COUNTERFACTUAL SELF-EVOLVING AGENTS FOR EVIDENCE-GROUNDED REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-play proposer–solver methods improve reasoning by generating tasks and learning from verified solutions. However, for evidence-identifiable tasks, where case-specific evidence and domain knowledge determine a checkable answer, self-play requires generating plausible cases whose answers can be independently verified. We introduce counterfactual self-evolution, which generates counterfactual context for reconsidering the original case. A trainable Proposer constructs targeted evidence edits and describes potential outcome changes with causal explanations. We handcraft an expert-verified counterfactual instruction-tuning dataset to teach the Proposer to generate high-quality counterfactuals across a broad range of action–outcome scenarios. Each counterfactual instruction-tuning example specifies an edit within a defined category and explains its hypothesized causal effect on the decision, teaching the Proposer to reason systematically about what changes and why. We instruction-tune the Proposer on these examples, then formulate a fine-tuning reward that integrates feedback from the Solver and Verifier. Across diverse counterfactual scenarios, this reward favors high-quality counterfactuals and warranted revisions, while penalizing changes that overturn correct decisions. The counterfactual context aims to correct errors and strengthen confidence in correct decisions. Accepted counterfactuals accumulate in memory that supplies in-context evidence to the frozen Solver; the Solver adapts through evolving context rather than weight updates. We apply the framework to clinical reasoning, fact verification, and business reasoning. Our evaluation tracks performance over successive rounds as counterfactual memory grows, including transfer to harder cases. Our method achieves superior results across diverse frontier models.[1]

1 INTRODUCTION

Language-model agents are being applied to tasks ranging from everyday information seeking to high-stakes decision support. For example, they gather evidence from documents to answer multi-hop questions (Yao et al., 2023; Zhou et al., 2026) and interpret case-specific findings to answer medical questions (Tang et al., 2024; Wang et al., 2026). Many reasoning tasks require drawing a verifiable conclusion from case-specific evidence, using domain knowledge and task-specific rules. Examples range from document-based question answering (Zhou et al., 2026) and fact verification (Yao et al., 2023) to clinical reasoning (Tang et al., 2024; Wang et al., 2026) and business reasoning (Chen et al., 2021). Such evidence may be distributed across patient records, multiple documents, or tables and charts. Agents can be knowledge-rich without being process-aware: recalling facts and rules does not ensure that they understand how an environment evolves or how actions affect later observations. Self-evolution offers a way for agents to learn these dynamics from interaction and feedback and adapt their reasoning to changing conditions (Gao et al., 2026). Several high-stake agentic frameworks lack a mechanism for self-evolution: they do not use experience from earlier cases to improve how they solve new ones. Take medical agents as an example, MedAgents uses specialist discussion (Tang et al., 2024), MDAgents uses complexity-adaptive collaboration (Kim et al., 2024), and MedAgent-Pro combines guideline retrieval with visual tools (Wang et al., 2026). These workflows coordinate specialist discussion, retrieval, and tool use within a case, but they do

[1] All results can be reproduced at `anonymous.4open.science/r/evo-med-agent-6381/README.md`.

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.

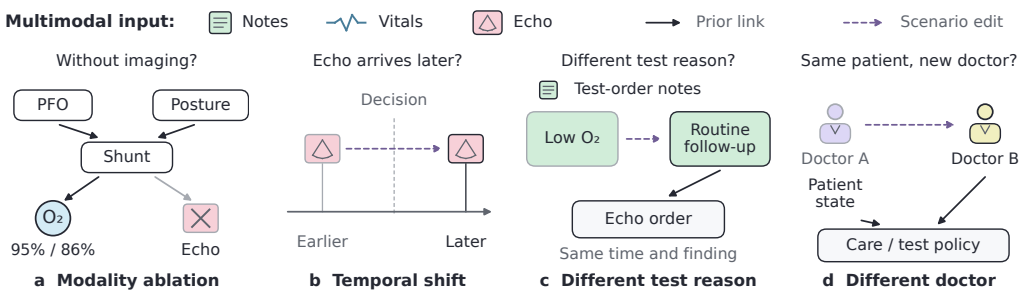


Figure 1: **A multimodal clinical example with diverse counterfactual context.** Using a case from our differential diagnosis evaluation (Strickland et al., 2023), we illustrate four types of counterfactual context for the agent. (a) Removing imaging hides a heart-scan result while retaining oxygen readings and prior knowledge of how an opening in the heart (PFO) affects blood flow. (b) Shifting the scan result's arrival time tests when its evidence is available to the agent. (c) Changing the test reason replaces concern about low oxygen with a routine checkup in the ordering note, keeping the test time and result fixed. (d) Changing the doctor varies the care and test-ordering policy for the same patient. These contexts prompt the agent to reason about which evidence is available, when and why it was collected, and how clinicians' choices shape it. Verified counterfactuals support self-evolution by providing feedback to train the Proposer and reusable context to help the Solver correct errors and preserve sound decisions on later cases.

not, by themselves, provide a mechanism for continual improvement. The broader challenge is to turn experience into reliable supervision without requiring new human labels (Madaan et al., 2023; Zhang et al., 2026). Probing an agent's own decision process can expose weaknesses in how it uses evidence and provide feedback for correcting mistakes while preserving sound decisions.

Counterfactual (CF) reasoning offers a principled way to probe a decision: vary selected aspects of the evidence and test how the conclusion depends on them. Recent research has explored this idea through counterfactual data augmentation, which generates controlled variations of observed examples to improve generalization and reduce reliance on spurious correlations (Feder et al., 2023). Other work uses language models to propose health interventions and augment sensor data (Soumma et al., 2026). More recently, counterfactuals have been incorporated into multi-agent clinical reasoning (You et al., 2026) and self-evolution through training on verified reasoning traces (Fan & Li, 2026). However, these studies do not directly address when counterfactual feedback should correct an agent's original decision and when it should preserve it, without updating the agent's underlying model. A plausible counterfactual that changes an answer does not establish that the original answer was wrong. Many reasoning tasks provide enough evidence to determine a definite answer given domain knowledge and task rules. We call these **evidence-identifiable** tasks. Figure 1 illustrates how counterfactuals test a decision's dependence on evidence availability, timing, test-ordering reasons, and clinician choices. Verified alternatives provide feedback beyond the observed case. Retaining this feedback as context supports self-evolution by helping agents correct recurring reasoning errors while preserving well-supported decisions.

To address these limitations, we propose counterfactual self-evolution (CF-Evolve), a Proposer–Verifier–Solver framework that improves decision-making through evolving counterfactual context (Figure 2). The Proposer generates multiple targeted case variations, the Verifier assesses their plausibility against a knowledge base of clinical guidelines, medical references, and workflow protocols, and the Solver uses accepted context to correct errors while reinforcing existing well-supported decisions. All three agents share an LLM backbone; only the Proposer is trained, while the Solver and Verifier remain frozen, and no stronger teacher is queried during the evolution loop. We construct a curated counterfactual instruction-tuning dataset spanning diverse application scenarios across binary classification, multiple-choice reasoning, and regression tasks. Its structured examples train the Proposer to reason about counterfactuals, apply domain knowledge, follow a consistent output format, and judge when evidence warrants changing or preserving a decision. Our instruction targets teach the Proposer to construct counterfactuals, trace their hypothesized causal mechanisms, and justify whether the original decision should change. Beyond instruction tuning, we tailor the Proposer's fine-tuning objective to integrate Verifier feedback with Solver decision quality. As we

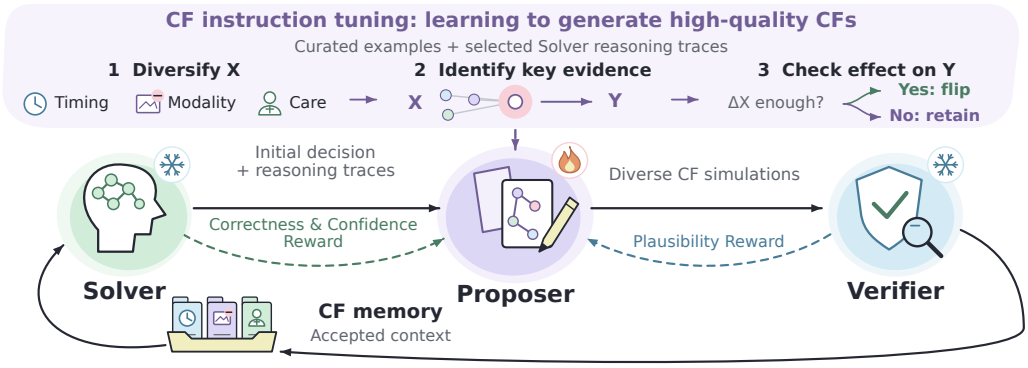


Figure 2: **Counterfactual self-evolution framework.** The Proposer generates counterfactuals from the Solver's decision and selected reasoning traces. The Verifier checks plausibility, and accepted counterfactuals enter memory as reusable Solver context. The Proposer learns through CF instruction tuning and feedback combining Solver's correctness/confidence with Verifier's plausibility.

progressively inject counterfactual context, we track performance over successive rounds and find that the framework learns to solve increasingly difficult held-out cases. We also find that counterfactual instruction tuning can approach the performance of task-specific Solver fine-tuning with **substantially fewer training examples**, while leaving the Solver's parameters unchanged. For example, on MIMIC-IV in-hospital mortality, a Proposer trained on around $3{,}000$ counterfactual examples supplies additional context that brings the frozen Solver's performance close to directly fine-tuning the Solver on $41{,}886$ examples. CF-Evolve thus uses about $1/14$ as many tuning examples. Analyses also show that the generated counterfactual mix differs across tasks, indicating that useful evidence edits are task-dependent.

2 Related Works

Self-evolving agents adapt outputs, memory, code, or model parameters through feedback. Self-Refine iteratively revises outputs through self-feedback (Madaan et al., 2023), while Reflexion retains verbal reflections to guide later attempts without updating model weights (Shinn et al., 2023). In clinical reasoning, GSEM organizes experiences into a graph and uses outcome feedback to calibrate their reliability and joint retrieval (Han et al., 2026a). Our focus is learning to generate feedback that tests how decisions depend on evidence. Other approaches ground improvement in external evaluation. The Darwin Gödel Machine modifies agent code and evaluates changes on coding benchmarks (Zhang et al., 2026), and CORAL supports collaborative discovery through shared persistent memory and separate evaluators (Qu et al., 2026). Propose, Solve, Verify trains a difficulty-aware problem proposer and solver through formally verified self-play (Wilf et al., 2026). Many evidence-based decisions lack formal correctness checks. Our Verifier instead checks plausibility and evidence consistency, while Proposer training combines this feedback with Solver's decision.

Counterfactual methods use controlled changes for explanation, augmentation, and reasoning. FIZLE prompts LLMs to generate text counterfactuals that probe black-box classifiers (Bhattacharjee et al., 2024). CATO uses causal assumptions to vary spurious attributes for robust learning (Feder et al., 2023), while SenseCF fine-tunes LLMs for health intervention design and sensor-data augmentation (Soumma et al., 2026). For clinical reasoning, You et al. (2026) edit findings and measure diagnostic confidence shifts to guide specialist discussion in a training-free framework. Counterfactuals also support learning across attempts: EvoCF derives constraints from execution failures, generates alternative multi-agent plans, and ranks them using retrieved experience (Chi et al., 2026); I-EDI filters reasoning traces through executable counterfactual tests and structural consistency checks before distilling them into the solver (Fan & Li, 2026). CF-Evolve learns a Proposer that edits case evidence and explains whether the original decision warrants correction or preservation. Training targets distinguish a response to edited evidence from a justified revision of the original decision. Accepted feedback becomes reusable context for the frozen Solver. Each memory note records its mechanism, applicability conditions, and failure boundaries, allowing the Solver to assess whether feedback from an earlier case is relevant before revising current decision.

3 COUNTERFACTUAL SELF-EVOLVING AGENTS

We now discuss the proposed CF-Evolve method. We begin with a motivating example, then explain counterfactual generation and verification. We next describe CF instruction tuning, feedback-based Proposer learning, and the use of accepted counterfactuals as Solver memory.

Problem Setup and Motivating Example. An evidence-identifiable task asks a question q whose evidence E, together with domain knowledge K, determines a unique, checkable answer y. We write this relation as $f(q, E; K) = y$, where f denotes the task's answer model. We initialize the Solver with a pretrained reasoning model (Qwen2.5-7B-Instruct; Yang et al., 2025) that produces a zero-shot prediction and reasoning trace from q and E. Counterfactual context can prompt the Solver to reconsider how evidence supports its decision. Figure 3 illustrates this on in-hospital mortality (IHM) prediction. In the illustrated case, severe acidosis leads the zero-shot Solver to predict death with probability 0.85. A counterfactual asks whether the acidosis reflects a reversible metabolic process rather than ongoing shock, while preserving the observed findings. The Solver lowers its death probability to 0.14, correctly predicting survival. The comparison in Figure 3c shows that crafted CF context improves the Solver's test accuracy. The performance gain motivates learning CF context that helps the Solver reconsider its decisions using the available evidence.

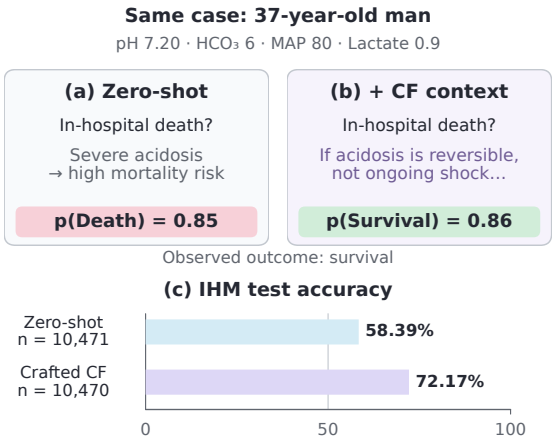


Figure 3: **CF context supports reconsideration.** (a,b) Crafted CF context corrects an IHM prediction. (c) Test accuracy with zero-shot inference and crafted CF context.

3.1 INSTRUCTION TUNING FOR GROUNDED COUNTERFACTUAL REASONING

Structural causal formulation. We specify the semantics of a CF using an acyclic structural causal model $M = (U, V, F, P_U)$, with background variables U, endogenous task variables V, mechanisms F, and background distribution P_U (Pearl, 2009a). Each variable satisfies $V_j = f_j(\mathrm{Pa}_j, U_j)$, where Pa_j denotes its parents. An action intervention replaces f_A by $A = a'$ while retaining the other mechanisms and the same background state u. Consequently,

$$V_j^{\mathrm{do}(A=a')}(u) = V_j(u) \quad \text{for } V_j \notin \{A\} \cup \mathrm{Desc}_M(A). \tag{1}$$

Here $\mathrm{Desc}_M(A)$ denotes the strict descendants of A. The equation states that a variable keeps its original value if the intervention cannot affect it through any causal path. It specifies what an admissible intervention preserves; the correction and preservation analysis below then asks whether the resulting context improves the Solver's answer to the original case. Appendix C.2 formalizes the intervention $A = a'$ and proves that unaffected variables stay fixed.

What does a useful counterfactual test? Our goal is to design a Proposer that generates CF context to expose reasoning errors while preserving decisions supported by the original evidence. Drawing on causal reasoning (Pearl, 2009b;a), we distinguish three tests of evidence, explanation, and action mechanisms in a structural causal model M. **Observe** ($E \to E'$) changes which observations are available, or when they become available, to test the Solver's dependence on particular evidence. **Explain** ($H \to H'$) uses abduction to compare causal hypotheses H and H' for the same evidence E, testing its interpretation while keeping the observed facts fixed. **Intervene** ($M \to M_{\mathrm{do}(A=a')}$) replaces the mechanism for action A with $A = a'$, leaving other mechanisms unchanged, to test the consequences of a different treatment, support state, or timing. We aim for balanced coverage of these three types in the Proposer's generated context.

Correction and preservation. For a binary task, let $\hat{y}$ be the original Solver decision and $\hat{y}^+$ its decision on the original case after reading CF context. Relative to the known training label y, there are four outcomes: (1) correct-to-wrong, (2) correct-to-correct, (3) wrong-to-correct, and (4) wrong-to-wrong. Types 2 and 3 are the desired preservation and correction behaviors (Figure 4a). Their

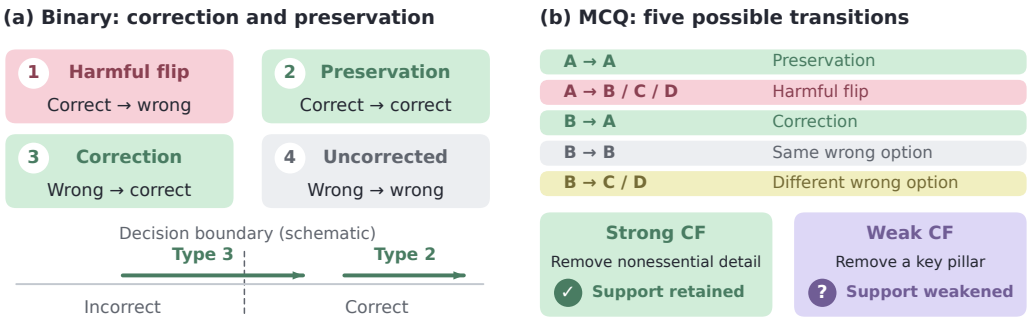


Figure 4: **Decision scenarios emphasized in CF instruction tuning.** (a) Binary decision transitions favor correction (Type 3) and preservation (Type 2, strengthen the confidence). (b) Multiple-choice questions (MCQ) also allow switches between wrong options; A denotes the correct option. Strong CFs remove nonessential evidence while retaining diagnostic support; weak CFs remove a diagnostic pillar and weaken support without specifying a replacement answer.

importance follows from the exact decomposition

$$\Pr(\hat{y}^+ = y) - \Pr(\hat{y} = y) = \Pr(\hat{y} \neq y, \hat{y}^+ = y) - \Pr(\hat{y} = y, \hat{y}^+ \neq y). \tag{2}$$

The accuracy gain equals the probability of correcting a wrong answer minus the probability of overturning a correct one. A flip-only objective rewards both terms, although they affect accuracy in opposite directions. The early failures in Appendix C.1 illustrate why a changed decision alone is insufficient. The Proposer must therefore assess the support removed, the support that remains, and whether either justifies revising the original decision. These outcome types guide CF instruction-tuning data construction and evaluation.

CF instruction-tuning setup. CF instruction tuning teaches the Proposer to apply domain knowledge, follow the required output format, and reason about whether changing key evidence warrants revising a decision. We construct supervision from training instances and the target Solver's own predictions, confidence scores, and reasoning traces. Cases with related evidence patterns or reasoning errors form complementary groups: a corrective example identifies an error using the available evidence, a hard preservation example uses counterevidence to rule out an analogous correction, and an anchor preservation example retains a well-supported decision. We add "`no_cf`" targets when no admissible edit is supported. Labels guide curation but are excluded from Proposer inputs, and factual claims in each target must be supported by the instance's available evidence. Each sample pairs the task, available evidence, and original Solver draft with a structured target: an evidence audit, a local hypothetical edit, its expected effect, an applicability boundary, and optional Solver memory. A separate flip-warrant gate controls whether CF context is used to revise the Solver's decision; corrective, preservation, and abstention targets teach the Proposer to judge when the evidence warrants such a change. Appendix C.4 details data construction and checks, Appendix C.5 explains the target fields, and Appendix C.7 gives the training objective and settings.

We next analyze the behavior these instruction targets aim to teach. The two propositions below separate net accuracy gain from harmful reversals and characterize when CF context is worth using.

Proposition 1 (Causal benefit of CF context). For fixed Solver and CF-generation procedures, let $R_t = \mathbf{1}\{\hat{y}(t) = y\}$ denote Solver correctness without ($t = 0$) or with ($t = 1$) CF context, holding the original case fixed. Correction $b = \Pr(R_1 = 1, R_0 = 0)$ is a probability of necessity and sufficiency (Tian & Pearl, 2000); harm is $h = \Pr(R_1 = 0, R_0 = 1)$. For marginal accuracies $p_t = \Pr(R_t = 1)$, the sharp bounds are

$$\max(0, p_1 - p_0) \leq b \leq \min(p_1, 1 - p_0), \qquad h = b - (p_1 - p_0). \tag{3}$$

Even with higher accuracy, some originally correct answers may be harmed; paired correction and harm rates reveal what the net gain conceals.

Proposition 2 (When to propose). Let X collect the available evidence, original Solver draft, and candidate CF, excluding the reference answer, and let $b(X)$ and $h(X)$ be the conditional correction and harm probabilities. Suppose abstention retains the original decision, correction earns utility

$\kappa_+ > 0$, harm costs $\kappa_- > 0$, and unchanged correctness has zero incremental utility. Then the inject-or-abstain rule maximizing expected incremental utility is

$$g^*(X) = \mathbf{1}\{\kappa_+ b(X) > \kappa_- h(X)\}, \tag{4}$$

with ties resolved by abstention. The rule **uses CF context only when the expected benefit of correction exceeds the expected cost of harm**; with equal weights, it selects positive expected accuracy gain. It describes the ideal behavior our instruction targets and flip warrant seek to approximate, without assuming that the implemented gate estimates calibrated response probabilities. Appendix C.3 gives the assumptions and proofs of both propositions.

3.2 Self-Evolution via Feedback from Multi-Agents

Instruction tuning does not directly test whether new proposals are plausible or help the Solver answer correctly. Verifier assessments and Solver feedback computed from training labels further train the Proposer, while the Solver revises its answers using the resulting CF context. In later memory-evolution rounds, no agent is retrained; only the bank of accepted CFs grows.

Verifier feedback. The Verifier checks proposed edits against retrieved domain knowledge. As an example, our clinical implementation uses Chroma with nine topic-specific PubMed Central collections, DailyMed drug labels, Surviving Sepsis Campaign guidelines, and ACR imaging criteria, covering major ICU physiological systems. Topic routing and semantic retrieval supply evidence to a judge that returns PASS, FLAG, or REJECT verdicts. Appendix D details retrieval and scoring.

Solver feedback and memory. The memory bank organizes factual cases, verified CFs, and knowledge gaps with task tags and provenance. Reusable notes retain a mechanism, applicability conditions, and failure boundaries. The frozen Solver reads the original case and draft alongside the current CF note and separately marked retrieved experience. Its revised answer distribution p^c is compared with the original p^0. On training cases, the reference answer y supplies both a correctness signal r_C and an uncertainty signal r_U: the clipped reduction in true-answer log loss. Thus confidence is rewarded when it moves toward the correct answer. To prevent context explosion, the memory bank imposes a fixed capacity on the CF context. The feedback objective therefore optimizes the selection, ordering, and combination of CF contexts within this budget, favoring context that is relevant to the current case and useful for improving the Solver's decision.

Feedback-based Proposer optimization. We formulate the collective-feedback objective as a correctness-veto format. For training pairs $(x, y) \sim \mathcal{D}_{\text{train}}$ and CFs $c \sim \pi_\theta(\cdot \mid x)$, let $g = \mathbf{1}\{\hat{y}^c = y\}$ indicate a correct revised answer and $s = \lambda_U r_U + \lambda_V r_V + r_F$ combine auxiliary feedback, with $\lambda_C > 0$ and $\lambda_U, \lambda_V \geq 0$. Then the objective is:

$$\max_\theta \quad J(\theta) = \mathbb{E}[\lambda_C r_C + g\, s + (1 - g)\min(s, 0)]\,, \tag{5}$$

where r_C is positive for correction and preservation and negative for harmful reversals and unresolved errors. The uncertainty $r_U = \text{clip}(\log p^c(y) - \log p^0(y), -2, 2)$ rewards increased probability of the true answer and penalizes decreases, with its magnitude capped at two. Verifier support is scored by r_V, and r_F combines the implemented direction/schema bonuses and invalid-output penalties. When the revised answer is wrong, the combined auxiliary contribution is capped at zero, so the total reward remains negative. Labels are only used in reward computation.

Memory evolution lets the frozen Solver reuse validated experience on new cases without retraining. Starting from $\mathcal{M}_0 = \emptyset$, each round applies the trained agents and CF admission gate to disjoint new evolution cases, storing accepted notes $\mathcal{A}_r$ as $\mathcal{M}_r = \mathcal{M}_{r-1} \cup \mathcal{A}_r$ for $r = 1, \ldots, R$. The Solver receives selected CF context within a fixed budget. Context evolution has several dimensions: the number of notes within that budget, diversity of CF types, coverage of source subgroups, and the quality and relevance of the selected context. Feedback-based Proposer optimization targets context quality by rewarding its contribution to the Solver's decision. Fixed evaluation sets track each difficulty levels against its memory-free reference. The following bound assesses whether these changes yield more corrections than harmful reversals, accounting for sampling uncertainty.

Proposition 3 (Reliable memory evolution). For R prespecified rounds and S level, let $\Delta_{r,s}$ be the population accuracy gain from $\mathcal{M}_{r-1}$ to $\mathcal{M}_r$ under a fixed Solver context budget, estimated by the mean paired gain $\widehat{\Delta}_{r,s}$ on n_s independent cases. If the entire memory trajectory, including its

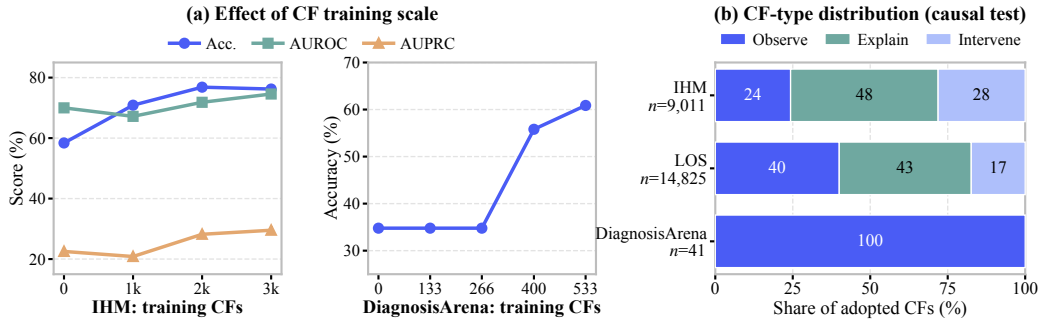


Figure 5: **CF instruction tuning produces high-quality CF context.** (a) We instruction-tune a Qwen2.5-7B Proposer on increasing numbers of training CFs to guide a frozen Solver. (b) We categorize the adopted CF context by the causal test it performs mentioned in Section 3.1: *Observe* changes the available evidence or its timing; *Explain* changes the causal hypothesis while holding evidence fixed; and *Intervene* changes an action. Scores are shown on a 0–100 scale.

context-selection and ordering rules, is independent of the evaluation sets, then for $\delta \in (0, 1)$,

$$L_{r,s} = \widehat{\Delta}_{r,s} - \sqrt{\frac{2\log(RS/\delta)}{n_s}}, \qquad \Pr(\Delta_{r,s} \geq L_{r,s}\ \forall r, s) \geq 1 - \delta. \tag{6}$$

Thus $\min_s L_{r,s} > 0$ certifies improvement across difficulty level. Appendix H gives the assumptions and proof using Hoeffding's inequality (Hoeffding, 1963). Specifically, the bound subtracts a margin for sampling uncertainty from the observed gain at each difficulty level. If every remainder is positive, the context update improves all levels with confidence at least $1 - \delta$.

4 Experiments

We evaluate CF-Evolve across healthcare, business, and fact verification domains. We ask the following research questions (RQ): (**RQ1**) How does counterfactual instruction tuning for Proposer improve CF quality? (**RQ2**) How robust is our framework across base models and component ablations, and how does it compare with strong baselines? (**RQ3**) Can CF-driven self-evolution progressively address harder problems across diverse situations?

Datasets and Baselines. We evaluate on the MIMIC-IV ecosystem (Johnson et al., 2023) for multimodal prediction of in-hospital mortality (IHM) and length of stay (LOS), DiagnosisArena (Zhu et al., 2026) for differential diagnosis, the MMMU Business section (Yue et al., 2024), with an additional MMMU/MMMU-Pro business evaluation for memory transfer, and HoVer (Jiang et al., 2020) for multi-hop fact verification. These tasks span binary classification, MCQ, and regression. We identify three baseline groups: (i) neither CF nor self-evolution, including zero-shot and task-fine-tuned base models, plus FuseMoE (Han et al., 2024) and MILM (Chung et al., 2026) for clinical prediction; (ii) CF without self-evolution, using a static retrieval model to supply relevant CFs from a fixed bank as Solver context (Zhou et al., 2023); and (iii) self-evolution without CF, such as GSEM (Han et al., 2026a), which updates an experience memory through feedback.

Experiment Setup. We evaluate Qwen2.5-7B (Yang et al., 2025), Llama3-8B (Meta AI, 2024), and MedGemma-4B (Sellergren et al., 2026). Task- and model-specific LoRA Proposers condition on the evidence and the target Solver's initial prediction and reasoning, while the Solver and Verifier remain frozen. Validation data select MCQ checkpoints and CF-acceptance thresholds. We report ranking and classification metrics for IHM, absolute error for LOS, and accuracy for MCQ and fact verification; paired corrected and harmed counts measure decision revisions.

4.1 CF Instruction Tuning Shapes Decision Quality (RQ1)

Figure 5 shows that high-quality CF context depends on both supervision and the available causal tests. With limited supervision, higher IHM accuracy accompanies lower AUROC and AUPRC,

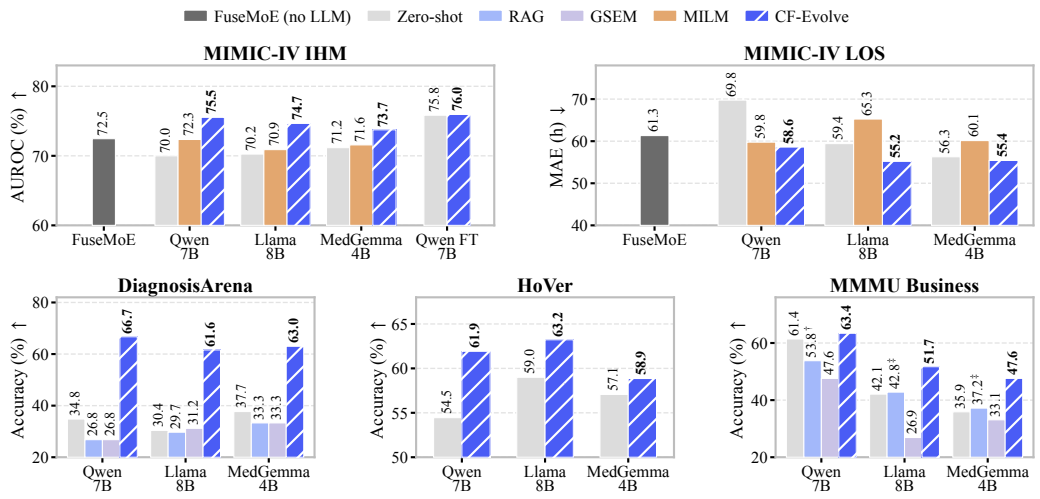


Figure 6: **Comprehensive evaluations of CF-Evolve.** We instruction-tune diverse base models to generate CF context that guides a frozen Solver, followed by memory evolution to optimize the involved context. Results show that adding high-quality CF context consistently improves performance across tasks. Task-specific baselines are shown only where applicable.

meaning that predicted risks distinguish deaths from survivors less well. The 1k set is randomly sampled, whereas the 2k–3k sets use curated examples, so gains may reflect better examples as well as more data. On DiagnosisArena, the gain from 400 to 533 training records comes entirely from fewer harmful revisions: both runs correct 36 errors, but the latter preserves every initially correct answer (Appendix B.3). This supports our emphasis on teaching the Proposer when to preserve a decision alongside how to correct it. The CF-type distribution also reflects the task and its edit interface: clinical outputs can test alternative explanations and actions, while DiagnosisArena uses executable edits to observed evidence. These tests probe the assumptions supporting the original decision while keeping hypothetical changes distinct from observed facts. Our framework accommodates these different tests within the same correction–preservation objective, allowing the Proposer to improve the guidance supplied to a frozen Solver.

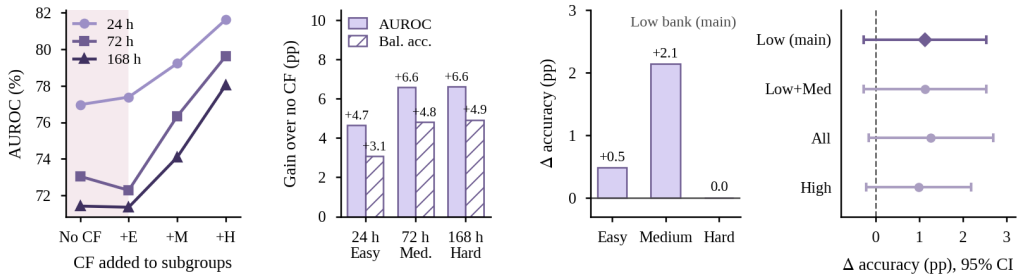
4.2 CF-Evolve Generalizes across Tasks and Base Models (RQ1-2)

Figure 6 evaluates whether CF-Evolve remains useful when the task and base model change. We compare each configuration against its corresponding baselines. MCQ configurations use separately trained Proposers and validation-selected acceptance policies, allowing the same framework to accommodate different Solver behaviors.

CF-Evolve improves the reported metric in every task–model setting shown and outperforms the competing baselines where they are evaluated. Gains span clinical prediction, diagnostic reasoning, and non-medical reasoning, supporting the framework's applicability beyond a single domain. The Proposer tests whether changing a finding or an assumption would change the answer, then explains what this reveals about the Solver's original reasoning. The Solver uses that explanation to reconsider its answer without updating its weights. This feedback yields more consistent MCQ gains than the RAG and GSEM baselines. Even for the Qwen Solver fine-tuned on 41,886 examples, a Proposer trained on 3,000 CF examples slightly improves IHM AUROC. This supports CF instruction tuning as a sample-efficient complement to direct Solver training. Full metric vectors, paired correction–harm counts, and task-specific configurations can be found in Appendix E.

4.3 CF-Evolve Progressively Solves Harder Problems (RQ3)

We further evaluate whether expanding CF context, and thus exposure to alternative evidence–decision and action–outcome relations, helps a frozen Solver solve harder problems.



(a) CF curriculum **(b) Gain vs. difficulty** **(c) Gain vs. difficulty** **(d) Overall memory gain**

Figure 7: **Evaluate CF-Evolve on harder problems.** (a, b) CF-Evolve accumulates memory across patient-difficulty subgroups for clinical mortality prediction at increasing prediction horizons. (c, d) CF-Evolve accumulates memory across CF banks for MMMU/MMMU-Pro business questions of increasing difficulty. Across both domains, richer context yields limited gains on easier problems but increasingly benefits harder ones. Easy decisions already have sufficient evidence, whereas difficult cases benefit from counterfactual experience that supplies missing or ambiguous reasoning paths.

Harder patient cases. From the same first-24-hour record, we predict mortality at increasingly difficult 24-, 72-, and 168-hour horizons (Appendix G.1). Patient-specific CF raises AUROC by 4.65, 6.57, and 6.62 points, respectively; balanced accuracy follows the same pattern (Figure 7b). The hardest task's AUROC with CF exceeds the easiest task's baseline. Within each task, we progressively supply CF to confidence-defined Easy, Medium, and Hard patient groups. The Medium and Hard rounds account for at least 91% of the final AUROC gain, whereas Easy-only CF changes little (Figure 7a). This pattern suggests that CF is most useful when the Solver needs help resolving uncertain evidence, supporting targeted context expansion for harder decisions.

Harder business reasoning. We evaluate CF memory on unseen MMMU/MMMU-Pro business questions with predefined Easy, Medium, and Hard difficulty levels. We group source cases by initial Solver confidence to form four banks: Low, Low+Medium, All, and High-only. All four yield positive accuracy gains, although their confidence intervals include zero (Figure 7d). The validation-selected Low bank raises accuracy from 56.48% to 57.61%, helping Medium questions most (Figure 7c); larger banks yield small gains on Hard questions (Appendix G.2). The memory audit suggests a reason for this uneven transfer: most stored CFs preserve the source answer and contain source-specific numbers. Such examples can reinforce familiar reasoning, but may offer limited guidance when a new question requires different decision dependencies. Together, these results support expanding relevant CF coverage for difficult cases and suggest that transfer depends on capturing reusable reasoning, beyond increasing the number of memories.

5 Conclusion and Future Work

We introduced counterfactual self-evolution, combining grounded counterfactual instruction tuning with memory-based self-evolution for a frozen Solver. The Proposer learns to test how decisions depend on evidence, helping the Solver correct reasoning errors and retain answers supported by the original evidence. Different CF categories distinguish changes to observations, causal hypotheses, and actions. The Proposer is refined by collecting feedback from Solver's correctness and uncertainty combined with Verifier's feedback. Accepted counterfactual accumulates as reusable context, so the Solver adapts without further fine-tuning. Our analysis clarifies intervention semantics and gives conditions under which correction outweighs harm and memory updates improve performance. The results suggest that broader CF coverage helps difficult clinical decisions, while transfer across cases depends on whether memory captures reusable reasoning. Future work could estimate correction and harm probabilities to guide counterfactual selection and use uncertainty bounds to control memory updates. A second direction is to learn structural causal models that constrain edits and actively select informative counterfactuals to resolve uncertain mechanisms. They could improve how agents acquire and transfer experience as task environments change.

AI USE STATEMENT

We used generative AI tools to assist with literature search, implementation of our method, and manuscript revision. Language models serve as the Proposer, Solver, and Verifier in our framework. For the MCQ experiments, Claude Opus 4.6, with Sonnet used for some supplementary examples, assisted the verification of the correctness of counterfactual instruction-tuning samples; a frozen Qwen3.6-27B model provided text descriptions of MMMU images. The authors take responsibility for the accuracy and integrity of the final text, claims, code, figures, and proofs, including all material produced with AI assistance.

ETHICS STATEMENT

This study uses existing clinical records and released benchmarks, without prospective patient recruitment or clinical intervention. MIMIC-IV contains de-identified records from a single US medical center (Johnson et al., 2023); access requires PhysioNet credentialing, research ethics training, and a signed data use agreement. Patient-derived data and artifacts remain subject to the source data's access and sharing restrictions. MIMIC-IV results may reflect that center's population and practices, and benchmark performance does not establish fairness or transfer to other settings. Counterfactual explanations and automated verification can be inaccurate; verifier acceptance does not establish causal validity or guarantee a correct decision. The outputs of our framework should not guide patient care without independent validation and qualified clinical oversight.

REPRODUCIBILITY STATEMENT

Section 3 defines the agents, training objective, and memory evolution, and Section 4 specifies the datasets, models, baselines, and evaluation metrics. Appendix C documents CF data construction, input restrictions, structured targets, and training hyperparameters; Appendix B.1 details MCQ data splits and validation procedures. Appendix D describes feedback and memory processing, with retrieval corpora, embedding models, verification procedures, and acceptance policies in Appendix D.1. The theoretical assumptions and proofs are provided in Appendices C.2, C.3, and H. The implementation is available in the anonymous repository linked in the abstract.

REFERENCES

Pedram Akbarian, Huy Nguyen, Xing Han, and Nhat Ho. Quadratic gating mixture of experts: Statistical insights into self-attention. *arXiv preprint arXiv:2410.11222*, 2024. URL https://arxiv.org/abs/2410.11222.

Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. Zero-shot LLM-guided counterfactual generation: A case study on NLP model evaluation. In *2024 IEEE International Conference on Big Data (BigData)*, pp. 1243–1248, 2024. doi: 10.1109/BigData62323.2024.10825537. URL https://doi.org/10.1109/BigData62323.2024.10825537.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, 2021. doi: 10.18653/v1/2021.emnlp-main.300. URL https://aclanthology.org/2021.emnlp-main.300/.

Haotian Chi, Zeyu Feng, Xingrui Yu, Linbo Luo, Yew-Soon Ong, Ivor Tsang, Hechang Chen, Yi Chang, and Haiyan Yin. EvoCF: Multi-agent collaboration via agentic memory-driven evolutionary counterfactual planning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026. URL https://openreview.net/forum?id=FsSirPJy9U.

Hsing-Huan Chung, Shijun Li, Yoav Wald, Xing Han, Suchi Saria, and Joydeep Ghosh. MILM: Large language models for multimodal irregular time series with informative sampling. *arXiv preprint arXiv:2605.13711*, 2026. URL https://arxiv.org/abs/2605.13711.

Runze Fan and Yong Li. I-EDI: Robust self-evolution agents via verifiable counterfactual simulation. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 90–98, 2026. doi: 10.24963/ijcai.2026/11. URL https://doi.org/10.24963/ijcai.2026/11.

Amir Feder, Yoav Wald, Claudia Shi, Suchi Saria, and David Blei. Data augmentations for improved (large) language model generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 70638–70653, 2023. doi: 10.52202/075280-3096. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/df88b275bef31ac96c85f0c4013734fc-Paper-Conference.pdf.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Qihan Ren, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=CTr3bovS5F.

Xiao Han, Yuzheng Fan, Sendong Zhao, Haochun Wang, and Bing Qin. GSEM: Graph-based self-evolving memory for experience augmented clinical reasoning. *arXiv preprint arXiv:2603.22096*, 2026a. URL https://arxiv.org/abs/2603.22096.

Xing Han, Huy Nguyen, Carl Harris, Nhat Ho, and Suchi Saria. FuseMoE: Mixture-of-experts transformers for fleximodal fusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 67850–67900, 2024. doi: 10.52202/079017-2167. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/7d62a85ebfed2f680eb5544beae93191-Paper-Conference.pdf.

Xing Han, Hsing-Huan Chung, Joydeep Ghosh, Paul Pu Liang, and Suchi Saria. Massively multimodal foundation models: A framework for capturing interactions with specialized mixture-of-experts. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=qF9WJxvHX8.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. doi: 10.1080/01621459.1963.10500830.

Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. HoVer: A dataset for many-hop fact extraction and claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 3441–3460. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.findings-emnlp.309. URL https://aclanthology.org/2020.findings-emnlp.309/.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10:1, 2023. doi: 10.1038/s41597-022-01899-x. URL https://doi.org/10.1038/s41597-022-01899-x.

Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Hyeonhoon Lee, Marzyeh Ghassemi, Cynthia Breazeal, and Hae Won Park. MDAgents: An adaptive collaboration of LLMs for medical decision-making. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 79410–79452, 2024. doi: 10.52202/079017-2522. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/90d1fc07f46e31387978b88e7e057a31-Paper-Conference.pdf.

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 (NeurIPS)*, volume 36, pp. 46534–46594, 2023. doi: 10.52202/075280-2019. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91edff07232fb1b55a505a9e9f6c0ff3-Paper-Conference.pdf.

Meta AI. Meta Llama 3 model card, 2024. URL https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct.

Huy Nguyen, Xing Han, Carl William Harris, Suchi Saria, and Nhat Ho. On expert estimation in hierarchical mixture of experts: Beyond softmax gating functions. *arXiv preprint arXiv:2410.02935*, 2024. URL https://arxiv.org/abs/2410.02935v2.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2 edition, 2009a. ISBN 9780521895606. doi: 10.1017/CBO9780511803161.

Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009b. doi: 10.1214/09-SS057.

Ao Qu, Han Zheng, Zijian Zhou, Yihao Yan, Yihong Tang, Shao Yong Ong, Fenglu Hong, Kaichen Zhou, Chonghe Jiang, Minwei Kong, Jiacheng Zhu, Xuan Jiang, Sirui Li, Cathy Wu, Bryan Kian Hsiang Low, Jinhua Zhao, and Paul Pu Liang. CORAL: Towards autonomous multi-agent evolution for open-ended discovery. *arXiv preprint arXiv:2604.01658*, 2026. URL https://arxiv.org/abs/2604.01658.

Andrew Sellergren, Sahar Kazemzadeh, Tiam Jaroensri, et al. MedGemma technical report. *arXiv preprint arXiv:2507.05201*, 2026. doi: 10.48550/arXiv.2507.05201. URL https://arxiv.org/abs/2507.05201.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 8634–8652, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Shovito Barua Soumma, Asiful Arefeen, Stephanie M Carpenter, Melanie Hingle, and Hassan Ghasemzadeh. Counterfactual modeling with fine-tuned llms for health intervention design and sensor data augmentation. *IEEE Open Journal of Engineering in Medicine and Biology*, 7: 232–240, 2026. doi: 10.1109/OJEMB.2026.3697994. URL https://doi.org/10.1109/OJEMB.2026.3697994.

Emily Strickland, Zachary Kerosky, Joseph Krivda, and Takor Arrey-Mbi. Platypnea-orthodeoxia syndrome: A rare cause of hypoxia. *Annals of Internal Medicine: Clinical Cases*, 2(2):e221124, 2023. doi: 10.7326/aimcc.2022.1124. URL https://doi.org/10.7326/aimcc.2022.1124.

Xiangru Tang, Anni Zou, Zhuosheng Zhang, Ziming Li, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. MedAgents: Large language models as collaborators for zero-shot medical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 599–621, 2024. doi: 10.18653/v1/2024.findings-acl.33. URL https://aclanthology.org/2024.findings-acl.33/.

Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence policy improvement. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2380–2388. PMLR, 2015. URL https://proceedings.mlr.press/v37/thomas15.html.

Jin Tian and Judea Pearl. Probabilities of causation: Bounds and identification. *Annals of Mathematics and Artificial Intelligence*, 28(1–4):287–313, 2000. doi: 10.1023/A:1018912507879.

Ziyue Wang, Junde Wu, Linghan Cai, Chang Han Low, Xihong Yang, Qiaxuan Li, and Yueming Jin. MedAgent-Pro: Towards evidence-based multi-modal medical diagnosis via reasoning agentic workflow. In *International Conference on Learning Representations*, pp. 114360–114389, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/ba1c5356d9164bb64c446a4b690226b0-Paper-Conference.pdf.

Alex Wilf, Pranjal Aggarwal, Bryan Parno, Daniel Fried, Louis-Philippe Morency, Paul Pu Liang, and Sean Welleck. Propose, solve, verify: Self-play through formal verification. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Vuaq1qXCki.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025. doi: 10.48550/arXiv.2412.15115. URL https://arxiv.org/abs/2412.15115.

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

Zhiwen You, Xi Chen, Aniket Vashishtha, Simo Du, Gabriel Erion-Barner, Hongyuan Mei, Hao Peng, and Yue Guo. Improving clinical diagnosis with counterfactual multi-agent reasoning. *arXiv preprint arXiv:2603.27820*, 2026. URL https://arxiv.org/abs/2603.27820.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9556–9567, 2024. doi: 10.1109/CVPR52733.2024.00913. URL https://openaccess.thecvf.com/content/CVPR2024/html/Yue_MMMU_A_Massive_Multi-discipline_Multimodal_Understanding_and_Reasoning_Benchmark_for_CVPR_2024_paper.html.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Gödel machine: Open-ended evolution of self-improving agents. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=pUpzQZTvGY.

Wenxuan Zhou, Sheng Zhang, Hoifung Poon, and Muhao Chen. Context-faithful prompting for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 14544–14556. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.968. URL https://aclanthology.org/2023.findings-emnlp.968/.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=XY8AaxDSLb.

Yakun Zhu, Zhongzhen Huang, Linjie Mu, Yutong Huang, Wei Nie, Jiaji Liu, Shaoting Zhang, Pengfei Liu, and Xiaofan Zhang. DiagnosisArena: Benchmarking diagnostic reasoning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3074–3098. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.151. URL https://aclanthology.org/2026.findings-acl.151/.

Supplementary Material for "Counterfactual Self-Evolving Agents for Evidence Grounded Reasoning"

A Extended Related Works

A.1 Feedback and Memory for Self-Improvement

A central distinction in self-improvement is what changes after feedback: an answer, stored experience, an executable program, or model parameters. Output refinement and episodic memory use language as the feedback medium. Self-Refine revises a current answer through self-critique (Madaan et al., 2023), while Reflexion carries verbal reflections into later attempts (Shinn et al., 2023). In clinical reasoning, GSEM organizes experience into a graph and calibrates retrieval through outcome-dependent updates to node quality and edge weights (Han et al., 2026a). These mechanisms make feedback reusable, but feedback quality remains central: retaining a critique does not establish that it identifies an error in the original decision. Our Proposer is therefore trained to connect an evidence edit to a hypothesized mechanism and to assess whether the original evidence warrants correction or preservation.

Externally evaluable tasks provide stronger grounds for accepting updates. Code evolution and collaborative discovery use coding benchmarks or separate task evaluators to assess candidate solutions, as in the Darwin Gödel Machine and CORAL (Zhang et al., 2026; Qu et al., 2026). Propose, Solve, Verify uses formal verification to support self-play between a problem proposer and solver (Wilf et al., 2026). Our clinical Verifier instead checks evidence consistency and plausibility using retrieved knowledge; its judgments are not formal correctness certificates. We train the Proposer using Solver and Verifier feedback, then evolve memory while keeping agent parameters fixed. This separates learning to produce useful feedback from accumulating accepted experience.

A.2 Counterfactual Objectives and Agent Evolution

Counterfactual generation serves different objectives depending on what is altered and how the result is used. For model explanation, FIZLE uses localized text edits to probe black-box classifiers (Bhattacharjee et al., 2024). For robust learning, CATO uses causal assumptions to guide augmentation that varies spurious attributes while preserving the target (Feder et al., 2023). SenseCF learns to generate health-related counterfactuals for intervention design and sensor-data augmentation (Soumma et al., 2026). These uses establish the value of controlled changes, but evaluating an altered input answers a different question from deciding whether the original prediction should be revised. Our instruction targets require the Proposer to return to the original case and examine the support that remains, rather than treating a prediction change alone as evidence of improvement.

Counterfactuals can also guide search and select experience for learning. EvoCF induces constraints from execution failures, generates alternative multi-agent plans through counterfactual mutations, and evaluates candidates using retrieved experience (Chi et al., 2026). It provides a close precedent for combining counterfactual search with persistent memory, with plans as the object of intervention. I-EDI instead tests reasoning traces through executable counterfactual perturbations and structural consistency checks, then distills accepted traces into the solver (Fan & Li, 2026). Our interventions concern case evidence and its interpretation; the learned component is the Proposer, while the Solver receives accepted counterfactual feedback as context. The distinction is thus the adaptation target and the requirement to justify correction or preservation on the original case, rather than counterfactual generation or memory alone.

A.3 Evidence Use in Clinical Reasoning

Clinical agent systems improve evidence use through complementary forms of coordination. Role-based discussion and adaptive team formation organize specialist perspectives (Tang et al., 2024; Kim et al., 2024); guideline retrieval and multimodal tools connect reasoning to external clinical evidence (Wang et al., 2026). Counterfactual diagnostic discussion makes the dependence on individual findings more explicit: You et al. (2026) edit findings and use resulting confidence differences

to guide specialist deliberation and revise diagnoses. This training-free framework is particularly relevant to our within-case feedback loop. Our additional focus is learning the Proposer's feedback policy and retaining accepted experience for subsequent cases.

Evidence representation is a separate challenge in irregular clinical records. FuseMoE integrates missing and irregularly sampled modalities through expert routing (Han et al., 2024), whereas MILM serializes observations as time-ordered triplets and uses staged training to learn sampling patterns before incorporating measured values (Chung et al., 2026). These methods motivate our predictive baselines and the need to preserve temporal and missingness information. Complementary advances improve attention and expert routing: quadratic-gating theory motivates nonlinear transformations of the attention value matrix (Akbarian et al., 2024), and hierarchical MoE analysis motivates Laplace gating to accelerate expert estimation (Nguyen et al., 2024). For multimodal representations, MERGE guides specialized experts using temporal redundancy, uniqueness, and synergy between modalities (Han et al., 2026b). Our contribution concerns how a Solver examines and reuses such evidence through counterfactual feedback, complementing improvements to the underlying representation.

B Additional Experimental Results

B.1 Experimental Setup for MCQ Evaluations

This section describes DiagnosisArena and MMMU Business. Each dataset–model pair uses a separately trained Proposer and a frozen Solver. Clinical IHM/LOS training is described in Section C.7.

Data and validation. DiagnosisArena contains 915 four-choice diagnosis questions. MMMU Business contains 967 strict A–D questions from Accounting, Economics, Finance, Management, and Marketing. Table 1 gives the benchmark splits. Supervision constructed from training questions is further divided into Proposer fitting and internal-validation records. Internal validation is used for checkpoint selection; the external benchmark validation set is used for CF-acceptance settings. Test questions are excluded from both fitting sets. One supervision record can contain multiple CFs, so record counts differ from CF and atomic-edit counts.

Table 1: **Benchmark splits for MCQ evaluation.**

Dataset	Training	Validation	Test
DiagnosisArena	640	137	138
MMMU Business	677	145	145

Image and Proposer inputs. For MMMU Business, the recorded Qwen3.6-27B captioning configuration converts images into fixed descriptions of visible text, numbers, tables, axes, and spatial relations without reference answers. The text-based Solvers receive these descriptions rather than raw images. At inference, each Proposer receives the question, options, and its Solver's initial answer and reasoning. DiagnosisArena additionally supplies the top-two logit margin and an evidence catalog with IDs and exact source text, to describe decision uncertainty and locate executable edits. Reference answers, record IDs, reference CFs, and post-CF Solver answers are excluded from Proposer inputs. Humans design the CF supervision and human experts verify it, with Claude Opus 4.6 assisting verification through AWS Bedrock. Training answers support this offline review and are excluded at inference. Solver and Verifier parameters remain frozen.

Outcome measures. Accuracy is exact A–D match over the full test set. We also count *corrections* (wrong to correct) and *harmful revisions* (correct to wrong), because changing an answer is not necessarily useful. For B initially correct answers, C corrections, H harmful revisions, and n test questions,

$$N_{\text{correct,final}} = B + C - H, \qquad \Delta_{\text{pp}} = 100(C - H)/n.$$

The correction rate is $C/(n - B)$ and preservation is $(B - H)/B$, when the denominators are nonzero. Changes between two wrong options are counted separately, and valid-output counts are reported where available. Paired statistics use the initial predictions recorded in the same full-system run as the revised predictions, including its output-processing protocol. A separate zero-shot result need not be that run's starting point.

Comparison scope. Cross-model evaluation tests the framework with a separate Proposer and retrieval configuration for each base model, rather than zero-shot transfer of one shared adapter. Seed averages and best recorded complete runs are identified separately in the relevant results.

B.2 Full IHM Component Comparison (RQ1)

Table 2 reports all four IHM metrics for the component comparison summarized in Figure 5(a). CF instruction tuning is implemented as supervised fine-tuning (SFT). The SFT Proposer is trained on 3,000 training CFs; SFT+GRPO is its step-64 continuation after one RL epoch. These parameter updates precede the memory-evolution rounds, during which both models remain frozen. "Qwen SFT" denotes a task-fine-tuned Solver. The zero-shot, SFT, and SFT+GRPO runs share all 10,471 IHM test cases, the frozen Qwen Solver, and the evaluation implementation. Appendix D specifies the reported reward.

Table 2: **Component comparison on full-test IHM (RQ1).** SFT Proposer and SFT+GRPO are evaluated with the frozen Qwen2.5-7B Solver; the Solver-specific row uses a Proposer conditioned on the SFT Solver's draft. Higher is better.

Proposer	Solver	Accuracy	AUROC	AUPRC	F1
None (zero-shot)	Qwen	.5839	.7000	.2252	.3052
SFT	Qwen	.7372	.7543	.2985	**.3723**
None	Qwen SFT	**.8786**	.7584	.3128	.1216
Solver-specific SFT	Qwen SFT	.8630	**.7595**	.3212	.3521
SFT + GRPO	Qwen	.8417	.7553	**.3237**	.3720

B.3 DiagnosisArena Instruction Tuning and Training Scale

Learned guidance improves the frozen Solver. Experiment 4 evaluates a Proposer trained on human-designed, expert-verified CF supervision with Claude-assisted verification. The frozen Qwen2.5-7B-Instruct Solver initially answers 48 of 138 test questions correctly (34.8%). The three Proposer runs reach 60.9%, 63.0%, and 58.7%, with 34–41 corrections and 0–2 harmful revisions each (Table 3). Mean accuracy is 60.9% with a sample standard deviation of 2.2 percentage points. All three reported seeds improve the frozen Solver, supporting learned CF guidance as a route to better decisions without updating Solver parameters.

Table 3: **DiagnosisArena CF instruction-tuning results.**

Configuration	Test correct	Accuracy (%)	Corrected	Harmed
Zero-shot Solver	48	34.8	—	—
CF-IT, seed 20260728	84	60.9	36	0
CF-IT, seed 20260730	87	63.0	41	2
CF-IT, seed 20260731	81	58.7	34	1
Three-seed mean	84.0	60.9 ± 2.2	37.0	1.0
Validation-selected consensus	87	**63.0**	39	**0**

Consensus retains corrections with fewer harmful changes. All values in Table 3 are test results. CF-IT means CF instruction tuning; the mean and standard deviation use only the three individual runs. The consensus combines their outputs using a rule selected on the external 137-question validation set and frozen before testing. It reaches 87/138 correct answers by correcting 39 of the 90 initially wrong answers and preserving all 48 initially correct ones. Seed 20260730 reaches the same accuracy with 41 corrections and two harms. Thus, equal accuracy can reflect different correction–preservation balances. Construction and training details appear in Section C.8.

Training scale and preservation. Experiment 5 uses nested subsets of 133, 266, 400, and 533 records, stratified by `CROSSED`/`NOT_CROSSED`, with the same 93-record internal-validation set and a fixed five-epoch recipe. Table 4 reports one completed run per size, not a seed average. All

outcome columns refer to the same 138-question test set, starting from 48 correct answers. The last column counts switches from one wrong option to another.

Table 4: **DiagnosisArena training-scale results.**

Training records	Test correct	Accuracy (%)	Corrected	Harmed	Different wrong option
133	48	34.8	1	1	0
266	48	34.8	0	0	0
400	77	55.8	36	7	6
533	84	**60.9**	36	**0**	1

The 133- and 266-record runs remain at 34.8% accuracy. The 400- and 533-record runs reach 55.8% and 60.9%. Both correct 36 errors, but harmful revisions fall from seven to zero. Their 5.1-percentage-point difference therefore comes from preserving more initially correct answers, rather than a larger correction count. Fixed epochs also give larger datasets more optimizer updates, so this is a comparison of overall training scale under that recipe. The observed result supports evaluating CF guidance through both correction and preservation.

C Counterfactual Instruction Tuning

C.1 Failure modes of early counterfactual construction

Two early approaches motivated instruction tuning: heuristic construction could produce ungrounded edits, and flip-reward learning did not distinguish correction from harm. Their failures show why the Proposer must learn whether an evidence change warrants reconsideration before optimizing its effect on the Solver.

The training-free baseline selected channels by matching keywords in the Solver's reasoning. For example, mentions of acidosis could select bicarbonate or anion gap, while mentions of infection could select white blood cells. When the trace supplied no time interval, the heuristic used the final third of the available trajectory. The selected channels and interval were passed to the Proposer with a predefined output template. This procedure conflated how often a feature was mentioned with whether it was decisive, and left the meaning of numeric edits insufficiently constrained. Figure 8a shows a resulting mismatch between the raw proposal and the rendered intervention. These examples belong to an early prototype that used longer trajectories; the current protocol restricts patient evidence to the prediction window.

The second failure concerns the learning objective. A reward for $\hat{y}^+ \neq \hat{y}$ gives positive feedback to both a correction and a harmful reversal. In the early flip-reward RL attempt, these opposing transitions canceled the accuracy benefit. Figure 8b gives two pilot examples of the underlying behavior. They illustrate the distinction in Equation 2, rather than estimating transition frequencies or isolating the causal effect of RL training. The instruction targets therefore teach the Proposer to inspect residual support and to preserve a sound decision even when a plausible CF can be generated.

C.2 Structural semantics of counterfactual edits

Let $M = (U, V, F, P_U)$ be an acyclic structural causal model with equations $V_j = f_j(\mathrm{Pa}_j, U_j)$. A background realization u determines one factual world. The intervened model $M_{a'}$ replaces only the equation for action A with $A = a'$; other structural functions and the background realization remain unchanged. The notation $V_j^{\mathrm{do}(A=a')}(u)$ denotes the solution for V_j in $M_{a'}$. This is the structural semantics of counterfactuals described by Pearl (2009a, Chapter 7).

For observed evidence $E = e$ and a query variable $Z \in V$, abduction–action–prediction takes the form

$$\Pr_M(Z_{a'} = z \mid E = e) = \int \mathbf{1}\{Z_{M_{a'}}(u) = z\}\, dP_M(u \mid E = e). \tag{7}$$

Abduction updates the background distribution using the factual evidence, action replaces the designated mechanism, and prediction evaluates the query in the modified model. The same background

(a) Heuristic construction failure

Solver draft. Survival, $p_{\text{death}} = 0.35$. The trace mentions stable vital signs, mild hypoxemia, hyperglycemia, and acid–base abnormalities.

Raw proposal.

```
Glucose:  150
Heart Rate:  80
time_window:  [34, 50]
predicted_outcome_proba:  0.78
```

Rendered intervention. The template treats the two values as increments: glucose 111 → 261 mg/dL and heart rate 69.2 → 149 bpm, over hours 34–50.

What fails. A value–increment ambiguity produces large simultaneous changes. The raw proposal predicts mortality, yet the assembled record contains `y_cf=0`. A well-formed template has not established a coherent intervention or effect.

Training requirement. Specify one edit, its semantics, held-fixed evidence, and the effect supported by that edit.

(b) A flip can help or harm

Correction (Type 3). Observed outcome: survival. Diastolic blood pressure (DBP) is persistently low: last 52.0, mean 54.3 mmHg. The CF raises DBP from 52 to 70 mmHg.

$$p_{\text{death}}: \quad 0.637 \longrightarrow 0.148$$

The Solver changes from an incorrect death prediction to survival.

Harmful reversal (Type 1). Observed outcome: survival. Last DBP is 49.0 but mean DBP is 65.0 mmHg; motor response and the reported metabolic measurements support the original survival judgment. The CF raises DBP from 49 to 60 mmHg.

$$p_{\text{death}}: \quad 0.029 \longrightarrow 0.593$$

The Solver changes from correct survival to an incorrect death prediction.

Training requirement. A similar edited channel can have different relevance across cases. Check the trajectory and remaining support before proposing a revision.

Figure 8: **Early CF failures motivate grounded, selective revision.** (a) A heuristic proposal becomes an inconsistent intervention. (b) Two pilot cases show opposite accuracy effects despite both changing the Solver's decision. Probabilities refer to the original patient before and after CF-context injection, not measured outcomes of the hypothetical intervention.

realization couples the factual and hypothetical worlds. The Proposer's structured target follows this organization as a language-based audit; the present implementation does not estimate the functions F, the posterior $P_M(u \mid E = e)$, or numerical instance-level counterfactual outcomes from this equation.

Local preservation. Let $\text{Desc}_M(A)$ denote the strict descendants of A. For every $V_j \notin \{A\} \cup \text{Desc}_M(A)$, Equation 1 holds at every fixed u.

Proof. Order the variables topologically. A variable outside A and its descendants has no parent in that set, since such a parent would make it a descendant. Its structural equation is unchanged. Starting from the roots, its parents retain their factual values by induction, and its background inputs retain their values because u is fixed. Applying the same function to the same inputs yields the same output. No independence assumption among the background variables is needed. Descendants may change when their inputs change, so replacing one mechanism does not imply changing only one observed value.

The three CF families act on different parts of this reasoning process. Observe changes the Solver's access to a fixed record; it can be represented by a visibility variable B_j in $O_j = h_j(V_j, B_j)$, where O_j is the exposed observation. Changing B_j does not change the underlying state V_j. Explain compares hypotheses about latent causes using the same evidence, through quantities such as $P(H \mid E = e)$; selecting an alternative explanation is not automatically an intervention on H. Intervene changes an action mechanism in M and permits downstream propagation. These distinctions specify the intended constraints for the `preserved`, `evidence_delta`, and mechanism fields. They do not establish that a generated causal hypothesis is true; the decision analysis below evaluates its effect on the Solver's answer to the original case.

C.3 CAUSAL BENEFIT AND SELECTIVE PROPOSAL

We analyze the causal effect of supplying CF context to a frozen Solver. The unit is an original task instance with a fixed reference answer y. A fixed Proposer procedure generates a candidate from the available evidence and Solver draft, without access to y. The two procedures are $T = 0$, retaining the original decision, and $T = 1$, revising that decision after reading the candidate. Both answer the original question using the original record; the hypothetical clinical outcome is not the evaluation label. Thus the estimand concerns the effect of context on Solver correctness, not the effect of a clinical treatment on a patient.

Let $R_t = \mathbf{1}\{\hat{y}(t) = y\}$ be the potential correctness under procedure t. The Solver, Proposer, and memory state are fixed during this comparison, with no updates between evaluation cases. Deterministic execution defines both responses for each case. With stochastic execution, a specified joint randomization scheme defines their pairing; correction and harm probabilities refer to that scheme. Marginal accuracy differences do not require a unique pairing. Running both procedures on the same labeled cases measures the joint responses under the chosen protocol. Comparing different case sets would not establish that joint distribution. To isolate the CF's content from additional computation, an evaluation can further match the revision procedure with a neutral-context control.

Proposition 1: correction and harm from marginal accuracies. Write $p_t = \Pr(R_t = 1)$, $b = \Pr(R_0 = 0, R_1 = 1)$, and $h = \Pr(R_0 = 1, R_1 = 0)$. Then Equation 3 gives the sharp bounds on b and the corresponding value of h. Here b is the probability that context provision is both necessary and sufficient for correctness, an application of the probabilities-of-causation framework of Tian & Pearl (2000). The bound is a specialization of their Equation 23, rather than a new general identification result. No monotonicity assumption that context cannot harm a correct answer is imposed.

Proof. Let $s = \Pr(R_0 = 1, R_1 = 1)$. The joint response probabilities are

$$\begin{array}{c|cc} & R_1 = 0 & R_1 = 1 \\ \hline R_0 = 0 & 1 - p_0 - p_1 + s & p_1 - s \\ R_0 = 1 & p_0 - s & s \end{array}. \tag{8}$$

Nonnegativity requires

$$\max(0, p_0 + p_1 - 1) \le s \le \min(p_0, p_1). \tag{9}$$

Substituting $b = p_1 - s$ yields the stated bounds, while $h = p_0 - s = b - (p_1 - p_0)$. Every s in this interval defines a valid joint distribution, establishing sharpness. For example, $p_0 = 0.70$ and $p_1 = 0.80$ permit $b \in [0.10, 0.30]$ and $h \in [0, 0.20]$. These illustrative values show why a positive accuracy difference does not determine the harm rate. Paired evaluation additionally estimates the conditional correction rate $b/(1 - p_0)$ and preservation rate $1 - h/p_0$, when the denominators are nonzero.

The correctness-based decomposition applies to any finite answer space. Binary predictions satisfy $\Pr(\hat{y}(1) \neq \hat{y}(0)) = b + h$. In MCQ tasks, there is an additional term:

$$\Pr(\hat{y}(1) \neq \hat{y}(0)) = b + h + \Pr(R_0 = R_1 = 0,\, \hat{y}(1) \neq \hat{y}(0)). \tag{10}$$

Switching between wrong options therefore increases the flip rate without improving accuracy, as illustrated in Figure 4b. Strong/weak CF supervision addresses a different distinction: removing nonessential evidence should retain diagnostic support, whereas removing a diagnostic pillar should weaken it. The latter does not, by itself, identify a new correct option.

Proposition 2: optimal selective proposal. Let X contain all information used by the gate before injection, including the available evidence, original Solver draft, and candidate CF, but excluding the reference answer. Define $b(X) = \Pr(R_0 = 0, R_1 = 1 \mid X)$ and $h(X) = \Pr(R_0 = 1, R_1 = 0 \mid X)$. Consider an inject-or-abstain policy $g(X) \in \{0, 1\}$, where abstention retains the original decision exactly. Relative to abstention, let correction earn $\kappa_+ > 0$, harm cost $\kappa_- > 0$, and unchanged correctness have zero incremental utility. Among policies using X, Equation 4 maximizes expected incremental utility, with ties resolved by abstention.

Proof. Conditioning on X gives

$$\mathbb{E}[U_g \mid X] = g(X)\left[\kappa_+ b(X) - \kappa_- h(X)\right]. \tag{11}$$

The maximizing choice is $g(X) = 1$ exactly when the bracket is positive; taking expectations preserves this pointwise optimum. For equal weights, define $\tau(X) = b(X) - h(X)$. The gated correctness satisfies $R_g = (1 - g(X))R_0 + g(X)R_1$, hence

$$\mathbb{E}[R_{g^*}] - \mathbb{E}[R_0] = \mathbb{E}[\max\{\tau(X), 0\}] \geq 0. \tag{12}$$

This is a population statement under the true conditional response probabilities, not a guarantee for an estimated gate or for each individual case.

To expose the quantities a gate must assess, write

$$\begin{aligned} w(X) &= \Pr(R_0 = 0 \mid X), \\ \alpha(X) &= \Pr(R_1 = 1 \mid R_0 = 0, X), \\ \beta(X) &= \Pr(R_1 = 0 \mid R_0 = 1, X). \end{aligned} \tag{13}$$

Then $b(X) = w(X)\alpha(X)$ and $h(X) = [1 - w(X)]\beta(X)$, so proposing is preferred when

$$\kappa_+ w(X)\alpha(X) > \kappa_- [1 - w(X)]\beta(X). \tag{14}$$

Conditional probabilities within a zero-probability stratum can be assigned arbitrarily because their multiplier vanishes. Original-answer uncertainty alone does not determine this inequality: the prospect of correcting an error and the risk of damaging a correct decision also matter. The instruction targets teach these distinctions through corrective, preservation, and `no_cf` examples. The implemented Proposer does not explicitly estimate all three probabilities, and its reported confidence is not assumed calibrated. Applying the guarantee in Equation 12 to a learned gate would require additional estimation and validation. Training labels may supervise these response types, but they remain unavailable to the Proposer at inference.

C.4 Data construction and input boundaries

The current clinical input serializes the first 24 hours of each patient record into baseline demographics and history, physiological summaries, support and medication states, outputs, devices, transitions, and compressed radiology findings available before the cutoff. Unobserved channels are omitted, empty sections indicate that no information is available, and singleton measurements retain their observation count. The Proposer then receives the target Solver's discrete prediction, continuous probability, and full reasoning trace, followed by the CF audit request. The fields `y_true`, `planner_correct`, `teacher_signal`, and `hospital_expire_flag` are excluded at prompt construction.

Training cases cover errors in evidence use, including omitted findings, incorrect numerical readings, temporal order, support state, and relationships between findings. Corrective, hard preservation, and anchor preservation records are selected from different training stays with related phenotypes or error mechanisms. Matching selects relevant cases; it does not permit copying another patient's measurements into a target. All factual numbers must be traceable to the current patient input, and hypothetical changes must be marked explicitly. If an observed outcome cannot be explained from the prediction-window evidence, the target expresses uncertainty or abstains instead of inventing a corrective mechanism. Data checks cover numerical grounding, hidden-outcome absence, unique stays, and separation of training and test patients. Development audits used test reasoning to identify abstract error types; test patient facts and outcomes were not copied into instruction-tuning messages. Test labels were reserved for scoring, but test reasoning was therefore not wholly unexamined during development.

The XML contract supports six operation values. `action_change` modifies a treatment or support state; `temporal_shift` changes an event's timing or duration; `cause_swap` substitutes a hypothesized explanation of an observed phenomenon; `modality_ablation` changes access to an information channel; `evidence_addition` introduces an explicitly hypothetical finding; and `no_cf` abstains. These operations instantiate the three conceptual families in Section 3.1; the family of a timing edit depends on whether it changes the clinical process or only access to evidence. A non-abstaining target names the operation, states a single local change, identifies the unaffected evidence paths, explains the mechanism, and returns to the actual case to assess whether a revision is warranted. The Solver-visible memory contains only the relevant current facts, the relation to reconsider, and its applicability boundary. It remains empty for abstention.

C.5 Structured causal reasoning target

The target follows the abduction–action–prediction organization of structural counterfactual reasoning (Pearl, 2009a, Chapter 7), adapted to a language-based evidence audit. In `<think>`, the Proposer first reconstructs the explanation linking the observed evidence to the Solver's decision, locates an error or fragile premise, and considers counterevidence. This is the abductive step; a plausible explanation is kept distinct from an observed fact. In `<x_cf>`, it specifies one patient-specific hypothetical change, the evidence held fixed, and the resulting local difference. For an action intervention, only unaffected paths are held fixed; downstream changes belong to the hypothesized effect. In `<y_cf>`, it traces that effect and states the direction, magnitude, and confidence of the change in decision support, without assigning a flipped label. Finally, `<fragility_signal>` compares the affected premise with residual support and states where the argument stops applying. The Proposer returns to the actual record before writing an optional `<memory>` note for the Solver. Abstention leaves memory empty. Figure 9 summarizes the format; Appendix C.6 gives corrective and preservation examples.

Structured clinical CF target

Input: current evidence + Solver draft

```
<cf>
  <cf_type>operation</cf_type>
  <think>evidence; counterevidence</think>
  <x_cf>IF edit; preserve unaffected paths;
   describe evidence change</x_cf>
  <y_cf>mechanism; direction; magnitude;
   confidence</y_cf>
  <fragility_signal>boundary</fragility_signal>
  <memory>note or empty</memory>
</cf>
```

Figure 9: **Structured clinical CF instruction target.** Current evidence and the original Solver draft condition an evidence audit, a hypothetical edit with unaffected paths held fixed, its expected effect, and an applicability boundary. Optional memory returns to the original case; abstention leaves it empty. The schema is condensed and is not a complete training record.

C.6 Corrective and preservation target examples

Figure 10 gives condensed excerpts from the prototype instruction targets. The corrective example audits a factual error in the original draft; the preservation example tests whether removing one evidence channel leaves enough support. These excerpts illustrate the intended training behavior and do not report post-CF Solver performance. The corrective prototype used a 48-hour record, whereas the current clinical configuration uses the 24-hour input boundary described above.

An action-based example in the current training corpus uses a different reasoning test. A patient with worsening hypoxemia receives escalating ventilatory support: PEEP increases from 5 to 12 cmH_2O and inspired oxygen from 0.50 to 0.60, while PaO_2 changes only from 43 to 44 mmHg. The original Solver predicts survival with death probability 0.35. The CF asks what would happen without the escalation, holding the pre-intervention state fixed, then returns to the observed lack of improvement to challenge the Solver's treatment of support as reassurance. A complementary preservation case has brief postoperative vasopressor use, lactate decreasing from 3.3 to 1.0 mmol/L, and improving oxygenation. Its target preserves the original survival prediction rather than treating any vasopressor exposure as evidence of ongoing deterioration. Both examples come from training patients, and no hypothetical outcome is presented as observed.

(a) Corrective target (Type 3)

<think>
The draft attributes high mortality risk to mean MAP 56.2 mmHg. The record instead gives mean MAP 79.1 and last MAP 92 mmHg. Creatinine is 3.7 → 3.7 → 3.6 mg/dL, and GCS motor response remains 6. The low-mean-MAP premise is a factual misreading.

<y_cf>
Correcting that premise weakens the draft's mortality argument. Reassess the contribution of the remaining findings before changing the decision.

<fragility_signal>
The original argument relies heavily on the misread value. A stable creatinine sequence must remain an observation, not be rewritten as an unobserved diagnosis.

<memory>
Recheck the cited MAP against the actual mean and latest value; reconsider the shock argument using the current record.

(b) Preservation target (Type 2)

<think>
The survival judgment is supported by neurological recovery, extubation at hour 17, formal spinal clearance, stable hemodynamics, and an unremarkable metabolic panel.

<x_cf>
IF the GCS trajectory were unavailable, preserve the extubation, clearance, hemodynamic, and laboratory observations. Only neurological visibility changes.

<y_cf>
Removing this channel leaves several observations supporting survival. In this target, the ablation has little effect on the original decision's support.

<fragility_signal>
Robust to the proposed ablation; the remaining support does not warrant a reversal.

<memory></memory>
Preserve the decision; inject no correction.

Figure 10: **Complementary instruction targets.** (a) A corrective reasoning excerpt identifies a misread measurement. (b) A preservation excerpt tests evidence removal and retains the original decision. The examples are condensed and omit fields not needed to show the contrast; they are not complete training records.

C.7 Optimization and training checks

IHM and LOS use separate task-specific adapters and training records with the same Qwen2.5-7B-Instruct backbone. Table 5 reports the documented full-scale clinical configuration. The Qwen chat template places the task contract in the system turn, patient evidence and the original Solver draft in the user turn, and the complete XML target followed by EOS in the assistant turn. Prompt and padding labels are masked with -100. Every remaining assistant token contributes equally to Equation 15; there is no separate loss on `x_cf` or extra weighting of reasoning spans in this configuration.

For input u_i and target token sequence z_i, we minimize the assistant-only autoregressive loss

$$\mathcal{L}_{\text{CF-IT}}(\theta) = -\frac{1}{\sum_i |z_i|} \sum_i \sum_{t=1}^{|z_i|} \log p_\theta(z_{i,t} \mid u_i, z_{i,<t}). \tag{15}$$

Setting	Value
Base model	Qwen2.5-7B-Instruct
Training scale	3,000 records; 2 epochs
Adapter	LoRA rank 16, scaling 32, dropout 0.05
Target projections	Attention: `q, k, v, o`; MLP: `gate, up, down`
Trainable parameters	40,370,176
Learning rate and schedule	10^{-4}; 3% warm-up; cosine decay
Effective batch size	16 (device batch 1, accumulation 16)
Sequence budget	4,096 tokens, retaining the complete XML target
Precision and memory	BF16; gradient checkpointing
Checkpointing	Save each epoch; retain the final adapter
Frozen components	Base backbone and Solver

Table 5: Full-scale clinical Proposer instruction-tuning settings.

Before training, checks confirm the intended projection coverage, trainable parameter count, and complete target visibility. Training-case replay then checks XML completeness, termination, fac-

tual grounding, correction versus preservation, and empty memory for `no_cf`. Replay verifies that the interface has been learned; generalization is assessed separately on held-out cases. Earlier experiments also considered span-weighted assistant losses, but these are distinct from the uniform objective reported here.

C.8 MCQ Supervision and Executable Counterfactuals

The MCQ implementation trains the Proposer to inspect its Solver's reasoning, propose a local hypothetical edit, and explain the resulting decision boundary. It uses executable JSON targets, separately from the clinical XML configuration above.

C.8.1 Training labels and CF roles

Offline, the frozen Solver answers each training question. Correct initial answers are assigned `NOT_CROSSED` (Type 2) for preservation supervision; incorrect answers are assigned `CROSSED` (Type 3) for corrective supervision. These labels describe training roles. Actual correction and harm are measured from downstream paired answers. At inference, the Proposer predicts its verdict without reference answers.

Table 6 distinguishes intended CF effects. A *sufficient corrective* CF addresses decisive evidence to make the reference answer clearly best; an *insufficient corrective* CF concerns the same boundary but leaves the answer non-unique or the case undetermined. Both start from an incorrect Solver answer, whereas a safe-preserving CF retains support for an already correct answer. Human experts verify these sufficiency judgments with Claude assistance. They describe corrective targets, separately from Figure 4's Strong/Weak ablations, which test whether diagnostic support survives evidence removal.

Table 6: **CF roles in MCQ supervision.**

CF role	Initial answer	Intended effect
Safe-preserving	Correct	Preserve support for the reference answer after a non-decisive evidence change.
Sufficient corrective	Incorrect	Address the key reasoning error and make the reference answer clearly best.
Insufficient corrective	Incorrect	Address the same diagnostic boundary while leaving the reference answer non-unique.

C.8.2 Human design and expert verification

Candidate design. Humans design the CF cases, and human experts verify their correctness. Claude Opus 4.6, accessed through AWS Bedrock, assists verification. Each case includes the original evidence, A–D options, Solver answer and reasoning, and the training reference answer. The canonical DiagnosisArena candidate pool contains five safe-preserving CFs for Type 2, or three sufficient-intent and two insufficient-intent corrective CFs for Type 3. Each candidate explains the relevant evidence and diagnostic boundary; the contrast concerns diagnostic sufficiency, not wording or length.

Apply, review, and select. Each candidate is mechanically applied to a fresh copy of the original case. Expert review, assisted by Claude, checks medical coherence, role consistency, answer leakage, grammar, contradictions, and grounding in the edited case. The verification record includes decisive evidence spans and execution checks. Accepted supervision contains one safe-preserving CF for Type 2 or one sufficient/insufficient corrective pair for Type 3; records with no acceptable candidate are excluded. Final records are assembled locally. Solver replay is not used to assign supervision roles.

C.8.3 Executable edits and quality checks

A CF targets one evidence concept using the operations in Table 7. Multiple atomic edits are allowed when the same fact is repeated or a directly conflicting statement must also be updated. This preserves the intended local change while keeping the edited record internally consistent.

Table 7: **Executable CF edits in DiagnosisArena.**

Operation	Required fields	Effect
`delete`	Exact `target_span`	Remove the specified source text.
`add`	Exact `anchor_span`, `position`, `new_text`	Insert explicitly hypothetical evidence at the specified location.
`replace`	Exact `target_span`, `new_text`	Replace a finding with a hypothetical alternative.

Source spans must match the designated `case`, `exam`, or `tests` section verbatim and uniquely. Options remain unchanged; inserted text cannot reveal the reference diagnosis, option letter, or answer-selection instruction. Mechanical checks cover schema, source matches, execution, text integrity, and explicit leakage. Failed outputs may be retried with recorded errors, while raw outputs, applied cases, and review decisions are retained.

Canonical corpus. Of 636 eligible records, 626 remain after construction and verification (98.4%). They comprise 533 fitting records (363 corrective, 170 preservation) and 93 internal-validation records (63 corrective, 30 preservation). The 426 corrective pairs and 200 safe-preserving CFs yield 1,052 CF objects.

C.8.4 Solver-specific CF supervision

Each dataset–model pair uses its own Solver drafts: the same question may need correction for one Solver and preservation for another. Tables 8 and 9 report the supervision pools before oversampling. "Paired Type 3" is a subset of Type 3, not additional records. "Train / val." refers to Proposer fitting and internal validation, not the external benchmark validation set.

Table 8: **DiagnosisArena supervision pools.**

Base model	Type 2 preserve	Type 3 correct	Paired Type 3	Pool	Train / val.
Qwen2.5-7B	200	426	426	626	533 / 93
Llama3-8B	204	428	324	632	569 / 63
MedGemma-4B	255	375	296	630	567 / 63

DiagnosisArena Qwen uses complete sufficient/insufficient pairs for all corrective records. Llama and MedGemma combine paired records with supplementary human-designed, expert-verified examples, with Claude Sonnet 4.6 assisting verification through Bedrock. These examples supply one primary corrective or safe-preserving CF and undergo local executable-edit and leakage checks; they do not require a five-candidate pool. Paired coverage is 324/428 Type 3 records for Llama and 296/375 for MedGemma.

Table 9: **MMMU Business supervision pools.**

Base model	Type 2 preserve	Type 3 correct	Pool	Train / val.
Qwen2.5-7B	352	181	533	480 / 53
Llama3-8B	216	228	444	400 / 44
MedGemma-4B	213	248	461	415 / 46

Business uses one primary CF per record without a paired insufficient contrast. Its supervision covers formulas, variable roles, units, time bases, table or graph interpretation, and option mapping. Both corrective and preservation targets remain part of every model's supervision.

C.8.5 PROPOSER OUTPUT AND OPTIMIZATION

The Proposer learns a structured target rather than just an answer letter (Table 10). A missing contrast is `null`; `UNRESOLVED` provides an abstention option. Reference answers, archive IDs, reference CFs, and post-CF Solver outputs are not Proposer inputs. The generated `proposed_answer` remains an answer-supervised output target, not a supplied test answer.

Table 10: **MCQ Proposer target fields.**

Target field	Learning target
`boundary_verdict`	Predict `CROSSED`, `NOT_CROSSED`, or `UNRESOLVED`.
`proposed_answer`	Predict the proposed A–D answer.
`closest_competitor`	Identify the competing option relevant to the evidence boundary.
`think`	Explain the reasoning error or retained support and cite relevant evidence IDs.
`x_cf.primary`, `x_cf.contrast`	Generate the primary executable CF and an optional insufficient contrast.
`y_cf`	Explain the expected effects on decision support.
`fragility`	Assess the original decision's robustness to the proposed evidence change.

DiagnosisArena optimization. The documented Qwen2.5-7B-Instruct recipe uses 4-bit NF4 QLoRA with double quantization, rank 16, scaling 32, and dropout 0.08. Adapters cover attention `q/k/v/o` and MLP `gate/up/down` projections. The learning rate is 5×10^{-5} with cosine decay. Micro-batch size one and four gradient-accumulation steps give effective batch size four. Training allows up to five epochs with early-stopping patience two; the retained messages fit the 8,192-token budget without target truncation.

Prompt and padding labels are masked with -100, focusing supervision on the assistant CF output. The MCQ loss weights are 10 for `boundary_verdict` and `proposed_answer`, 6 for `closest_competitor`, 3 for `x_cf`, 2 for `y_cf` and `fragility`, and 1.5 for `think`. This emphasizes decision and edit fields while retaining reasoning supervision. These MCQ settings are separate from the uniform clinical XML loss above.

Table 11 reports the model-specific fitting configurations. Training records count distinct records; the next column includes repeats from oversampling, not newly constructed cases. The last column is the stopping epoch, not necessarily the best checkpoint's epoch. These early-stopped configurations are separate from Experiment 5's fixed-five-epoch series.

Table 11: **Model-specific Proposer training settings.**

Base model	Training records	After oversampling	Stopping epoch
DiagnosisArena			
Qwen2.5-7B	533	703	4
Llama3-8B	569	749	3
MedGemma-4B	567	567	3
MMMU Business			
Qwen2.5-7B	480	648	3
Llama3-8B	400	400	3
MedGemma-4B	415	415	3

C.8.6 WHAT REACHES THE SOLVER

Strict edit replay. The CF is applied to the case, examination, or test text. The Solver receives the edited sections and original A–D options, without the Proposer's reasoning audit, proposed answer, expected answer, reference answer, or record ID. This evaluates the Solver on the edited input.

Full closed-loop evaluation. An accepted current-instance CF is converted into evidence-boundary context for the original question. The frozen Solver uses this guidance to reconsider its initial answer. Mechanical checks establish executability, retrieval-based verification assesses domain support, and the validation-selected policy controls adoption. Full-system correction and harm counts measure this revision of the original answer, not the offline sufficiency label.

C.8.7 Examples of Counterfactual Supervision

Preservation. In one canonical DiagnosisArena example, a hypothetical replacement makes an imaging finding more alarming: remodeling and thinning become expansile remodeling with cortical destruction. The example retains the original histological evidence, including rare mitoses and no cytologic atypia, as support for preservation. The intended lesson is to weigh a changed finding against the evidence that still supports the answer.

Corrective sufficiency contrast. Another example concerns a Solver that underweights a *KRIT1* variant and familial brain hemorrhage. The sufficient CF makes inheritance explicit and introduces hypothetical shared-variant evidence in relatives. The insufficient CF changes only the lack of improvement after prolonged compression-stocking use. Expert verification, assisted by Claude, identifies the former as sufficient and the latter as relevant but insufficient for the diagnostic boundary. These are supervision examples; hypothetical findings are not added to the factual record. Together, they illustrate the intended distinction between justified reconsideration and preservation.

D Feedback from Verifier and Proposer

Verifier construction and coverage. The clinical Verifier first checks target ranges and deterministic safety constraints, then extracts the proposed condition, intervention, and relevant evidence. A topic router selects prebuilt Chroma collections. The configured PubMed Central (PMC) collections cover ICU prognosis, hemodynamics and vasopressors, consciousness, infection, respiratory support, hematology, metabolism, cardiac rhythm, and renal function. Acid–base queries share the metabolic collection. Three additional collections contain DailyMed drug labels, Surviving Sepsis Campaign guidelines, and American College of Radiology appropriateness criteria. These sources cover clinical literature, drug safety, sepsis management, and imaging decisions; they are not an exhaustive knowledge base for every specialty or task.

The implementation embeds condition, intervention, outcome, and critical-feature queries with Qwen3-Embedding-0.6B. It retrieves up to four chunks per query and collection, deduplicates them, and limits the judge's evidence bundle to 24 items, retaining source identifiers. The Qwen2.5-7B judge cites retrieved evidence and returns `PASS` when the assessed claims are supported, `FLAG` for incomplete support without contradiction, and `REJECT` for contradicted claims. Intervention plausibility and a hypothesized outcome can be evaluated separately; a plausibility verdict does not establish the unobserved outcome as fact. The three-level plausibility score is $r_V = 1$ for `PASS`, 0.5 for `FLAG`, and 0 for `REJECT`. Deterministic safety rejection blocks the edit, and the grounded verifier pipeline additionally penalizes unsafe or insufficiently supported proposals. These verifier-side checks are distinct from the downstream flip-warrant gate.

Memory organization and use. The bank stores typed factual-case, verified-CF, and gap records with task/evidence tags, source identity, and verifier metadata. Tag overlap and text similarity retrieve a small relevant subset. Before reuse, a note is reduced to a conditional mechanism, its applicability requirements, counterexamples, and provenance. The Solver prompt separates current observations, the original Solver argument, the current-instance CF review, and retrieved training experience. Donor measurements are not presented as current observations. The current-instance note can refer to observed facts but keeps hypothetical changes explicit. If no usable note passes the gate, the Solver receives a fixed instruction to retain its initial assessment. Memory retrieval and the current-instance reviewer note are distinct components; the latter can be used without a cross-instance bank.

Training reward and the reported configuration. The reported clinical RL run uses an additive outcome-guided reward with weights $(\lambda_C, \lambda_U, \lambda_V) = (2, 1, 0)$, without the correctness veto in

Equation 5:

$$R_{\text{reported}} = 2r_C + \text{clip}\big(\log p^c(y) - \log p^0(y), -2, 2\big) + 0.2r_{\text{direction}} + 0.1r_{\text{schema}} - r_{\text{violation}}. \tag{16}$$

For the before/after correctness pairs $(0,1)$, $(1,0)$, $(1,1)$, and $(0,0)$, r_C equals 3, -4, 0.75, and -1, respectively. Probabilities are clipped to $[10^{-4}, 1-10^{-4}]$ before taking logarithms. The confidence term is a reduction in true-answer log loss, not an entropy bonus: becoming more confident in a wrong answer does not earn credit. In the binary clinical configuration, the decision threshold is 0.5.

The direction indicator is one when the reviewer note contains an explicit `INCREASE`, `DECREASE`, or `MAINTAINED` instruction. It rewards an explicit format, not medical validity. The schema indicator requires nonempty reasoning, edit, effect, and memory fields with no prohibited outcome disclosure. A malformed or label-leaking output is replaced with the fixed retain note and incurs $r_{\text{violation}} = 2$. If the Solver probability cannot be parsed, the implementation falls back to $p^c = p^0$ and applies the same penalty. Otherwise the violation penalty is zero. Reference labels are kept in reward-side metadata and never supplied to the Proposer prompt.

The Proposer starts from its instruction-tuned checkpoint and samples four candidates per training instance. Each candidate is scored through the frozen Solver, and group-relative policy updates compare their rewards. The documented clinical run uses 512 training patients and 128 disjoint validation patients; test cases are excluded from RL training. The separate DiagnosisArena RL implementation retains the same correctness-transition rewards but uses a clipped change in the correct-option logit margin for confidence and task-specific JSON/edit validity checks.

D.1 MCQ RAG Verifier and Model-Specific Indexes

Role and inputs. Retrieval serves different roles in Experiment 8: the retrieval-only baseline supplies context directly to the Solver, whereas the retrieval stage described here supplies evidence to a Verifier assessing a newly proposed CF. Mechanical checks establish executability, the Verifier assesses domain support, and the acceptance policy decides whether the guidance reaches the frozen Solver.

The Verifier receives the original question and options, edited input, CF operation and claimed effect, Proposer rationale, and retrieved evidence. It does not receive reference answers, post-CF Solver answers, correction/harm labels, or test scores. It returns an evidence-support judgment rather than a benchmark answer.

D.1.1 Model-specific retrieval indexes

Within each dataset, all three configurations use the same knowledge texts, independently encoded with their assigned embedding model (Table 12). Vectors are L2-normalized and stored in separate Chroma indexes; documents, IDs, and metadata remain aligned. Queries follow the corresponding format: Qwen uses an instruction with `Query:`, Nemotron uses `query:`/`passage:`, and EmbeddingGemma uses its task and document prefixes. Shared texts keep the source knowledge comparable, while each index uses its own embedding space; the retrieved passages need not be identical.

Table 12: **Embedding models for MCQ retrieval.**

Base model	Embedding checkpoint	Dim.
Qwen2.5-7B	`Qwen/Qwen3-Embedding-0.6B`	1,024
Llama3-8B	`nvidia/llama-nemotron-embed-1b-v2`	2,048
MedGemma-4B	`google/embeddinggemma-300m`	768

D.1.2 Knowledge bases

Medical knowledge. DiagnosisArena uses 39 collections and 84,572 chunks: 36 topical PubMed Central collections plus DailyMed, Surviving Sepsis Campaign guidelines, and American College

of Radiology Appropriateness Criteria. Topics span major clinical areas. Llama and MedGemma re-encode the same source passages, IDs, and metadata without adding benchmark answers or Solver outputs. Table 13 reports counts per model-specific index; these are the MCQ indexes, not the earlier IHM/LOS configuration.

Table 13: **Knowledge-base size per MCQ index.**

Knowledge corpus	Collections	Chunks
DiagnosisArena medical	39	84,572
MMMU Business	5	16,302

Business textbooks. Six OpenStax textbooks cover financial and managerial accounting, economics, finance, management, and marketing. The accounting books share one collection. Fixed PDFs are stored with SHA-256 hashes and converted using `pdftotext -layout`. Deterministic cleaning removes front matter, assessment and answer-key pages, and repeated page boilerplate. Text is split into approximately 900-character chunks with 150-character overlap, without joining across removed-page gaps.

To reduce benchmark overlap, question and option text from all 967 Business items is compared with the corpus using normalized exact 12-word n-grams. Twenty-four overlapping chunks are removed. Benchmark text is used only for exclusion; reference answers and Solver outputs are not added to the corpus. Table 14 reports the retained text shared by the three indexes.

Table 14: **Business corpus after cleaning.**

Business collection	Chunks
`business_accounting`	4,189
`business_economics`	4,767
`business_finance`	2,789
`business_management`	2,182
`business_marketing`	2,375
Total	16,302

D.1.3 Query construction and retrieval

Medical retrieval. Four queries cover the original condition, the intervention's diagnostic meaning, the claimed outcome or preservation, and the decisive feature. A topic router selects collections by disease, organ system, drug, or imaging topic. The base configuration retrieves up to four chunks per query and collection, deduplicates them, and supplies at most 24 passages with source IDs to the Verifier.

Business retrieval. The question's subject selects one collection. Queries cover the required concept or formula, the edit's effect, the claimed conclusion, and the decisive quantity. Retrieval applies a distance threshold of 0.75, deduplicates by chunk ID, and selects up to three qualifying passages with the smallest distances. The displayed score $1/(1+d)$ ranks stored retrieval distances d; it is not a calibrated probability.

D.1.4 Verifier judgments and safeguards

Each configuration uses its corresponding frozen base model in a Verifier call at temperature zero. It assesses the edited condition, intervention, and claimed consequence. Medical `PASS`/`SUPPORT` indicates relevant support; `FLAG`/`INSUFFICIENT` indicates partial, indirect, or non-unique support; `REJECT`/`CONTRADICT` indicates contradiction. Missing evidence produces uncertainty, not automatic contradiction.

Business verification also checks formulas, variable roles, units, time bases, aggregation levels, and linked totals or ratios. A text edit can be executable yet violate an accounting identity. `PASS` requires support for the condition, intervention, and outcome; contradiction gives `REJECT`, and

incomplete or non-unique support gives `FLAG`. Unparseable final outputs are recorded as `ERROR`. Deterministic formatting repairs do not change verdicts or business claims; at most one strict-JSON retry is permitted, with raw responses retained.

D.1.5 Acceptance and reproducibility

A verdict does not overwrite the Solver answer. Each acceptance policy is selected on validation and frozen for test. It may require support, veto an explicit rejection, or ignore the RAG verdict; a non-adopted proposal leaves the initial answer unchanged. Corpus and embedding manifests, source and retrieved-passage IDs, raw judgments, and retrieval summaries make the decisions traceable. The policy results in Section E.8 evaluate how this evidence signal is used. This separates an executable edit from a domain-supported claim and from a proposal selected for downstream use.

E Extended RQ2 Analyses

E.1 Robustness across base models

Table 15: **Extended robustness and baseline comparisons (RQ2).** Metrics follow Figure 6, with additional clinical baselines. LOS is regression. A dash marks a matched result still in progress. Section E.6 specifies the MCQ acceptance policies, retrieval baselines, and Business output processing.

Dataset / task	Metric	Base model	Zero-shot	RAG	GSEM	MILM	**Ours**
MIMIC-IV / IHM	AUROC	Qwen2.5-7B	0.7000	–	–	0.7235	**0.7553**
		Llama3-8B	0.7025	–	–	0.7089	**0.7466**
		MedGemma-4B	0.7117	–	–	0.7157	**0.7374**
MIMIC-IV / LOS	MAE ↓	Qwen2.5-7B	69.76	–	–	59.77	**58.60**
		Llama3-8B	59.42	–	–	65.26	**55.20**
		MedGemma-4B	56.29	–	–	60.14	**55.42**
DiagnosisArena	Acc. (%)	Qwen2.5-7B	34.8	26.8	26.8	–	**66.7**
		Llama3-8B	30.4	29.7	31.2	–	**61.6**
		MedGemma-4B	37.7	33.3	33.3	–	**63.0**
HoVer	Acc. (%)	Qwen2.5-7B	54.45	–	–	–	**61.92**
		Llama3-8B	59.00	–	–	–	**63.22**
		MedGemma-4B	57.07	–	–	–	**58.86**
MMMU Business	Acc. (%)	Qwen2.5-7B	61.4	53.8†	47.6	–	**63.4**
		Llama3-8B	42.1	42.8‡	26.9	–	**51.7**
		MedGemma-4B	35.9	37.2‡	33.1	–	**47.6**
FuseMoE (no LLM)			IHM AUROC 0.7246; LOS MAE 61.33				

E.2 Proposer–Solver decomposition

The SFT and SFT+GRPO rows use the same configurations as Table 2; the heuristic row adds a training-free comparison.

Solver	Proposer context	Accuracy	AUROC	AUPRC
Qwen	None (zero-shot)	0.5839	0.7000	0.2252
Qwen	Heuristic CF	0.5839	0.6968	0.2225
Qwen	SFT Proposer	0.7372	0.7543	0.2985
Qwen	SFT + GRPO	**0.8417**	**0.7553**	**0.3237**
Qwen SFT	None	**0.8786**	0.7584	0.3128
Qwen SFT	Solver-specific SFT	0.8630	**0.7595**	**0.3212**

Table 16: Full-test IHM decomposition of Proposer instruction tuning, outcome-guided RL, and direct Solver fine-tuning. Comparisons are controlled within each Solver pair.

E.3 CF SUPERVISION SCALE AND OPERATION DISTRIBUTIONS

Figure 11(a) reports the IHM data-scale ablation. One thousand examples primarily teach the output behavior, while 2k and 3k examples provide enough corrective and preservation boundaries to improve accuracy and ranking metrics jointly. Figure 11(b) reports adopted CFs by unified edit-level type across tasks.

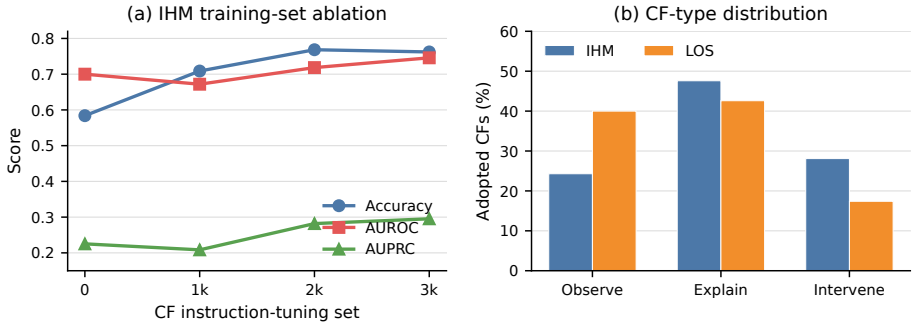


Figure 11: Analysis of CF supervision. (a) IHM performance with zero-shot inference and CF Proposers trained on 1k, 2k, and 3k examples. (b) Adopted CFs by unified edit-level type (Appendix E.4) differ across tasks. IHM excludes 1,460 `no_cf` outputs ($n = 9{,}011$ actionable CFs); LOS reports its full evaluation ($n = 14{,}825$).

E.4 UNIFIED CF TYPES ACROSS DATASETS

Each dataset exposes CF operations in its own output contract: clinical XML names semantic operations (Appendix C.4), whereas MCQ tasks name executable span edits (Table 7). Table 17 maps both onto the edit-level categories used in Figure 5b. These categories describe the operation, while Observe, Explain, and Intervene describe its causal purpose. A cause swap adds an alternative explanation for fixed evidence (Explain), so it counts as Add. Withholding or adding evidence, including changing when a finding is observed, tests Observe; changing a treatment or support action tests Intervene. Span edits test Observe and are reported as Mixed when they combine different atomic operations. Clinical CFs use one local edit. The mapping uses operation labels, and `no_cf` abstentions are excluded.

Table 17: **Unified edit-level CF types and their dataset-specific operations.**

Edit type	What changes	Clinical (MIMIC-IV)	Span edit (MCQ)
Remove	An evidence item or channel is withheld	`modality_ablation`	`delete`
Add	Hypothetical evidence or an alternative explanation is introduced	`evidence_addition`, `cause_swap`	`add`
Replace	A finding, observation time, treatment, or support state is substituted	`action_change`, `temporal_shift`	`replace`
Mixed	Several atomic edits of different types	– (single edit)	combination

E.5 CROSS-MODEL IHM CONFIGURATIONS

Table 18 gives the full IHM metric vectors for the configurations in Table 15, with a separately trained Proposer for each base model. Its Qwen row matches the SFT row in Table 2. Qwen and Llama each use 10,471 cases in both runs; MedGemma uses 10,336 zero-shot and 10,299 CF cases. Figure 3(c) instead reports an earlier Qwen Proposer configuration trained on the same 3,000 CFs: accuracy 0.7217, AUROC 0.7518, and AUPRC 0.2881 on 10,470 CF cases, compared with 10,471 zero-shot cases.

Base model	Context	Accuracy	AUROC	AUPRC
Qwen2.5-7B	None / learned CF	0.5839 / **0.7372**	0.7000 / **0.7543**	0.2252 / **0.2985**
Llama3-8B	None / learned CF	0.6065 / **0.7424**	0.7025 / **0.7466**	0.2268 / **0.2830**
MedGemma-4B	None / learned CF	0.7620 / **0.8027**	0.7117 / **0.7374**	0.2344 / **0.2906**

Table 18: Full IHM metrics for Table 15.

E.6 Cross-Task and Cross-Model MCQ Results

Comparison settings. Experiment 8 compares zero-shot inference, retrieval without evolution, GSEM, and the full CF-guided system. DiagnosisArena uses static Top-1 CF-memory retrieval. On Business, † marks Qwen's Top-3 textbook retrieval and ‡ marks Llama and MedGemma's legacy RAG results. These baseline contexts are distinct from the full system's RAG Verifier (Section D.1). GSEM is an external baseline, not a within-framework removal of CF generation. Table 19 uses the recorded runs plotted in Figure 6, not seed averages. Business accuracy compares complete systems, including model-specific numerical recalculation or output recovery. Corrected and Harmed measure the subsequent paired CF revisions within each run.

DiagnosisArena acceptance policies. The Qwen result (66.7%, 47 corrections, three harms) uses the validation-selected "Ignore RAG signal" policy in Table 22: at least one candidate vote and reliability at least 0.96, with mechanical checks active. RAG verdicts do not determine adoption in this configuration. Llama and MedGemma use validation-calibrated gates requiring an executable `CROSSED` proposal, a reliability threshold (approximately 0.8785 and 0.8175, respectively), agreement between the revised Solver answer and the Proposer answer, and calibrated confidence changes of at least -0.05 and -0.15. These policies do not require a positive RAG verdict. Section E.8 compares alternative uses of Verifier feedback and their correction–preservation tradeoffs.

Table 19: **MCQ results across tasks and base models.**

	Accuracy (%)				Answer changes	
Base model	Zero-shot	RAG	GSEM	**Ours**	Corrected	Harmed
DiagnosisArena ($n = 138$)						
Qwen2.5-7B	34.8	26.8	26.8	**66.7**	47	3
Llama3-8B	30.4	29.7	31.2	**61.6**	43	0
MedGemma-4B	37.7	33.3	33.3	**63.0**	35	3
MMMU Business ($n = 145$)						
Qwen2.5-7B	61.4	$53.8^{\dagger}$	47.6	**63.4**	4	1
Llama3-8B	42.1	$42.8^{\ddagger}$	26.9	**51.7**	23	13
MedGemma-4B	35.9	$37.2^{\ddagger}$	33.1	**47.6**	3	1

Paired gains across models. The full system has the highest recorded accuracy among the listed methods in each dataset–model setting. Table 20 uses each full-system run's initial predictions, which can differ from a separate zero-shot run because of output processing. Every row satisfies $B + C - H = N_{\text{correct,final}}$, with gains computed from unrounded counts. DiagnosisArena gains range from 23.2 to 31.9 percentage points, with 35–47 corrections and 0–3 harms. Llama corrects 43 errors and preserves all initially correct answers in this run.

Business system-level gains over the separate zero-shot runs are 2.1, 9.7, and 11.7 percentage points for Qwen, Llama, and MedGemma, computed from unrounded counts. The corresponding paired CF gains are 2.1, 6.9, and 1.4 points. For MedGemma, output recovery raises the starting count from 52 to 67 correct answers; CF revision then yields $67 + 3 - 1 = 69$. Thus the main comparison reports total system gains, while Table 20 measures CF's incremental effect. Across all six paired runs, corrections exceed harms, with the largest gains on DiagnosisArena.

Table 20: **Paired correction and preservation results.** Initial correct is measured after each run's output processing; Gain measures the subsequent CF revision step. Separate zero-shot accuracies appear in Table 19.

Base model	Initial correct	Final correct	Corrected	Harmed	Gain (pp)
DiagnosisArena ($n = 138$)					
Qwen2.5-7B	48	92	47	3	+31.9
Llama3-8B	42	85	43	0	+31.2
MedGemma-4B	55	87	35	3	+23.2
MMMU Business ($n = 145$)					
Qwen2.5-7B	89	92	4	1	+2.1
Llama3-8B	65	75	23	13	+6.9
MedGemma-4B	67	69	3	1	+1.4

E.7 Adopted CF Operations and Evidence-Use Errors

Experiment 9 examines the DiagnosisArena consensus configuration from Experiment 4. Of 41 adopted interventions, 39 correct a wrong answer, two switch between wrong options, and none harms an initially correct answer (Table 21). The 95.1% correction share uses adopted interventions as its denominator, not all test questions. Replace is the most frequent primary operation; these frequencies describe adoption rather than an operation-specific ablation.

Table 21: **Adopted DiagnosisArena CF interventions.**

Category	Count	Share (%)
Adopted outcomes ($n = 41$)		
Wrong → correct	39	95.1
Wrong → different wrong	2	4.9
Correct → wrong	0	0.0
Primary CF operation ($n = 41$)		
Replace	19	46.3
Add	11	26.8
Mixed	11	26.8

Evidence-use errors dominate contributing memories. Among 104 contributing seed-memory records, 46 concern missed evidence and 36 concern misweighted evidence: together, 78.8%. The remaining 13 concern insufficient specificity and nine incorrect medical inference (Figure 12). Multiple seed memories may contribute to one intervention; the 104 records are not independent test questions. This distribution is consistent with the intended role of CF guidance: prompting the Solver to reconsider which findings support its answer and how much weight they deserve.

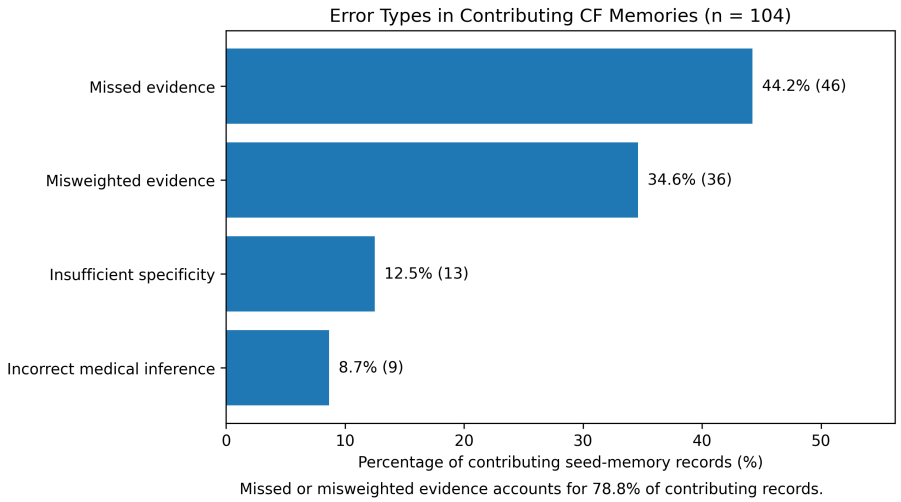


Figure 12: **Error types in contributing CF memories.** The 104 DiagnosisArena seed-memory records include 44.2% missed evidence and 34.6% misweighted evidence.

E.8 CF Acceptance Policies Using RAG Verifier Feedback

Experiment 7 compares how RAG judgments enter CF-acceptance decisions with Qwen2.5-7B on DiagnosisArena. All policies share the three-seed candidate pool, retrieved evidence, frozen Solver replays, and Verifier judgments. Mechanical checks remain active. Each policy selects its own reliability, vote, and margin conditions on 137 validation questions, then is frozen for the 138-question test. This compares complete acceptance policies rather than a single Verifier on/off switch.

Policies ignore RAG judgments, veto an explicit `REJECT`, or accept `PASS`/`FLAG` verdicts in combination with their other conditions. `FLAG` denotes incomplete support, not full verification. The two named single-vote variants use the shared candidate pool, not newly trained single-seed Proposers. In Table 22, accuracy and preservation are percentages. Adopted, Corrected, Harmed, and Preserved refer to test; preservation is measured among the 48 initially correct answers. Each row is one policy result, not an average across independent policy runs.

Table 22: **RAG-Verifier acceptance policies on DiagnosisArena.**

Policy	Val. acc.	Test acc.	Adopted	Corrected	Harmed	Preserved (%)
Ignore RAG signal	67.9	**66.7**	55	47	3	93.8
Veto explicit REJECT	56.9	52.9	32	27	2	95.8
Positive-only (PASS/FLAG)	52.6	48.6	21	19	**0**	**100.0**
Single-vote + REJECT veto	67.2	**66.7**	51	46	2	95.8
Single-vote + positive support	66.4	64.5	44	41	**0**	**100.0**

Correction and preservation. The "Ignore RAG signal" row supplies the Qwen headline result in Figure 6 and Table 19. The single-vote REJECT-veto policy matches the 66.7% accuracy of ignoring RAG judgments. Its 46 corrections and two harms yield the same net gain of 44 correct answers as 47 corrections and three harms, with 51 rather than 55 adopted interventions. The single-vote positive-support policy reaches 64.5%, corrects 41 errors, and preserves all initially correct answers in this test run. These outcomes show how validation-selected policies can retain substantial correction benefits while choosing a more conservative revision pattern. The other policies provide the full comparison: benefits depend on how the verdict and selection conditions are combined, not simply on whether a Verifier is present.

F Comparison with Direct Solver Fine-Tuning

Experiment 11 asks where to apply supervision: directly to the Solver, or to a Proposer guiding a frozen Solver. Data-matched Solver-SFT uses the same training question IDs as the corresponding Proposer. On DiagnosisArena, the baseline uses answer-only SFT. On Business, Qwen and MedGemma use SFT adapters evaluated with their zero-shot reasoning and parsing protocol; Llama uses reasoning-aligned SFT on the same 400 training questions. The Ours column repeats the best recorded complete Experiment 8 runs, not seed means. Business results include numerical recalculation or output recovery, so this remains a system-level comparison (Table 23).

Table 23: **Comparison with direct Solver fine-tuning (accuracy, %).**

Base model	Zero-shot	Solver-SFT	Ours
DiagnosisArena ($n = 138$)			
Qwen2.5-7B	34.8	**74.6**	66.7
Llama3-8B	30.4	**79.0**	61.6
MedGemma-4B	37.7	**70.3**	63.0
MMMU Business ($n = 145$)			
Qwen2.5-7B	61.4	62.1	**63.4**
Llama3-8B	42.1	49.7	**51.7**
MedGemma-4B	35.9	44.1	**47.6**

Both approaches improve over the listed zero-shot results. Solver-SFT is higher on DiagnosisArena, while the frozen-Solver systems are higher on Business. The comparison supports a practical alternative to updating the decision-making model: learning CF guidance can yield useful full-system gains while leaving Solver parameters unchanged.

G Additional Results for RQ3

G.1 Patient-specific CF curriculum

We define three task-difficulty levels by prediction horizon: from the same first-24-hour record, the frozen Solver predicts mortality within 24 (Easy), 72 (Medium), or 168 hours (Hard). On a fixed paired cohort of 2,000 patients, zero-shot AUROC decreases from 0.7697 to 0.7305 and 0.7142, establishing increasingly difficult tasks without changing the observed input. Within each task, we further partition patients into Easy, Medium, and Hard tertiles using only the frozen zero-shot confidence. Starting with no CF, we progressively make one patient-specific CF available to the Easy, then Medium, and finally Hard subgroup; patients not yet covered retain their exact zero-shot probability. Thus each round expands the applicable CF context without updating either model or revealing additional patient observations. Because mortality prevalence rises with the horizon, we compare tasks mainly with AUROC and balanced accuracy, which do not depend on prevalence, and report Macro-AUPRC and Macro-F1 as secondary metrics.

Figure 13 shows little or slightly negative change when only Easy experience is added, followed by consistent gains as Medium and Hard experience becomes available. At the final round, AUROC reaches 0.8162, 0.7962, and 0.7804 for Easy, Medium, and Hard prediction, corresponding to gains of 0.0465, 0.0657, and 0.0662. Balanced-accuracy gains similarly increase from 0.0307 on Easy to 0.0481 and 0.0493 on Medium and Hard. The ordered increase shows that additional CF context contributes little to already easy decisions but becomes increasingly useful for harder patient subgroups and harder tasks. The AUROC gap between the 24-hour and 168-hour tasks shrinks from 0.0555 without CF to 0.0358 with full CF. The Medium and Hard rounds account for at least 91% of the final AUROC gain at every horizon, and at 72 hours CF on the Easy subgroup alone lowers AUROC by 0.0076 and balanced accuracy by 0.0077. Full CF context also improves Macro-AUPRC ($+0.060$ to $+0.070$) and Macro-F1 ($+0.084$ to $+0.097$) at every horizon, but these prevalence-dependent metrics do not show larger gains at longer horizons.

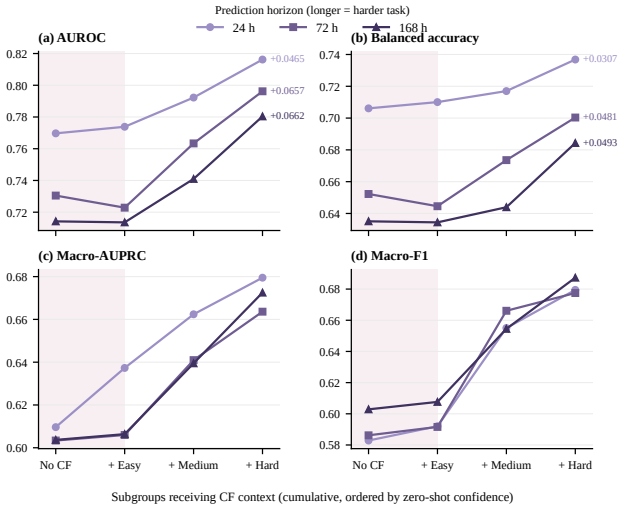


Figure 13: **Difficulty-ordered accumulation of CF context.** Patients are grouped by frozen zero-shot confidence, and CF context is admitted cumulatively from Easy to Medium to Hard. Adding Easy context alone has little effect, whereas the larger gains emerge after Medium and Hard cases are included, especially for the longer-horizon tasks. Darker lines are longer (harder) horizons; the shaded region marks the Easy-only round, and labels in (a) and (b) give the final gain over no CF.

G.2 Held-out CF memory transfer

Setup. All cases that already have a CF form the memory pool (517 source cases); the remaining cases are split into a validation set (177) and a frozen test set (710). Train, validation, and test share no record IDs, no test case contributes its own CF or memory, and no memory source appears in the test set. The test set contains 480 MMMU and 230 MMMU-Pro questions (207 Easy, 327 Medium, 176 Hard). We compare an empty bank with four banks built from source cases of increasing Solver confidence: Low (189 memories), Low+Medium (338), All (517), and High-only (179). Each non-empty condition retrieves the top-1 memory. The Solver, prompt, parser, and safety gate are frozen before the test run, and the Low bank, best on validation, is the main configuration; the other banks are ablations.

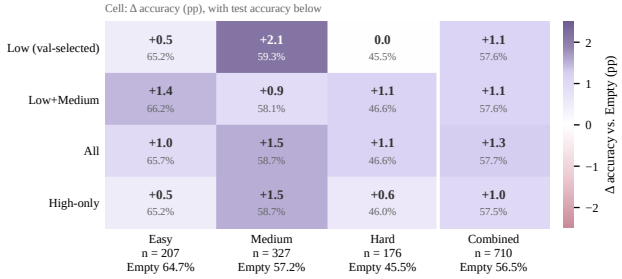


Figure 14: **Held-out accuracy with CF memory.** Each cell shows the change in test accuracy over the empty bank (bold) and the resulting test accuracy (below).

Accuracy by difficulty. All four banks improve overall accuracy over the empty bank (56.5%), reaching 57.5–57.7% and finishing within two test questions of each other (Figure 14). Growing the bank from 189 to 517 memories therefore adds little. The main bank gains most on Medium questions (+2.1 pp) and leaves Hard questions unchanged; the larger banks add two correct Hard answers (+1.1 pp).

Corrections, damage, and uncertainty. The net gain hides changes in both directions (Figure 15). Each bank fixes roughly two wrong answers for every correct answer it breaks; the main bank fixes

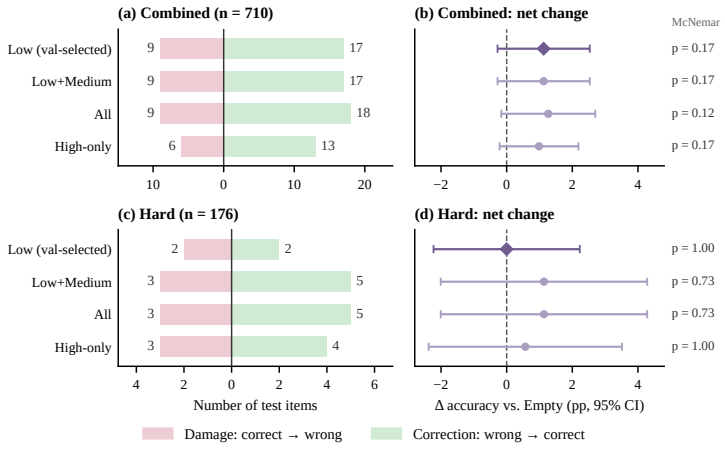


Figure 15: **Memory fixes and breaks answers, and the net gain is uncertain.** (a, c) Corrections (wrong → correct) and damage (correct → wrong) relative to the empty bank. (b, d) Net change in test accuracy with 95% paired confidence intervals and exact McNemar p-values.

17 and breaks 9, and the High-only bank makes fewer changes in both directions (13 and 6). All confidence intervals include zero, both overall and on Hard questions.

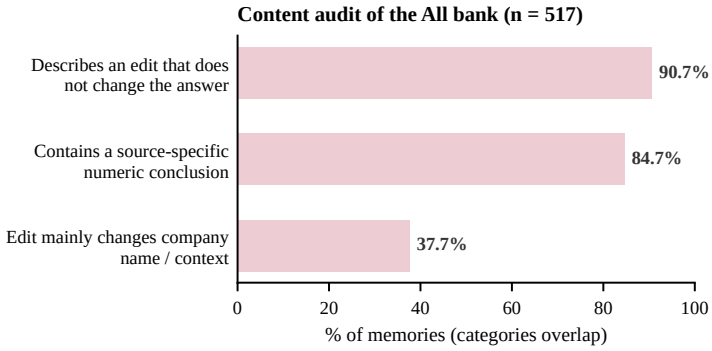


Figure 16: **Content audit of the full memory bank.** Share of the 517 memories with each property; a memory can fall into several categories.

Memory content. An audit of the full bank (Figure 16) shows that 90.7% of memories describe an edit that did not change the answer, 84.7% contain numeric conclusions tied to their source question, and 37.7% mainly change a company name or context. Such memories give the Solver little it can reuse on a new question, so the bottleneck is the information in each memory rather than the number of memories.

Limitations. Memory sources were selected because a CF already existed, not at random, and all memories come from MMMU while the test set also includes MMMU-Pro.

H Analysis of Memory Evolution

We specialize the principle of high-confidence policy improvement (Thomas et al., 2015) to the memory trajectory in Section 3.2. The result compares complete memory snapshots, including their effects on retrieval and Solver decisions. Passing the existing gate for an individual CF does not imply that the resulting bank improves accuracy.

Assumptions for Proposition 3. Let $\mathcal{H}$ contain the initial model configuration, evolution data, and all choices and randomness used to construct $\mathcal{M}_0, \ldots, \mathcal{M}_R$. Successive rounds may depend on earlier memories and feedback from the evolution pool. The model weights remain fixed, and each round adds notes from disjoint new cases. Proposer optimization precedes these rounds, and the teacher is used only for the initial instruction-tuning warm-up. Fix the round budget R and the S target strata before evaluation. For each stratum s, let $\mathcal{D}_s$ be its fixed task distribution and $\mathcal{V}_s = \{Z_{si}\}_{i=1}^{n_s}$ an i.i.d. sample independent of $\mathcal{H}$. Evaluation inputs, labels, and scores do not determine memory construction, retrieval settings, or subsequent evolution choices. Memory is read-only during evaluation; the context budget and the retrieval, selection, and ordering rules are fixed before evaluation. Use deterministic decoding or a fixed stochastic protocol with independent randomness across sampled units; all expectations below include that randomness. Correlated records from one patient are not separate independent units. Different strata may share units, and rounds reuse the same cases; independence is required within each stratum, not across rounds or strata.

For $Z = (x, y)$, write $C_r(Z) = \mathbf{1}\{\hat{y}(x; \mathcal{M}_r) = y\}$ and, conditional on $\mathcal{H}$, define

$$\begin{aligned} a_{r,s} &= \mathbb{E}_{Z \sim \mathcal{D}_s}[C_r(Z)], \\ \Delta_{r,s} &= a_{r,s} - a_{r-1,s}, \\ \widehat{\Delta}_{r,s} &= \frac{1}{n_s} \sum_{i=1}^{n_s} \big[C_r(Z_{si}) - C_{r-1}(Z_{si})\big]. \end{aligned} \tag{17}$$

Under the specified pairing, let $b_{r,s} = \Pr(C_{r-1} = 0, C_r = 1)$ and $h_{r,s} = \Pr(C_{r-1} = 1, C_r = 0)$ within stratum s. As in Equation 2, $\Delta_{r,s} = b_{r,s} - h_{r,s}$ and $\widehat{\Delta}_{r,s} = \widehat{b}_{r,s} - \widehat{h}_{r,s}$. The bound therefore asks whether the observed excess of corrections over harmful reversals exceeds sampling uncertainty.

Proposition 3: simultaneous improvement bounds. Under these assumptions, for any prespecified $\delta \in (0, 1)$, the bounds in Equation 6 satisfy

$$\Pr(\Delta_{r,s} \geq L_{r,s} \text{ for all } r = 1, \ldots, R,\ s = 1, \ldots, S) \geq 1 - \delta. \tag{18}$$

On this event, any transition with $\min_s L_{r,s} > 0$ improves population accuracy in every target stratum. If the condition holds at every round, every stratum's population accuracy increases along the trajectory. No such increase is assumed in establishing the bound.

Proof. Condition on $\mathcal{H}$, which fixes the memory trajectory independently of the evaluation samples. For any (r, s), the differences $D_{si} = C_r(Z_{si}) - C_{r-1}(Z_{si})$ are independent across i, have mean $\Delta_{r,s}$, and lie in $[-1, 1]$. Hoeffding's inequality (Hoeffding, 1963) gives

$$\Pr\Big(\widehat{\Delta}_{r,s} - \Delta_{r,s} > \epsilon \mid \mathcal{H}\Big) \leq \exp\left(-\frac{n_s \epsilon^2}{2}\right). \tag{19}$$

Set $\epsilon = \sqrt{2 \log(RS/\delta)/n_s}$. The probability that any particular lower bound exceeds its true gain is at most $\delta/(RS)$. A union bound over the RS comparisons gives Equation 18 conditional on $\mathcal{H}$. Averaging over $\mathcal{H}$ gives the unconditional statement. The union bound does not require independent comparisons, so reusing evaluation cases across rounds, or patients across strata, does not invalidate it. Positivity of the relevant lower bounds proves the improvement statements.

Acceptance interpretation and scope. The result supplies a sufficient acceptance criterion for a prespecified memory transition. It does not establish that our CF admission gate enforces that criterion, nor that a failed certificate implies an update is harmful. Reusing the frozen evaluation sets is compatible with the proof only while the memory trajectory remains independent of them. If their scores trigger rollback, change retrieval, or guide later proposals, the resulting trajectory requires a different argument, for example fresh independent validation batches with round-wise error budgets summing to δ. Data used for such selection are validation data, and final testing must remain separate.

The reported RQ3 experiment (Figure 13) cumulatively enables patient-specific CF context for confidence-defined Easy, Medium, and Hard subgroups on a fixed paired cohort. Both models remain frozen, and patients not yet covered retain their zero-shot probabilities. This tests expanding CF coverage without an acceptance or rollback step. Proposition 3 applies to memory trajectories

satisfying the independence assumptions above; its accuracy bound does not certify the reported AUROC or balanced-accuracy gains.

Each bound applies to its own $\mathcal{D}_s$. A hard or rare stratum may differ from the evolution pool and still be covered when it has its own independent evaluation sample. Improvement on an easier stratum alone does not certify improvement on a harder one. The theorem concerns mean correctness under fixed stratum distributions; it does not guarantee zero harm to individual cases, valid hypothetical outcomes, or improvement in AUROC, F1, or other non-additive metrics. An uncertainty band across model seeds is also distinct from this bound on sampling error across task instances.