure mode. In closed-ended or scaffolded settings, models may start from reasonable performance and improve slightly after correction (Renze and Guven, 2024; Tie et al.). Here, both initial constraint satisfaction and subsequent correction were weak. The result is therefore not merely that reflection gives small gains; it is that, in this open-ended constraint-enforcement setting, reflection often behaved like another statistical attempt rather than detection of an error type followed by targeted repair.

Reasoning-labelled models also did not show a reliable advantage. Longer traces and reasoning-oriented branding should therefore not be taken as evidence that these systems will enforce constraints more robustly in open-ended revision. Evaluation should measure not only whether a second pass improves final success, but whether it reduces the specific failure mode that made the first pass invalid.

6 Conclusion

Current LLM reflection can improve open-ended generation, but improvement is not the same as correction. In a compact task where a candidate item must satisfy CRT-style structure while avoiding exact copies and superficial variants of known CRT items, eight frontier models started from low first-pass performance, improved only modestly after generic non-diagnostic reflection, and often repeated the original violation. This combination is the key result: unlike settings where models start from reasonable known-answer performance and gain slightly from correction, here both initial constraint satisfaction and subsequent correction were weak. Fluent self-critique and second-pass gains therefore do not show that the model detected the failed constraint and applied the relevant corrective rule. Future evaluations should track whether revision removes the specific error that made the first output invalid, especially when there is no answer key, executable test, or diagnostic error label.

7 Limitations

Limitations and We study one task family, so the results should not be read as a universal claim about all open-ended generation. The value of the task is diagnostic rather than exhaustive: it isolates a compact setting where success requires an intuitive-error structure, a single reflective answer, simplicity, and exclusion from known CRT items. Evaluation also relied on LLM-based scoring with moderate human agreement, especially for category-level judgements. We therefore treat exact category proportions cautiously and provide human-audit analyses in the supplement. These limitations qualify the breadth and precision of the estimates, but they do not remove the main pattern: modest improvement co-occurs with substantial

same-error persistence under generic feedback.

References

Rakefet Ackerman and Valerie A. Thompson. 2017. Meta-Reasoning: Monitoring and Control of Thinking and Reasoning. *Trends in Cognitive Sciences*, 21(8):607–617.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025a. Towards Reasoning Era: A Survey of Long Chain-of-Thought for Reasoning Large Language Models. *Preprint*, arXiv:2503.09567.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023. Teaching Large Language Models to Self-Debug. *Preprint*, arXiv:2304.05128.

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. 2025b. Reasoning Models Don't Always Say What They Think. *Preprint*, arXiv:2505.05410.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2024. Chain-of-Verification Reduces Hallucination in Large Language Models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 3563–3578, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. *Preprint*, arXiv:2305.11738.

Juyeon Heo, Christina Heinze-Deml, Oussama Elachqar, Kwan Ho Ryan Chan, Shirley Ren, Udhay Nallasamy, Andy Miller, and Jaya Narain. 2025. Do LLMs "know" internally when they follow instructions? *Preprint*, arXiv:2410.14516.

Arian Hosseini, Alessandro Sordoni, Daniel Toyama, Aaron Courville, and Rishabh Agarwal. 2024. Not All LLM Reasoners Are Created Equal. *Preprint*, arXiv:2410.01748.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large Language Models Cannot Self-Correct Reasoning Yet. *Preprint*, arXiv:2310.01798.

Bowen Jiang, Yangxinyu Xie, Zhuoqun Hao, Xiaomeng Wang, Tanwi Mallick, Weijie J. Su, Camillo J. Taylor, and Dan Roth. 2024. A Peek into Token Bias: Large Language Models Are Not Yet Genuine Reasoners. *Preprint*, arXiv:2406.11050.

Subbarao Kambhampati, Karthik Valmeekam, Siddhant Bhambri, Vardhan Palod, Lucas Saldyt, Kaya Stechly, Soumya Rani Samineni, Durgesh Kalwar, and Upasana Biswas. 2025. Position: Stop Anthropomorphizing Intermediate Tokens as Reasoning/Thinking Traces! *arXiv preprint*.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. 2024. When Can LLMs Actually Correct Their Own Mistakes? A Critical Survey of Self-Correction of LLMs. *Preprint*, arXiv:2406.01297.

Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, Ryan Bloom, Thomas Broadley, Haoxing Du, Brian Goodrich, Nikola Jurkovic, Luke Harold Miles, Seraphina Nix, Tao Lin, Neev Parikh, and 6 others. 2026. Measuring AI Ability to Complete Long Software Tasks. *Preprint*, arXiv:2503.14499.

Xiaotian Lu, Jiyi Li, Koh Takeuchi, and Hisashi Kashima. 2024. AHP-Powered LLM Reasoning for Multi-Criteria Evaluation of Open-Ended Responses. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1847–1856, Miami, Florida, USA. Association for Computational Linguistics.

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.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models. *Preprint*, arXiv:2410.05229.

David Rein, Joel Becker, Amy Deng, Seraphina Nix, Chris Canal, Daniel O'Connel, Pip Arnott, Ryan Bloom, Thomas Broadley, Katharyn Garcia, Brian Goodrich, Max Hasin, Sami Jawhar, Megan Kinniment, Thomas Kwa, Aron Lajko, Nate Rush, Lucas Jun Koba Sato, Sydney Von Arx, and 3 others. 2025. HCAST: Human-Calibrated Autonomy Software Tasks. *Preprint*, arXiv:2503.17354.

Matthew Renze and Erhan Guven. 2024. Self-Reflection in LLM Agents: Effects on Problem-Solving Performance. *Preprint*, arXiv:2405.06682.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. *Preprint*, arXiv:2303.11366.

Keela S. Thomson and Daniel M. Oppenheimer. 2016. Investigating an alternate form of the cognitive reflection test. *Judgment and Decision Making*, 11(1):99–113.

Guiyao Tie, Zenghui Yuan, Zeli Zhao, Chaoran Hu, Tianhe Gu, Ruihang Zhang, Sizhe Zhang, Junran Wu, Xiaoyue Tu, Ming Jin, Qingsong Wen, Lixing Chen, Pan Zhou, and Lichao Sun. Can LLMs Correct Themselves? A Benchmark of Self-Correction in LLMs.

Yongqi Tong, Dawei Li, Sizhe Wang, Yujia Wang, Fei Teng, and Jingbo Shang. 2024. Can LLMs Learn from Previous Mistakes? Investigating LLMs' Errors to Boost for Reasoning. *Preprint*, arXiv:2403.20046.

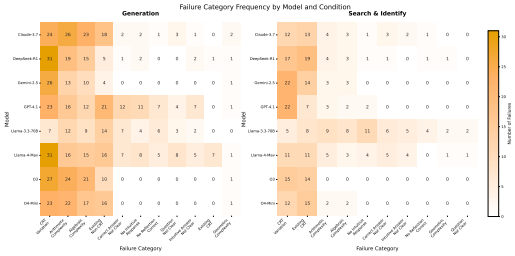
Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. https://arxiv.org/abs/2201.11903v6.

Zonghai Yao, Aditya Parashar, Huixue Zhou, Won Seok Jang, Feiyun Ouyang, Zhichao Yang, and Hong Yu. 2025. MCQG-SRefine: Multiple Choice Question Generation and Evaluation with Iterative Self-Critique, Correction, and Comparison Feedback. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10728–10777, Albuquerque, New Mexico. Association for Computational Linguistics.

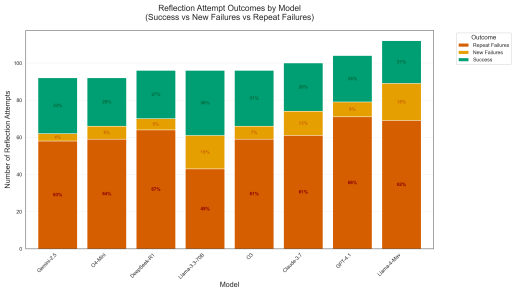
Model	Initial M	Initial SD	Post-reflection M	Post-reflection SD
claude-3-7-extended	0.219	0.209	0.422	0.221
deepseek-reasoner-nvidia	0.250	0.232	0.453	0.163
gemini-2-5-pro-preview	0.281	0.088	0.516	0.087
gpt-4.1	0.188	0.222	0.383	0.270
llama-3-3-70b	0.250	0.232	0.523	0.198
llama4-maverick	0.125	0.189	0.305	0.321
o3	0.250	0.189	0.484	0.205
o4-mini	0.281	0.248	0.484	0.236

Extended Data Table 1: Per-model pass-rates (M, SD) at initial generation and after reflection. Values are per-session averages; for reflection, pass-rates are averaged within session across strategies to avoid variance inflation. Omnibus tests found no between-model differences at either round.

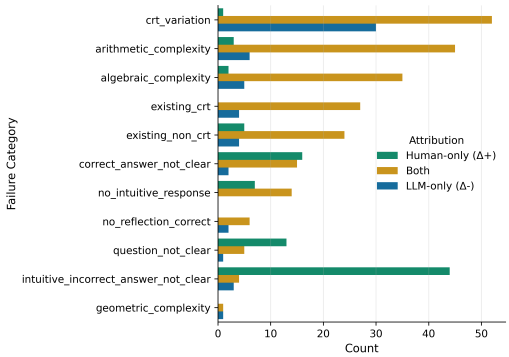
A Extended Data



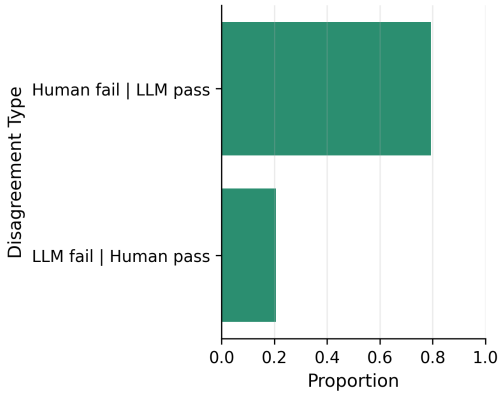
Extended Data Fig. 1: Failure-category frequencies by model and condition. Rows and columns are ordered by overall frequency.



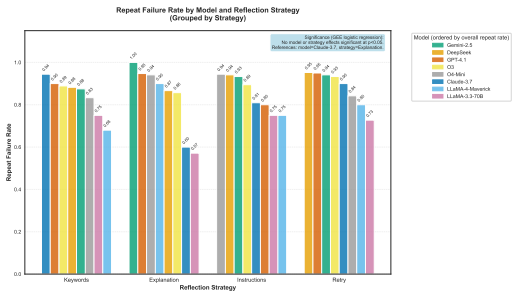
Extended Data Fig. 2: Reflection outcomes, grouped as repeat of the original failure category, introduction of a new failure category, or correction to a valid item.



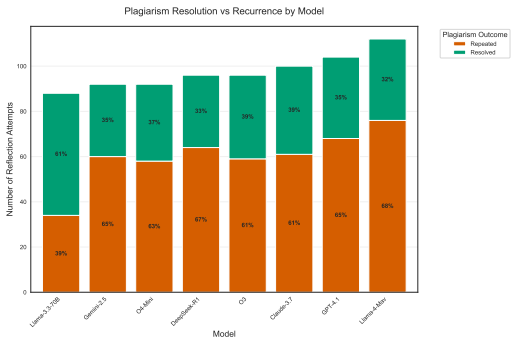
Extended Data Fig. 3: Human–LLM evaluator agreement on category assignments in the stratified audit sample. Agreement is moderate; see Section B.1 for κ values and audit details.



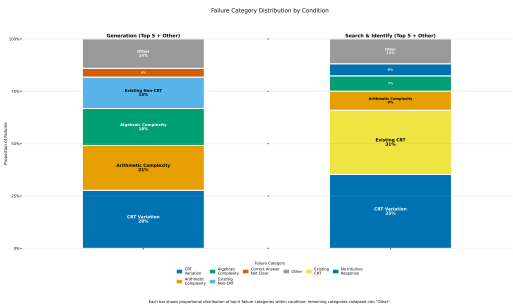
Extended Data Fig. 4: Breakdown of human–LLM disagreements in the stratified audit sample.



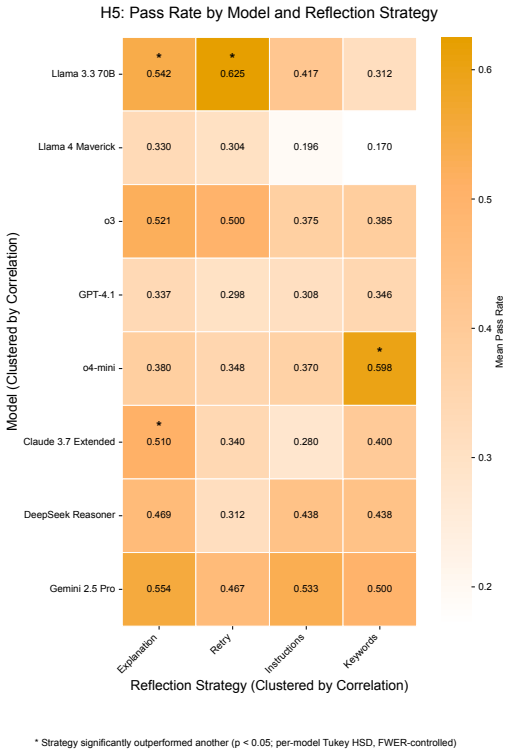
Extended Data Fig. 5: Repeat-failure rate at reflection, grouped by strategy with models colour-coded. No single strategy eliminates repeats across models.



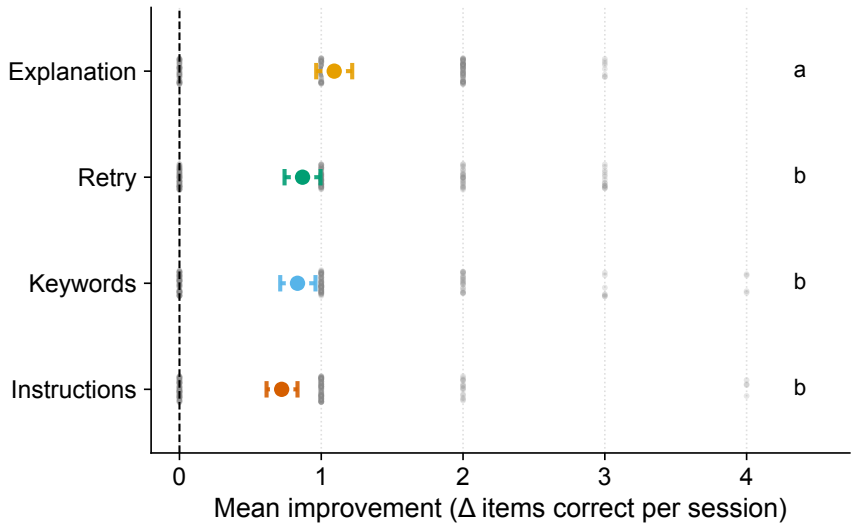
Extended Data Fig. 6: Plagiarism recidivism: share of reflection attempts that plagiarise again, conditional on initial plagiarism. Models are ordered by repeat proportion.



Extended Data Fig. 7: Top five failure categories by condition. Generation is dominated by *CRT-variation* and arithmetic/algebraic complexity, whereas search–identify shifts mass toward *existing CRT* and *CRT-variation*.



Extended Data Fig. 8: Pass-rates by reflection strategy within model (clustered). Explanation leads on average, but gains are model-idiosyncratic rather than evidence of a globally dominant strategy.



Extended Data Fig. 9: **Reflection-strategy effects.** Mean improvement in valid items per session (Δ out of 4) with 95% bootstrapped confidence intervals. Rug marks show per-session values. Compact letter display indicates Holm-adjusted groupings; strategies sharing a letter are not significantly different at $\alpha = .05$.

B Supplementary Methods and Analyses

B.1 Task, scoring, and audit details

Session accounting. Each model completed eight sessions. Within a session, items that failed the baseline attempt were re-attempted under four framings: Explanation, Keywords, Instructions, and Retry. Thus, each model contributed $8 \times 4 = 32$ post-initial re-answer attempts.

Sampling settings. Models were queried through their standard public APIs using ordinary generative settings rather than deterministic decoding. This preserves the open-ended character of the task and avoids constraining models into an artificially narrow decoding regime.

System persona. Following Renze & Guven (Renze and Guven, 2024), each session used a constant expert persona prompt: 'You are a cognitive science expert and psychometrician with experience designing and validating Cognitive Reflection Test (CRT) items.' The intent was methodological rather than substantive: the task presupposes knowledge of canonical CRT items and exclusion criteria, so we retained a domain-relevant cue rather than testing persona effects.

Novelty screening rationale. We used an LLM-based screen to flag both CRT overlap and widely shared non-CRT trick questions, rather than attempting to curate a closed-book list that would inevitably be incomplete. This was treated as a pragmatic first-pass filter for high-frequency reused riddles. Human audit was then used to assess the fidelity of this screening procedure.

Scoring note. The tri-level Novelty score and the underlying 0–5 complexity rubric were retained for descriptive checks, but all confirmatory analyses used the binary decision rule reported in the main text. Thus, only Novelty $= 1.0$ counted as pass, and any item with total complexity ≥ 3 failed the complexity screen.

Human audit and evaluator agreement. The human audit was designed to assess evaluator fidelity rather than replace the full automated pipeline. A stratified 20% sample (strata: model $\times$ condition $\times$ reflection) was double-coded by two graduate students. Pre-adjudication agreement was $\kappa = 0.55$. All disagreements were reconciled through discussion, producing one final human label set with full agreement between the human coders. Agreement between this reconciled human label set and the collective LLM labels was $\kappa = 0.54$. Agreement summaries are shown in Extended Data Fig. 3 and Extended Data Fig. 4.

Pass-rate formalisation. Let $\mathrm{Correct}_{\mathrm{initial}}$ denote the number of valid baseline items in a session, $\mathrm{Correct}_{\mathrm{reflection}}$ the number of newly recovered valid items under a given reflection framing, and $\mathrm{Total}_{\mathrm{initial}} = 4$ the number of items requested per session. Then

$$\mathrm{PassRate}_{\mathrm{initial}} = \frac{\mathrm{Correct}_{\mathrm{initial}}}{\mathrm{Total}_{\mathrm{initial}}}$$

and

$$\mathrm{PassRate}_{\mathrm{reflection}} = \frac{\mathrm{Correct}_{\mathrm{initial}} + \mathrm{Correct}_{\mathrm{reflection}}}{\mathrm{Total}_{\mathrm{initial}}}.$$

This treats reflection as part of one continuous session-level attempt: items already correct at baseline remain counted as correct, and reflection contributes only newly recovered valid items among those that initially failed.

B.2 General analysis conventions

The primary inferential unit was the *session* (4 items/session). Initial pass-rate was defined as the proportion of the four baseline items judged valid. Post-reflection pass-rate was defined as the proportion valid after the reflection phase, counting items already correct at baseline plus any failed items subsequently recovered. Unless otherwise noted, confirmatory estimates are reported with 95% confidence intervals. Confirmatory and exploratory analyses are distinguished explicitly.

B.3 Supplementary analyses

SA1. Robustness checks for H1 (mixed-effects + paired test)

Methods. For each session, we paired the initial pass-rate with the mean post-reflection pass-rate across the four reflection strategies. We estimated the overall effect of reflection with a linear mixed-effects model in which generation round (initial vs. reflection) was entered as a fixed effect and session contributed both random intercepts and random slopes, thereby accounting for repeated within-session observations and allowing heterogeneity in reflection gains across sessions. As a complementary paired analysis, we also conducted a paired t-test on per-session pass-rates and report the corresponding paired effect size (d_z).

Results. Reflection improved pass-rate under both specifications. The mixed-effects estimate was $\beta_{\text{reflection}} = 0.216$, 95% CI $[0.199, 0.232]$, $p < 0.001$. The paired analysis gave the same conclusion, $t(63) = 14.40$, $p < 0.001$, with $d_z = 1.80$. Per-model descriptives are shown in Extended Data Table 1.

SA2. H1b permutation benchmark for 'repeat the original failure'

Methods. We tested whether reflection failures disproportionately reproduced the *same* failure category observed at baseline. The primary model pooled reflection attempts across tasks and models and estimated repeat probability using a session-clustered logistic regression with reflection strategy as a fixed effect. To determine whether the observed repeat rate exceeded what would be expected from the empirical mix of failure categories, we also constructed a stratified permutation benchmark. Within each task $\times$ strategy cell, reflection-round category labels were permuted across attempts (10,000 draws), preserving the empirical category mix while breaking any link to the original failure.

Results. Across 567 reflection attempts, the repeat-category rate was 85.36%. Relative to the stratified benchmark, repetition was higher: observed 85.36% vs. benchmark 74.69% (95% permutation interval 72.13–77.25), an excess of $+10.68$ percentage points; permutation $p = 0.0001$. Strategy terms were not reliably different in the pooled logistic model (all $p \geq .30$). Related distributions are shown in Extended Data Fig. 1, Extended Data Fig. 2, and Extended Data Fig. 5.

SA3. Human–LLM evaluator agreement

Methods. On the stratified audit sample, we computed Cohen's κ for the human double-code and then computed agreement between the reconciled human label set and the collective LLM label set. For descriptive interpretation, disagreements were also grouped into broad boundary types.

Results. Agreement was moderate. Human pre-adjudication agreement was $\kappa = 0.55$, and agreement between the reconciled human labels and the collective LLM labels was $\kappa = 0.54$. Most disagreements occurred near category boundaries rather than between clearly dissimilar labels. Agreement summaries are shown in Extended Data Fig. 3–4.

SA4. Strategy effects at reflection (exploratory)

Methods. To compare reflection framings, we analysed reflection-only data with a mixed-effects model in which strategy was entered as a fixed effect and session as a random intercept. The outcome was Δ pass-rate, operationalised as recovered valid items out of four. Strategy effects were estimated relative to Explanation, pairwise contrasts were Holm-adjusted, and within-model contrasts used Tukey HSD.

Results. Explanation produced the largest mean gain on average, but its advantage was modest rather than transformative. Retry was statistically indistinguishable from Instructions and Keywords after multiplicity control, and within-model contrasts favouring Explanation were non-significant in most models. Strategy gains were therefore model-idiosyncratic rather than evidence of a single robustly superior reflective framing. See Extended Data Fig. 8.

SA5. Repeat-failure heterogeneity (logistic models)

Methods. We examined whether repeat failures varied systematically by strategy or model using session-clustered logistic models for the outcome 'repeat same category'. Reference levels were model = Claude-3.7 and strategy = Explanation. We summarised marginal effects with 95% confidence intervals.

Results. Repeat odds varied across model–strategy cells in descriptive terms, but no single strategy eliminated repeats across models. In pooled models, strategy terms were not reliably different, and model effects were limited. The descriptive pattern is shown in Extended Data Fig. 5.

SA6. Failure prevalence by condition (descriptive)

Methods. We summarised human-coded failure categories by model and condition. These analyses were descriptive and intended to characterise the dominant failure modes rather than test a confirmatory contrast.

Results. Generation was dominated by *CRT-variation* and arithmetic/algebraic complexity. Search–identify shifted mass toward *existing CRT* and *CRT-variation*. These descriptive patterns are shown in Extended Data Fig. 1 and Extended Data Fig. 7.

C Prompt Templates

This appendix summarises the prompt templates used by the experiment controller. The full YAML files, exact JSON schemas, and complete known-CRT database are included with the supplementary repository. Runtime placeholders such as `failed_questions`, `task_type`, `criteria`, and `known_crt_database` were interpolated by the controller.

C.1 System Role and Shared Reminder

System role. Models were assigned a constant domain role: "You are a cognitive science expert and psychometrician specialising in Cognitive Reflection Test research." The system prompt also required pure JSON output with no prose, Markdown, comments, or schema restatement.

Shared task reminder. Models were instructed to think carefully, work step by step, and explicitly provide both the intuitive–incorrect and reflection-based correct answer for each item.

C.2 Initial Task Prompts

Generation condition. The model was asked to generate four unique CRT questions. Each question had to be newly generated, include both the intuitive–incorrect and correct answer, and avoid copying or rephrasing existing CRT items, including the original CRT, CRT2, CRT-L, and other published variants.

Search–identify condition. The model was asked to identify four pre-existing trick questions or brain teasers findable verbatim online, while excluding existing CRT items and CRT variants. Each identified item still had to include both the intuitive–incorrect and correct answer.

Optional prompt blocks. Prompt subconditions added one of three blocks: *complexity*, which emphasised reasoning over arithmetic and required explanation of the lure and reflective solution; *practical*, which told the model not to fill the quota with invalid items; and *examples*, which showed canonical CRT items as explicit do-not-copy material.

Initial output schema. Initial responses were required as JSON with a `questions` array. Each item included an `id`, `condition`, `scratchpad`, `text`, `intuitive_answer`, `correct_answer`, and an `explanation` object containing the intuitive reasoning and reflection path. Search–identify responses also included a `source` field.

C.3 Self-Reflection Elicitation Prompts

Generic self-reflection frame. For failed items, the model was shown the failed questions and told that they did not meet the required criteria. It was not shown the evaluator's diagnostic failure category. The model then generated self-reflective text intended to help a future attempt replace the failed item.

Reflection strategies. Three self-reflection framings were used. *Keywords* asked the model to list likely error types in the failed questions. *Explanation* asked the model to explain why each failed question did not meet the criteria. *Instructions* asked the model to write a step-by-step guide for producing valid replacements. A fourth condition, *Retry*, skipped self-generated reflective advice and re-presented only the generic failure verdict and original constraints.

Reflection output schemas. The Keywords prompt returned a JSON list of error categories. The Explanation prompt returned a JSON explanation string. The Instructions prompt returned a JSON list of instruction strings.

C.4 Reflection Stage Generation Prompts

Retry prompt. The model was told that its previous questions failed and was asked to produce one replacement question for each failed question, using the original task criteria.

Strategy-guided replacement prompt. For Keywords, Explanation, and Instructions, the model was additionally shown its own previously generated reflective text and instructed to use it when producing replacement questions. The number of requested replacements matched the number of failed baseline items.

C.5 Evaluator Prompt

Evaluator role and inputs. Evaluators were given the candidate item, condition, and embedded known-CRT database. They scored each item against validity, clarity, mathematical complexity, and novelty.

Failure categories. Evaluator outputs were constrained to the following categories: `no_intuitive_response`, `no_reflection_correct`,

question_not_clear, intuitive_incorrect_answer_not_clear, correct_answer_not_clear, arithmetic_complexity, algebraic_complexity, geometric_complexity, crt_variation, existing_non_crt, and existing_crt.

Novelty rule. In Generation, exact or near-exact CRT copies, superficial CRT variants, and obvious reuse of widely circulated non-CRT riddles failed. In Search–identify, pre-existing non-CRT trick questions were admissible, but exact CRT overlap and superficial CRT variation still failed.

C.6 Known CRT Database

The evaluator prompt included an embedded database of known CRT items and variants for novelty comparison, including the original CRT, CRT2, CRT-L, CRT-V, CRT-D, CRT7, and an extended set of CRT-like logic items. The complete database is included in the supplementary repository. The key protocol point is that models in the examples subcondition were explicitly shown canonical CRT items as do-not-copy material, while evaluators used the database to flag exact copies, near-copies, and superficial variants.

D Related Work Comparison

Extended Data Table 2: Contrast with adjacent work. Open-endedness alone is not enough: the present study adds an auditable exclusion/constraint task and tests whether the same failure recurs after generic feedback.

Work / approach	**Open-ended?**	**Auditable constraint?**	**Feedback / scaffold**	**Repeated failure type check?**	**Main contrast with ours**
Closed-ended self-correction (Huang et al., 2024; Kamoi et al., 2024)	No	Yes, via answer key	Correctness signal or narrow answer space	No	Usually tests final-answer accuracy; ours tests construction without answer options, executable tests, or oracle error labels.
CorrectBench-style self-correction (Tie et al.)	Partly	Yes, mostly known-answer tasks	Intrinsic, external, or fine-tuned correction	Mostly no	Broad benchmark coverage, but focuses on solve rate/pass@k; ours tracks whether the *same failure category* persists after reflection.
Tool/verifier-based refinement (Dhuliawala et al., 2024; Gou et al., 2024)	Partly	Yes	Tools, retrieval, tests, or diagnostic evidence	No	Shows refinement under external support; ours removes diagnostic support to isolate generic-feedback correction.
Open-ended answer evaluation (Lu et al., 2024)	Yes	Mostly subjective/rubric-based	LLM-derived or LLM-applied criteria	No	Addresses scoring of diverse answers; ours uses explicit item constraints so violations can be classified and followed across attempts.
Agentic autonomy benchmarks (Rein et al., 2025; Kwa et al., 2026)	Yes	Task/environment-specific	Environment state, files, tests, interaction traces	Usually no	Measures task completion in rich environments; ours uses a compact controlled task where the violated constraint is observable.
Question generation with refinement (Yao et al., 2025)	Yes, grounded	Domain-quality constraints	Expert prompts, scaffolded critique, comparison feedback	No	Optimises domain MCQ quality; ours tests unscaffolded exclusion and constraint enforcement under non-diagnostic feedback.
Ours	**Yes**	**Yes: explicit CRT structure + exclusion**	**Generic failure only**	**Yes**	Separates pass-rate gains from corrective revision by asking whether reflection eliminates the violated constraint.