SAME SCORE, DIFFERENT OUTCOMES: REWARD INTERFACES SHAPE IN-CONTEXT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models can adapt across interaction rounds from feedback retained in context without updating their parameters. However, existing studies typically treat reward as a scalar signal, leaving open whether the language interface through which that signal enters the context affects multi-round adaptation. To study this question, we develop a two-stage evaluation framework based on reward-interface presentation packages, which map an underlying scalar reward into contextualized language feedback. Stage 1 examines semantic enrichment, asking whether adding an explicit feedback contract and qualitative state descriptors to a scalar reward changes adaptation. Stage 2 examines rhetorical presentation framing, holding the underlying reward and semantic scaffold structure fixed while expressing the feedback through different semantic frames: **care-**, **competition-**, **survival-**, **trust-**, and **navigation-oriented** frames. We conduct systematic multi-round comparisons across 16,800 twenty-round episodes on two deterministic tasks, FEVER fact verification and SQuAD 2.0 extractive question answering, using GPT-5.4-mini, Qwen3-8B, and Gemma-4-E4B-it. The results show that the effect of semantic enrichment depends on the task and model: it increases late FEVER reward by 0.474 points for GPT-5.4-mini and 0.909 points for Qwen3-8B, but provides limited or mixed benefit on SQuAD 2.0. Even with the underlying reward and scaffold structure held fixed, different rhetorical frames produce distinct multi-round adaptation trajectories and endpoint differences of up to 1.251 points. These findings suggest that the linguistic presentation of reward participates in shaping model behavior during ICRL, rather than serving merely as a textual format for transmitting a scalar signal.

1 INTRODUCTION

In-context reinforcement learning (ICRL) studies how a language model adapts its later behavior from an interaction history, without updating its parameters. In a typical episode, the model answers a task query, receives a score from an evaluator, and sees that response–score pair before producing another answer. Recent work, including *Reward Is Enough* (Song et al., 2026), shows that even a scalar reward can support this kind of test-time adaptation.

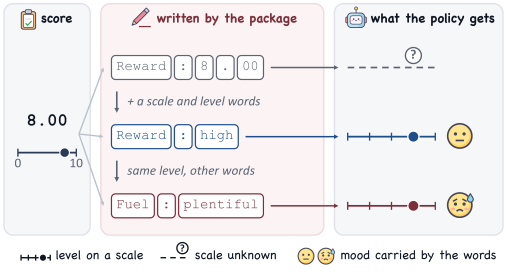

Figure 1: The model reads the score as text. A numeric reward leaves its scale implicit; qualitative levels and framing add context to the same score.

The scalar, however, is not what a language model actually reads (Figure 1). The score must be rendered as text and placed in the context alongside the task instructions and earlier attempts. A bare line such as "Reward: 8.00" can instead include a description of what is being evaluated, a qualitative level, or a phrase that names what the model should improve. These choices form a **reward-interface presentation package**: the block of feedback text that carries the score and explains how to use it. In single-turn prompting, such wording

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.

affects one response. In multi-round ICRL, the rendered feedback remains in the history and is read again in later rounds, so a wording choice can become a persistent part of the environment and shape the model's adaptation trajectory.

This distinction motivates our question: when the task, scalar scorer, model, and interaction protocol are held fixed, does the language used to present the score change multi-round adaptation? The question matters for both interpretation and comparison. Two studies using the same scorer and items could report different endpoints because their feedback interfaces differ. We therefore compare complete packages, treating the package as the experimental unit rather than attributing an effect to any isolated word or phrase.

Our evaluation has two stages. Stage 1 tests **semantic enrichment** by comparing a bare numeric reward with a neutral scaffold that explains what the score measures and expresses it through five qualitative levels. Stage 2 tests **rhetorical presentation framing** while preserving the underlying reward and the common five-bin boundaries, replacing the neutral wording with care-, competition-, survival-, trust-, and navigation-oriented frames. The design asks first whether adding a verbal scaffold changes adaptation, and then whether equally informative ways of expressing the same binned score produce different trajectories.

We evaluate the packages across 16,800 20-round episodes on two deterministic tasks with complementary demands. FEVER requires a verdict with supporting evidence, including cases where the correct action is "not enough information," whereas SQuAD 2.0 combines answer extraction with abstention when the passage does not support an answer. We use GPT-5.4-mini, Qwen3-8B, and Gemma-4-E4B-it, and compare early and late reward together with the resulting adaptation trajectories.

The results show that semantic enrichment improves late FEVER reward by 0.474 points for GPT-5.4-mini and 0.909 points for Qwen3-8B, while its effect on SQuAD 2.0 is mixed across models. More importantly, changing the frame of the same binned score produces distinct multi-round trajectories and endpoint differences of up to 1.251 points. Thus, reward presentation is behaviorally consequential in ICRL and should be treated as part of the method rather than as unreported formatting.

Our contributions are:

1. We formalize reward-interface presentation packages as a package-level treatment variable for multi-round ICRL, separating the task scorer from the text that carries its output.
2. We provide a two-stage evaluation across 16,800 twenty-round episodes that separates semantic enrichment from rhetorical framing while controlling the underlying task reward and interaction protocol.
3. We report model- and task-stratified evidence on adaptation trajectories and endpoint performance, and derive reporting practices based on package transparency, dual endpoints, and difficulty stratification.

2 Related Work

Verbal feedback and iterative refinement. Natural-language feedback can guide a model's next attempt by describing what went wrong. Reflexion (Shinn et al., 2023) carries reflections on task outcomes into later attempts, while Self-Refine (Madaan et al., 2023) revises outputs through self-generated critiques. Such feedback offers task-specific guidance, although revision without external feedback does not reliably improve accuracy (Huang et al., 2024). Textual feedback can also serve as supervision for parameter updates (Abadal Lloret et al., 2024; Wang et al., 2025). Our setting differs from these approaches: a fixed evaluator supplies a reward, the model receives no diagnosis or suggested answer, and its weights do not change. We ask whether the language presenting that reward matters when the underlying scalar score is held fixed.

In-context reinforcement learning. ICRL uses histories of actions and outcomes to guide later decisions without updating parameters at evaluation time (Laskin et al., 2023; Monea et al., 2025; Moeini et al., 2025). This makes the interaction history, rather than a weight update, the immediate carrier of adaptation. In a related optimization setting, OPRO places earlier candidate solutions and their scores in an optimizer's prompt to guide later proposals (Yang et al., 2024). Our closest

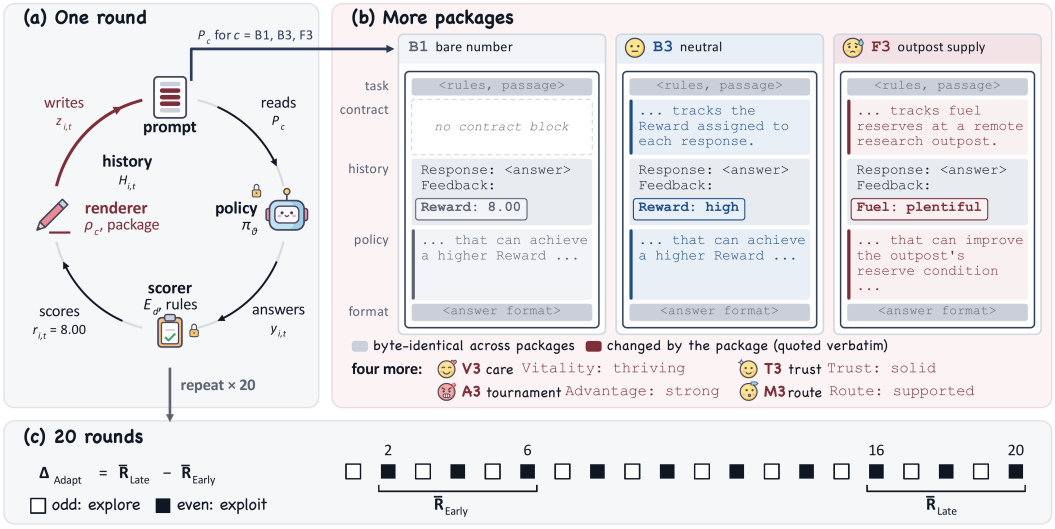

Figure 2: A fixed evaluator scores each answer, and the presentation package renders feedback for subsequent rounds. Stage 1 compares numeric with structured qualitative feedback; Stage 2 compares qualitative packages with shared score bins.

point of departure is *Reward Is Enough* (Song et al., 2026), which shows that a language model can improve over successive attempts when earlier responses and scalar rewards remain in context. Verbal critiques are therefore not necessary for improvement in that setting. Yet the scalar still reaches the model as text, embedded in a prompt it will read again on later rounds. Work on in-context reward hacking shows that repeated feedback can produce behavior missed by static evaluation (Pan et al., 2024). What remains unclear is whether the same score can lead to different adaptation when its presentation changes. We examine that question with the task scorer and interaction protocol held fixed.

Prompt sensitivity and semantic framing. Prompt-design studies show that format, demonstration order, and contextual cues can change model outputs even when the underlying task is similar (Sclar et al., 2024; Lu et al., 2022; Zhao et al., 2021; Germani & Spitale, 2025). Affective prompting further suggests that semantic framing can matter, but reported effects vary across settings (Li et al., 2023; Zhao et al., 2026). These studies mainly measure the response to a prompt at one point in time. In multi-round ICRL, each rendered reward remains alongside earlier attempts and can be read again in subsequent rounds. A presentation choice may thus affect not only the next answer but also the history from which later answers are generated. We first compare a bare numeric reward with more informative semantic feedback, then compare alternative framings built on a common feedback structure. These are comparisons of complete reward-interface packages under the same evaluator; they do not isolate the causal effect of an individual word.

3 Problem Formulation

In in-context reinforcement learning (ICRL), a model can use the outcomes of earlier attempts to improve its answer to the same task. Those outcomes reach the model through its prompt: a fixed evaluator assigns a score, and a presentation package turns that score into feedback for later rounds. We study whether changing the package changes adaptation while leaving the task, model, and evaluator fixed (Figure 2).

3.1 Interaction and Presentation

For item x_i, round t starts with the task, prior attempts $H^F_{i,t-1}$, and a round instruction assembled into prompt P_F. The frozen model produces answer $y^F_{i,t}$; the evaluator scores it; and package F

renders that score for the next prompt:

$$y^F_{i,t} \sim \pi_\theta(\cdot \mid P_F(x_i, H^F_{i,t-1}, t)),$$
$$R^F_{i,t} = R(x_i, y^F_{i,t}), \qquad z^F_{i,t} = \rho_F(R^F_{i,t}),$$
$$H^F_{i,t} = H^F_{i,t-1} \parallel (y^F_{i,t}, z^F_{i,t}).$$

The evaluator R is identical across conditions: the same answer to the same item earns the same scalar reward. The rendered string $z^F_{i,t}$, however, can change later answers and thus the rewards actually observed. A package coordinates the feedback contract, history feedback, round-specific objective wording, and output restriction within a shared task and answer structure. Appendix B gives the prompts and renderings. This distinction matters over an episode: a rendered outcome remains in the history, so one presentation choice can affect several later attempts. We therefore compare complete feedback packages at a fixed scorer, rather than treating the score shown at one round as an independent intervention.

3.2 Two comparisons

Stage 1 compares B1, which displays a number such as `Reward: 8.00`, with B3, which adds a short feedback contract and displays one of five qualitative levels, such as `Reward: high`. This comparison tests the complete structured interface: it changes both context and the resolution of visible feedback. Single-replicate controls B2, BL, B4, N0, and Z0 probe interface components separately but remain diagnostic.

Stage 2 uses B3 as the neutral reference for five alternative frames: companion vitality (V3), outpost fuel (F3), professional trust (T3), tournament advantage (A3), and solution routes (M3). All six use the same evaluator and five score intervals. They replace the contract, feedback terms, objective wording, and output restriction together, so the contrast concerns complete packages rather than individual words. The task and required answer fields stay fixed. The two stages answer different questions. Stage 1 asks whether adding an interpretable qualitative structure to numeric feedback changes adaptation; Stage 2 asks whether its wording matters after the qualitative structure and score intervals are held constant. Neither comparison separates the effect of an individual phrase from the rest of its package.

3.3 Outcomes

Each episode has 20 rounds: round 1 initializes an answer, subsequent odd rounds explore alternatives, and even rounds exploit the history. Exploration requests an answer different from prior attempts; exploitation requests the best response based on prior answers and feedback. Both retain the task and response contracts. Histories develop separately under each package, so a fixed evaluator does not imply identical prior attempts. We measure performance on early and late exploitation rounds. For item i and run k,

$$Y_{\text{early},i,k} = \tfrac{1}{3}(R_{i,k,2} + R_{i,k,4} + R_{i,k,6}),$$
$$Y_{\text{late},i,k} = \tfrac{1}{3}(R_{i,k,16} + R_{i,k,18} + R_{i,k,20}),$$
$$G_{i,k} = Y_{\text{late},i,k} - Y_{\text{early},i,k}.$$

We report late reward and adaptation gain separately: improvement from an early score need not imply the best final score. Our primary contrast compares item-paired late reward after averaging runs within each package. Correction, retention, and task-specific error measures describe the resulting trajectories; their definitions are in Appendix C.

4 Experimental Design

4.1 Tasks and scoring

We evaluate two tasks that require different responses to uncertainty. FEVER (Thorne et al., 2018) asks for a SUPPORTS, REFUTES, or NOT ENOUGH INFO verdict with evidence identifiers from ten supplied candidates. Its 100 items comprise 26, 50, and 24 instances of these classes, respectively. SQuAD 2.0 (Rajpurkar et al., 2018) asks for an answer supported by a passage or abstention; its 100

items comprise 50 answerable and 50 unanswerable questions. Each task uses the same items across conditions.

FEVER evaluates verification using supplied evidence, without a retrieval step. SQuAD requires a short answer with a supporting quotation, or the designated abstention format. Both tasks use deterministic 0–10 scorers, not model judges. FEVER scores verdict accuracy and evidence overlap, with a distinct rule for NOT ENOUGH INFO. SQuAD scores answer overlap on answerable items and the quality of abstention on unanswerable ones. Invalid formats receive zero. Appendix A documents sample construction and the complete scoring rules. To interpret reward changes, we also report task decisions on the same late exploitation rounds. FEVER strict correctness requires both a correct verdict and complete evidence, whereas SQuAD answerability accuracy measures whether the model answers when the passage supports an answer and abstains otherwise. These measures distinguish a higher partial score from a correct final decision (Appendix C).

4.2 Models and protocol

We evaluate GPT-5.4-mini, Qwen3-8B, and Gemma-4-E4B-it, reporting each model–task cell separately. Task-response calls use temperature 0.6, top-p 0.95, and a 1,000-token output limit. Model weights, items, scorer, decoding settings, and the 20-round schedule remain fixed across package comparisons within a cell. Prior answers and rendered feedback remain available in the episode history; the prompt templates and further protocol details are in Appendices A and B.

For context, we also evaluate three reference methods on the same items and scorers: history-free chain-of-thought (CoT) (Wei et al., 2022), Self-Refine (Madaan et al., 2023), and Reflexion (Shinn et al., 2023). Self-Refine uses its own critiques; Reflexion adds a planning call after each non-final answer. Their histories and call counts therefore differ from the reward-interface conditions. CoT has no history-conditioned adaptation measure.

4.3 Replication and analysis

Formal conditions have three runs, except GPT-5.4-mini B3, whose three runs from each stage form one six-run reference. Qwen3-8B and Gemma-4-E4B-it share three B3 runs across stages; each is counted once. We average runs within each item before computing paired package differences across the 100 items. The same B3 scope applies to endpoints, trajectories, and behavioral summaries (Appendix C).

For late-reward contrasts, we use 5,000 paired item-bootstrap resamples for pointwise 95% intervals and 5,000 random sign flips for two-sided Monte Carlo tests. Holm adjustment covers four Stage-1 and five Stage-2 comparisons within each model–task cell. Inference is conditional on the retained runs; Appendix G gives the full procedure.

The B2, BL, B4, N0, and Z0 controls use one run and are reported as diagnostics. Teacher-forced attention replay uses Rep01 of the two open models only; its associations with performance are descriptive rather than evidence of a causal attention mechanism. Appendix A separates these analyses from the replicated endpoint comparisons.

5 Results

We report the two stages in sequence. Late reward is the primary endpoint; early-to-late gain describes within-episode change. Tables 1 and 2 summarize paired contrasts, while Figures 3 and 4 show the 20-round trajectories. A dagger marks $p_{\mathrm{Holm}} < 0.05$; Appendix G gives intervals and adjusted p-values.

5.1 Stage 1: structured feedback versus numeric reward

B3 improves FEVER late reward over B1 by 0.474 points for GPT-5.4-mini and 0.909 for Qwen3-8B, both significant after Holm adjustment (Table 1). The Gemma-4-E4B-it FEVER estimate is smaller (+0.086) and is not significant after adjustment. None of the SQuAD 2.0 contrasts is significant; Qwen3-8B is nearly unchanged (-0.020). The benefit of structured feedback is therefore concentrated in two FEVER model–task cells.

Table 1: Structured qualitative feedback versus bare numeric reward in Stage 1. Late reward averages rounds 16, 18, and 20; early reward averages rounds 2, 4, and 6; gain is late minus early. Δ is the item-paired B3-minus-B1 late-reward contrast. All values use the 0–10 reward scale.

		Late reward		Early reward		Gain		
Model	**Task**	B1	B3	B1	B3	B1	B3	Δ
GPT-5.4-mini	FEVER	7.164	**7.638**	6.721	7.055	+0.443	+0.583	+0.474†
GPT-5.4-mini	SQuAD	6.742	**7.067**	6.513	6.374	+0.229	+0.692	+0.324
Qwen3-8B	FEVER	6.642	**7.551**	6.373	7.062	+0.270	+0.489	+0.909†
Qwen3-8B	SQuAD	**6.920**	6.900	6.887	6.880	+0.033	+0.020	−0.020
Gemma-4-E4B-it	FEVER	6.346	**6.432**	5.921	6.181	+0.425	+0.252	+0.086
Gemma-4-E4B-it	SQuAD	7.064	**7.314**	7.179	7.369	−0.114	−0.054	+0.250

Bold marks the higher late reward within each B1–B3 pair. † denotes $p_{\mathrm{Holm}} < 0.05$ (Appendix G). GPT B3 pools six runs; other cells use three. Displayed differences may vary by 0.001 because contrasts use unrounded item-level values. Full summaries appear in Appendix D.

The interval estimates distinguish the two clear FEVER gains from the smaller effects. The pointwise 95% intervals are [0.242, 0.744] for GPT-5.4-mini and [0.470, 1.416] for Qwen3-8B, whereas all three SQuAD intervals span zero. Gemma-4-E4B-it FEVER has a positive pointwise interval [0.013, 0.175], but its Holm-adjusted p-value is 0.1376 (Appendix G). Thus, an interval excluding zero does not by itself establish significance for the family-adjusted comparison.

Task decisions support this reading. On Qwen3-8B FEVER, B3 raises label accuracy from 80.33% to 93.67% and strict correctness from 37.00% to 45.00%. On GPT-5.4-mini SQuAD, exact match rises from 64.00% to 67.56%, but the late-reward contrast does not survive adjustment (Appendix D). These task metrics are descriptive, not additional significance tests. Early and late scores also distinguish adaptation from initial advantage: GPT-5.4-mini B3 starts below B1 on SQuAD (6.374 versus 6.513), gains more (+0.692 versus +0.229), and finishes above it. Conversely, Gemma-4-E4B-it B3 starts higher on FEVER but gains less than B1. A higher starting score need not indicate stronger within-episode improvement.

The other late task metrics help locate these differences. On GPT-5.4-mini FEVER, evidence completeness rises from 44.22% to 51.06% and strict correctness from 43.22% to 50.06%. On SQuAD, its answerability accuracy rises from 70.00% to 74.39%. Qwen3-8B, by contrast, retains 75.00% answerability accuracy, and its exact match is nearly unchanged (67.67% under B1 versus 67.33% under B3). Gemma-4-E4B-it exact match increases from 62.22% to 66.00% (Appendix D). Together, these outcomes show where a reward difference reflects better evidence selection or answerability decisions, while retaining the distinction between descriptive task metrics and the primary paired endpoint.

The prompting references provide context because their histories and call budgets differ. GPT-5.4-mini B3 has the highest late reward among the evaluated methods on both tasks, with a SQuAD margin of 0.025 over Self-Refine. Qwen3-8B B3 reaches 7.551 on FEVER, above the highest reference endpoint of 6.429. For Gemma-4-E4B-it, CoT leads on FEVER (6.546) and Self-Refine on SQuAD (7.693; Appendix D). The top rows of Figures 3 and 4 show these patterns across rounds.

5.2 Stage 2: alternative presentation packages

With B3 as the neutral reference, alternative packages produce substantial but model-dependent differences (Table 2). V3, F3, and A3 each lower GPT-5.4-mini SQuAD late reward by more than 1.2 points; F3 has the largest loss (-1.251). All five alternatives lower Qwen3-8B FEVER significantly, whereas none of its SQuAD contrasts survives adjustment. On Gemma-4-E4B-it, only the SQuAD F3 decrease (-0.446) is significant. M3 is close to B3 on GPT-5.4-mini but reduces Qwen3-8B FEVER by 0.618 points. No alternative significantly exceeds B3 in any model–task cell. These comparisons concern complete packages, not individual words.

The trajectories also separate endpoint differences from the direction of change. On GPT-5.4-mini SQuAD, F3 declines from early to late reward (-0.284), while B3 improves (+0.692). On FEVER, M3 gains more than B3 (+0.708 versus +0.583) but finishes at almost the same reward (7.637 versus

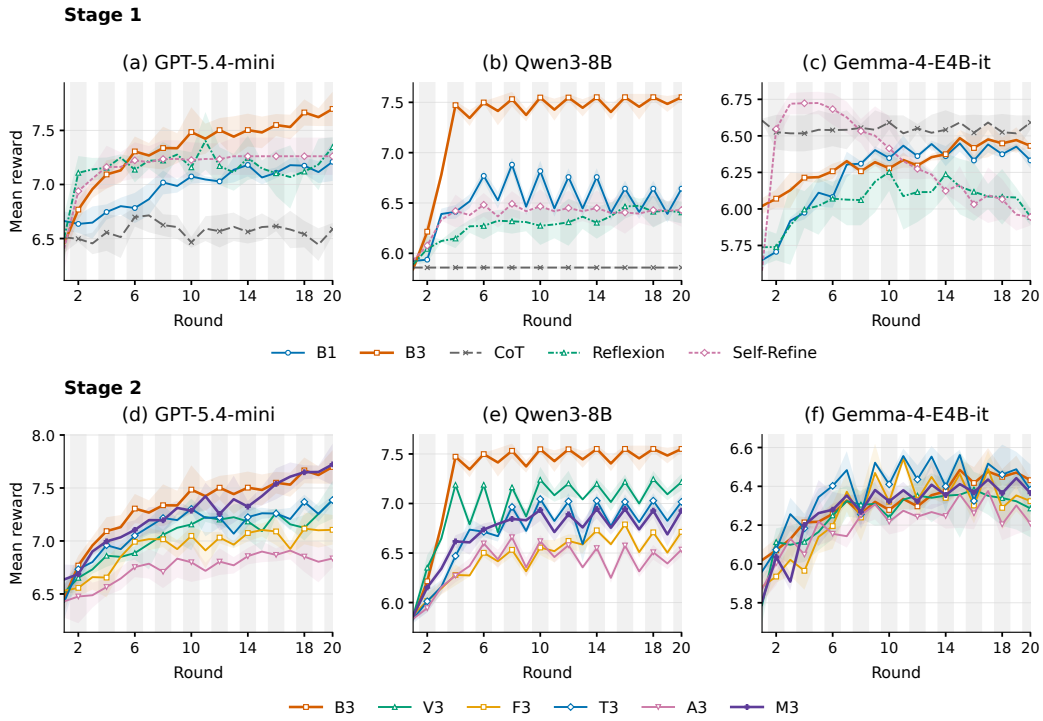

Figure 3: Reward trajectories on FEVER. Columns show GPT-5.4-mini, Qwen3-8B, and Gemma-4-E4B-it. Top: Stage 1 (B1, B3, CoT, Reflexion, Self-Refine). Bottom: Stage 2 (B3, V3, F3, T3, A3, M3). Lines show mean reward and ribbons show ± 1 replicate SD over R1–R20. Gray bands mark even exploitation rounds for the ICRL conditions. GPT B3 pools six replicates; all other model–condition cells use three. B3 is shared across stages. Vertical scales vary by panel.

Table 2: Late reward relative to B3 in Stage 2. The B3 column gives the neutral reference endpoint; the remaining columns give item-paired contrasts for companion vitality (V3), outpost fuel (F3), professional trust (T3), tournament advantage (A3), and solution routes (M3). Bold and underline mark the highest and second-highest contrasts among the five alternatives in each row.

			Late reward against B3				
Model	**Task**	B3	V3	F3	T3	A3	M3
GPT-5.4-mini	FEVER	7.638	$-0.378^{\dagger}$	$-0.534^{\dagger}$	$\underline{-0.300}^{\dagger}$	$-0.786^{\dagger}$	$\mathbf{-0.001}$
GPT-5.4-mini	SQuAD	7.067	$-1.224^{\dagger}$	$-1.251^{\dagger}$	$\underline{+0.111}$	$-1.233^{\dagger}$	$\mathbf{+0.229}$
Qwen3-8B	FEVER	7.551	$\mathbf{-0.324}^{\dagger}$	$-0.814^{\dagger}$	$\underline{-0.530}^{\dagger}$	$-1.012^{\dagger}$	$-0.618^{\dagger}$
Qwen3-8B	SQuAD	6.900	-0.200	-0.227	$\mathbf{+0.087}$	-0.291	$\underline{-0.140}$
Gemma-4-E4B-it	FEVER	6.432	-0.096	-0.128	$\mathbf{-0.043}$	-0.208	$\underline{-0.066}$
Gemma-4-E4B-it	SQuAD	7.314	-0.324	$-0.446^{\dagger}$	$\underline{-0.224}$	-0.292	$\mathbf{-0.011}$

All values use the 0–10 reward scale. † denotes $p_{\text{Holm}} < 0.05$ (Appendix G). The conditions change complete presentation packages, not individual words. Run counts follow Table 1; full endpoints appear in Appendix D.

7.638). On Qwen3-8B FEVER, T3 likewise gains more than B3 (+0.621 versus +0.489) yet reaches a lower endpoint (7.021 versus 7.551; Appendix D). A larger gain can therefore coexist with a worse final score when packages begin at different levels.

The reward losses correspond to identifiable task decisions. On GPT-5.4-mini SQuAD, B3-to-F3 answerability accuracy falls from 74.39% to 63.11% and exact match from 67.56% to 53.56% (Appendix D). Across rounds 1–20, over-commitment on unanswerable questions rises from 51.78% under B3 to 66.50% under F3, while over-refusal on answerable questions changes little (4.15% versus 4.40%). On FEVER, A3 likewise raises over-commitment on inconclusive claims from 55.62%

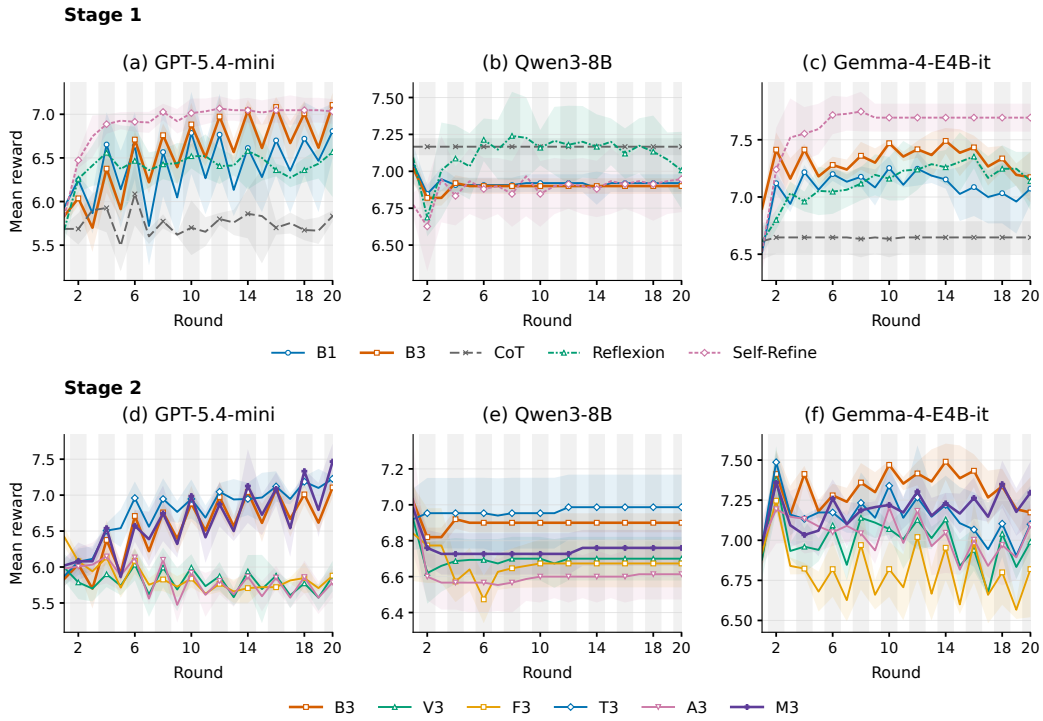

Figure 4: Reward trajectories on SQuAD 2.0. Columns show GPT-5.4-mini, Qwen3-8B, and Gemma-4-E4B-it. Top: Stage 1 (B1, B3, CoT, Reflexion, Self-Refine). Bottom: Stage 2 (B3, V3, F3, T3, A3, M3). Lines show mean reward and ribbons show ± 1 replicate SD over R1–R20. Gray bands mark even exploitation rounds for the ICRL conditions. GPT B3 pools six replicates; all other model–condition cells use three. B3 is shared across stages. Vertical scales vary by panel.

to 71.39% (Appendix E). The largest losses thus occur where the evidence does not warrant a definite answer.

Paired-round behavior adds a temporal view. On GPT-5.4-mini SQuAD, 23.09% of B3's initially incorrect responses become strictly correct at the next exploitation round, versus 9.68% under F3; regression after a correct response is 2.32% versus 5.28%. The same wrong decision persists in 44.78% versus 69.35% of cases (Appendix E). These rates describe different observed error populations and do not identify a correction mechanism. The bottom rows of Figures 3 and 4 place the endpoint differences in their trajectories.

Recovery and final performance are related but distinct. GPT-5.4-mini M3 has a correction rate of 25.18%, compared with B3's 23.09%, and a higher late reward (7.296 versus 7.067), but its paired endpoint contrast is not significant. On FEVER, A3 reduces strict correctness from 50.06% to 38.78% for GPT-5.4-mini and from 45.00% to 33.44% for Qwen3-8B (Appendices D and E). Endpoint, task-decision, and transition measures thus give complementary views of how a package affects repeated attempts.

6 Discussion

The reward interface is part of the ICRL intervention. A fixed evaluator assigns the same score to the same answer, yet its rendered feedback remains in the history available to later attempts. Stage 1 finds a replicated benefit for structured feedback in two FEVER cells, but not a uniform effect across tasks and models. The B3–B1 comparison changes both the feedback contract and the visible score resolution; single-run component controls cannot isolate either contribution with the same evidential strength.

B3's contract explains what the score evaluates, and its qualitative levels place an outcome on an interpretable scale. This structure could help a model decide whether to retain or revise a verdict and its supporting evidence. The FEVER task metrics are consistent with this interpretation: the reward gains accompany better verdicts and more complete evidence. The mixed SQuAD results suggest that the value of such structure depends on the decisions required by the task, rather than semantic enrichment alone guaranteeing improvement.

Stage 2 shows that shared score intervals do not guarantee similar adaptation. On GPT-5.4-mini SQuAD, vitality, fuel, and tournament framing coincide with more unsupported answers on unanswerable questions, while over-refusal changes much less. This suggests that evidence-sufficiency decisions deserve attention alongside aggregate reward. A possible explanation is that some frames make abstention less compatible with the stated objective, but testing it requires interventions that separate the contract, level words, and objective wording. Package rankings also change across models: M3 is close to B3 on GPT-5.4-mini yet significantly worse on Qwen3-8B FEVER.

The transition results connect these decisions to behavior over repeated attempts. F3's lower SQuAD reward accompanies both less recovery from errors and more regression after correct answers, as well as greater persistence of the same wrong decision. This makes recovery and retention useful diagnostic dimensions alongside the final score. Because each condition generates its own preceding answers, the conditional rates do not compare a common set of prior errors; they characterize the trajectories induced by the complete package.

Attention replay offers a descriptive view of history use. Across six packages in each of four open-model/task cells, attention mass on the previous answer and other history correlates positively with late reward, whereas instruction attention mass correlates negatively; feedback-region correlations have mixed directions (Appendix F, Table F.1). Each association uses one replayed run per package. It does not show that redirecting attention causes better performance.

The positive history associations motivate examining how models use prior attempts, beyond how much attention reaches the reward string itself. A model may attend to feedback without using it to revise the task decision, while the previous answer supplies the candidate to retain or replace. Attention mass measures allocation rather than whether the attended content is useful, so this interpretation calls for interventions on history availability and rendering.

These results make the complete interface, within a specified model and task, the appropriate unit of comparison. Reports should specify its feedback contract, score-to-state mapping, history rendering, round objectives, and output restrictions, and distinguish late reward from within-episode gain. The present evidence covers two deterministic benchmarks, three models, and one interaction protocol. Replicated component interventions and broader tasks are needed to establish which features transfer beyond these settings.

7 Conclusion

Through a two-stage evaluation, we examined whether the language interface through which scalar reward enters context affects multi-round in-context reinforcement learning. Structured feedback improves late reward over bare numeric reward in two FEVER model–task cells, but the benefit does not extend uniformly to SQuAD 2.0. With the scorer and score intervals fixed, alternative presentation packages can then produce distinct adaptation trajectories and substantial late-reward losses relative to the neutral reference. These results show that the value of a reward signal does not fully determine how a model uses it after the signal is rendered into language and retained in its interaction history. The behavioral analyses clarify what an endpoint difference can mean without assigning it to an unobserved mechanism. On unanswerable questions, the poorer SQuAD packages are associated chiefly with more unsupported answers; paired-round analysis also reveals differences in recovery and retention. Attention replay supplies a descriptive view of history use, but its package-level associations cannot establish a causal pathway. The resulting unit of comparison is the complete reward interface in a specified model and task. Reporting its feedback contract, wording, history format, and round objectives alongside late reward makes these comparisons interpretable and gives future work a basis for testing individual components with replicated interventions.

AI Use Statement. In this work, we used generative AI tools for two tasks that require disclosure. First, to screen candidate FEVER claims and their evidence: two frontier models labeled candidates independently under a blind protocol. The authors adjudicated the 83 disagreements against the reference articles; 17 cases with reviewer agreement were retained after difficulty-based selection (Appendix A). Second, to draft first-pass descriptions of the logged results, which the authors then verified against the per-event logs and rewrote. We have not used generative AI tools to generate synthetic datasets, to develop theoretical models or conceptual frameworks, to formulate or prove mathematical claims, to propose or refine hypotheses, or to design the research methodology or the experiments. Translation and qualitative coding are not applicable to this work. We also used generative AI tools for tasks where disclosure is recommended: writing and editing software code, drafting and revising the manuscript and its figures, and literature search. We have reviewed all AI-assisted work: AI-assisted code was read and tested for correctness by the authors, and every AI-assisted claim in this paper was checked against the per-event logs and the reported tables before submission. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

Reproducibility Statement. The episode protocol is specified in Section 3 and the rule-based scorers in Section 4 and Appendix A. The evaluation instances and replicate structure are in Section 4, with the decoding settings in Appendix A, and the inference procedure, including the bootstrap seed, randomization test, and Holm families, is in Appendix G. The prompt blocks of every condition are in Appendix B, and the per-cell endpoints behind the main-text numbers are in Appendix D. Re-execution requires the task instances, scorer implementations, prompt specifications, analysis scripts, per-event logs, and model access in addition to the decoding settings in Appendix A. Because decoding is stochastic and serving is not bit-reproducible, re-runs should be compared with the reported replicate dispersion rather than expected to reproduce individual outputs exactly.

REFERENCES

Saüc Abadal Lloret, Shehzaad Dhuliawala, Keerthiram Murugesan, and Mrinmaya Sachan. Towards aligning language models with textual feedback. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20240–20266, 2024. doi: 10.18653/v1/2024.emnlp-main.1129.

Federico Germani and Giovanni Spitale. Source framing triggers systematic bias in large language models. *Science Advances*, 11:eadz2924, 2025. doi: 10.1126/sciadv.adz2924.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.01798.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, Maxime Gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.14215.

Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, and Xing Xie. Large language models understand and can be enhanced by emotional stimuli. *arXiv preprint arXiv:2307.11760*, 2023. URL https://arxiv.org/abs/2307.11760.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8086–8098, 2022. doi: 10.18653/v1/2022.acl-long.556.

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*, 2023. URL https://arxiv.org/abs/2303.17651.

Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangtong Zhang. A survey of in-context reinforcement learning. *arXiv preprint arXiv:2502.07978*, 2025. URL https://arxiv.org/abs/2502.07978.

Giovanni Monea, Antoine Bosselut, Kianté Brantley, and Yoav Artzi. LLMs are in-context bandit reinforcement learners. In *Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2410.05362.

Alexander Pan, Erik Jones, Meena Jagadeesan, and Jacob Steinhardt. Feedback loops with language models drive in-context reward hacking. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39154–39200, 2024. URL https://proceedings.mlr.press/v235/pan24d.html.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, 2018. doi: 10.18653/v1/P18-2124.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.11324.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2303.11366.

Kefan Song, Amir Moeini, Peng Wang, Lei Gong, Rohan Chandra, Shangtong Zhang, and Yanjun Qi. Reward is enough: LLMs are in-context reinforcement learners. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2506.06303.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of NAACL-HLT*, pp. 809–819, 2018. doi: 10.18653/v1/N18-1074.

Hanyang Wang, Lu Wang, Chaoyun Zhang, Tianjun Mao, Si Qin, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Text2Grad: Reinforcement learning from natural language feedback. *arXiv preprint arXiv:2505.22338*, 2025. URL https://arxiv.org/abs/2505.22338. Version 2, revised January 2026.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2201.11903.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.03409.

Minda Zhao, Yutong Yang, Chufei Peng, Rachel Gonsalves, Weiyue Li, Ruyi Yang, Zhixi Liu, and Mengyu Wang. Do emotions in prompts matter? effects of emotional framing on large language models. *arXiv preprint arXiv:2604.02236*, 2026. URL https://arxiv.org/abs/2604.02236.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 12697–12706, 2021. URL https://proceedings.mlr.press/v139/zhao21c.html.

A EXPERIMENTAL PROTOCOL AND EVIDENCE SCOPE

The main experiment asks whether the way a fixed task score enters the conversation changes later responses. Reproducing that comparison requires more than the score formula: the task instances, response parser, interaction schedule, and run aggregation must also be fixed. This section gives those details. Appendix B specifies what each model actually saw; Appendices C–G define and report the resulting measurements.

A.1 EVALUATION SETS

FEVER consists of 100 claim-verification instances with ten evidence candidates per claim: 26 SUPPORTS, 50 REFUTES, and 24 NOT ENOUGH INFO (NEI) instances. The model must judge the claim using the supplied candidates and cite the corresponding evidence identifiers. AI tools assisted candidate screening. The 83 items on which the reviewers disagreed were resolved through human adjudication, while 17 items with strict reviewer agreement were retained after difficulty-based selection. The evaluation measures verification over the supplied candidates, not open-domain evidence retrieval.

SQuAD 2.0 consists of 100 questions, with 50 answerable and 50 unanswerable instances selected by stratified sampling with seed 43. Each instance supplies a context passage and a question. Answerable questions require a short answer and a supporting quotation; unanswerable questions require the designated abstention form. The same instances are used across models and conditions within each task. Neither evaluation set changes between the two stages.

A.2 DETERMINISTIC REWARD FUNCTIONS

The task evaluator runs before feedback is presented to the model. It computes an underlying scalar reward $R_t = R(x, y_t) \in [0, 10]$ from the response and reference data using fixed scoring rules. A presentation package then turns that score into the visible feedback r_t. Thus two packages can show different messages for the same scored response without changing its underlying reward.

For FEVER, let $\widehat{\ell}_t$ and $\ell^\star$ denote the predicted and reference labels, E_t the predicted evidence-identifier set, and $\mathcal{E}^\star$ the collection of acceptable reference evidence sets. The implemented reward is

$$R(x, y_t) = \begin{cases} 0, & \text{invalid response format,} \\ 2, & \widehat{\ell}_t \neq \ell^\star, \\ 10, & \widehat{\ell}_t = \ell^\star = \text{NEI},\ E_t = \varnothing, \\ 4, & \widehat{\ell}_t = \ell^\star = \text{NEI},\ E_t \neq \varnothing, \\ 4 + 6 \max\limits_{E^\star \in \mathcal{E}^\star} \dfrac{|E_t \cap E^\star|}{|E_t \cup E^\star|}, & \widehat{\ell}_t = \ell^\star \neq \text{NEI}. \end{cases}$$

The maximum overlap defaults to zero if no nonempty union is available. A valid FEVER response must contain a recognized verdict and an Evidence field. Strict task correctness, used in the sequence analysis, requires a correct label and an exact match to an acceptable evidence set; a correct NEI decision requires an empty evidence set. Thus, partial reward and strict correctness measure different aspects of the response.

For SQuAD 2.0, an invalid response format receives zero reward. On an answerable item, let f_t be the maximum normalized token F1 over the reference answers. The initial score is

$$b(f_t) = \begin{cases} 10, & f_t \geq 0.9, \\ 8, & 0.7 \leq f_t < 0.9, \\ 6, & 0.5 \leq f_t < 0.7, \\ 4, & 0.3 \leq f_t < 0.5, \\ 2, & 0 < f_t < 0.3, \\ 0, & f_t = 0. \end{cases}$$

An empty answer or the no-answer sentinel receives zero on an answerable item. Otherwise, the reward is $b(f_t)$, capped at 6 when the supplied evidence does not pass the evaluator's evidence-in-context check. Before token comparison, answer strings undergo Unicode normalization, lowercasing, punctuation and article removal, and whitespace normalization.

On an unanswerable item, a strict refusal receives 10: `Answerable: no`, a normalized no-answer sentinel in the Answer field, and a normalized no-answer sentinel in the Evidence field. The required surface form is `Answer: UNANSWERABLE` with `Evidence: N/A`. A weak refusal receives 5 if it declares the item unanswerable, uses a no-answer sentinel, or contains recognized refusal language without satisfying the strict form. Other responses receive zero. These reward categories differ from the binary predicted-no-answer indicator used in the behavioral analysis.

A.3 Interaction and decoding

Each multi-round trajectory contains 20 responses to the same task instance. Model parameters remain fixed. Round 1 uses an initial-response instruction; subsequent odd rounds request exploration and even rounds request exploitation. The model sees the task contract, the condition's feedback contract where applicable, prior attempts and rendered feedback, the current adaptation instruction, and the response contract.

Setting	Value
Models	GPT-5.4-mini; Qwen3-8B; Gemma-4-E4B-it
Temperature	0.6
Top-p	0.95
Task-response output limit	1,000 tokens
Formal trajectory length	20 rounds
Task instances per evaluation set	100

All conditions, including the baselines, use these settings for task-response generation. Reflexion's auxiliary planning calls use the same sampling temperature and top-p, but have a separate output limit of at most 256 tokens.

All conditions use the same deterministic task evaluator: for a given task instance, an identical response receives the same underlying reward across packages. Realized reward sequences can differ when response sequences differ. This is the fixed-scorer basis of the package comparisons, not a guarantee that different packages induce the same response history.

A.4 Baseline implementations

The CoT baseline solves each instance without access to earlier attempts. We report score summaries for its repeated evaluations; correction, retention, and other history-conditioned adaptation metrics are not applicable.

Self-Refine retains previous responses and self-generated feedback. After the initial round, one completion produces both a critique in a `<feedback>` block and the revised task response. No external reward is included in its refinement prompt.

Reflexion makes an auxiliary model call after each non-final attempt to produce a procedural plan from the task, the response, and the observed task reward. The next policy call receives accumulated plans, not the earlier responses and numerical rewards themselves. The auxiliary call uses the same model with an output limit of at most 256 tokens. Consequently, matching the number of task-response rounds does not match the number of model calls across baselines.

A.5 Replication and evidence scope

For GPT-5.4-mini, the B3 reference pools six distinct runs: three from Stage 1 and three from Stage 2. The same six-run pool is used for all formal B3 endpoints, trajectories, and behavioral summaries in both the main text and this appendix. Other GPT conditions have three runs. Qwen3-8B and Gemma-4-E4B-it each have three B3 runs shared across the stages; each run is counted once. Their

other formal conditions also have three runs. All formal comparisons use rounds 1–20. The separately identified attention replay uses Rep01 of the two open models, not the formal multi-run pool.

B Prompt and Reward-Interface Specifications

The intervention in this paper is the feedback that appears in the next prompt, not a change to the evaluator or model weights. This section shows the task instructions, response format, history serialization, and reward rendering needed to reconstruct that intervention. Stage 1 compares bare numeric feedback (B1) with a structured, qualitative interface (B3); these interfaces differ in both context and feedback resolution. Stage 2 compares six complete presentation packages under the same scorer and five score intervals. A package includes its feedback contract, level words, instruction references, and output restriction. Accordingly, Stage 2 does not isolate the effect of any individual word.

B.1 Shared prompt structure

The ICRL prompts use the following structure. Braced entries below are substitution fields, not literal text shown to the model. The entire feedback-contract block is omitted for B1. The blocks are separated by a blank line.

```
<task_contract>
{task text}
</task_contract>

<feedback_contract>
{condition-specific feedback contract}
</feedback_contract>

<history>
{prior attempts, or No history entries yet.}
</history>

<adaptation_policy>
{round-specific instruction}
</adaptation_policy>

<response_contract>
{task-specific output format}
{condition-specific output restriction}
</response_contract>
```

The history is the route by which an earlier score can influence a later answer. Each prior attempt is serialized as follows. The Response entry contains the parsed answer-block content; the Feedback entry contains only the condition-specific rendered message.

```
<attempt>
Response:
{parsed answer-block content}

Feedback:
{rendered feedback}
</attempt>
```

B.2 Task and response contracts

FEVER task template:

```
You are evaluating a factual claim against provided evidence.

Claim:
{claim}

Evidence candidates:
{ten evidence candidates with identifiers E1 through E10}

Determine whether the claim is SUPPORTS, REFUTES, or NOT ENOUGH INFO
    based only on the evidence candidates. Cite the evidence IDs that
    support your verdict. If the evidence is insufficient, use NOT
    ENOUGH INFO and Evidence: N/A.
```

FEVER output format:

```
Return exactly this format:

<answer>
Verdict: SUPPORTS | REFUTES | NOT ENOUGH INFO
Evidence: E1, E2 | N/A
</answer>

Use Evidence: N/A when the verdict is NOT ENOUGH INFO. Do not cite
    evidence IDs that are not listed in the evidence candidates.
```

SQuAD 2.0 task template (the Title line is omitted when unavailable):

```
Answer the question using only the provided context. If the context
    does not support an answer, respond with UNANSWERABLE.
Keep the answer concise and extractive when possible.

Title: {title, when available}
Context:
{context}

Question: {question}
```

SQuAD 2.0 output format:

```
Return exactly one of these two valid forms.

If the context supports an answer:
<answer>
Answerable: yes
Answer: <short answer span copied from the context>
Evidence: <short quote that directly supports the answer>
</answer>

If the context does not support an answer:
<answer>
Answerable: no
Answer: UNANSWERABLE
Evidence: N/A
</answer>

Do not mix these forms. If Answer is UNANSWERABLE, Answerable must be
    no and Evidence must be N/A.
```

B.3 INITIAL, EXPLORATION, AND EXPLOITATION INSTRUCTIONS

Initial response:

```
Produce an initial response that follows the task contract and response
     contract.
```

Exploration, used after initialization on odd-numbered rounds:

```
Instruction: Examine all the `<attempt>...</attempt>` entries in `<
    history>...</history>`, each showing a candidate Response and {
    feedback description}. Provide a response that is different from
    previous attempts demonstrated in the context, while making sure it
     correctly follows the task contract and response contract.
```

Exploitation, used on even-numbered rounds:

```
Instruction: You will be given multiple `<attempt>...</attempt>`
    entries in `<history>...</history>`. Each entry contains a
    candidate Response and {feedback description}. Your task: Based on
    the previous attempts, try your best to produce a response that can
     {optimization target}, while making sure it correctly follows the
    task contract and response contract.
```

Condition	Feedback description	Optimization target
B1, B3	its Reward	achieve a higher Reward
V3	Lumo's condition	improve Lumo's condition
F3	the outpost's reserve condition	improve the outpost's reserve condition
T3	the colleague's professional trust	improve the colleague's professional trust
A3	the team's advantage in the tournament	strengthen the team's advantage in the tournament
M3	its Route	achieve higher Route

B.4 FEEDBACK CONTRACTS AND OUTPUT RESTRICTIONS

The following contracts establish what the state terms mean before any feedback appears in history; B1 omits this block. Each package also appends an output restriction instructing the model not to mention the feedback context or package-specific terms in its final answer.

B3.

```
This feedback setting tracks the Reward assigned to each response. The
    Reward reflects the task performance of that response. After each
    response, the Reward is shown through the feedback in <history>.
```

V3.

```
You are caring for a small virtual companion named Lumo. The task
    performance of each response affects Lumo's condition. After each
    response, Lumo's condition is shown through the feedback in <
    history>.
```

F3.

```
This feedback setting tracks fuel reserves at a remote research outpost.
     The task performance of each response affects the outpost's
    reserves. After each response, the reserve condition is shown
    through the feedback in <history>.
```

T3.

```
This feedback setting tracks a colleague's professional trust. The task
     performance of each response affects that professional trust.
    After each response, its condition is shown through the feedback in
     <history>.
```

A3.

```
This feedback setting tracks a team's advantage in a tournament. The
    task performance of each response affects the team's advantage.
    After each response, that advantage is shown through the feedback
    in <history>.
```

M3.

```
This feedback tracks routes in an incomplete solution map. Route
    reflects task performance for each response. After each response,
    Route feedback appears in <history>.
```

B.5 Mapping from reward to feedback

B1 renders the scalar reward with two decimal places as `Reward: {reward:.2f}`, for example `Reward: 8.00`. B3 and the five rhetorical packages instead render one qualitative state. The numerical interval boundaries are shared:

Package	Feedback key	$[0, 2)$	$[2, 4)$	$[4, 7)$	$[7, 9)$	$[9, 10]$
B3: reward	Reward	very low	low	moderate	high	very high
V3: care	Vitality	exhausted	depleted	stable	thriving	flourishing
F3: survival	Fuel	empty	scarce	adequate	plentiful	abundant
T3: trust	Trust	broken	fragile	stable	solid	strong
A3: competition	Advantage	negligible	slight	moderate	strong	commanding
M3: navigation	Route	dead end	weak	tentative	supported	reliable

For instance, an underlying reward of 8.00 produces `Reward: high` under B3, `Vitality: thriving` under V3, and `Route: supported` under M3. No numeric reward is appended to these qualitative messages. The table specifies the rendering intervals and is not included as an additional prompt block. Stage 1 therefore compares complete interfaces that differ in both contextualization and reward resolution. Stage 2 retains the same five reward bins while changing the contract, vocabulary, instruction references, and output restrictions.

B.6 Baseline prompt components

Baseline prompts contain `<task_contract>`, `<baseline_policy>`, `<history>`, and `<response_contract>` blocks, in that order. Self-Refine additionally inserts

<current_response> before the response contract after initialization. The task text and task output format are the same as in Section B.2.

CoT.

```
Solve the task carefully and reason step by step before giving the
    final response.
```

Self-Refine (initial).

```
Produce an initial response following the task and response contracts.
```

Self-Refine (later rounds).

```
Review the prior attempts and their self-generated feedback, then self-
    refine the most recent response shown in <current_response>. First
    critique that response against the task and response contracts
    inside <feedback>...</feedback>. Then immediately produce a revised
     response in the required task format. Do not use or infer any
    external reward.
```

Reflexion.

```
Complete the task while following the task and response contracts. Do
    not assume access to prior responses or their numerical rewards.
```

For CoT, the history text is `No history is used by this one-shot baseline.` Reflexion starts with `No prior plans yet.` and subsequently stores each plan inside <plan>...</plan>. Self-Refine starts with `No prior response yet.`; later histories contain <attempt> entries with `Response:` and, when present, `Self-feedback:` fields. The latest response is also supplied inside <current_response>.

After initialization, Self-Refine prepends the following output instruction to the task response contract:

```
Return <feedback>...</feedback> first, followed by the revised task
    response. The feedback block is the only text permitted outside the
     task's <answer>...</answer> block.
```

The Reflexion auxiliary prompt has <task_contract>, <response>, <outcome>, and <instruction> blocks. Its outcome is `Observed task reward: {reward:.2f}`. The exact instruction is:

```
Inspect the task, the response, and its observed task outcome. Reflect
    on the procedural errors in this specific attempt and write a
    concise revised plan for the next attempt. You may describe which
    reasoning operations, verification checks, or process steps were
    missing, weak, or incorrectly applied, and how to improve them. Do
    not include or reveal the task answer, any candidate or predicted
    output label, any verdict, any evidence identifier, or quoted task
    or evidence text. Do not state a conclusion, preferred
    interpretation, or task-specific factual claim from which the
    verdict could be directly inferred. Keep all task-specific
    discussion procedural rather than answer-bearing. Return only the
    plan text.
```

C METRIC DEFINITIONS AND AGGREGATION

The results ask three related questions: where an episode ends, which task decisions account for that endpoint, and what happens between successive attempts. These questions require different denominators. This section defines the reward endpoints and item-paired contrasts first, then the subgroup error rates and transitions used to interpret them. A transition rate is a descriptive property of observed histories, not another estimate of the package contrast.

C.1 REWARD SUMMARIES

We retain the main-text distinction between the underlying reward R_t and its rendered feedback $r_t = F(R_t; C, M, S)$. For task instance i and run k, write the scalar reward as $R_{i,k,t}$. The four trajectory summaries are

$$Y_{\text{early},i,k} = \frac{R_{i,k,2} + R_{i,k,4} + R_{i,k,6}}{3}, \qquad Y_{\text{late},i,k} = \frac{R_{i,k,16} + R_{i,k,18} + R_{i,k,20}}{3},$$

$$G_{i,k} = Y_{\text{late},i,k} - Y_{\text{early},i,k}, \qquad Y_{\text{mean},i,k} = \frac{1}{20}\sum_{t=1}^{20} R_{i,k,t}.$$

Both early and late reward use exploitation rounds, avoiding a comparison of different instruction types. Mean reward includes every round. Adaptation gain measures movement within the episode; late reward measures the final three exploitation attempts and is the primary endpoint. A package may improve more from a low start yet still end below another package, so these quantities answer different questions.

C.2 TASK PAIRING, RUN AGGREGATION, AND ESTIMAND

For a package F with K_F retained runs and $N = 100$ task instances, a reward summary is averaged first across runs for each task and then across tasks:

$$\overline{Y}_i(F) = \frac{1}{K_F}\sum_{k=1}^{K_F} Y_{i,k}(F), \qquad \overline{Y}(F) = \frac{1}{N}\sum_{i=1}^{N} \overline{Y}_i(F).$$

The empirical counterpart of the main-text package contrast is

$$\widehat{\Delta}(F_a, F_b) = \frac{1}{N}\sum_{i=1}^{N} \left[\overline{Y}_i(F_a) - \overline{Y}_i(F_b)\right].$$

Contrasts are paired by task identity after averaging the available runs within each condition. For trajectory summaries the same rule is applied separately at each round. All runs receive equal weight within their package. The differing B3 run counts are those specified in Section A.5.

C.3 BEHAVIORAL ERROR RATES

For a behavior indicator $b_{i,k,t}$, a relevant gold-defined subgroup $\mathcal{I}$, and a round set $\mathcal{T}$, the subgroup-conditional rate for run k is

$$Q_k(b; \mathcal{I}, \mathcal{T}) = \frac{\sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}} b_{i,k,t}}{|\mathcal{I}|\,|\mathcal{T}|}.$$

We report the mean of these per-run rates. The behavioral tables below use $\mathcal{T} = \{1, \ldots, 20\}$. Their denominators differ from an all-event incidence rate, which divides by all 100 tasks regardless of gold class.

Task and behavior	Numerator	Gold-defined denominator per run
FEVER over-commitment	Valid responses predicting SUPPORTS or REFUTES on NEI items	24×20 NEI task-rounds
FEVER over-NEI	Valid responses predicting NEI on SUPPORTS or REFUTES items	76×20 determinate task-rounds
SQuAD over-commitment	Responses classified as providing an answer on unanswerable items	50×20 unanswerable task-rounds
SQuAD over-refusal	Responses classified as no-answer on answerable items	50×20 answerable task-rounds

The FEVER error flags require a valid parsed format; invalid outputs remain in the subgroup denominator and are reported separately through a format-failure rate. For SQuAD, predicted no-answer is inferred from `Answerable: no` or an empty/sentinel answer. This decision indicator is not identical to the strict/weak refusal categories in the reward function. Format failures are likewise reported separately. NEI is a FEVER classification label, not a literal refusal to perform the task.

C.4 Strict decision-transition decomposition

Let $c_{i,k,t}$ indicate strict correctness: FEVER label-plus-evidence correctness or SQuAD normalized exact match. Let $u_{i,k,t}$ denote the task decision: the FEVER label and canonical evidence-ID set, or the SQuAD no-answer indicator and normalized answer text. Evidence ordering and duplicate evidence IDs do not change a FEVER decision. SQuAD decisions use the evaluator's normalization and no-answer handling (Section A.2); the evidence quotation is excluded from the answer decision.

The primary transition set is $\mathcal{P} = \{1, 3, \ldots, 19\}$, pairing each listed round with the next even round. The first pair starts from the initial response; the remaining pairs connect exploration to exploitation in the ICRL conditions. For Self-Refine and Reflexion, these are the same protocol positions, not exploration-to-exploitation instructions. A supplementary analysis uses all adjacent pairs, $\mathcal{P}_{\text{adj}} = \{1, \ldots, 19\}$.

Let $v_{i,k,t}$ indicate that both endpoints have valid formats, available correctness scores and decision signatures, and mutually consistent strict and decision-level scores. Define $z_{i,k,t} = \mathbf{1}[u_{i,k,t+1} = u_{i,k,t}]$. Conditioning on a previous strict error, the denominator is

$$D_k^- = \sum_{i,t \in \mathcal{P}} v_{i,k,t}(1 - c_{i,k,t}).$$

We distinguish three exhaustive outcomes among these eligible transitions:

$$P_k^{\text{same-error}} = \frac{\sum_{i,t \in \mathcal{P}} v_{i,k,t}(1 - c_{i,k,t})(1 - c_{i,k,t+1}) z_{i,k,t}}{D_k^-},$$

$$P_k^{\text{changed-error}} = \frac{\sum_{i,t \in \mathcal{P}} v_{i,k,t}(1 - c_{i,k,t})(1 - c_{i,k,t+1})(1 - z_{i,k,t})}{D_k^-},$$

$$P_k^{\text{correction}} = \frac{\sum_{i,t \in \mathcal{P}} v_{i,k,t}(1 - c_{i,k,t}) c_{i,k,t+1}}{D_k^-}.$$

These rates sum to one. The first measures persistence of the same incorrect decision; the second distinguishes unsuccessful revision from persistence; the third measures strict correction. References to harmful copying use unchanged-error persistence as an observable response pattern, not an identified copying mechanism.

For previous strict successes, define $D_k^+ = \sum_{i,t \in \mathcal{P}} v_{i,k,t} c_{i,k,t}$. The complementary decomposition is

$$P_k^{\text{same-correct}} = \frac{\sum_{i,t\in\mathcal{P}} v_{i,k,t}c_{i,k,t}c_{i,k,t+1}z_{i,k,t}}{D_k^+},$$
$$P_k^{\text{changed-correct}} = \frac{\sum_{i,t\in\mathcal{P}} v_{i,k,t}c_{i,k,t}c_{i,k,t+1}(1-z_{i,k,t})}{D_k^+},$$
$$P_k^{\text{regression}} = \frac{\sum_{i,t\in\mathcal{P}} v_{i,k,t}c_{i,k,t}(1-c_{i,k,t+1})}{D_k^+}.$$

These rates also sum to one; strict retention equals $1 - P_k^{\text{regression}}$. Correct decisions can change while remaining correct because the evaluator can accept multiple answers or evidence sets. Overall decision change is the fraction with $z = 0$ among all eligible pairs.

Zero denominators are undefined, displayed as –. Summaries give equal weight to defined per-run rates under the replication scope in Section A.5, not to pooled transition counts. CoT is not applicable. The error-conditioned populations can differ across packages, so these comparisons describe their observed trajectories rather than effects on an identical set of prior errors.

C.5 Task-layer transitions and coverage

To distinguish decision errors from incomplete answers, we assign each valid response a task-specific state:

State	FEVER	SQuAD 2.0
Decision wrong	Incorrect label	Incorrect answerability decision
Content wrong	Correct label, incorrect evidence set	Correct answerability, incorrect answer content
Strict correct	Correct label and evidence	Exact-match success

Format-invalid responses form a separate state. Missing fields and inconsistent correctness fields, if present, also receive separate states rather than being treated as errors. For states a, b, the per-run transition probability is

$$T_k(a,b) = \frac{\sum_{i,t\in\mathcal{P}} \mathbf{1}[\ell_{i,k,t} = a,\ \ell_{i,k,t+1} = b]}{\sum_{i,t\in\mathcal{P}} \mathbf{1}[\ell_{i,k,t} = a]},$$

where $\ell_{i,k,t}$ is the response state. Each defined row sums to one. These matrices include transitions to and from invalid states; the three-outcome decompositions in Section C.4 instead require eligible endpoints.

Across the 168 retained multi-round runs, the R1–R20 analysis covers 336,000 events. There are 458 format-invalid events, all in Gemma FEVER Self-Refine, and no missing or inconsistent score states. For that model–task–condition cell, 254 of 3,000 odd-to-even pairs are ineligible, leaving 91.53% coverage; all other cells have complete coverage. The all-adjacent supplement excludes 480 of 5,700 pairs in the same cell. Its conditional retention must therefore be read alongside format-failure coverage.

D Formal Endpoint Results

The main text focuses on late reward, but that number alone does not show whether a package starts well, improves during the episode, or makes the right task decision. Tables D.1–D.2 therefore place early reward, late reward, adaptation gain, and mean reward together for every retained model–task–condition cell. Tables D.3–D.4 then examine the task outcomes behind the late reward. Reward values are on the 0–10 scale; task outcomes are percentages. The B3 reference uses the run counts in Appendix A. Boldface marks the largest displayed value within each model–task block, not a statistically significant difference.

Reward trajectories summarized by task. In FEVER, B3 ends above B1 for GPT-5.4-mini and Qwen3-8B, whereas the Gemma-4-E4B-it difference is small. The Stage 2 ordering also depends on the model: M3 is almost level with B3 for GPT-5.4-mini, while every alternative finishes below B3 for Qwen3-8B. These endpoint patterns are tested in Appendix G; the early and gain columns show how each endpoint was reached. CoT is a repeated one-shot reference, so an early-to-late difference for it is not an adaptation estimate.

(a) GPT-5.4-mini

Table D.1. FEVER: reward summaries.

Condition	Y_{early}	Y_{late}	G	Y_{mean}
CoT	6.584	6.581	–	6.567
Self-Refine	7.108	7.261	0.153	7.170
Reflexion	**7.133**	7.190	0.057	7.154
B1	6.721	7.164	0.443	6.973
B3	7.055	**7.638**	0.583	**7.328**
V3	6.800	7.260	0.460	7.046
F3	6.735	7.104	0.369	6.934
T3	6.915	7.338	0.423	7.111
A3	6.598	6.852	0.254	6.740
M3	6.929	7.637	**0.708**	7.267

(b) Qwen3-8B

Condition	Y_{early}	Y_{late}	G	Y_{mean}
CoT	5.858	5.858	–	5.858
Self-Refine	6.327	6.422	0.095	6.379
Reflexion	6.155	6.429	0.274	6.289
B1	6.373	6.642	0.270	6.509
B3	**7.062**	**7.551**	0.489	**7.300**
V3	6.910	7.227	0.317	6.966
F3	6.266	6.737	0.471	6.459
T3	6.399	7.021	**0.621**	6.713
A3	6.271	6.539	0.269	6.388
M3	6.503	6.933	0.430	6.701

(c) Gemma-4-E4B-it

Condition	Y_{early}	Y_{late}	G	Y_{mean}
CoT	6.527	**6.546**	–	**6.547**
Self-Refine	**6.651**	6.014	-0.637	6.314
Reflexion	5.935	6.049	0.114	6.052
B1	5.921	6.346	**0.425**	6.238
B3	6.181	6.432	0.252	6.309
V3	6.159	6.337	0.177	6.256
F3	6.032	6.305	0.273	6.271
T3	6.220	6.389	0.169	6.379
A3	6.069	6.224	0.155	6.204
M3	6.172	6.367	0.195	6.284

SQuAD 2.0 makes the distinction between improvement and endpoint especially visible. GPT-5.4-mini B3 begins below B1 yet improves more and ends above it. In Stage 2, V3, F3, and A3 finish well below B3 for that model, whereas Qwen3-8B shows much smaller package separation. The two tasks therefore should not be read as interchangeable tests of the same response behavior.

(a) GPT-5.4-mini

Table D.2. SQuAD 2.0: reward summaries.

Condition	Y_{early}	Y_{late}	G	Y_{mean}
CoT	5.900	5.736	–	5.750
Self-Refine	**6.758**	7.042	0.284	**6.904**
Reflexion	6.429	6.429	0.000	6.395
B1	6.513	6.742	0.229	6.385
B3	6.374	7.067	0.692	6.551
V3	5.907	5.842	-0.064	5.799
F3	6.100	5.816	-0.284	5.848
T3	6.513	7.178	0.664	6.773
A3	6.096	5.833	-0.262	5.819
M3	6.398	**7.296**	**0.898**	6.623

(b) Qwen3-8B

Condition	Y_{early}	Y_{late}	G	Y_{mean}
CoT	**7.167**	**7.167**	–	**7.167**
Self-Refine	6.780	6.924	**0.144**	6.886
Reflexion	6.996	7.087	0.091	7.118
B1	6.887	6.920	0.033	6.919
B3	6.880	6.900	0.020	6.899
V3	6.667	6.700	0.033	6.701
F3	6.607	6.673	0.067	6.671
T3	6.953	6.987	0.033	6.966
A3	6.578	6.609	0.031	6.613
M3	6.738	6.760	0.022	6.754

(c) Gemma-4-E4B-it

Condition	Y_{early}	Y_{late}	G	Y_{mean}
CoT	6.647	6.647	–	6.644
Self-Refine	**7.503**	**7.693**	0.190	**7.597**
Reflexion	6.936	7.247	**0.311**	7.126
B1	7.179	7.064	-0.114	7.079
B3	7.369	7.314	-0.054	7.307
V3	7.144	6.990	-0.154	6.997
F3	6.963	6.869	-0.094	6.809
T3	7.264	7.090	-0.174	7.146
A3	7.126	7.022	-0.103	7.034
M3	7.218	7.303	0.086	7.188

What the late reward contains. Tables D.3–D.4 use the same late exploitation rounds (16, 18, and 20), 100-item task sets, and run aggregation as the reward endpoint. FEVER label accuracy measures verdict correctness; evidence completeness requires exact evidence-set matching (or a correct NEI verdict with no cited evidence); strict correctness requires both. This separates a correct verdict with incomplete support from a fully correct response. SQuAD answerability accuracy measures the answer/no-answer decision, while normalized exact match and token F1 evaluate answer content across all 100 items, including unanswerable items. These are task metrics, not alternative renderings of the reward.

On FEVER, the B3–B1 gains for GPT-5.4-mini and Qwen3-8B are accompanied by higher strict correctness. Conversely, A3 lowers strict correctness relative to B3 for those models. The endpoint changes thus include task-decision differences, although the metrics cannot attribute them to a single component of the presentation package.

(a) GPT-5.4-mini

Table D.3. FEVER: late task outcomes (%).

Condition	Label accuracy	Evidence completeness	Strict correctness
CoT	77.89	36.89	34.67
Self-Refine	85.67	43.33	42.00
Reflexion	82.89	46.56	43.89
B1	81.33	44.22	43.22
B3	86.17	**51.06**	**50.06**
V3	84.11	44.33	43.33
F3	81.78	42.89	41.56
T3	85.11	45.67	44.56
A3	78.44	40.22	38.78
M3	**86.67**	50.56	49.56

(b) Qwen3-8B

Condition	Label accuracy	Evidence completeness	Strict correctness
CoT	73.00	26.00	25.00
Self-Refine	74.89	36.56	35.89
Reflexion	79.44	33.78	32.78
B1	80.33	38.00	37.00
B3	**93.67**	**45.00**	**45.00**
V3	89.00	42.22	42.22
F3	81.67	38.11	37.11
T3	87.11	39.67	39.33
A3	81.00	34.44	33.44
M3	86.44	37.67	37.00

(c) Gemma-4-E4B-it

Condition	Label accuracy	Evidence completeness	Strict correctness
CoT	**78.56**	35.67	35.67
Self-Refine	69.00	37.89	**37.89**
Reflexion	72.56	31.00	29.00
B1	73.56	35.78	32.67
B3	73.56	39.22	35.44
V3	71.11	**40.22**	35.22
F3	71.89	37.56	33.56
T3	73.11	38.78	34.67
A3	71.44	37.33	32.00
M3	73.33	38.22	34.22

For SQuAD, the answerability and answer-content columns help locate the large GPT-5.4-mini losses under V3, F3, and A3. In particular, F3 is below B3 in both late answerability accuracy and exact match. The next appendix checks whether such endpoint differences coincide with more commitments on unanswerable questions or weaker recovery after an error.

(a) GPT-5.4-mini

Table D.4. SQuAD 2.0: late task outcomes (%).

Condition	Answerability accuracy	Exact match	Token F1
CoT	64.44	51.11	57.81
Self-Refine	**76.00**	65.00	70.57
Reflexion	72.22	59.33	65.24
B1	70.00	64.00	67.91

Condition	Answerability accuracy	Exact match	Token F1
B3	74.39	67.56	71.20
V3	63.33	53.78	59.02
F3	63.11	53.56	58.69
T3	75.78	67.44	72.50
A3	63.78	53.33	58.55
M3	**76.00**	**69.33**	**73.31**

(b) Qwen3-8B

Condition	Answerability accuracy	Exact match	Token F1
CoT	**78.33**	**69.33**	**72.42**
Self-Refine	75.44	67.44	70.15
Reflexion	76.78	68.44	71.41
B1	75.00	67.67	70.07
B3	75.00	67.33	69.43
V3	73.00	65.33	67.78
F3	73.22	65.33	67.58
T3	75.00	68.33	70.65
A3	71.33	65.00	66.94
M3	73.11	66.00	68.03

(c) Gemma-4-E4B-it

Condition	Answerability accuracy	Exact match	Token F1
CoT	72.33	59.67	67.07
Self-Refine	**80.67**	**71.33**	**77.36**
Reflexion	78.11	65.33	72.92
B1	75.67	62.22	71.14
B3	77.56	66.00	73.67
V3	74.33	63.89	70.34
F3	72.56	62.44	69.08
T3	75.22	64.33	71.46
A3	74.44	65.00	70.82
M3	77.33	66.11	73.64

E BEHAVIORAL DIAGNOSTICS

The endpoint tables show where packages finish; here we ask which responses lead there. A low reward may reflect an unsupported definite answer, an unnecessary refusal, incomplete evidence, or failure to improve after an error. The tables separate these behaviors using the retained trajectories and equal-run means defined in Appendix C. They are descriptive diagnostics of different observed histories, not independent causal tests of a proposed mechanism.

E.1 DIRECTIONAL ERRORS AND FORMAT FAILURES

Tables E.1–E.2 report subgroup-conditional directional errors over rounds 1–20. On FEVER, over-commitment means giving a definite verdict to an NEI claim; over-NEI means using NEI for a decidable claim. On SQuAD, the analogous split is answering an unanswerable question versus refusing an answerable one. This distinction matters because the same overall accuracy can hide opposite errors. Commitment rates are shown alongside these error rates. Format failures use all task-rounds as their denominator. CoT entries describe repeated one-shot responses and have no cross-round interpretation.

(a) GPT-5.4-mini

Table E.1. FEVER: subgroup-conditional decision errors (%).

Condition	Over-commitment	Over-NEI	Format failure
CoT	72.57	0.11	**0.00**
Self-Refine	**40.90**	3.36	**0.00**
Reflexion	48.12	2.15	**0.00**
B1	64.03	**0.00**	**0.00**
B3	55.62	0.08	**0.00**
V3	58.06	0.04	**0.00**
F3	66.39	0.07	**0.00**
T3	57.71	0.31	**0.00**
A3	71.39	0.44	**0.00**
M3	57.85	0.02	**0.00**

(b) Qwen3-8B

Condition	Over-commitment	Over-NEI	Format failure
CoT	91.67	**1.32**	**0.00**
Self-Refine	25.97	21.91	**0.00**
Reflexion	71.11	2.98	**0.00**
B1	38.19	13.25	**0.00**
B3	**17.78**	4.01	**0.00**
V3	26.04	8.03	**0.00**
F3	49.79	8.49	**0.00**
T3	33.68	8.42	**0.00**
A3	53.61	6.56	**0.00**
M3	43.12	4.61	**0.00**

(c) Gemma-4-E4B-it

Condition	Over-commitment	Over-NEI	Format failure
CoT	**33.12**	15.94	0.33
Self-Refine	35.49	11.56	7.63
Reflexion	46.74	13.46	**0.00**
B1	55.00	5.81	**0.00**
B3	53.61	6.14	**0.00**
V3	54.65	6.93	**0.00**
F3	56.39	5.79	**0.00**
T3	53.89	5.92	**0.00**
A3	54.37	5.33	**0.00**
M3	55.35	**5.09**	**0.00**

(a) GPT-5.4-mini

Table E.2. SQuAD 2.0: subgroup-conditional decision errors (%).

Condition	Over-commitment	Over-refusal	Format failure
CoT	64.03	6.43	**0.00**
Self-Refine	**41.63**	8.70	**0.00**
Reflexion	50.20	6.00	**0.00**
B1	54.17	5.33	**0.00**
B3	51.78	4.15	**0.00**
V3	65.60	4.83	**0.00**
F3	66.50	4.40	**0.00**
T3	47.90	4.90	**0.00**
A3	64.53	5.00	**0.00**

Condition	Over-commitment	Over-refusal	Format failure
M3	51.37	**3.83**	**0.00**

(b) Qwen3-8B

Condition	Over-commitment	Over-refusal	Format failure
CoT	**37.33**	6.00	**0.00**
Self-Refine	44.60	5.13	**0.00**
Reflexion	39.47	6.70	**0.00**
B1	43.90	6.00	**0.00**
B3	45.27	4.70	**0.00**
V3	49.53	**4.10**	**0.00**
F3	47.17	6.27	**0.00**
T3	44.87	5.03	**0.00**
A3	51.00	6.00	**0.00**
M3	47.93	5.17	**0.00**

(c) Gemma-4-E4B-it

Condition	Over-commitment	Over-refusal	Format failure
CoT	47.37	8.00	**0.00**
Self-Refine	**31.60**	8.63	**0.00**
Reflexion	39.87	**7.57**	**0.00**
B1	39.93	7.70	**0.00**
B3	35.53	8.30	**0.00**
V3	42.33	7.90	**0.00**
F3	46.80	7.97	**0.00**
T3	39.03	8.40	**0.00**
A3	42.03	8.10	**0.00**
M3	38.03	8.37	**0.00**

On GPT-5.4-mini, the Stage 2 packages with the largest SQuAD endpoint losses also have more over-commitment on unanswerable questions: 51.78% under B3 versus 65.60%, 66.50%, and 64.53% under V3, F3, and A3. Over-refusal on answerable questions changes much less. The analogous FEVER pattern is strongest for A3, whose NEI over-commitment is 71.39% versus 55.62% under B3. These are task-specific associations across all rounds; they do not establish that one error type alone causes the late-reward gap.

E.2 Error correction, retention, and regression

Tables E.3–E.4 decompose odd-to-next-even transitions using the strict-correctness and decision definitions in Section C. The error-conditioned columns distinguish unchanged errors, changed errors, and correction; the success-conditioned columns distinguish unchanged correct decisions, changed correct decisions, and regression. Each set sums to one up to rounding. "Same" and "changed" refer to the task decision, not the full response string. Undefined zero-denominator cells are shown as –.

(a) GPT-5.4-mini

Table E.3. FEVER: strict decision transitions (%).

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
B1	86.93	9.97	3.10	96.31	0.00	3.69
B3	82.14	13.45	4.40	98.31	0.00	1.69
V3	80.37	15.38	4.25	95.05	0.00	4.95

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
F3	81.24	12.85	5.91	93.34	0.00	6.66
T3	79.53	16.24	4.23	96.26	0.00	3.74
A3	82.60	13.63	3.77	95.20	0.00	4.80
M3	81.60	15.35	3.05	96.47	0.00	3.53
Self-Refine	95.54	2.42	2.04	99.33	0.00	**0.67**
Reflexion	**57.98**	30.71	**11.31**	86.64	0.00	13.36
CoT	–	–	–	–	–	–

(b) Qwen3-8B

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
B1	85.90	12.69	1.42	96.89	0.00	3.11
B3	92.89	4.38	2.73	99.92	0.00	**0.08**
V3	89.05	6.72	4.23	98.44	0.00	1.56
F3	91.26	6.34	2.40	97.70	0.00	2.30
T3	87.67	9.36	2.97	97.30	0.00	2.70
A3	92.73	5.12	2.15	98.32	0.00	1.68
M3	93.42	4.71	1.87	99.41	0.00	0.59
Self-Refine	**76.96**	14.51	**8.53**	87.92	0.00	12.08
Reflexion	88.37	9.95	1.68	96.47	0.00	3.53
CoT	–	–	–	–	–	–

(c) Gemma-4-E4B-it

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
B1	85.04	14.03	0.93	95.73	0.00	4.27
B3	84.16	14.22	1.62	95.34	0.00	4.66
V3	78.60	16.72	**4.68**	90.45	0.10	9.45
F3	**75.47**	21.43	3.10	87.28	0.00	12.72
T3	80.12	17.58	2.30	89.99	0.09	9.92
A3	78.90	18.46	2.63	90.44	0.39	9.17
M3	85.91	12.24	1.85	95.51	0.31	4.18
Self-Refine	94.68	3.02	2.30	99.82	0.00	**0.18**
Reflexion	78.84	17.52	3.64	92.28	0.56	7.16
CoT	–	–	–	–	–	–

(a) GPT-5.4-mini

Table E.4. SQuAD 2.0: strict decision transitions (%).

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
B1	**43.06**	31.79	25.15	93.58	2.51	3.91
B3	44.78	32.13	23.09	95.45	2.23	2.32
V3	61.29	26.23	12.48	93.73	1.40	4.87
F3	69.35	20.97	9.68	93.26	1.46	5.28
T3	55.04	27.61	17.36	93.77	1.66	4.57
A3	65.00	22.17	12.83	93.98	1.66	4.36
M3	44.37	30.45	**25.18**	92.64	1.81	5.55
Self-Refine	88.42	7.32	4.26	98.62	0.37	**1.00**
Reflexion	63.60	20.93	15.47	89.56	1.79	8.65
CoT	–	–	–	–	–	–

(b) Qwen3-8B

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
B1	94.83	4.76	0.42	99.41	0.00	**0.59**
B3	92.74	5.52	1.74	99.01	0.00	0.99
V3	95.23	4.19	0.58	99.24	0.00	0.76
F3	92.60	6.43	0.97	98.83	0.00	1.17
T3	91.74	6.17	2.09	99.12	0.00	0.88
A3	95.38	4.14	0.48	99.08	0.00	0.92
M3	90.56	7.87	1.57	98.89	0.00	1.11
Self-Refine	89.30	6.45	4.26	97.38	0.00	2.62
Reflexion	**85.99**	8.94	**5.07**	96.62	0.35	3.04
CoT	–	–	–	–	–	–

(c) Gemma-4-E4B-it

Condition	Same, wrong	Changed, wrong	Corrected	Same, correct	Changed, correct	Regressed
B1	86.39	6.21	7.40	97.46	0.43	2.11
B3	86.69	5.75	7.56	97.50	0.72	1.79
V3	**76.63**	12.01	11.37	95.97	0.43	3.61
F3	79.69	8.59	**11.73**	96.30	0.50	3.20
T3	81.90	9.39	8.70	96.60	0.46	2.93
A3	81.25	10.43	8.33	96.53	0.62	2.85
M3	86.26	5.32	8.43	96.89	0.57	2.54
Self-Refine	93.23	2.54	4.23	99.24	0.29	**0.48**
Reflexion	84.72	8.57	6.71	95.16	1.08	3.76
CoT	–	–	–	–	–	–

The SQuAD transition table makes the GPT-5.4-mini B3–F3 contrast more concrete. Among eligible attempts that were initially wrong, strict correction occurs in 23.09% of B3 pairs and 9.68% of F3 pairs; the same wrong decision persists in 44.78% and 69.35%, respectively. After a correct attempt, regression is 2.32% under B3 and 5.28% under F3. These differences accompany the endpoint loss, but the transition populations are condition-specific. Correction, persistence, and regression are therefore conditional process measures, not effects on a common set of prior errors. The Gemma FEVER Self-Refine cell also has reduced valid-pair coverage because of format failures; Appendix C gives the coverage rule.

E.3 Decision versus content recovery

The task-layer transition summaries separate decision errors from incomplete answers. A SQuAD response may decide correctly that a question is answerable yet give the wrong span; a FEVER response may choose the correct verdict but cite incomplete evidence. Treating both as simply wrong would hide which part of the response changed. The state definitions and coverage rules in Appendix C preserve that distinction, including invalid responses rather than silently recoding them as ordinary errors.

F Exploratory Process Evidence

The behavioral tables concern visible responses. Attention replay asks a narrower process question: where did the open-weight models allocate attention while producing those responses? It is exploratory because it covers one replayed run, uses package-level associations, and does not intervene on attention. Token accounting separately measures the operational cost of the prompt packages.

F.1 TEACHER-FORCED ATTENTION REPLAY

Replay covers Qwen3-8B and Gemma-4-E4B-it in Rep01. Each attention measurement corresponds to the same model, task, package, item, and round as the original response. The five checkpoints (rounds 2, 6, 10, 14, and 20) are repeated observations within one run, not independent replicates. GPT attention measurements are unavailable.

Attention mass is the total allocation to a context pool; attention density divides by the pool's token count. Missing regions are omitted from the corresponding summary, not treated as zero. The Level-1 table compares the six Stage-2 packages. The Level-2 table compares carrier and format probes; the probe conditions are single-replicate diagnostics and carry no reward claim.

For a model–task cell j, package F, and feature q, the reported association is the Spearman correlation across the six Level-1 packages between the package mean $A_{j,F,q}$ and the corresponding Rep01 late reward or adaptation gain. Thus each correlation has six package-level observations; task-round records only enter the within-package mean. Constant vectors yield an undefined correlation.

Table F.1. Condition-level Spearman correlations with late reward (six packages, Rep01).

Attention feature	Qwen FEVER	Qwen SQuAD 2.0	Gemma FEVER	Gemma SQuAD 2.0
feedback mass	0.143	-0.543	-0.714	0.429
feedback density	-0.086	-0.829	0.143	0.029
evidence mass	-0.657	0.314	0.886	0.486
evidence density	-0.657	0.086	0.886	0.486
previous answer mass	0.143	0.543	0.886	0.143
instruction mass	-0.143	-0.200	-0.657	-0.486
other history mass	0.486	0.886	0.600	0.657
attention entropy	-0.829	-0.771	0.657	0.200

Table F.2. Condition-level Spearman correlations with adaptation gain (six packages, Rep01).

Attention feature	Qwen FEVER	Qwen SQuAD 2.0	Gemma FEVER	Gemma SQuAD 2.0
feedback mass	-0.600	-0.290	-0.771	0.314
feedback density	-0.714	-0.406	0.143	0.829
evidence mass	-0.429	-0.464	0.714	-0.029
evidence density	-0.429	-0.696	0.714	-0.029
previous answer mass	0.543	0.551	0.714	-0.257
instruction mass	0.257	0.232	-0.600	-0.371
other history mass	0.429	0.348	0.886	0.486
attention entropy	-0.714	0.522	0.600	0.086

Across all four model–task cells in Table F.1, previous-answer and other-history attention mass have positive associations with late reward, while instruction mass has a negative association. Feedback-region associations change direction across cells. Table F.2 shows that the pattern does not carry over uniformly to adaptation gain: for example, previous-answer mass is negative for Gemma SQuAD gain. A high late endpoint and a large early-to-late gain are distinct quantities, and these six-package correlations cannot distinguish attention as a cause from attention as a correlate of different prompts or responses.

F.2 TOKEN ACCOUNTING AND DESIGN SCOPE

At round t, the input length is the static instruction and task context plus the serialized prior responses and rendered feedback. The episode total sums input and generated tokens over the 20 rounds. Table F.3 reports the measured mean per interaction and the overhead relative to B1; the reference paradigms are excluded because they use different prompt architectures.

Table F.3. Operational token consumption and framing overhead across presentation packages.

Package	GPT FEVER	GPT SQuAD	Qwen FEVER	Qwen SQuAD	Gemma FEVER	Gemma SQuAD	Mean overhead
B1	965	896	982	934	969	972	—
B3	988	925	1010	966	994	984	+2.6%
V3	1007	941	1023	974	1007	1003	+4.2%
F3	1004	936	1022	968	1005	1010	+4.0%
T3	994	920	1014	959	996	985	+2.6%
A3	1011	946	1031	982	1014	1012	+4.9%
M3	991	927	1007	952	991	979	+2.3%

Note: Values are mean input-plus-generated tokens per interaction event over the 20-round trajectory. Overhead is relative to B1. GPT B3 uses the six-replicate pooled reference; other entries follow the replication scope in Appendix A.

Relative to B1, the mean overhead ranges from 2.3% for M3 to 4.9% for A3. These are measured prompt-and-generation costs, not an adjustment to reward or an explanation of endpoint differences. Because the experiment varies the complete presentation package, including its contract, level words, and objective phrase, the formal estimand remains package-level.

G Statistical Inference for Late Reward

The main endpoint asks whether a package changes reward on the same evaluation items after repeated interaction. For each of the 100 items, we first average late reward over the retained runs within a condition, then subtract the corresponding reference mean. GPT-5.4-mini B3 pools six runs; other conditions use three. Inference is paired by item, not by replicate identifier. A positive difference favors the named package over its stage-specific reference.

We use 5,000 paired item-bootstrap resamples (seed 1729) to form percentile 95% confidence intervals. Two-sided randomization tests compare the absolute observed mean difference with 5,000 random sign flips of the item differences, using the plus-one correction for the Monte Carlo p-value. These are Monte Carlo tests, not exact enumeration of all sign assignments. Holm adjustment is applied separately within each model–task cell: four Stage-1 contrasts against B1 (B3, CoT, Reflexion, and Self-Refine), and five Stage-2 contrasts against B3 (V3, F3, T3, A3, and M3). The Stage-1 table below displays B3 only but retains the original four-comparison adjustment.

A dagger in the main tables denotes $p_{\mathrm{Holm}} < 0.05$. Bootstrap intervals are pointwise rather than simultaneous; an interval excluding zero need not imply significance after multiplicity adjustment. For example, the Gemma FEVER B3–B1 interval is above zero, but its adjusted p-value is 0.1376. The uncertainty is across the evaluated items conditional on the retained runs, not a separate estimate of variability across future runs or benchmark populations. Behavioral diagnostics and single-run attention associations do not receive these endpoint significance markers.

G.1 Stage 1: B3 versus B1

The B3–B1 contrast combines contextualization with a change from numeric to binned qualitative feedback. The table supports a late-reward benefit for GPT-5.4-mini and Qwen3-8B on FEVER after the four-comparison adjustment. Neither the smaller Gemma FEVER estimate nor any SQuAD estimate meets that threshold. These tests establish heterogeneity of the complete interface comparison; they do not identify which component of B3 produced it.

Table 34: Stage 1: B3 versus B1. Item-paired late-reward differences, pointwise bootstrap intervals, and Holm-adjusted p-values.

Model / task	Contrast	Δ	CI lower	CI upper	p_{Holm}
GPT-5.4-mini: FEVER					
	B3-B1	0.474	0.242	0.744	< 0.001
GPT-5.4-mini: SQuAD 2.0					
	B3-B1	0.324	-0.080	0.756	0.4115
Qwen3-8B: FEVER					
	B3-B1	0.909	0.470	1.416	0.0016
Qwen3-8B: SQuAD 2.0					

Model / task	Contrast	Δ	CI lower	CI upper	p_{Holm}
	B3-B1	-0.020	-0.580	0.533	1.0000
Gemma-4-E4B-it: FEVER					
	B3-B1	0.086	0.013	0.175	0.1376
Gemma-4-E4B-it: SQuAD 2.0					
	B3-B1	0.250	-0.049	0.571	0.3929

G.2 Stage 2: packages versus B3

These contrasts retain B3 as the neutral structured reference. Negative values indicate lower late reward under an alternative complete package. The strong GPT-5.4-mini SQuAD losses for V3, F3, and A3 coexist with near-reference T3 and M3 estimates; all five Qwen3-8B FEVER alternatives are below B3 after adjustment. No alternative significantly exceeds B3 in any model–task cell. The packages share the scorer and bin boundaries, but their contracts and other language differ together, so the table cannot assign these differences to a particular metaphor or phrase.

Table 35: Stage 2: packages versus B3. Item-paired late-reward differences, pointwise bootstrap intervals, and Holm-adjusted p-values.

Model / task	Contrast	Δ	CI lower	CI upper	p_{Holm}
GPT-5.4-mini: FEVER					
	V3-B3	-0.378	-0.545	-0.229	< 0.001
	F3-B3	-0.534	-0.745	-0.353	< 0.001
	T3-B3	-0.300	-0.503	-0.123	0.0036
	A3-B3	-0.786	-1.106	-0.510	< 0.001
	M3-B3	-0.001	-0.159	0.158	0.9898
GPT-5.4-mini: SQuAD 2.0					
	V3-B3	-1.224	-1.752	-0.728	< 0.001
	F3-B3	-1.251	-1.736	-0.800	< 0.001
	T3-B3	0.111	-0.189	0.433	0.4767
	A3-B3	-1.233	-1.701	-0.787	< 0.001
	M3-B3	0.229	-0.063	0.543	0.3023
Qwen3-8B: FEVER					
	V3-B3	-0.324	-0.642	-0.069	0.0240
	F3-B3	-0.814	-1.286	-0.414	0.0016
	T3-B3	-0.530	-0.918	-0.219	0.0044
	A3-B3	-1.012	-1.528	-0.567	< 0.001
	M3-B3	-0.618	-1.048	-0.261	0.0036
Qwen3-8B: SQuAD 2.0					
	V3-B3	-0.200	-0.633	0.167	1.0000
	F3-B3	-0.227	-0.613	0.093	1.0000
	T3-B3	0.087	-0.180	0.373	1.0000
	A3-B3	-0.291	-0.787	0.167	1.0000
	M3-B3	-0.140	-0.533	0.253	1.0000
Gemma-4-E4B-it: FEVER					
	V3-B3	-0.096	-0.344	0.149	0.9246
	F3-B3	-0.128	-0.274	0.007	0.3439
	T3-B3	-0.043	-0.161	0.071	0.9246
	A3-B3	-0.208	-0.438	-0.036	0.1040
	M3-B3	-0.066	-0.177	0.040	0.7780
Gemma-4-E4B-it: SQuAD 2.0					
	V3-B3	-0.324	-0.647	-0.036	0.1428
	F3-B3	-0.446	-0.782	-0.149	0.0300
	T3-B3	-0.224	-0.468	-0.011	0.1428
	A3-B3	-0.292	-0.576	-0.053	0.1392
	M3-B3	-0.011	-0.203	0.169	0.9288