CLAIMSIFT: SELECTIVE ANSWER REVISION IN LARGE LANGUAGE MODELS UNDER MIXED FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) can revise answers using feedback. With mixed correct and incorrect feedback proposals, however, successful revision requires judging the proposals, updating dependent items, and preserving initially correct unrelated items. Final Accuracy measures the revised answer but obscures these decisions. Moreover, complete revisions with the same revision reward can cause different damage to initially correct items. This raises the question of how to learn selective revision while accounting for this damage. We introduce ClaimSift, a ProofWriter- and FOLIO-based benchmark for evaluating complete revisions, and propose Preservation-Aware Revision Optimization (PARO). PARO combines revision reward with explicit preservation cost, while a shared penalty weight adjusts to observed risk relative to fixed risk budgets. Experiments on ProofWriter show that PARO achieves the highest Overall Success among evaluated Qwen3-8B trained revision methods, exceeding TD-DPO and DPO by 9.75 and 9.59 percentage points, respectively, and improving Content Preservation over DPO by 3.42 points.

1 INTRODUCTION

Answer revision allows large language models (LLMs) to correct errors using feedback. For answers containing several answer items, a feedback proposal may concern one item while the proposed change affects dependent items. The model must therefore judge feedback proposals, determine which answer items require changes, and preserve initially correct unrelated items.

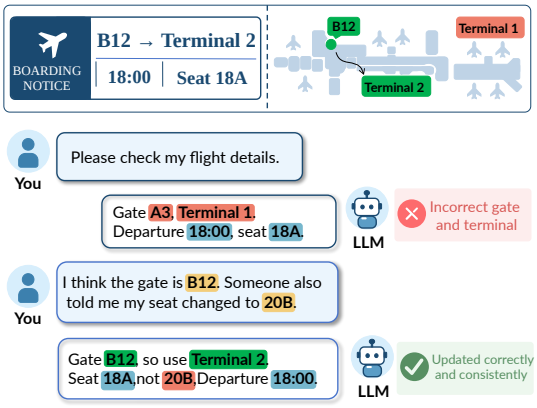

Figure 1: Selective revision: update the gate and dependent terminal, reject the incorrect seat change, and preserve the departure time.

Existing revision methods use iterative feedback, verbal reflection, and learned correction to improve answers (Kumar et al., 2025; Madaan et al., 2023; Shinn et al., 2023). However, feedback proposals can be correct or incorrect (Sharma et al., 2024; Yun et al., 2025), and correcting the target item may require changes to dependent items (Zhong et al., 2023). Initially correct unrelated items should remain unchanged (Gao et al., 2023). For example, in Figure 1, correcting the gate requires updating the terminal, while the incorrect seat proposal must be rejected and the departure time preserved. Following every feedback proposal can introduce errors, while updating only the target item can leave dependent items incorrect.

Selective revision therefore depends on proposal judgment and answer dependencies, which Final Accuracy alone does not distinguish. Our six-model diagnostic in Section 3.2 shows that high Target

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.

Correction can coexist with low Overall Success, while high Content Preservation can accompany little Target Correction. Moreover, complete revisions with the same revision reward can cause different damage to initially correct items. This raises a central question: how should a model learn the necessary changes while accounting for damage to initially correct items?

We propose Preservation-Aware Revision Optimization (PARO), a framework that learns from alternative complete revisions using revision reward and explicit preservation cost. The preservation cost counts damage to all initially correct items, including those within the necessary revision scope. PARO compares candidate returns through a Leave-one-out Baseline, while a shared penalty weight adjusts to observed risk relative to fixed risk budgets. This changes how preservation cost contributes to subsequent candidate returns.

To evaluate selective revision, we introduce ClaimSift, a benchmark based on ProofWriter and FOLIO with 3,958 candidate constructed problems (Section 4). Overall Success requires all applicable checks to pass within the same complete revision. On ProofWriter, PARO improves Overall Success over TD-DPO and DPO by 9.75 and 9.59 percentage points, respectively, leading the evaluated Qwen3-8B trained revision methods. Fixed penalty comparisons further examine correction and preservation.

Our contributions are threefold:

- We introduce ClaimSift to jointly evaluate Proposal Judgment, Target Correction, Proposal Resistance, Dependency Propagation, and Content Preservation.
- We propose PARO, combining revision reward with explicit preservation cost and an adaptive penalty weight.
- We evaluate revision methods and preservation controls, and analyze Target Correction, Content Preservation, and Overall Success across models.

2 Related Work

Answer Revision. Self-Refine and Reflexion use feedback to guide subsequent responses (Madaan et al., 2023; Shinn et al., 2023). Learned correctors and reinforcement learning train correction (Kumar et al., 2025; Welleck et al., 2023), while ReFeed considers multiple feedback dimensions (Yun et al., 2025). Sycophancy studies examine agreement at the expense of correctness (Ma et al., 2026; Sharma et al., 2024). ClaimSift evaluates proposal judgment and the resulting complete revision together. Table 1 compares evaluation coverage.

Table 1: Evaluation coverage of related settings and ClaimSift.

	Interaction			**Decision trace**		**Revision scope**	
Setting	Initial	Items	Mixed	Judge	Consistency	Propagate	Preserve
Self-Refine (Madaan et al., 2023)	✓	△	×	×	×	×	×
RefineBench (Lee et al., 2026)	✓	✓	×	×	×	×	△
ReFeed (Yun et al., 2025)	✓	✓	△	△	×	×	△
Rational updating (Ma et al., 2026)	✓	×	×	×	×	×	×
MQuAKE (Zhong et al., 2023)	×	×	×	×	×	✓	×
ClaimSift	✓	✓	✓	✓	✓	✓	✓

Coverage concerns evaluation targets, not model quality. Initial: prior answer; Items: multiple assessed units; Mixed: correct and incorrect proposals in one turn; Judge: proposal judgments; Consistency: judgment–revision consistency. Propagate/Preserve use evaluator-defined dependent/unrelated item sets. ✓: explicit; △: partial or setting-dependent (including checklist units or non-target quality dimensions); ×: not an explicit evaluation target.

Evaluation of Selective Revision. RefineBench uses checklists to evaluate refinement (Lee et al., 2026). RARR repairs unsupported statements while preserving content (Gao et al., 2023), and MQuAKE evaluates dependent consequences of knowledge edits (Zhong et al., 2023). ClaimSift

combines proposal judgment, dependency propagation, and content preservation against fixed evidence within the same revision case.

Learning from Revision Outcomes. DPO learns response preferences (Rafailov et al., 2023). IPO, TDPO, and SimPO use alternative objectives (Gheshlaghi Azar et al., 2024; Meng et al., 2024; Zeng et al., 2024), while TD-DPO emphasizes dialogue response differences (Lai et al., 2026). PARO compares complete revisions using a leave-one-out baseline (Ahmadian et al., 2024) and adapts a preservation penalty through fixed risk budgets, following constrained optimization (Achiam et al., 2017; Moskovitz et al., 2024). Appendix I provides further discussion.

3 What Makes Revision Selective?

3.1 Selective Revision

Final Accuracy summarizes the revised answer without separately evaluating proposal judgment, necessary correction, and content preservation. Even for the same feedback proposal, an initially incorrect target item may require correction, whereas an initially correct target item may require preservation. Selective revision evaluates these decisions together with changes to dependent items. A constructed problem contains context x, answer items I, and ground truth y^*. A *revision case* combines x, initial answer $y^{(0)}$, and feedback scenario f. The model returns proposal judgments d and revised answer $y^{(f)}$. Ground truth and dependencies support scoring and are not model inputs.

For target item t, the necessary revision scope S_t contains the target item and its dependent items. Initially correct unrelated items form

$$U_t^{(0)} = \{\, i \in I \setminus S_t \;\mid\; y_i^{(0)} = y_i^* \,\}, \tag{1}$$

These items must remain unchanged during revision. A feedback proposal is correct when its proposed value agrees with ground truth.

We evaluate selective revision through checks on the same complete revision. Target Correction checks initially incorrect target items under correct proposals; Proposal Resistance checks initially correct target items under incorrect proposals. Dependency Propagation checks initially incorrect dependent items when an initially incorrect target receives a correct proposal. Content Preservation checks retention of $U_t^{(0)}$. The first two metrics count eligible cases; the latter two count eligible items. For successful and eligible counts $a_{m,s}, b_{m,s}$ of metric m and source s, we pool events as

$$\mathrm{Score}_m = \frac{\sum_s a_{m,s}}{\sum_s b_{m,s}}. \tag{2}$$

Scores are percentages. Parse failures fail applicable checks; empty denominators give N/A.

Overall Success requires all applicable answer checks, retention of initially correct items within S_t, correct proposal judgments, and Judgment–Revision Consistency. ACCEPT requires the proposed value; REJECT requires a different value. Unlike Final Accuracy, Overall Success permits preexisting errors outside the applicable requirements. Dependency Propagation applies only under correct target proposals. Appendix B.5 gives the full criteria.

3.2 Observations

We examine how Target Correction, Content Preservation, and Overall Success vary across six models using saved ProofWriter responses. Five resamples of source groups allocate 394 candidate slots each, retaining paired variants, feedback scenarios, and admission outcomes without new model calls. We pool successful and eligible counts within each resample, then report means and sample standard deviations across resamples. Admission and decoding differ across models, so these profiles describe revision behavior rather than a controlled model ranking. Appendix C.1 details the settings and scores. Figure 2 shows two observations.

Observation 1: Target Correction Does Not Establish Overall Success. Figure 2 shows that Target Correction can greatly exceed Overall Success. For Llama-3.1-8B-Instruct, Target Correction reaches 88.67%, while Overall Success is 2.60% and Dependency Propagation is 34.29%. Correcting the target item therefore leaves other requirements unresolved: dependent items may remain

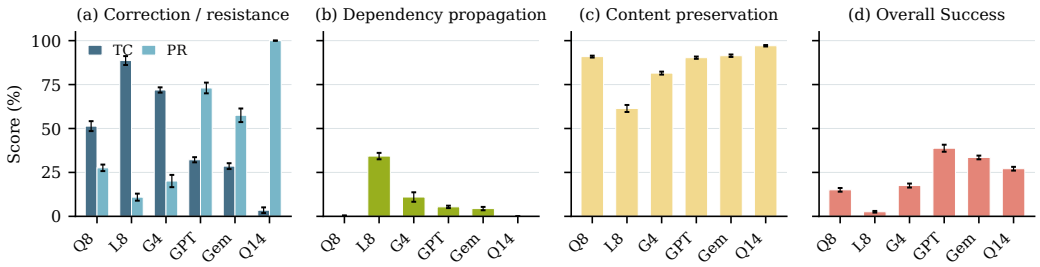

Figure 2: Six-model revision profiles: mean ± SD over five resamples of source groups at 394 candidate slots. Admission and decoding differ across models; abbreviations and scores appear in Appendix C.1.

incorrect, proposal judgments may be incorrect, and initially correct items may be damaged. Overall Success requires these checks to pass within the same revision case.

Observation 2: Content Preservation Does Not Establish Target Correction. High Content Preservation does not necessarily accompany high Target Correction. Qwen3-14B achieves 97.09% Content Preservation, but only 3.49% Target Correction and 0.00% Dependency Propagation. Its Overall Success is 27.10%. Preserving initially correct unrelated items can therefore coexist with leaving initially incorrect items unchanged. These profiles show why selective revision must evaluate correction and preservation together: high performance on either requirement alone does not establish that the complete revision succeeds.

4 CLAIMSIFT CONSTRUCTION

To examine correction and preservation within the same complete revision, we construct ClaimSift from ProofWriter and FOLIO. It contains 3,958 candidate constructed problems across both domains. We describe its source problems, feedback scenarios, and revision cases below.

Source Problem Construction. ClaimSift uses ProofWriter's facts, rules, and derivations (Tafjord et al., 2021) and FOLIO's first-order reasoning annotations (Han et al., 2024). Each constructed problem contains one target item, two dependent items, and three unrelated items. Paired variants reverse target and dependent labels while preserving queries, item positions, and unrelated labels. Both variants remain in one split before model-specific admission. Dependency validation requires blocking and restoring annotated support, checking alternative derivations, and preserving unrelated labels. Identifiers are shuffled and deletion counts matched. Appendix B.1 details both domains' semantic checks.

Feedback Scenarios. ClaimSift comprises four feedback scenarios per admitted problem: a correct target proposal, an incorrect target proposal, and a correct target proposal paired with either a correct or an incorrect unrelated proposal. Both paired scenarios address the same initially correct unrelated item, with the target proposal first. Context and initial answer remain fixed, and proposal correctness labels are hidden. These scenarios distinguish proposal judgment from item action (Table 8). A correct proposal can require KEEP, while rejecting an incorrect proposal can require UPDATE.

Revision Case Construction. The candidate inventory contains 1,964 ProofWriter and 1,994 FOLIO constructed problems. Admission requires at least two initially correct unrelated items and, when the target is initially incorrect, at least one initially incorrect dependent item. One or both paired variants may qualify, so admitted labels need not remain balanced. Each admitted problem contributes four revision cases to Overall Success; component metrics follow the eligibility criteria in Section 3.1. Appendix C details candidate, family, and admission counts in Table 14.

5 PARO: PRESERVATION-AWARE REVISION OPTIMIZATION

Motivated by the observations in Section 3.2, we propose Preservation-Aware Revision Optimization (PARO) to learn correction and preservation together. As illustrated in Figure 3, PARO comprises Plan-SFT Initialization, Risk Estimation, Reward Computation, and Constrained Optimization. Reference revisions establish fixed risk budgets. Current-policy revisions provide candidate returns for policy updates and observed risk for adjusting the next batch's penalty weight. Algorithm 1 gives the training sequence. Ground truth and dependencies support scoring; inference uses only context, initial answer, and feedback.

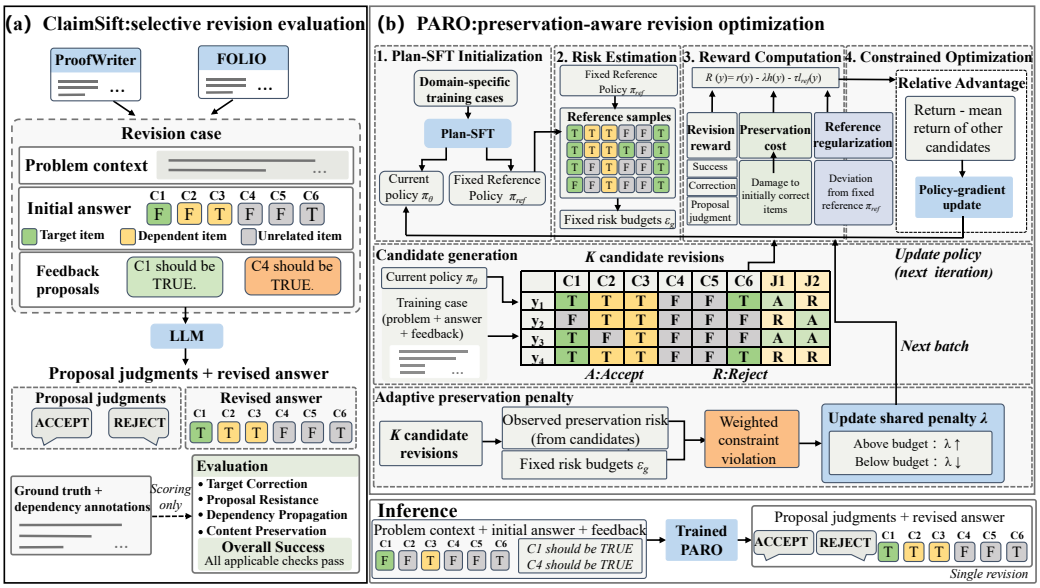

Figure 3: ClaimSift evaluation and PARO optimization. Ground truth and dependencies support scoring, not model inputs. Candidate returns update the policy; observed risk adjusts the penalty for the next batch. Inference produces one revision.

5.1 PLAN-SFT INITIALIZATION

Plan-SFT supervises proposal judgments, Keep/Update decisions, and complete revised answers. Teacher judgments are correct, and teacher answers match ground truth. Each admitted training problem supplies one teacher response per feedback scenario. The checkpoint initializes current policy π_θ and fixed reference π_{ref}; subsequent optimization updates only π_θ. Keep/Update labels are auxiliary: evaluation checks judgments and realized answer values. Appendix A details the initialization.

5.2 RISK ESTIMATION

PARO estimates preservation risk from the fixed reference policy before optimization. For value group $g \in \{\mathrm{T}, \mathrm{F}\}$, let D_g count initially correct training items. Sampling $K > 1$ reference revisions per case provides KD_g preservation opportunities. With H_g^{ref} newly damaged items, the fixed risk budget is

$$\epsilon_g = \frac{H_g^{\text{ref}}}{KD_g}. \tag{3}$$

Here, ϵ_g is the fraction of reference preservation opportunities that produce new damage. Both budgets and their denominators remain fixed as the current policy changes. The groups contribute to pooled preservation risk in proportion to their opportunity counts.

5.3 Reward Computation

For training case p_q, the current policy samples K complete responses y_{qk} containing proposal judgments, actions, and revised answers; $y^{(f)}$ denotes answer values alone. Candidates share context, initial answer, and feedback. Let s indicate Overall Success, c count corrected initially incorrect items, and j count correct proposal judgments. For N training cases and fixed opportunity counts D_s, D_c, D_j, revision reward is

$$r(y) = N\left(\alpha_s \frac{s}{D_s} + \alpha_c \frac{c}{D_c} + \alpha_j \frac{j}{D_j}\right). \tag{4}$$

Nonnegative coefficients $\alpha_s, \alpha_c, \alpha_j$ weight the normalized events. Overall Success rewards satisfying all applicable checks, while correction and judgment counts distinguish partially successful revisions. The correction count covers initially incorrect items and differs from Target Correction, whose eligibility depends on the target and feedback scenario.

Equal rewarded outcomes can still conceal different damage. Two failed revisions may correct one item, judge both proposals correctly, and miss the same dependent update, while only one damages an initially correct item. With h_g newly damaged initially correct items of value g, preservation cost is

$$h(y) = \frac{N(h_{\mathrm{T}} + h_{\mathrm{F}})}{D_{\mathrm{T}} + D_{\mathrm{F}}}. \tag{5}$$

This cost covers all initially correct items, including those within the necessary revision scope; Content Preservation evaluates the unrelated subset. Correcting an initially incorrect item creates no preservation cost. Candidate return combines revision reward, preservation cost, and reference regularization:

$$R(y) = r(y) - \lambda h(y) - \tau \ell_{\mathrm{ref}}(y). \tag{6}$$

Here, λ, τ weight cost and regularization. The sampled sequence log-ratio $\ell_{\mathrm{ref}}(y) = \log \pi_\theta(y \mid p_q) - \log \pi_{\mathrm{ref}}(y \mid p_q)$ sums token log-probabilities and can be negative. With equal rewards, equal reference log-ratios, and $\lambda > 0$, the less damaging candidate receives higher return. Increasing λ strengthens this difference, while candidates with equal costs retain their ordering when other terms are fixed. Appendix E.3 derives the general ordering threshold.

5.4 Constrained Optimization

PARO updates the policy and shared penalty weight from the same sampled revisions. Candidate returns determine which revisions to reinforce, while observed risk adjusts the contribution of preservation cost to subsequent returns.

Policy Update. The Leave-one-out Baseline (Ahmadian et al., 2024) compares each candidate with the other candidates from the same case. The Relative Advantage is

$$A_{qk} = R(y_{qk}) - \frac{1}{K-1} \sum_{k' \neq k} R(y_{qk'}). \tag{7}$$

Positive Relative Advantage identifies a return above the other candidates' mean. For the $B = |\mathcal{B}|$ cases of one training problem, the policy loss averages their BK candidates:

$$\mathcal{L}_{\mathrm{policy}} = -\frac{1}{BK} \sum_{q=1}^{B} \sum_{k=1}^{K} \mathrm{stopgrad}(A_{qk}) \log \pi_\theta(y_{qk} \mid p_q). \tag{8}$$

Advantages remain fixed during differentiation. AdamW updates the policy with gradient clipping. Zero-gradient policy steps are skipped, but sampled damage still updates the penalty. Equal candidate returns give zero Relative Advantages, so the penalty can change even when candidate comparisons supply no policy-gradient signal.

Penalty Update. For P training problems, $N = PB$. Batch damage $H_g^{\mathcal{B}}$ gives $\widehat{r}_g = PH_g^{\mathcal{B}}/(KD_g)$, scaled to the full training denominator. This scaled contribution can exceed one.

With fixed-order batches, it varies with both problem composition and policy. Starting at λ_0, a projected update (Moskovitz et al., 2024) uses step size η and cap $\lambda_{\max}$:

$$\lambda \leftarrow \mathrm{clip}_{[0,\lambda_{\max}]}\left[\lambda + \eta \frac{\sum_g D_g(\widehat{r}_g - \epsilon_g)}{D_{\mathrm{T}} + D_{\mathrm{F}}}\right]. \tag{9}$$

A positive pooled constraint violation increases λ; a negative value decreases it, subject to clipping. Weighting by D_g matches preservation cost, allowing lower risk in one group to offset excess in the other. Separate Penalties updates group weights independently with the pooled cost denominator (Appendix A.2). This update specifies the response to risk, without guaranteeing that each group satisfies its budget. Appendix E derives the normalization and constraint interpretation. At inference, the policy produces judgments and one revised answer without reference sampling, ground-truth scoring, or penalty updates.

6 Experiments and Analysis

6.1 Experimental Settings

We evaluate Qwen3-8B revision methods on ProofWriter and FOLIO. Each of five independent repetitions trains a Plan-SFT initialization, which all trained revision methods share within that repetition before separate training. Comparisons use fixed evaluation populations and shared initial answers within each dataset/backbone panel. Tables report mean ± sample SD across training runs; differences use unrounded rates. The six-model diagnostic instead resamples saved responses.

DPO, TDPO1, SimPO, IPO, and TD-DPO share teacher pairs. Preservation controls remove the cost, fix its weight at 0.5, 1.0, or 2.0, or use Separate Penalties. Tuned Fixed Penalty selects by mean Development Overall Success, then mean Content Preservation, then smaller weight, and remains fixed on Test. Appendix A details method configurations and sampling costs.

6.2 Main Results

Tables 2 and 3 compare Overall Success and its component requirements. Correction cases have initially incorrect targets and correct target proposals; Resistance cases have initially correct targets and incorrect target proposals.

Table 2: Overall Success by revision requirement.

	ProofWriter			FOLIO		
Method	Correction	Resistance	Overall	Correction	Resistance	Overall
Keep All	0.00 ±0.00	**100.00** ±0.00	**67.00** ±0.00	0.00 ±0.00	**100.00** ±0.00	**70.55** ±0.00
Plan-SFT	40.40 ±3.34	39.31 ±4.94	40.20 ±3.76	34.14 ±4.80	38.35 ±5.33	35.89 ±4.58
DPO	44.07 ±1.48	44.43 ±1.45	44.88 ±2.57	43.66 ±2.96	46.97 ±1.85	45.02 ±1.69
TD-DPO	46.07 ±3.44	43.74 ±4.17	44.72 ±3.00	41.47 ±2.13	44.13 ±2.58	42.69 ±3.95
w/o Preservation	41.73 ±2.53	34.29 ±3.70	39.54 ±3.09	38.32 ±2.53	35.96 ±2.44	38.33 ±1.29
Tuned Fixed	<u>53.33</u> ±1.94	54.29 ±4.63	53.45 ±2.58	<u>51.43</u> ±6.94	56.97 ±5.96	52.94 ±5.90
PARO	**54.40** ±2.72	<u>54.68</u> ±3.54	<u>54.47</u> ±2.98	**53.99** ±2.75	<u>60.18</u> ±3.73	<u>55.66</u> ±3.70

Percentages, mean ± SD under Section 6.1. All columns use the same Overall Success predicate on different subsets. Bold/underline mark the best/second distinct means among the listed rows.

ProofWriter: 300 Correction, 203 Resistance, 709 Other; 1212 total; FOLIO: 273 Correction, 218 Resistance, 745 Other; 1236 total. Tuned Fixed denotes Tuned Fixed Penalty with Development-selected λ (ProofWriter: 1.0; FOLIO: 1.0). Full Other counts appear in Appendix F.2.

PARO achieves 54.47% Overall Success on ProofWriter, leading evaluated Qwen3-8B trained revision methods by 9.59 points over DPO and 9.75 over TD-DPO. Target Correction remains similar to DPO, while Content Preservation improves by 3.42 points. The improvement therefore accompanies stronger preservation, resistance, and propagation. Overall Success also improves within both groups: from 44.07% to 54.40% on Correction cases and from 44.43% to 54.68% on Resistance cases.

Table 3: Qwen3-8B ProofWriter comparison: 303 admitted problems and 1,212 revision cases per seed.

Method	TC ↑	PR ↑	DP ↑	CP ↑	OS ↑
Plan-SFT	79.00 ±3.88	<u>83.05</u> ±4.88	59.61 ±2.91	90.68 ±0.42	40.20 ±3.76
DPO	84.80 ±2.22	72.32 ±2.59	66.16 ±4.82	92.77 ±1.15	44.88 ±2.57
TDPO1	**87.20** ±0.61	77.54 ±3.39	<u>67.84</u> ±3.71	<u>93.83</u> ±0.84	46.22 ±2.79
SimPO	85.67 ±0.88	77.34 ±5.01	65.66 ±3.55	93.37 ±0.63	45.35 ±2.58
IPO	<u>86.00</u> ±2.15	76.16 ±1.66	64.99 ±2.57	93.14 ±0.79	<u>46.49</u> ±2.73
TD-DPO	83.80 ±2.69	79.51 ±2.89	64.59 ±2.85	93.16 ±0.93	44.72 ±3.00
PARO	84.93 ±2.30	**92.71** ±1.76	**72.49** ±2.25	**96.19** ±0.15	**54.47** ±2.98

Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

Keep All reaches 67.00% Overall Success but succeeds on no Correction cases and all Resistance cases. Its aggregate success depends on initial target correctness. The grouped comparison separates retaining correct answers from correcting errors.

6.3 Preservation Cost

Tables 4 and 5 examine the contribution of preservation cost.

Table 4: ProofWriter preservation controls.

Method	TC ↑	CP ↑	OS ↑
w/o Pres.	**92.40** ±1.77	89.70 ±1.75	39.54 ±3.09
Fixed 0.5	79.33 ±2.13	<u>96.64</u> ±0.22	51.95 ±2.84
Fixed 1.0	77.47 ±2.59	96.17 ±0.57	<u>53.45</u> ±2.58
Fixed 2.0	75.20 ±2.82	**97.01** ±0.52	52.16 ±2.05
Sep. Pen.	81.80 ±2.30	96.31 ±0.73	53.18 ±3.03
PARO	<u>84.93</u> ±2.30	96.19 ±0.15	**54.47** ±2.98

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

Table 5: FOLIO with dependency annotations.

Method	TC ↑	CP ↑	OS ↑
Plan-SFT	74.21 ±7.45	88.06 ±1.89	35.89 ±4.58
DPO	<u>80.07</u> ±2.70	90.53 ±0.58	45.02 ±1.69
w/o Pres.	**86.15** ±2.91	87.56 ±0.81	38.33 ±1.29
Fixed 0.5	76.41 ±1.97	<u>94.55</u> ±0.70	48.77 ±5.25
Fixed 1.0	71.50 ±3.71	94.10 ±1.32	<u>52.94</u> ±5.90
Fixed 2.0	70.40 ±5.40	**94.69** ±1.30	51.00 ±3.84
PARO	79.05 ±3.10	94.50 ±0.79	**55.66** ±3.70

Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

TC: Target Correction; CP: Content Preservation; OS: Overall Success. w/o Pres.: w/o Preservation; Sep. Pen.: Separate Penalties; Each seed uses 1,212 ProofWriter and 1,236 FOLIO revision cases.

Removing the cost increases Target Correction but lowers Content Preservation and Overall Success in both domains. All three fixed penalties improve the latter two metrics over w/o Preservation. Explicit cost therefore contributes beyond the preservation checks already included in Overall Success reward, while its weight affects the balance between correction and preservation.

6.4 Adaptive Weighting

Compared with Fixed 1.0 on ProofWriter, PARO raises Target Correction from 77.47% to 84.93% at nearly unchanged Content Preservation (96.17% versus 96.19%). Overall Success improves by 1.02 points. Fixed 0.5 and Fixed 2.0 achieve higher Content Preservation (96.64% and 97.01%), but lower Target Correction (79.33% and 75.20%). PARO thus improves Overall Success without leading every component metric. This comparison shows an average advantage over the evaluated fixed settings; an independent effect of adaptation is not statistically established.

On FOLIO, PARO reaches 55.66% Overall Success, exceeding DPO by 10.65 points despite lower Target Correction. Llama results further show gains of 3.82 and 1.18 points over Tuned Fixed Penalty on ProofWriter and FOLIO, respectively, using each backbone's own evaluation population (Appendix G).

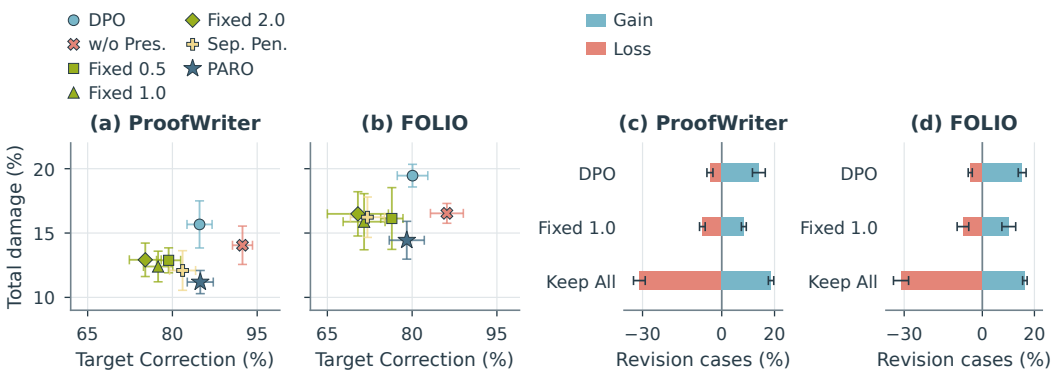

Figure 4: Correction, damage, and paired Overall Success for Qwen3-8B. (a,b) Target Correction versus damage to all initially correct items; lower damage is better. Both axes show one sample SD. (c,d) Gain: only PARO succeeds; Loss: only the comparator succeeds, plotted to the left of zero. Both use shared revision cases. Error bars show one sample SD across five independent training runs. Fixed 1.0 is selected on Development for the paired comparisons. w/o Pres.: w/o Preservation; Sep. Pen.: Separate Penalties.

6.5 Behavioral Analysis

Figure 4 relates Target Correction to damage across all initially correct items and compares paired Overall Success. PARO reduces total damage from 15.67% to 11.19% on ProofWriter and from 19.46% to 14.44% on FOLIO relative to DPO, while retaining similar Target Correction. Unlike Content Preservation, this measure also covers initially correct items within the necessary revision scope. Fixed 2.0 has less unrelated damage on ProofWriter, but more total damage than PARO. The two measures therefore describe different aspects of preservation.

Net gains are positive against DPO and Tuned Fixed Penalty in both domains, but negative against Keep All. Lower damage therefore does not alone determine Overall Success. Proposal judgments add another requirement: PARO's 766.6 average Answer-scope Success cases yield 660.2 Overall Success cases on ProofWriter. Reassigning proposal judgments to agree with revised answers retains PARO's advantage over Tuned Fixed Penalty in both domains (Table 38). This postprocessing analysis holds answers fixed, separating the comparison from the models' original proposal judgments.

Replacing a correct unrelated proposal with an incorrect one reduces successful cases by an average of 15.2 for PARO, 18.4 for DPO, and 33.2 for w/o Preservation per run on ProofWriter. This paired comparison isolates proposal correctness on shared cases (Figure 8).

Appendix analyses distinguish judgment postprocessing from feedback-order and balanced-feedback rules. The rules expose feedback cues but do not establish trained-model robustness. Risk-control simulations and dense-reward analysis likewise remain separate from measured training trajectories and matched Dense-reward RL results, which are not reported (Appendices B.4 and H).

7 Conclusion

We study selective revision under mixed correct and incorrect feedback. ClaimSift jointly evaluates Proposal Judgment, Target Correction, Proposal Resistance, Dependency Propagation, and Content Preservation within complete revisions. PARO combines revision reward with explicit preservation cost and adapts a shared penalty weight to observed risk. Experiments on ProofWriter and FOLIO show improved Overall Success among the evaluated trained revision methods, while component and paired analyses characterize the remaining trade-off between correction and preservation.

AI USE STATEMENT

Generative AI tools assisted with manuscript writing, polishing, and translation; literature retrieval, discovery, and organization; and synthetic dataset generation. They also supported methodological discussion, code development, and figure preparation. The construction and validation criteria for ClaimSift revision cases are described in Section 4 and Appendix B.1. The authors are responsible for the experiments, reported results, citations, code, and scientific claims.

ETHICS STATEMENT

We confirm that this research complies with all applicable ethical guidelines. No specific ethical concerns were identified in this study. Use and redistribution of ProofWriter and FOLIO remain subject to the original licenses and terms of use.

REPRODUCIBILITY STATEMENT

The anonymous code repository is at `https://anonymous.4open.science/r/ClaimSift-7C16`. Section 6.1 and Appendix D describe the training repetitions, shared evaluation populations, and statistical summaries. Algorithms 2 and 1 specify evaluation and optimization. Method comparisons are computed from model responses; no success quota is imposed and no proposal judgment is flipped to force an aggregate score. The six-model motivation diagnostic uses the separate source-group resampling procedure in Appendix C.1. Integer event counts and paired outcomes support verification of the reported statistics.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *ICML*, volume 70, pp. 22–31. PMLR, 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *ACL*, pp. 12248–12267, 2024. doi: 10.18653/v1/2024.acl-long.662. URL https://doi.org/10.18653/v1/2024.acl-long.662.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *ICLR*, 2024. URL https://arxiv.org/abs/2304.05128.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *NIPS*, volume 30, pp. 4299–4307, 2017. URL https://papers.nips.cc/paper/7017-deep-reinforcement-learning.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. Explaining answers with entailment trees. In *EMNLP*, pp. 7358–7370. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.585. URL https://aclanthology.org/2021.emnlp-main.585/.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y. Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. RARR: Researching and revising what language models say, using language models. In *ACL*, pp. 16477–16508, 2023. doi: 10.18653/v1/2023.acl-long.910. URL https://doi.org/10.18653/v1/2023.acl-long.910.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Rémi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *AISTATS*, volume 238 of *Proceedings of Machine Learning Research*, pp. 4447–4455. PMLR, 2024. URL https://proceedings.mlr.press/v238/gheshlaghi-azar24a.html.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *ICLR*, 2024. URL https://arxiv.org/abs/2305.11738.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander Fabbri, Wojciech Maciej Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. FOLIO: Natural language reasoning with first-order logic. In *EMNLP*, pp. 22017–22031, 2024. doi: 10.18653/v1/2024.emnlp-main.1229. URL https://aclanthology.org/2024.emnlp-main.1229/.

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

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *ICLR*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can LLMs actually correct their own mistakes? A critical survey of self-correction of LLMs. *TACL*, 12:1417–1440, 2024. doi: 10.1162/tacl_a_00713. URL https://doi.org/10.1162/tacl_a_00713.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *ICLR*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/871ac99fdc5282d0301934d23945ebaa-Abstract-Conference.html.

Shuzhong Lai, Junhong Lai, Chenxi Li, Qing Zhou, Haifeng Li, Gang Pan, Lin Yao, and Yueming Wang. TD-DPO: Difference-aware preference optimization for mitigating sycophancy in clinical autism intervention dialogue. *arXiv preprint arXiv:2607.18304*, 2026. doi: 10.48550/arXiv.2607.18304. URL https://arxiv.org/abs/2607.18304v1. Version 1.

Young-Jun Lee, Seungone Kim, Byung-Kwan Lee, Minkyeong Moon, Yechan Hwang, Jong Myoung Kim, Graham Neubig, Sean Welleck, and Ho-Jin Choi. RefineBench: Evaluating refinement capability of language models via checklists. In *ICLR*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/835b6c04ca2daaa1a682c2eaad4a70ea-Abstract-Conference.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *ICLR*, 2024. URL https://arxiv.org/abs/2305.20050.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *ACL*, pp. 3214–3252. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Huanhuan Ma, Weizhi Xu, Yifan Wei, Liuji Chen, Liang Wang, Qiang Liu, Shu Wu, and Liang Wang. EX-FEVER: A dataset for multi-hop explainable fact verification. In *Findings of ACL*, pp. 9340–9353. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.556. URL https://aclanthology.org/2024.findings-acl.556/.

Huanhuan Ma, Henry Peng Zou, Chengze Li, Enze Ma, Yunyue Su, and Philip S. Yu. Sycophancy suppression can impair rational updating: Anti-sycophancy should preserve the ability to update. *arXiv preprint arXiv:2608.26511*, 2026. URL https://arxiv.org/abs/2608.26511v1. Version 1.

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 *NeurIPS*, volume 36, pp. 46534–46594, 2023. doi: 10.52202/075280-2019. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *NeurIPS*, volume 37, pp. 124198–124235, 2024. doi: 10.52202/079017-3946. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e099c1c9699814af0be873a175361713-Abstract-Conference.html.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *EMNLP*, pp. 12076–12100, 2023. doi: 10.18653/v1/2023.emnlp-main.741. URL https://doi.org/10.18653/v1/2023.emnlp-main.741.

Ted Moskovitz, Aaditya K. Singh, DJ Strouse, Tuomas Sandholm, Ruslan Salakhutdinov, Anca D. Dragan, and Stephen Marcus McAleer. Confronting reward model overoptimization with constrained RLHF. In *ICLR*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/5eee634cb9729b8bcc2ec9f2a46a74ae-Abstract-Conference.html.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, volume 35, pp. 27730–27744, 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, volume 36, pp. 53728–53741, 2023. doi: 10.52202/075280-2338. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *ICLR*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/0105f7972202c1d4fb817da9f21a9663-Paper-Conference.pdf.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*, volume 36, pp. 8634–8652, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of ACL-IJCNLP*, pp. 3621–3634. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-acl.317. URL https://aclanthology.org/2021.findings-acl.317/.

Gladys Tyen, Hassan Mansoor, Victor Carbune, Peter Chen, and Tony Mak. LLMs cannot find reasoning errors, but can correct them given the error location. In *Findings of ACL*, pp. 13894–13908, 2024. doi: 10.18653/v1/2024.findings-acl.826. URL https://doi.org/10.18653/v1/2024.findings-acl.826.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. Generating sequences by learning to self-correct. In *ICLR*, 2023. URL https://openreview.net/forum?id=hH36JeQZDaO.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *ICLR*, 2024. URL https://arxiv.org/abs/2305.13300.

Kailai Yang, Zhiwei Liu, Qianqian Xie, Jimin Huang, Erxue Min, and Sophia Ananiadou. Selective preference optimization via token-level reward function estimation. In *EMNLP*, pp. 7032–7056. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.359. URL https://aclanthology.org/2025.emnlp-main.359/.

Taewon Yun, Jihwan Oh, Hyangsuk Min, Yuho Lee, Jihwan Bang, Jason Cai, and Hwanjun Song. ReFeed: Multi-dimensional summarization refinement with reflective reasoning on feedback. In *COLM*, 2025. URL https://openreview.net/forum?id=6BGDGKZN7q.

Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. In *ICML*, pp. 58348–58365. PMLR, 2024. URL https://proceedings.mlr.press/v235/zeng24c.html.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In *EMNLP*, pp. 15686–15702. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.971. URL https://aclanthology.org/2023.emnlp-main.971/.

TABLE OF CONTENTS FOR THE APPENDIX

The appendix provides experimental details, task construction and evaluation criteria, and benchmark statistics. It then describes the training runs and statistical summaries and analytical properties underlying the comparisons. Complete results include grouped outcomes, damage and error analysis, and backbone comparisons. Additional mechanism analyses and detailed related work follow.

			Page
A	**Experimental Details**		**15**
	A.1	Initialization and Preference Objectives	15
	A.2	Preservation Controls and Optimization Settings	15
	A.3	Tuned Fixed Penalty and Inference	17
B	**Task Construction and Evaluation Criteria**		**17**
	B.1	Paired Evidence and Dependency Requirements	18
	B.2	Admission and Available Information	19
	B.3	Prompts and Output Format	19
	B.4	Input Cues and Proposal Order	20
	B.5	Eligible Units and Overall Success	21
	B.6	Input and Output Examples	22
C	**Benchmark Statistics**		**23**
	C.1	Model Coverage and Motivation Diagnostic	24
D	**Training Runs and Evaluation Records**		**25**
	D.1	Shared Inputs and Unmodified Model Responses	25
	D.2	Recomputation and Statistical Summaries	25
E	**Analytical Properties**		**26**
	E.1	Component Measures and Overall Success	26
	E.2	Bounds for Unconditional Strategies	26
	E.3	Preservation Cost and Candidate Ordering	27
	E.4	Risk Normalization and Penalty Updates	27
F	**Complete Results and Error Analysis**		**28**
	F.1	Method Comparisons and Preservation Controls	28
	F.2	Overall Success by Revision Requirement	28
	F.3	Initial Correctness and Preservation Trade-offs	29
	F.4	Answer Quality and Proposal Judgments	30
	F.5	New Damage and Error Categories	32
	F.6	Tuned Fixed Penalty	34
	F.7	Feedback Scenarios and Overall Success	34
	F.8	Paired Outcomes and Correction Need	34
	F.9	Integer Counts and Dependency Coverage	36
	F.10	Overall Success Across Training Runs	37
G	**Additional Backbone Comparisons**		**37**
H	**Additional Mechanism Analyses**		**38**
	H.1	Balanced Feedback Design	39
	H.2	Judgment Postprocessing	40
	H.3	Dense Correctness and Damage	41
	H.4	Budget and Realized Risk	41
	H.5	Fixed Penalties at Matched Risk	41
I	**Detailed Related Work**		**42**
	I.1	Feedback and Self-correction	42
	I.2	Unreliable Feedback and Evidence Conflicts	42
	I.3	Fine-grained Evaluation and Revision	42
	I.4	Repair Without Regressions	42
	I.5	Reasoning Sources and Preference Optimization	42

A Experimental Details

This section details the implementation, initialization, optimization objectives, preservation controls, Development selection, and inference used in the method comparisons.

Reference implementation. The Plan-SFT reference implementation for Qwen3-8B and Llama-3.1-8B-Instruct uses an NVIDIA GeForce RTX 5090. Training uses PyTorch 2.8.0, Transformers 4.57.1, PEFT 0.17.1, TRL 0.24.0, and Accelerate 1.15.0; revision evaluation uses vLLM 0.11.0. Both use Tokenizers 0.22.2 and Safetensors 0.8.0. Base weights use unquantized bfloat16, and comparison adapters use float32. Plan-SFT uses gradient checkpointing and supervises response tokens, including EOS, while masking prompt and padding tokens. Training sequence length is capped at 1,024 tokens. Table 6 gives the optimization settings.

A.1 Initialization and Preference Objectives

Within each dataset/backbone panel, each of the five repetitions trains Plan-SFT once. All trained revision methods in that repetition start from its checkpoint. Plan-SFT uses teacher responses with correct proposal judgments and complete answers matching ground truth. Item actions follow the difference between initial and revised values. Each admitted training problem supplies four teacher revisions, one per feedback scenario. Five controlled error types cover proposal judgment, action consistency, target correctness, dependency correctness, and unrelated preservation. Applying them to all four cases yields twenty preference pairs per problem, shared by all offline objectives.

Minimal preference pair on a shared input

Use the evidence, initial answer, and feedback of Appendix B.6. The chosen teacher returns the correct answer TTTFFF with judgments (ACCEPT, REJECT). Its actions are (UPDATE, UPDATE, KEEP, KEEP, KEEP, UPDATE).

Field	Chosen	Rejected
F1 judgment	ACCEPT	REJECT
F2 judgment	REJECT	REJECT
Revised C1–C6	TTTFFF	TTTFFF
Actions	Identical	Identical

Only F1's judgment changes. Both answers are completely correct, but the rejected response misjudges the true proposal and contradicts its own final C1 value. The pair isolates a judgment error; it does not isolate judgment correctness from consistency.

The five error types operate on the same response structure: (i) change a proposal judgment, (ii) make a Keep/Update decision inconsistent with the revised answer, (iii) corrupt the target value, (iv) corrupt a required dependent value, and (v) corrupt an initially correct unrelated value. Value corruptions update the corresponding action when needed; an action error deliberately does not. These are controlled preference contrasts, not disjoint empirical failure classes. The example illustrates pair construction, not an additional measured training record. Prompt tokens remain masked in the response objective; the supervised suffix and end-of-sequence convention follow Appendix A.

DPO and IPO use coefficient 0.1; TDPO1 uses 0.1. SimPO uses length-normalized response log-probabilities with coefficient 2.5 and margin 0.55, without a reference term. TD-DPO uses coefficient 0.08 and the same predefined differing-token mask rather than a method-specific pair selection. The objective family changes while initialization, pair membership, response masking, and use of the final checkpoint stay fixed. These are configuration choices, not guarantees that all methods have equal compute or optimal hyperparameters.

A.2 Preservation Controls and Optimization Settings

Every online problem batch contains the four revision cases and $K = 4$ candidates per case. w/o Preservation sets the preservation penalty to zero but retains the Overall Success reward, including its preservation checks. w/o Adaptation holds the penalty weight at 0.5, 1.0, or 2.0. PARO updates the shared penalty weight. Separate Penalties assigns a weight to each label while retaining the

pooled opportunity denominator:

$$R_{\mathrm{sep}}(y) = r(y) - \frac{N\big(\lambda_{\mathrm{T}} h_{\mathrm{T}}(y) + \lambda_{\mathrm{F}} h_{\mathrm{F}}(y)\big)}{D_{\mathrm{T}} + D_{\mathrm{F}}} - \tau \ell_{\mathrm{ref}}(y). \tag{10}$$

The two weights start at the same initial value as PARO and use the same step size and cap, but respond independently to label-specific constraint violations:

$$\lambda_g^{+} = \mathrm{clip}_{[0,\lambda_{\max}]}\big[\lambda_g + \eta(\widehat{r}_g - \epsilon_g)\big], \qquad g \in \{\mathrm{T}, \mathrm{F}\}. \tag{11}$$

Here $\widehat{r}_g = PH_g^{\mathcal{B}}/(KD_g)$ is the same batch statistic used by PARO. The return divides both damage counts by the total opportunities, whereas the update uses each label's own normalized risk without pooling. When $\lambda_{\mathrm{T}} = \lambda_{\mathrm{F}} = \lambda$, Equation 10 reduces to Equation 6; subsequent label-specific updates can separate the weights. This control changes risk coupling without changing the scale assigned to a damage event at equal weights.

Table 6: Specified configuration. Data-dependent update counts, risk budgets, and runtimes are not inferred from dataset size.

Setting	Value
Plan-SFT epochs / learning rate	3 / 10^{-4}
Plan-SFT microbatch / gradient accumulation	1 / 8
Plan-SFT warm-up fraction / schedule	0.05 / linear
Optimizer / weight decay	AdamW / 0
LoRA rank / scaling / initialization dropout	16 / 32 / 0.05
Comparison learning rate / gradient norm cap	5×10^{-6} / 1.0
Comparison dropout	0
Offline warm-up fraction / schedule	0.05 / linear
Online learning-rate schedule	constant
Feedback cases / candidates per case	4 / 4
Reward weights $(\alpha_s, \alpha_c, \alpha_j)$	$(0.5, 0.25, 0.25)$
Reference coefficient τ	0.01
Controller $(\lambda_0, \eta, \lambda_{\max})$	$(1.0, 0.1, 10)$
Fixed weights	$\{0.5, 1.0, 2.0\}$

LoRA is applied to the query, key, value, and output projections in attention, and to the gate, up, and down projections in the feed-forward layers. Offline comparison uses all twenty pairs for one problem per batch and one pass over the admitted training problems. Its warm-up length is 5% of the planned problem batches, rounded to the nearest integer with a minimum of one, followed by linear decay. Online comparison uses a constant learning rate and one problem visit per pass. A zero policy gradient skips the optimizer step but not the penalty update in response to risk. Consequently, the number of visited problems is an upper bound on nonzero online updates, not an observed update count.

Reference calibration draws K candidates for each training case before policy optimization and fixes the resulting risk budgets. They are empirical quantities of that reference policy, not percentages that can be scaled from the number of tasks. The batch statistic $PH_g^{\mathcal{B}}/(KD_g)$ is a scaled risk contribution and can exceed one; it is not a probability within that batch. Training preservation cost covers all initially correct items, whereas evaluation CP covers only the unrelated subset.

Table 7: Logical one-pass workloads implied by admitted Training slots. These are per-pass candidate counts, not observed optimizer-step counts.

Panel	P	Teachers	Pairs	Reference	Online
ProofWriter / Qwen	888	3,552	17,760	14,208	14,208
ProofWriter / Llama	897	3,588	17,940	14,352	14,352
FOLIO / Qwen	971	3,884	19,420	15,536	15,536
FOLIO / Llama	956	3,824	19,120	15,296	15,296

$B = 4$, $K = 4$. Reference and Online each contain KBP candidate slots; they refer to distinct stages. Rows use backbone-specific admission.

Each row's teacher count is BP, pair count is $5BP$, and reference or one-pass online candidate count is KBP. The two candidate columns describe different stages and cannot be merged into one completed generation count. Optimizer steps also depend on scheduling and nonzero gradients. No throughput, wall time, memory usage, or final risk is extrapolated from these logical counts.

Algorithm 1 Preservation-Aware Revision Optimization

Input: P training problems with B revision cases each; Plan-SFT data; ground truth and event denominators; $K > 1$, reward weights $\alpha_s, \alpha_c, \alpha_j$, τ, λ_0, η, and $\lambda_{\max}$.
Output: Revision policy π_θ.

Plan-SFT Initialization
Train Plan-SFT; initialize π_θ and the fixed reference policy π_{ref}; set $\lambda = \lambda_0$.
Risk Estimation
Sample K reference revisions per training case; fix $\epsilon_g = H_g^{\mathrm{ref}}/(KD_g)$.
for each single-problem training batch $\mathcal{B}$ **do**
  Reward Computation
  for each revision case $p_q \in \mathcal{B}$ **do**
    Sample K revisions $y_{qk} \sim \pi_\theta(\cdot \mid p_q)$.
    Compute candidate returns R_{qk} by Equation 6.
  end for
  Constrained Optimization
  Compute Relative Advantages A_{qk} by Equation 7.
  Compute Equation 8; clip and apply a nonzero policy gradient.
  Estimate $\widehat{r}_g = PH_g^{\mathcal{B}}/(KD_g)$ from the same BK revisions.
  Update the shared penalty weight λ by Equation 9, even if the policy gradient is zero.
end for
return π_θ; inference uses a single revision and no hidden ground truth.

A.3 Tuned Fixed Penalty and Inference

The Tuned Fixed Penalty is selected using Development Overall Success, then Content Preservation, then the smaller weight to break a remaining tie. The chosen weight is recorded before Test; all three fixed settings remain in the complete comparisons. The selected row is not an additional trained method. PARO retains its specified controller settings rather than receiving a weight chosen on Test.

Within each panel, inference methods receive identical contexts, initial answers, and feedback. Each returns proposal judgments and one complete revised answer. Reference sampling and candidate comparison belong to training; inference uses a single revision. The context limit includes the prompt and reserved response space, with oversized prompts rejected rather than truncated.

Decoding configuration. Online sampling uses temperature 1.0, top-$p = 1.0$, disabled top-k filtering, and at most 192 new tokens per candidate, without a grammar constraint. ProofWriter revision evaluation instead uses greedy decoding, a 192-token output cap, and constraints on response syntax alone. Qwen thinking is disabled. Revision inference uses one GPU, bfloat16 weights, eager execution, no prefix caching, and a 1,024-token total context. Prompts exceeding this limit cause an error rather than truncation. These limits describe the existing ProofWriter implementation; FOLIO inputs with dependencies require separate validation of prompt and response lengths.

Random Seeds and Checkpoint Selection. Training and candidate-sampling seeds are recorded in the run configurations. Each comparison evaluates the final checkpoint without Test-based selection. The tables summarize five independent repetitions. Each repetition has its own Plan-SFT checkpoint, shared across the trained revision methods within that panel. The motivation diagnostic instead uses source-group resampling. These two sources of variability are reported separately.

B Task Construction and Evaluation Criteria

This section describes the construction and evaluation of revision cases in ProofWriter and FOLIO. We first define paired evidence and dependency requirements, then specify admission, prompts, and scoring. The worked cases connect these definitions to complete inputs and outputs. Table 14 reports

Table 8: Required behavior under the four target situations.

Initial target	Target proposal	Judgment	Target action	Dependent repair	Keep All
Incorrect	Correct	ACCEPT	UPDATE	Repair	Fail
Incorrect	Incorrect	REJECT	UPDATE	Not required	Fail
Correct	Correct	ACCEPT	KEEP	Not required	Pass
Correct	Incorrect	REJECT	KEEP	Not required	Pass

Dependent repair concerns initially incorrect dependent items. Initially correct items covered by the scoring criteria remain protected in every row. Keep All uses judgments based on agreement with the initial answer; Pass/Fail refers to Overall Success for admitted cases, with any unrelated proposal judged consistently. The second row requires changing the target to reject the incorrect proposal consistently.

the allocation and admission counts, and Appendix D describes the training runs and statistical summaries separately.

B.1 Paired Evidence and Dependency Requirements

Each constructed problem contains one target item, two dependent items, and three unrelated items. Its paired variants share queries and roles: target and dependent labels reverse, while unrelated labels remain unchanged. All variants from one family belong to the same split. We shuffle the visible item identifiers and reserve role labels and dependency annotations for evaluation.

Dependency construction uses the reasoning semantics of the source. Blocking the annotated support must prevent the dependent consequence; restoring that support must recover it. Unrelated conclusions must remain unchanged under both interventions. Checking alternative derivations distinguishes necessary support from a redundant proof. Matching the size and type of context edits across variants prevents the edit pattern from revealing the answer label.

ProofWriter's selected binary tasks use closed-world reasoning. A proposition is TRUE when derivable and FALSE otherwise; a negated query receives the opposite label. Negation in rule premises is interpreted under the selected rule semantics. Ground-truth computation and interventions must use the same semantics as the prompt, not a mixture of open- and closed-world rules.

FOLIO requirements. The FOLIO extension requires the same structure of target, dependent, and unrelated items, realized in first-order reasoning contexts. Neither failure to prove a statement nor an UNKNOWN source label is automatically FALSE. Only determinate binary tasks are included, with contradictions and unresolved queries excluded before admission. Proofs supporting the target, blocking and restoration checks, and unchanged unrelated conclusions are construction requirements. These checks are applied to the source-language constructions before initial-answer admission.

First-order construction: support, blocking, and restoration

Shared premises. Every red entity is kind, and every kind entity is red. Every kind entity is quiet, and every quiet entity is kind. Mira is not red.

Paired evidence. Variant A: Robin is red. Variant B: Robin is not red.

Queries. C1–C3 ask whether Robin is red, kind, and quiet; C4–C6 ask the same of Mira.

Evidence	C1–C3	C4–C6
Variant A	T T T	F F F
Remove Robin's evidence	U U U	F F F
Restore Robin is red	T T T	F F F
Variant B	F F F	F F F

T means entailed; F means its negation is entailed; U means neither is entailed. In Variant B, contraposition of kind $\Rightarrow$ red and quiet $\Rightarrow$ kind entails Robin's negative dependent labels. Without either Robin fact, both all-true and all-false Robin valuations satisfy the premises: the labels are UNKNOWN, not FALSE. Adding both Robin facts makes the premises inconsistent and excludes the context.

The example illustrates paired determinate labels and dependency blocking under first-order semantics. Enumerating its six ground atoms gives one satisfying valuation per paired variant, two after blocking, and none under contradictory evidence. Only the two binary variants proceed to initial-answer admission; the blocked context tests the dependency. Explicit equivalences make this illustrative proof inspectable. The example is not a source-dataset item or an inventory validation, and broader contexts require their own satisfiability, entailment, and alternative-support checks.

Validation sequence. A source-level construction follows six checks in order. First, the context must be satisfiable. Second, every evaluated query must be entailed or have its negation entailed; an unresolved query is UNKNOWN and cannot enter the binary task. Third, identify the support for the target and dependent conclusions. Fourth, block that support and check the dependent consequences against all remaining premises, including alternative derivations. Fifth, restore the original support and recover the original labels. Sixth, verify that all unrelated labels remain unchanged under both interventions. Only then apply the shared initial-answer admission rule. A solver timeout is unresolved rather than evidence of either entailment or its absence.

Table 9 extends the preceding example with two failure cases. Adding Robin is kind after removing Robin is red preserves every Robin label through the remaining implications, exposing redundant support. Removing Mira's negative premise leaves Robin's labels intact but makes all three unrelated queries UNKNOWN. The first intervention fails the necessity check; the second fails unrelated invariance. These finite examples verify the checks and their counterexamples, not the validity rate of FOLIO source constructions.

Table 9: Dependency checks on the first-order example.

Evidence intervention	Valuations	C1–C3	C4–C6
Supported	1	TTT	FFF
Blocked	2	UUU	FFF
Restored	1	TTT	FFF
Alternative support	1	TTT	FFF
Unrelated premise removed	2	TTT	UUU
Contradictory	0	–	–

Valuations counts satisfying assignments to the six named ground atoms. T/F/U denote entailed, negation entailed, and UNKNOWN. A contradictory context has no satisfying valuation. Alternative support adds Robin is kind after blocking Robin is red; the unrelated intervention removes Mira is not red.

B.2 Admission and Available Information

Admission uses only the shared initial answer and ground truth, before any method-specific revision. A problem requires at least two initially correct unrelated items. If its target is initially incorrect, at least one dependent must also be initially incorrect; an initially correct target does not need this additional error. Every admitted problem contributes all four feedback scenarios. A paired family can contribute one or two admitted variants, so the admitted target labels need not remain balanced.

The initial prompt supplies evidence, reasoning semantics, and six queries. The revision prompt adds the initial answer and feedback proposals. Ground truth, role assignments, dependency annotations, and proposal-correctness labels remain outside the inference input. All methods in a dataset/backbone panel share admission, initial states, feedback, and eligibility criteria.

B.3 Prompts and Output Format

The revision instruction requests an independent judgment for each proposal, a Keep/Update decision for each answer item, and a complete revised answer. The evidence determines correctness. A judgment about one proposal does not determine the judgment or action for another item.

The output format has three ordered fields: `feedback_verdicts`, `claim_actions`, and `revised_answer`. Proposal judgments are ACCEPT or REJECT; item actions are KEEP or UPDATE; answer values are TRUE or FALSE. The displayed proposal and item identifiers determine

field order. ACCEPT denotes agreement with a proposed value, not necessarily an UPDATE: a correct proposal can reaffirm an already correct answer. Keep/Update decisions are auxiliary outputs checked for format; metrics use judgments and realized answer values.

Missing fields, duplicate keys, invalid labels, and missing or reordered identifiers are parse failures. They stay in every applicable denominator and fail the corresponding checks. A metric with no eligible cases is N/A, not zero. Model responses are decoded and parsed into binary answer values and proposal judgments for scoring. The following rule analysis separates input cues from proposal judgment based on evidence.

Complete revision instruction template

Read the evidence using the stated reasoning semantics. Evaluate every listed feedback proposal against that evidence. Return a judgment for each proposal and one complete revised answer.

Evidence and semantics: *[context; source reasoning convention]*
Queries: *[C1 through C6, in displayed order]*
Initial answer: *[one TRUE/FALSE value per query]*
Feedback: *[F1 and, where present, F2: item and proposed value]*

Return one JSON object with exactly these ordered fields:

1. `feedback_verdicts`: ACCEPT or REJECT for every displayed proposal, in order.
2. `claim_actions`: KEEP if the final value equals its initial value; otherwise UPDATE, for C1–C6 in order.
3. `revised_answer`: TRUE or FALSE for C1–C6 in order.

Use no additional fields or explanatory text. ACCEPT means that the final value agrees with the proposal; REJECT means that it disagrees. A correct proposal may reaffirm an initial value, so ACCEPT does not necessarily mean UPDATE.

The template makes the revision task explicit; it is not a verbatim historical prompt. Its slots contain no item roles, ground truth, dependency annotations, or proposal-correctness flags. The initial-answer template omits the initial answer and feedback and requests only the ordered answer values. Source semantics are explicit: FALSE is not substituted for UNKNOWN in first-order inputs. In the construction protocol, unresolved reference queries are excluded before binary prompting.

B.4 Input Cues and Proposal Order

We compare two rules that judge proposals without consulting evidence. Position-only uses proposal position and proposal count. Initial-agreement accepts a proposal exactly when its value matches the initial answer at the addressed item. Keep All already uses this judgment rule, so Initial-agreement describes that strategy rather than an additional baseline.

Development selection. There are three position categories: a sole proposal, the first of two proposals, and the second of two proposals. Each maps to ACCEPT or REJECT, giving eight rules. We evaluate all eight on the shared Development cases and select the largest pooled Proposal Judgment rate. Ties use lexicographic order with REJECT before ACCEPT. Both domains select (REJECT, ACCEPT, REJECT). The chosen rule is held fixed for Test. Prediction uses neither item truth nor correctness labels; Development truth is used only to select the rule, and Test truth only to score it.

Order comparison. For every two-proposal Test case, we reverse the ordered list of (item, proposed value) pairs and assign the displayed proposal identifiers to their new positions. Initial answers, ground truth, and proposal contents stay fixed. Single-proposal cases are unchanged. The selected position rule is applied again without reselection. Initial-agreement follows the item identified by each proposal, so reversing the list reverses its judgments and leaves their correctness unchanged. Table 10 reports pooled and two-proposal rates; Table 11 supplies integer counts. Position-only predicts judgments alone, so no answer-level metric is assigned to it.

The position scores coincide across domains because each admitted problem contributes the same four proposal-correctness patterns. In the original order, the selected rule judges four of six proposals correctly; after reversal, it judges two. Initial-agreement instead depends on initial-target

Table 10: Proposal Judgment of rules based only on the input.

Domain / rule	All: orig.	All: rev.	Dual: orig.	Dual: rev.
ProofWriter / Position-only	66.67	33.33	75.00	25.00
ProofWriter / Initial-agreement	78.00	78.00	83.50	83.50
FOLIO / Position-only	66.67	33.33	75.00	25.00
FOLIO / Initial-agreement	80.37	80.37	85.28	85.28

Percentages. Original and reversed order share the same proposals; only two-proposal cases are reversed. The position rule is selected once on Development and held fixed. Initial-agreement is the judgment rule of Keep All. These are deterministic rule calculations, not model perturbation tests.

Table 11: Exact Proposal Judgment counts for the order comparison.

Domain / rule	All: orig.	All: rev.	Dual: orig.	Dual: rev.
ProofWriter / Position-only	1212/1818	606/1818	909/1212	303/1212
ProofWriter / Initial-agreement	1418/1818	1418/1818	1012/1212	1012/1212
FOLIO / Position-only	1236/1854	618/1854	927/1236	309/1236
FOLIO / Initial-agreement	1490/1854	1490/1854	1054/1236	1054/1236

Successful / eligible proposals, counted once per retained case rather than repeated across five seeds. No revised answer is assigned to Position-only.

correctness: all unrelated proposals address an initially correct item. If the fraction of correct initial targets is c, its pooled Proposal Judgment is $(4c+2)/6$, and its two-proposal rate is $(c+1)/2$. These identities explain both the repeated position scores and the domain-dependent agreement scores.

The rule comparison quantifies available input cues. Evaluating a trained policy under reordered proposals instead requires paired model responses with the same checkpoint and decoding settings. Binary labels also link agreement to correctness: a proposal matches the initial answer exactly when both have the same correctness status. Agreement and initial correctness are therefore not independent experimental factors.

B.5 Eligible Units and Overall Success

Let $\mathcal{Q}$ contain admitted revision cases. Define $\mathcal{C}$ as cases with an initially incorrect target and a correct target proposal; $\mathcal{R}$ contains initially correct targets under incorrect target proposals. For $q \in \mathcal{C}$, $E_D(q)$ contains initially incorrect dependent items. Let $U_q^{(0)}$ contain initially correct unrelated items, and $G_q^{(0)}$ initially correct in-scope items. With v_{qi} indicating final item correctness, the component rates are

$$\mathrm{TC} = \frac{\sum_{q\in\mathcal{C}} v_{qt}}{|\mathcal{C}|}, \qquad \mathrm{PR} = \frac{\sum_{q\in\mathcal{R}} v_{qt}}{|\mathcal{R}|}, \tag{12}$$

$$\mathrm{DP} = \frac{\sum_{q\in\mathcal{C}} \sum_{i\in E_D(q)} v_{qi}}{\sum_{q\in\mathcal{C}} |E_D(q)|}, \qquad \mathrm{CP} = \frac{\sum_{q\in\mathcal{Q}} \sum_{i\in U_q^{(0)}} v_{qi}}{\sum_{q\in\mathcal{Q}} |U_q^{(0)}|}. \tag{13}$$

Rates are displayed as percentages. TC and PR count cases; DP and CP count eligible case–item pairs. In-scope Preservation similarly counts correct items in $G_q^{(0)}$ and is distinct from CP, which is restricted to initially correct unrelated items.

Answer-scope Success requires the applicable target and dependent checks, every item in $U_q^{(0)}$, and every item in $G_q^{(0)}$ to pass. Proposal Judgment compares each judgment with the proposal's correctness under the ground truth. Judgment–Revision Consistency requires ACCEPT exactly when the final value equals the proposed value. Overall Success is the conjunction of Answer-scope Success, all correct judgments, and all consistent judgments on the same case. Final Accuracy instead requires all six final values to be correct, regardless of scope.

Under an incorrect proposal to an initially incorrect binary target, the proposal equals the initial value. Correct rejection with a consistent final answer then requires correcting the target, although these cases are outside TC's eligible set. DP counts dependent items in correction-eligible cases.

Extended coverage additionally checks initially incorrect dependents whenever the initial target is wrong, without changing the case denominator.

Table 12: Exact Test eligibility counts. Each row is shared by every method in that panel.

Panel	TC	PR	DP	CP	OS	PJ
ProofWriter / Qwen	300	203	357	3,256	1,212	1,818
ProofWriter / Llama	357	183	417	3,204	1,208	1,812
FOLIO / Qwen	273	218	318	3,324	1,236	1,854
FOLIO / Llama	339	200	414	3,392	1,252	1,878

TC/PR/OS count cases; DP/CP count eligible case–item pairs; PJ counts proposals. Different backbones have separate initial states and admission.

Algorithm 2 ClaimSift evaluation

Input: Constructed problems $\mathcal{X}$; initial policy π_0; revision methods $\mathcal{M}$; ground truth and checked dependencies.
Output: Successful and eligible counts for each metric and source.

for each problem $x \in \mathcal{X}$ **do**
  Generate one initial answer $y^{(0)} \sim \pi_0(x)$.
  Apply initial-answer admission; record excluded problems and continue with admitted problems.
  for each of the four feedback scenarios f **do**
    Construct $p = (x, y^{(0)}, f)$; fix eligible sets using the initial answer.
    for each method $M \in \mathcal{M}$ **do**
      Obtain judgments and the complete revised answer $(d, y^{(f)}) \leftarrow M(p)$.
      Check target, required dependent changes, and all initially correct items.
      Check proposal judgments and judgment–revision consistency.
      Count Overall Success iff eligible and every applicable criterion is satisfied.
      Accumulate component numerators and denominators; retain applicable failures.
    end for
  end for
end for
Pool counts across sources; return N/A for zero-denominator metrics.

B.6 Input and Output Examples

The following example presents the evidence, initial answer, feedback, and complete output together. It illustrates the scoring rules rather than a measured model response. The displayed item order makes the roles visible; benchmark inputs shuffle identifiers and omit role annotations.

Worked case: correct one proposal, reject the other

Evidence. Robin is red. Mira is blue. Every red entity is kind. Every kind entity is quiet.

Semantics. Closed world: a query is TRUE if derivable, and FALSE otherwise.

Queries and initial answer.

	Query	Initial		Query	Initial
C1	Robin is red	F	C4	Mira is red	F
C2	Robin is kind	F	C5	Mira is kind	F
C3	Robin is quiet	T	C6	Mira is quiet	T

Feedback. F1: C1 should be TRUE. F2: C5 should be TRUE.

Evaluator view. C1 is the target, C2–C3 its dependents, and C4–C6 unrelated. The reference is TTTFFF. Removing Robin's red fact blocks both dependent conclusions; restoring it recovers them and leaves Mira's labels unchanged.

One valid scoped response

```
{
  "feedback_verdicts": {"F1": "ACCEPT", "F2": "REJECT"},
  "claim_actions": {
    "C1": "UPDATE", "C2": "UPDATE", "C3": "KEEP",
    "C4": "KEEP", "C5": "KEEP", "C6": "KEEP"},
  "revised_answer": {
    "C1": "TRUE", "C2": "TRUE", "C3": "TRUE",
    "C4": "FALSE", "C5": "FALSE", "C6": "TRUE"}
}
```

Checks. TC: 1/1; DP: 1/1 (C2); CP: 2/2 (C4–C5). C3 remains correct. Both judgments are correct and consistent. PR is N/A because the target was initially incorrect.

Outcome. Answer-scope Success = 1; Overall Success = 1; Final Accuracy = 0. C6 retains a pre-existing unrelated error, which is outside Content Preservation's eligible set.

Four outcomes on the same input. The arrays below follow C1–C6; judgments follow F1–F2. A/R denote ACCEPT/REJECT. Colored entries identify the missed update or new damage. FA, AS, and OS use the definitions in Table 13.

(a) Overall Success

T T T F F T
Judgments: A, R **FA/AS/OS:** 0 / 1 / 1
C1 and C2 corrected; initially correct items kept.

(b) Missed dependency

T **F** T F F T
Judgments: A, R **FA/AS/OS:** 0 / 0 / 0
C2 remains wrong after C1 is corrected.

(c) New unrelated damage

T T T **T** F T
Judgments: A, R **FA/AS/OS:** 0 / 0 / 0
C4 changes from correct to incorrect.

(d) Incorrect rejection

T T T F F T
Judgments: R, R **FA/AS/OS:** 0 / 1 / 0
F1 is rejected although C1 adopts TRUE.

Table 13 retains the same ground truth and initial answer but supplies only F1 (C1=TRUE), isolating the checks without the second proposal. Its Final Accuracy row also repairs C6 and therefore passes Final Accuracy.

Table 13: Worked checks on one specified revision case. One denotes a pass; zero a failure.

Revision	C1–C6	Judgment	FA	AS	OS
Overall Success	TTTFFT	ACCEPT	0	1	1
Final Accuracy	TTTFFF	ACCEPT	1	1	1
Incomplete propagation	TFTFFT	ACCEPT	0	0	0
New unrelated damage	TTTTFT	ACCEPT	0	0	0
Incorrect rejection	TTTFFT	REJECT	0	1	0

FA: Final Accuracy; AS: Answer-scope Success; OS: Overall Success. Rows illustrate scoring rules, not competing methods.

C Benchmark Statistics

This section reports the constructed benchmark population and initial-answer admission. The 3,958 candidate constructed problems comprise 1,964 ProofWriter and 1,994 FOLIO problems. Each source problem contributes two linked variants. Training, Development, and Test contain 589/196/197 ProofWriter families and 598/199/200 FOLIO families, respectively, giving totals of 2,374, 790, and 794 candidate problems. The paired variants remain in the same split.

Table 14 provides the complete allocations and admission counts. The two backbones share candidate identities and ground-truth arrays but use separate initial answers. Admission precedes method-specific revision, so every method within a backbone receives the same cases. Backbone panels describe the same candidate inventory and are not added together. Paired target and dependent labels reverse while unrelated labels remain fixed; the corresponding source problems are checked for dependency necessity and unrelated-label invariance under the interventions specified in Appendix B.1.

Table 14: Candidate allocation and initial-answer admission.

Dataset / split	Candidates	Families	Qwen	Llama
ProofWriter / Training	1178	589	888	897
ProofWriter / Development	392	196	280	312
ProofWriter / Test	394	197	303	302
FOLIO / Training	1196	598	971	956
FOLIO / Development	398	199	323	314
FOLIO / Test	400	200	309	313

Each family contains two variants of one source problem. The inventory totals 3,958 candidates. Qwen and Llama columns count admitted problems; each yields four revision cases.

FOLIO with dependency annotations. The expanded FOLIO task defines one target, two dependents, and three unrelated items under a shared context. Its realization requires target-support proofs, blocking and restoration interventions, and checks that unrelated conclusions survive the interventions. Open-world UNKNOWN cases and contradictions must be resolved before selecting binary tasks, following Appendix B.1. The source-language problems undergo these semantic checks before initial-answer admission.

Training, Development, and Test are domain-specific dataset splits. FOLIO Training supplies Plan-SFT examples, preference data, and reference-risk calibration; Development selects the fixed penalty; Test evaluates the final methods without further selection. This panel uses domain-specific training and is distinct from zero-shot transfer of a ProofWriter-trained checkpoint.

Shared evaluation population. Method comparisons, auxiliary measures, feedback groups, and paired counts reuse the same answer and judgment states. Table 12 lists their eligibility units. Worked examples explain the scoring definitions separately from these comparisons.

C.1 Model Coverage and Motivation Diagnostic

The motivation panel aligns its candidate count with the 394 ProofWriter Test slots, not the full Training, Development, and Test inventory. For each of the same five declared seeds, we draw 197 source groups with replacement from the 256 groups represented in the saved diagnostic responses. A draw carries both variants, each model's initial-answer admission, all four revision cases, and every metric's successful and eligible integer counts. The source sequence is shared across models, including groups that admit no cases for a particular model. Joint outcomes are carried with the source block rather than inferred from marginal percentages.

Table 15 reports the mean and sample SD of the five resampled ratios. Counts are pooled within each draw before computing a ratio. Because eligibility varies across draws, averaging those ratios can differ from pooling all draws together. All seeds are retained, without selection for ranking or proximity to the original rates. Resampling preserves saved responses and does not create independent sources or training variation. Its source identities differ from those in the method comparison panels; only the candidate allocation is aligned.

Q8, L8, G4, GPT, Gem, and Q14 denote Qwen3-8B, Llama-3.1-8B-Instruct, Gemma-3-4B-IT, GPT-5-mini, Gemini-3.1-Flash-Lite, and Qwen3-14B. Saved local runs used an 8,192-token context and a 512-token output cap; API outputs had a 1,600-token cap including reasoning. The two API panels retain the separately recorded recovery of one invalid initial answer and its four revisions per model. Original first-pass records remain preserved. These differences and model-specific admission make the profiles behavioral diagnostics rather than a matched ranking of model quality.

Table 15: Six-model behavioral diagnostic at 394 ProofWriter candidate slots. Mean ± sample SD over five resamples of source groups, not training runs. Model-specific admission precludes a controlled model ranking.

Model	TC	PR	DP	CP	OS
Q8	$51.35_{\pm 2.82}$	$27.61_{\pm 1.86}$	$0.37_{\pm 0.16}$	$90.92_{\pm 0.53}$	$15.09_{\pm 1.05}$
L8	$88.67_{\pm 2.53}$	$10.92_{\pm 2.00}$	$34.29_{\pm 1.82}$	$61.40_{\pm 2.00}$	$2.60_{\pm 0.47}$
G4	$71.92_{\pm 1.50}$	$20.10_{\pm 3.49}$	$10.99_{\pm 2.70}$	$81.52_{\pm 0.84}$	$17.53_{\pm 1.13}$
GPT	$32.20_{\pm 1.47}$	$73.06_{\pm 3.05}$	$5.39_{\pm 0.68}$	$90.30_{\pm 0.65}$	$38.77_{\pm 2.00}$
Gem	$28.58_{\pm 1.66}$	$57.51_{\pm 3.87}$	$4.41_{\pm 0.96}$	$91.42_{\pm 0.70}$	$33.51_{\pm 1.06}$
Q14	$3.49_{\pm 1.61}$	$100.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$97.09_{\pm 0.41}$	$27.10_{\pm 1.07}$

Table 16: Admission after sampling of source groups. Each column contains 394 candidate slots; each admitted problem contributes four revision cases. Repeated source draws count as slots, not new independent sources.

Model / seed	11	23	37	53	71
Q8	244	279	267	288	269
L8	244	248	263	237	258
G4	241	243	273	240	233
GPT	253	247	267	262	251
Gem	341	357	336	341	343
Q14	282	317	288	296	293
Distinct source identities	138	133	138	130	141

The admission counts in Table 16 are model-specific. Each admitted problem contributes exactly four cases to Overall Success, while the component denominators follow their eligible cases and items. The method comparison panels below evaluate trained models separately from this diagnostic based on resampled source groups. No six-model FOLIO with dependency annotations panel is inferred from these ProofWriter responses.

D Training Runs and Evaluation Records

Within each dataset/backbone panel, five independent repetitions each train a Plan-SFT checkpoint. In each repetition, the trained revision methods share that initialization and then undergo separate method-specific training on the fixed dataset splits. Complete proposal judgments and revised answers are obtained from the corresponding evaluated models. Training-run variability is distinct from source-group resampling in the six-model motivation diagnostic.

D.1 Shared Inputs and Unmodified Model Responses

Candidate identities, ground truth, initial answers, feedback scenarios, and eligibility are shared across methods within each dataset/backbone panel. Each response is scored against the applicable case-level and item-level criteria. Overall Success follows from the conjunction of those checks; it is not assigned through a success quota. Responses and judgments are not changed to enforce a target success rate or method ranking.

Development selects the fixed penalty using Overall Success, then Content Preservation, then the smaller weight. The selected penalty is frozen for Test. All fixed settings remain reported. Keep All and Accept All apply deterministic input-only rules; Resolve then Judge reconstructs an answer and judges proposals by agreement with that answer. Rule-based baselines are distinguished from independently trained methods.

D.2 Recomputation and Statistical Summaries

Complete model responses are retained for each run. Integer scores, paired outcomes, initial-correctness groups, and extended dependency checks are recomputed from the same responses.

Measured method-comparison results summarize the five training runs; deterministic input-only strategies use the same fixed cases. Appendix C.1 gives the separate motivation-diagnostic resampling procedure. The worked examples explain the scoring rules independently of the aggregate measurements.

Means, standard deviations, and count tables. For per-seed percentage x_r, we use

$$\bar{x} = \frac{1}{R}\sum_{r=1}^{R} x_r, \qquad s = \sqrt{\frac{1}{R-1}\sum_{r=1}^{R}(x_r - \bar{x})^2}. \tag{14}$$

We set $R = 5$. Error bars represent one sample SD, not standard errors or confidence intervals. Zero SD for deterministic retention and acceptance rules is expected because their inputs are fixed. SDs are computed rather than constrained to a desired range. Fixed eligibility implies that the mean percentage equals the ratio of summed successes to summed opportunities. Those sums count five independent training runs, not new independent sources. Paired differences use seed-matched outcomes, and their SD is computed directly rather than subtracting marginal SDs. Per-seed Overall Success scores are retained in the CSV supplement.

E Analytical Properties

This section examines the relationship between component measures and Overall Success, bounds for unconditional strategies, and the role of preservation cost. We then analyze risk normalization and penalty updates. The results follow from the scoring and optimization definitions, independently of the reported method scores.

E.1 Component Measures and Overall Success

For two binary requirements on the same n cases, let a, b be their pass counts and z their joint count. Inclusion–exclusion gives

$$\max(0, a + b - n) \le z \le \min(a, b). \tag{15}$$

A joint pass belongs to both pass sets, giving the upper bound; their union has at most n elements, giving the lower bound. Two checks that each pass three of four cases can have either two or three joint passes. Individual averages do not determine coordination. Since TC, PR, DP, and CP have different eligibility units, their percentages cannot be multiplied to recover OS.

Final Accuracy and OS are not interchangeable. Correct answers with an incorrect judgment can fail OS. Conversely, a case passing Overall Success may leave a pre-existing unrelated error unchanged and fail Final Accuracy, as in Table 13. The auxiliary tables retain both measures rather than replacing one with the other.

E.2 Bounds for Unconditional Strategies

Fix the admitted problems and include all four feedback scenarios equally. Let u denote the fraction with an initially incorrect target. Keep All returns the initial answer unchanged; Accept All accepts and adopts every supplied proposal.

Keep All. For an initially incorrect target, all three correct-target scenarios require a change. The remaining incorrect-target proposal equals the initial value: rejecting it without changing that value is inconsistent, while accepting it is an incorrect judgment. Thus all four scenarios fail, even when Keep All is granted arbitrary judgments:

$$\mathrm{OS}_{\mathrm{keep}} \le 1 - u, \qquad \mathrm{TC}_{\mathrm{keep}} = 0 \quad \text{when correction opportunities exist.} \tag{16}$$

Perfect CP is therefore compatible with no correction ability. This upper bound grants favorable judgments on initially correct targets; it is not a guarantee for any context-blind implementation.

Accept All. Two scenarios contain an incorrect proposal. Accepting it fails Proposal Judgment regardless of other outputs, leaving at most two successful scenarios:

$$\mathrm{OS}_{\mathrm{accept}} \leq \tfrac{1}{2}, \qquad \mathrm{PR}_{\mathrm{accept}} = 0 \quad \text{when resistance opportunities exist.} \tag{17}$$

The bound assumes complete, equally weighted scenarios and allows arbitrary values for items without proposals. Literal copying can perform worse. These bounds do not exclude strategies that exploit proposal order or other format cues.

E.3 Preservation Cost and Candidate Ordering

Two revisions can repair the same errors and judge proposals correctly while missing the same dependent update. Both fail Overall Success, but their newly introduced damage may differ. Preservation cost distinguishes these candidates even when the binary success reward does not.

For two candidates in one case, let $\Delta r = r(y_a) - r(y_b)$, $\Delta h = h(y_a) - h(y_b)$, and $\Delta \ell = \ell_{\mathrm{ref}}(y_a) - \ell_{\mathrm{ref}}(y_b)$. Then

$$R(y_a) - R(y_b) = \Delta r - \tau \Delta \ell - \lambda \Delta h. \tag{18}$$

When $\Delta h > 0$, the more damaging candidate has higher return exactly when

$$\lambda < \lambda^* = \frac{\Delta r - \tau \Delta \ell}{\Delta h}. \tag{19}$$

Raising the weight can reverse its preference when this threshold lies within the controller's permitted range. Candidates with equal costs retain their ordering. Correcting an initially incorrect item creates no preservation cost; the cost counts new errors at initially correct items.

For $K > 1$ candidates, subtracting the other candidates' mean return gives

$$A_{qa} - A_{qb} = \frac{K}{K-1}\big(R(y_a) - R(y_b)\big). \tag{20}$$

Relative advantages therefore preserve return ordering. The identity characterizes the comparison signal; the resulting policy update also depends on sampled candidates and their gradients.

E.4 Risk Normalization and Penalty Updates

For positive denominators $D_{\mathrm{T}}, D_{\mathrm{F}}$, write $D = D_{\mathrm{T}} + D_{\mathrm{F}}$, $\bar{r} = \sum_g D_g r_g / D$, and $\bar{\epsilon} = \sum_g D_g \epsilon_g / D$. Averaging preservation cost over all N training cases yields

$$\frac{1}{N}\sum_q \mathbb{E}[h(y_q)] = \frac{\sum_g \sum_q \mathbb{E}[h_g(y_q)]}{D} = \bar{r}. \tag{21}$$

For a fixed policy and a uniformly sampled problem among P problems, $\mathbb{E}[PH_g^{\mathcal{B}}/(KD_g)] = r_g$. This expectation covers problem and candidate sampling. A fixed-order training pass with a changing policy does not meet the fixed-policy sampling premise at each batch; the identity explains normalization rather than claiming unbiased full-population risk estimates throughout training.

An interpretation as constrained optimization subtracts $\lambda(\bar{r} - \bar{\epsilon})$ from regularized revision reward. Since $\lambda\bar{\epsilon}$ is constant across candidates, it does not change their ordering. Equation 9 raises the next penalty above the pooled budget and lowers it below the budget, subject to projection. Pooling permits lower risk for one label to offset excess risk for the other. Separate Penalties retains the pooled cost scale in Equation 10 but responds to each label's constraint violation independently in Equation 11. From equal weights, the updates coincide when both label-specific constraint violations coincide; equal starting weights alone do not make the rules equivalent.

The update specifies the direction of risk response. Feasibility and optimality additionally depend on the finite cap, policy changes, and sampling. Controller trajectories and final-policy risk require measurement rather than inference from dataset size or final scores. Tables 4 and 27 compare revision quality under adaptive and fixed weights separately.

F Complete Results and Error Analysis

F.1 Method Comparisons and Preservation Controls

Tables 17 and 18 present every learned method and rule-based strategy on the shared states. PARO has the highest Overall Success among the compared trained revision methods, while other methods retain stronger component scores. Keep All achieves higher aggregate Overall Success in both datasets but no correction or propagation. Table 19 separates these behaviors by correction need; the complete comparisons retain every strategy under the original scoring rules.

Table 17: ProofWriter: complete comparison.

Method	TC ↑	PR ↑	DP ↑	CP ↑	OS ↑
Keep All	0.00 ±0.00	**100.00** ±0.00	0.00 ±0.00	**100.00** ±0.00	**67.00** ±0.00
Accept All	**100.00** ±0.00	0.00 ±0.00	0.00 ±0.00	90.69 ±0.00	33.50 ±0.00
Plan-SFT	79.00 ±3.88	83.05 ±4.88	59.61 ±2.91	90.68 ±0.42	40.20 ±3.76
DPO	84.80 ±2.22	72.32 ±2.59	66.16 ±4.82	92.77 ±1.15	44.88 ±2.57
TD-DPO	83.80 ±2.69	79.51 ±2.89	64.59 ±2.85	93.16 ±0.93	44.72 ±3.00
TDPO1	87.20 ±0.61	77.54 ±3.39	67.84 ±3.71	93.83 ±0.84	46.22 ±2.79
SimPO	85.67 ±0.88	77.34 ±5.01	65.66 ±3.55	93.37 ±0.63	45.35 ±2.58
IPO	86.00 ±2.15	76.16 ±1.66	64.99 ±2.57	93.14 ±0.79	46.49 ±2.73
w/o Preservation	<u>92.40</u> ±1.77	76.55 ±0.75	**76.86** ±1.84	89.70 ±1.75	39.54 ±3.09
Fixed 0.5	79.33 ±2.13	95.07 ±1.30	65.71 ±2.10	96.64 ±0.22	51.95 ±2.84
Fixed 1.0	77.47 ±2.59	95.47 ±1.68	72.27 ±2.22	96.17 ±0.57	53.45 ±2.58
Fixed 2.0	75.20 ±2.82	<u>98.03</u> ±1.21	67.56 ±2.57	<u>97.01</u> ±0.52	52.16 ±2.05
Separate Penalties	81.80 ±2.30	95.67 ±2.07	71.43 ±1.80	96.31 ±0.73	53.18 ±3.03
PARO	84.93 ±2.30	92.71 ±1.76	72.49 ±2.25	96.19 ±0.15	<u>54.47</u> ±2.98
Resolve then Judge	86.40 ±4.16	86.50 ±2.71	<u>73.28</u> ±5.09	90.79 ±0.54	42.66 ±2.95

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

Table 18: FOLIO: complete comparison.

Method	TC ↑	PR ↑	DP ↑	CP ↑	OS ↑
Keep All	0.00 ±0.00	**100.00** ±0.00	0.00 ±0.00	**100.00** ±0.00	**70.55** ±0.00
Accept All	**100.00** ±0.00	0.00 ±0.00	0.00 ±0.00	90.70 ±0.00	35.28 ±0.00
Plan-SFT	74.21 ±7.45	84.13 ±4.01	51.70 ±4.69	88.06 ±1.89	35.89 ±4.58
DPO	80.07 ±2.70	74.13 ±2.09	59.06 ±4.14	90.53 ±0.58	45.02 ±1.69
TD-DPO	79.34 ±2.18	83.12 ±3.53	59.12 ±3.47	91.53 ±2.18	42.69 ±3.95
TDPO1	80.73 ±2.88	76.79 ±1.73	59.50 ±5.32	91.85 ±0.78	43.93 ±2.41
SimPO	79.41 ±2.71	75.60 ±3.30	59.69 ±7.27	90.80 ±2.23	43.04 ±5.38
IPO	78.17 ±4.48	78.72 ±3.98	58.93 ±5.42	90.86 ±1.44	44.13 ±4.69
w/o Preservation	<u>86.15</u> ±2.91	77.89 ±2.98	**72.08** ±3.08	87.56 ±0.81	38.33 ±1.29
Fixed 0.5	76.41 ±1.97	94.86 ±2.41	59.94 ±5.68	94.55 ±0.70	48.77 ±5.25
Fixed 1.0	71.50 ±3.71	95.69 ±1.61	63.84 ±6.64	94.10 ±1.32	52.94 ±5.90
Fixed 2.0	70.40 ±5.40	<u>97.16</u> ±0.82	60.50 ±5.22	<u>94.69</u> ±1.30	51.00 ±3.84
Separate Penalties	72.09 ±3.01	95.60 ±1.32	62.08 ±6.51	94.22 ±0.85	48.77 ±2.89
PARO	79.05 ±3.10	94.77 ±1.54	66.29 ±4.54	94.50 ±0.79	<u>55.66</u> ±3.70
Resolve then Judge	82.20 ±3.68	83.94 ±5.30	<u>70.00</u> ±9.18	89.27 ±3.89	37.75 ±8.48

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

F.2 Overall Success by Revision Requirement

Table 19 gives the complete integer partition behind Table 2. Correction requires an initially incorrect target and a correct target proposal. Resistance requires an initially correct target and an incorrect target proposal. Other contains the two remaining combinations and includes cases requiring a target change to reject an incorrect proposal consistently. Within each seed, the three groups

are disjoint and cover every case; their successful and eligible counts separately sum to Overall. The reported Overall rate is their denominator-weighted combination, not the mean of their three percentages. The same identities hold after summing the five independent training runs.

Table 19: Complete Overall Success partition for all methods.

Domain / method	Correction	Resistance	Other	Overall
ProofWriter / Keep All	0/1500	1015/1015	3045/3545	4060/6060
ProofWriter / Accept All	0/1500	0/1015	2030/3545	2030/6060
ProofWriter / Plan-SFT	606/1500	399/1015	1431/3545	2436/6060
ProofWriter / DPO	661/1500	451/1015	1608/3545	2720/6060
ProofWriter / TD-DPO	691/1500	444/1015	1575/3545	2710/6060
ProofWriter / TDPO1	688/1500	467/1015	1646/3545	2801/6060
ProofWriter / SimPO	695/1500	447/1015	1606/3545	2748/6060
ProofWriter / IPO	711/1500	443/1015	1663/3545	2817/6060
ProofWriter / w/o Preservation	626/1500	348/1015	1422/3545	2396/6060
ProofWriter / Fixed 0.5	757/1500	551/1015	1840/3545	3148/6060
ProofWriter / Fixed 1.0	800/1500	551/1015	1888/3545	3239/6060
ProofWriter / Fixed 2.0	776/1500	590/1015	1795/3545	3161/6060
ProofWriter / Separate Penalties	799/1500	534/1015	1890/3545	3223/6060
ProofWriter / PARO	816/1500	555/1015	1930/3545	3301/6060
ProofWriter / Resolve then Judge	528/1500	449/1015	1608/3545	2585/6060
FOLIO / Keep All	0/1365	1090/1090	3270/3725	4360/6180
FOLIO / Accept All	0/1365	0/1090	2180/3725	2180/6180
FOLIO / Plan-SFT	466/1365	418/1090	1334/3725	2218/6180
FOLIO / DPO	596/1365	512/1090	1674/3725	2782/6180
FOLIO / TD-DPO	566/1365	481/1090	1591/3725	2638/6180
FOLIO / TDPO1	578/1365	504/1090	1633/3725	2715/6180
FOLIO / SimPO	575/1365	480/1090	1605/3725	2660/6180
FOLIO / IPO	591/1365	488/1090	1648/3725	2727/6180
FOLIO / w/o Preservation	523/1365	392/1090	1454/3725	2369/6180
FOLIO / Fixed 0.5	632/1365	571/1090	1811/3725	3014/6180
FOLIO / Fixed 1.0	702/1365	621/1090	1949/3725	3272/6180
FOLIO / Fixed 2.0	662/1365	632/1090	1858/3725	3152/6180
FOLIO / Separate Penalties	639/1365	578/1090	1797/3725	3014/6180
FOLIO / PARO	737/1365	656/1090	2047/3725	3440/6180
FOLIO / Resolve then Judge	384/1365	437/1090	1512/3725	2333/6180

Fixed λ: w/o Adaptation with penalty weight λ. Integer counts sum five independent training runs of the same cases. Correction, Resistance, and Other partition Overall; their denominator-weighted rates reproduce every aggregate. Complete per-seed metrics and paired transitions are exported as CSV.

F.3 Initial Correctness and Preservation Trade-offs

Let p be the assigned fraction of admitted problems with an initially correct target. We retain the same outcomes within each target-correctness group and give all four feedback scenarios equal weight. If s_r^{correct} and $s_r^{\text{incorrect}}$ denote their Overall Success rates in realization r, the reweighted rate is

$$S_r(p) = p\, s_r^{\text{correct}} + (1-p)\, s_r^{\text{incorrect}}.$$

Each group retains its original relative case weights, including the relationships among dependent and unrelated errors. This operation changes the evaluation mixture, not the answers or the policy. Evaluating at the original p recovers every run's reported Overall Success exactly. We first reweight each run and then compute its mean and sample SD, retaining the covariance between the two groups.

Figure 5 shows the full range of mixtures. Keep All has $s_r^{\text{incorrect}} = 0$ and $s_r^{\text{correct}} = 1$, giving $S_r(p) = p$. The mean PARO curve crosses Keep All at $p = 54.86\%$ for ProofWriter and $p = 55.62\%$ for FOLIO. The original mixtures, 67.00% and 70.55%, lie above these crossings. PARO exceeds the displayed fixed-weight configurations in mean across the range, while retaining both

gains and losses against Keep All. This is a sensitivity calculation on the model outputs defined in Section 6.1, not fresh model inference under distribution shift.

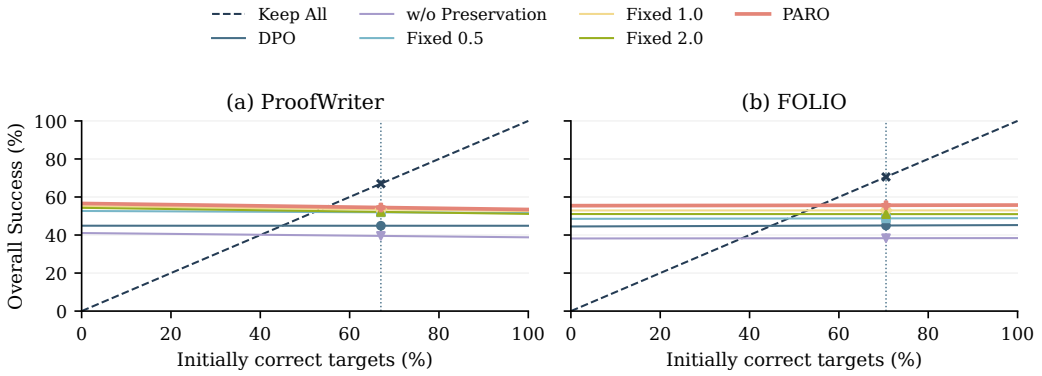

Figure 5: Overall Success under reweighting of the retained model outputs. Lines show means across the same five independent training runs. Markers and vertical dotted lines identify the original target-correctness mixture and recover the reported Overall Success. Group outcomes and the equal four-scenario mix are fixed; line slopes are mixture effects, not training trends.

Table 20: Overall Success under reweighting with fixed group scores.

Domain / method	$p = 0.2$	$p = 0.4$	$p = 0.5$	$p = 0.6$	$p = 0.8$
ProofWriter / Keep All	20.00 ±0.00	40.00 ±0.00	50.00 ±0.00	**60.00** ±0.00	**80.00** ±0.00
ProofWriter / DPO	44.90 ±2.70	44.89 ±2.59	44.89 ±2.56	44.89 ±2.56	44.88 ±2.62
ProofWriter / w/o Preservation	40.60 ±2.87	40.15 ±2.85	39.92 ±2.91	39.70 ±3.00	39.24 ±3.30
ProofWriter / Fixed 0.5	52.44 ±3.30	52.23 ±3.08	52.13 ±2.98	52.02 ±2.89	51.81 ±2.74
ProofWriter / Fixed 1.0	54.68 ±2.13	54.15 ±2.21	53.89 ±2.31	53.63 ±2.45	53.11 ±2.83
ProofWriter / Fixed 2.0	53.66 ±1.94	53.02 ±1.88	52.70 ±1.91	52.39 ±1.98	51.75 ±2.22
ProofWriter / PARO	**55.96** ±2.95	**55.33** ±2.89	**55.01** ±2.90	54.69 ±2.94	54.06 ±3.08
FOLIO / Keep All	20.00 ±0.00	40.00 ±0.00	50.00 ±0.00	**60.00** ±0.00	**80.00** ±0.00
FOLIO / DPO	44.65 ±2.16	44.80 ±1.91	44.87 ±1.82	44.94 ±1.74	45.08 ±1.66
FOLIO / w/o Preservation	38.23 ±1.41	38.27 ±1.30	38.29 ±1.27	38.31 ±1.27	38.35 ±1.34
FOLIO / Fixed 0.5	48.59 ±5.31	48.66 ±5.27	48.70 ±5.26	48.73 ±5.25	48.80 ±5.25
FOLIO / Fixed 1.0	52.92 ±6.34	52.93 ±6.15	52.94 ±6.06	52.94 ±5.98	52.95 ±5.83
FOLIO / Fixed 2.0	51.03 ±4.55	51.02 ±4.26	51.02 ±4.12	51.01 ±3.98	51.00 ±3.72
FOLIO / PARO	**55.50** ±3.12	**55.57** ±3.32	**55.60** ±3.44	55.63 ±3.56	55.69 ±3.83

Fixed λ: w/o Adaptation with penalty weight λ. Percentages, mean ± sample SD across the same five independent training runs. p is the assigned fraction of initially correct targets. Within-stratum case weights and the equal four-scenario mix are unchanged. These weighted rates are not integer success counts on new samples. Bold/underline mark the best/second distinct means within each domain and prevalence.

Figure 6 complements the reweighting analysis with the original-mixture profiles. The upper panels compare Target Correction with Content Preservation. The lower panels apply Overall Success separately to the Correction and Resistance groups defined in the main text. Rings mark nondominated mean points: no other displayed configuration is at least as high on both axes and strictly higher on one. Dominance is evaluated separately for each domain and metric pair. It is a relation among displayed means, not a statistical significance test. In particular, a fixed weight can retain higher preservation or resistance while PARO combines those behaviors with greater correction.

F.4 Answer Quality and Proposal Judgments

Tables 21 and 22 compare Final Accuracy, Answer-scope Success, Proposal Judgment, Judgment–Revision Consistency, and Overall Success. FA and AS count cases; PJ and JRC count proposals. Subtracting their percentages therefore does not give a case-level failure count. OS cannot exceed AS on the same cases, whereas FA and OS have no such ordering. Resolve then Judge has perfect

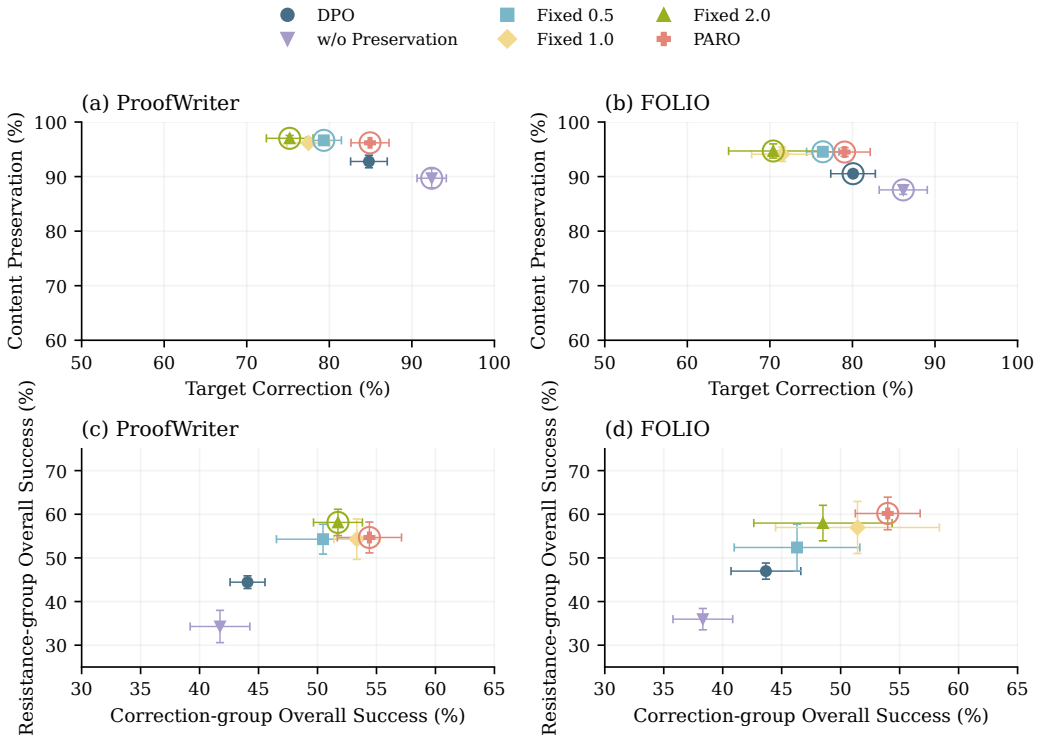

Figure 6: Correction and preservation profiles in the experimental evaluation. Panels (a,b) show component measures; (c,d) show group-specific Overall Success. Points use the original evaluation mixture. Horizontal and vertical bars give one sample SD across the same five independent training runs. Outer rings identify nondominated means among the six displayed method configurations; all three fixed weights are retained.

JRC by definition, leaving answer quality and judgment correctness as the remaining failure sources. The complete comparison tables retain all component scores.

Table 21: ProofWriter: auxiliary checks.

Method	FA ↑	AS ↑	PJ ↑	JRC ↑	OS ↑
Keep All	19.80 ±0.00	**75.25** ±0.00	78.00 ±0.00	**100.00** ±0.00	**67.00** ±0.00
Accept All	9.90 ±0.00	41.75 ±0.00	66.67 ±0.00	**100.00** ±0.00	33.50 ±0.00
Plan-SFT	42.94 ±3.28	48.12 ±2.53	82.05 ±1.47	73.27 ±1.91	40.20 ±3.76
DPO	48.48 ±3.10	53.45 ±3.27	87.12 ±0.61	77.80 ±1.00	44.88 ±2.57
TD-DPO	47.79 ±3.20	53.63 ±3.37	88.26 ±0.71	78.11 ±1.41	44.72 ±3.00
TDPO1	50.78 ±2.63	56.39 ±2.37	87.56 ±1.50	79.26 ±1.80	46.22 ±2.79
SimPO	49.01 ±1.97	54.41 ±1.93	86.88 ±1.39	78.49 ±1.64	45.35 ±2.58
IPO	49.16 ±2.18	54.79 ±1.76	87.32 ±1.11	78.14 ±1.23	46.49 ±2.73
w/o Preservation	49.98 ±3.15	56.01 ±3.34	80.89 ±1.24	73.32 ±1.62	39.54 ±3.09
Fixed 0.5	54.01 ±3.07	58.91 ±2.69	**91.30** ±0.84	81.65 ±0.95	51.95 ±2.84
Fixed 1.0	56.32 ±3.03	61.17 ±3.25	90.72 ±0.81	80.15 ±1.22	53.45 ±2.58
Fixed 2.0	54.11 ±2.24	58.66 ±2.14	90.54 ±1.11	80.15 ±1.65	52.16 ±2.05
Separate Penalties	<u>56.44</u> ±3.55	61.49 ±3.98	89.75 ±1.08	81.38 ±1.18	53.18 ±3.03
PARO	**58.33** ±2.72	<u>63.25</u> ±2.20	<u>91.21</u> ±1.02	<u>82.89</u> ±1.99	<u>54.47</u> ±2.98
Resolve then Judge	32.01 ±3.32	43.42 ±2.76	87.94 ±1.51	**100.00** ±0.00	42.66 ±2.95

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means. FA: Final Accuracy; AS: Answer-scope Success; PJ: Proposal Judgment; JRC: Judgment–Revision Consistency.

Table 22: FOLIO: auxiliary checks.

Method	FA ↑	AS ↑	PJ ↑	JRC ↑	OS ↑
Keep All	19.42 ±0.00	**77.91** ±0.00	80.37 ±0.00	**100.00** ±0.00	**70.55** ±0.00
Accept All	9.71 ±0.00	42.64 ±0.00	66.67 ±0.00	**100.00** ±0.00	35.28 ±0.00
Plan-SFT	37.59 ±4.75	41.84 ±4.58	80.26 ±3.73	71.42 ±3.49	35.89 ±4.58
DPO	46.18 ±1.76	49.64 ±1.98	87.93 ±0.97	76.57 ±1.69	45.02 ±1.69
TD-DPO	44.55 ±4.38	49.08 ±4.66	87.37 ±1.15	77.30 ±1.55	42.69 ±3.95
TDPO1	45.76 ±2.16	50.03 ±1.97	86.74 ±0.98	76.20 ±1.16	43.93 ±2.41
SimPO	44.45 ±5.83	48.19 ±5.74	87.28 ±2.67	76.35 ±2.79	43.04 ±5.38
IPO	45.42 ±5.05	49.34 ±5.09	86.77 ±1.12	76.25 ±2.62	44.13 ±4.69
w/o Preservation	46.59 ±1.76	51.72 ±1.77	81.13 ±0.99	72.58 ±1.21	38.33 ±1.29
Fixed 0.5	50.23 ±5.23	54.17 ±5.36	89.98 ±1.32	79.10 ±2.97	48.77 ±5.25
Fixed 1.0	<u>53.87</u> ±5.80	57.23 ±5.12	<u>91.27</u> ±1.31	79.32 ±2.89	52.94 ±5.90
Fixed 2.0	51.78 ±3.80	54.72 ±4.11	90.50 ±0.83	78.57 ±2.19	51.00 ±3.84
Separate Penalties	50.42 ±3.23	54.24 ±2.89	89.78 ±1.11	78.34 ±1.91	48.77 ±2.89
PARO	**56.83** ±4.01	<u>60.26</u> ±4.04	**91.48** ±0.92	<u>81.87</u> ±1.90	<u>55.66</u> ±3.70
Resolve then Judge	28.16 ±8.91	38.43 ±8.53	85.29 ±4.12	**100.00** ±0.00	37.75 ±8.48

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means. FA: Final Accuracy; AS: Answer-scope Success; PJ: Proposal Judgment; JRC: Judgment–Revision Consistency.

F.5 New Damage and Error Categories

Tables 23 and 24 report initially correct items made incorrect by revision, separating target/dependent items from unrelated items. In each seed, unrelated damage equals $100 - \mathrm{CP}$. Total damage weights the two groups by their eligible denominators rather than averaging their rates. The denominators repeat each shared initial answer across the four feedback scenarios. These final-state measures are distinct from training-batch risk and final-policy budget verification.

Table 23: New damage on initially correct items: ProofWriter.

Method	In scope ↓	Unrelated ↓	All ↓
DPO	28.50 ±3.07	7.23 ±1.15	15.67 ±1.83
w/o Pres.	**19.75** ±1.63	10.30 ±1.75	14.05 ±1.49
Fixed 0.5	27.34 ±2.26	<u>3.36</u> ±0.22	12.88 ±0.97
Fixed 1.0	25.41 ±2.66	3.83 ±0.57	12.40 ±1.19
Fixed 2.0	28.01 ±2.50	**2.99** ±0.52	12.92 ±1.30
Sep. Pen.	24.84 ±3.05	3.69 ±0.73	<u>12.09</u> ±1.54
PARO	<u>22.41</u> ±2.06	3.81 ±0.15	**11.19** ±0.90

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD of damaged-item percentages across five seeds; lower is better. Bold/underline mark the lowest/second distinct displayed means. Per-seed denominators: 2,144 in scope, 3,256 unrelated, and 5,400 total case–item pairs. Fixed: w/o Adaptation; w/o Pres.: w/o Preservation; Sep. Pen.: Separate Penalties.

FOLIO mechanism views. Figure 7 separates FOLIO damage by item group and reports paired Overall Success gains and losses. Scope and unrelated damage contributions use one common denominator, so their bar heights add to total damage. The paired comparison retains lost successes. Both views summarize the same model outputs.

Tables 25 and 26 instead use revision cases as the common denominator. A case may fail a required target check, miss a required dependent update, damage an initially correct item, misjudge a proposal, or contradict its judgment in the revised answer. Required target criteria use the eligible cases for TC or PR; dependent criteria use the eligible items for DP. A damaged correct target may fail both the target and damage checks. An incorrect proposal judgment may also be inconsistent, but neither property implies the other. Their union is exactly the set of failed Overall Success cases, checked separately for every method and seed; adding columns double-counts overlapping failures.

These frequencies are computed from the same unmodified model responses as the main tables. Failure categories can overlap; they describe observable scoring failures rather than, by themselves,

Table 24: New damage on initially correct items: FOLIO.

Method	In scope ↓	Unrelated ↓	All ↓
DPO	33.84 ±1.87	9.47 ±0.58	19.46 ±0.88
w/o Pres.	**22.42** ±1.95	12.44 ±0.81	16.53 ±0.77
Fixed 0.5	31.50 ±4.90	<u>5.45</u> ±0.70	16.13 ±2.40
Fixed 1.0	30.26 ±3.57	5.90 ±1.32	<u>15.88</u> ±2.18
Fixed 2.0	32.60 ±2.56	**5.31** ±1.30	16.49 ±1.72
Sep. Pen.	31.27 ±2.80	5.78 ±0.85	16.23 ±1.57
PARO	<u>27.31</u> ±2.85	5.50 ±0.79	**14.44** ±1.47

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD of damaged-item percentages across five seeds; lower is better. Bold/underline mark the lowest/second distinct displayed means. Per-seed denominators: 2,308 in scope, 3,324 unrelated, and 5,632 total case–item pairs. Fixed: w/o Adaptation; w/o Pres.: w/o Preservation; Sep. Pen.: Separate Penalties.

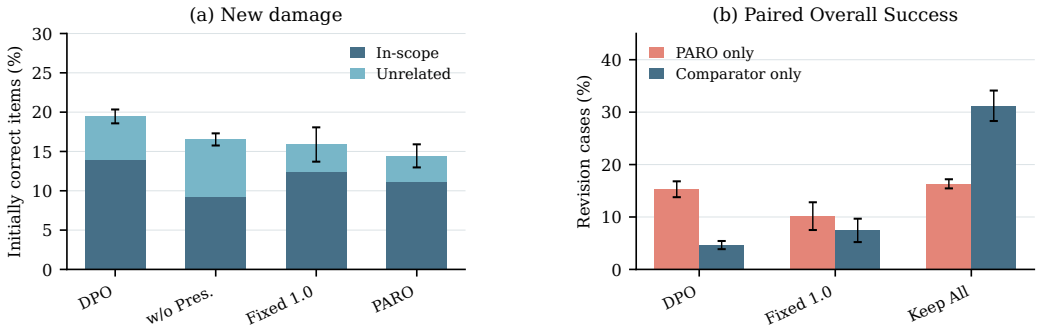

Figure 7: FOLIO damage by item group and paired Overall Success gains and losses. The fixed weight is selected on Development. Means and one-SD error bars reuse the five independent training runs; total damage is the error-bar quantity in (a).

Table 25: Overlapping case-level failure checks: ProofWriter.

Method	Tgt. ↓	Dep. ↓	Dmg. ↓	Judg. ↓	Cons. ↓
DPO	8.40 ±0.60	9.08 ±1.13	43.14 ±3.54	18.68 ±0.96	31.29 ±1.42
w/o Pres.	5.81 ±0.48	**6.27** ±0.55	41.06 ±3.30	27.61 ±1.49	37.54 ±1.75
Fixed 0.5	5.94 ±0.65	9.11 ±0.45	37.31 ±2.81	**12.81** ±1.28	<u>26.07</u> ±1.38
Fixed 1.0	6.34 ±0.73	7.57 ±0.56	35.07 ±2.94	13.70 ±1.08	28.22 ±1.44
Fixed 2.0	6.47 ±0.66	8.60 ±0.58	37.19 ±1.96	14.01 ±1.64	27.81 ±2.19
Sep. Pen.	<u>5.23</u> ±0.84	7.72 ±0.52	<u>34.82</u> ±4.06	15.02 ±1.48	26.35 ±1.63
PARO	**4.95** ±0.83	<u>7.31</u> ±0.74	**33.14** ±2.08	<u>13.09</u> ±1.48	**24.41** ±2.61

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD; percentages of all 1,212 cases per seed, not percentages of failures. Tgt.: required target check; Dep.: required dependent check; Dmg.: any new damage; Judg.: any wrong proposal judgment; Cons.: any judgment–revision mismatch. Categories overlap; their union is exactly the complement of Overall Success. Abbreviations and marking for metrics where lower values are better follow the new-damage tables.

Table 26: Overlapping case-level failure checks: FOLIO.

Method	Tgt. ↓	Dep. ↓	Dmg. ↓	Judg. ↓	Cons. ↓
DPO	8.96 ±0.93	9.58 ±0.89	47.96 ±2.03	17.18 ±1.58	31.73 ±2.12
w/o Pres.	6.96 ±0.85	**6.63** ±0.70	45.61 ±1.76	26.94 ±1.58	38.01 ±1.82
Fixed 0.5	6.12 ±0.79	9.29 ±1.22	42.64 ±5.13	14.45 ±1.79	28.58 ±3.56
Fixed 1.0	7.06 ±1.00	8.50 ±1.65	40.34 ±4.77	12.72 ±1.80	27.72 ±3.87
Fixed 2.0	7.04 ±1.24	9.03 ±1.04	42.39 ±3.65	13.85 ±1.15	28.90 ±3.25
Sep. Pen.	6.94 ±0.84	8.90 ±1.41	42.56 ±3.03	14.87 ±1.66	29.51 ±2.29
PARO	**5.55** ±0.85	7.82 ±0.86	**37.48** ±3.84	**12.44** ±1.26	**25.03** ±2.67

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD; percentages of all 1,236 cases per seed, not percentages of failures. Tgt.: required target check; Dep.: required dependent check; Dmg.: any new damage; Judg.: any wrong proposal judgment; Cons.: any judgment–revision mismatch. Categories overlap; their union is exactly the complement of Overall Success. Abbreviations and marking for metrics where lower values are better follow the new-damage tables.

the causal mechanism of model errors. The examples in Appendix B.6 explain the checks independently of these frequencies.

F.6 Tuned Fixed Penalty

Table 27 selects among fixed weights using mean Development OS, then mean CP, then the smaller weight. It contains a separate selection for each declared dataset/backbone panel. Each selected weight remains fixed across its five Test seeds. The same Test population is used for all fixed and adaptive rows within a panel. Test does not select a weight, and all fixed values remain visible. These comparisons do not measure hyperparameter-search cost or show that PARO has no tunable parameters.

Table 27: Fixed-weight selection using all five Development seeds.

Panel	0.5	1.0	2.0	Selected
ProofWriter / Qwen	51.96 ±2.90	**53.45** ±2.57	52.20 ±2.03	Fixed 1.0
ProofWriter / Llama	**49.66** ±1.72	49.12 ±2.63	45.22 ±3.72	Fixed 0.5
FOLIO / Qwen	48.79 ±5.29	**52.96** ±5.89	51.04 ±3.89	Fixed 1.0
FOLIO / Llama	43.33 ±3.13	44.76 ±5.23	**45.48** ±2.95	Fixed 2.0

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD OS, percent. Select by mean OS, then mean CP, then smaller weight. Test values never enter this selection. One selected weight per panel is retained in every paired Test analysis.

F.7 Feedback Scenarios and Overall Success

The four feedback scenarios share each problem and initial answer. CT and IT contain a correct or incorrect target proposal, respectively; CT+IU adds an incorrect unrelated proposal to a correct target proposal; CT+CU adds a correct unrelated proposal instead. Each scenario is evaluated using the corresponding model responses and the same scoring contract.

F.8 Paired Outcomes and Correction Need

Table 30 summarizes paired net differences. The accompanying counts partition each seed's cases into PARO-only, comparator-only, both-successful, and neither-successful outcomes. The difference between the first two counts, divided by panel size, equals the OS difference in each seed and hence in their means. This accounting uses paired states rather than independent observations for significance testing. Figure 9 displays both directions of change. Keep All retains a negative net difference for PARO, while the fixed-weight comparisons contain losses as well as gains.

Correction-eligible, resistance-eligible, and other cases partition OS in Table 19. Membership uses the initial answer and feedback, not method outcomes. Other combines initially correct targets under correct target proposals with initially incorrect targets under incorrect target proposals. In the

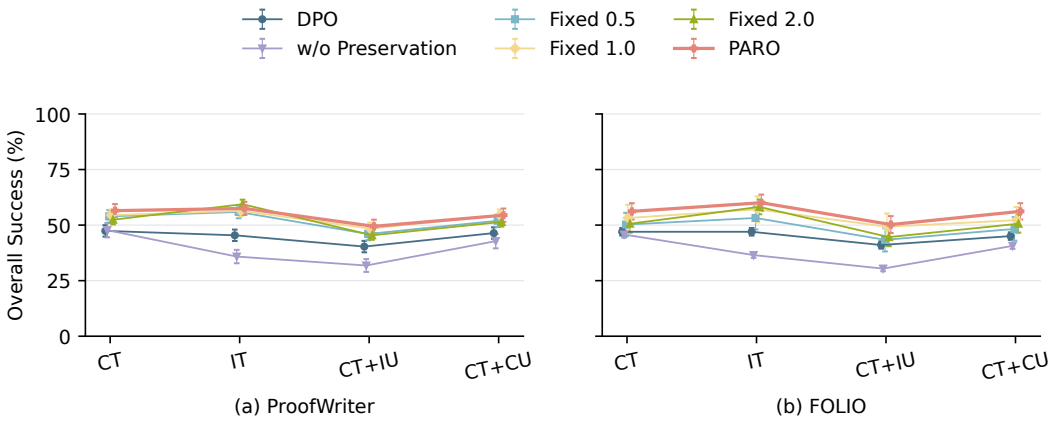

Figure 8: Mean Overall Success by feedback scenario with one-SD error bars. All fixed weights remain visible; the shared axes and methods match the feedback-scenario tables. Small horizontal offsets separate overlapping error bars.

Table 28: ProofWriter: Overall Success by feedback scenario.

Method	CT	IT	CT+IU	CT+CU
DPO	47.39 ±2.57	45.41 ±2.57	40.33 ±2.57	46.40 ±2.57
w/o Preservation	47.72 ±3.24	35.84 ±3.01	31.82 ±2.88	42.77 ±3.24
Fixed 0.5	53.93 ±2.84	55.91 ±2.84	46.01 ±2.84	51.95 ±2.84
Fixed 1.0	54.46 ±2.56	56.44 ±2.56	48.45 ±2.63	**54.46** ±2.56
Fixed 2.0	52.41 ±2.08	**59.41** ±2.03	45.41 ±2.02	51.42 ±2.08
PARO	**56.50** ±2.96	57.49 ±2.96	**49.44** ±3.01	**54.46** ±2.99

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD, percent. CT/IT: correct/incorrect target proposal; IU/CU: incorrect/correct unrelated proposal. Feedback scenarios share initial answers.

Table 29: FOLIO: Overall Success by feedback scenario.

Method	CT	IT	CT+IU	CT+CU
DPO	46.99 ±1.70	46.99 ±1.70	41.04 ±1.64	45.05 ±1.70
w/o Preservation	45.76 ±1.40	36.50 ±1.24	30.42 ±1.19	40.65 ±1.35
Fixed 0.5	50.23 ±5.29	53.20 ±5.19	43.37 ±5.23	48.28 ±5.29
Fixed 1.0	53.20 ±5.90	57.09 ±5.85	49.26 ±5.95	52.23 ±5.90
Fixed 2.0	50.61 ±3.99	58.19 ±3.31	44.60 ±4.08	50.61 ±3.99
PARO	**56.18** ±3.67	**60.06** ±3.67	**50.23** ±3.80	**56.18** ±3.67

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD, percent. CT/IT: correct/incorrect target proposal; IU/CU: incorrect/correct unrelated proposal. Feedback scenarios share initial answers.

Table 30: Paired Overall Success changes: PARO versus each comparator.

Panel / comparator	Gain	Loss	Net
ProofWriter / DPO	14.06 ±2.43	4.47 ±1.14	9.59 ±3.56
ProofWriter / Fixed 1.0	8.37 ±0.93	7.34 ±1.07	1.02 ±1.89
ProofWriter / Keep All	18.68 ±1.02	31.20 ±2.22	−12.52 ±2.98
FOLIO / DPO	15.29 ±1.51	4.64 ±0.78	10.65 ±2.25
FOLIO / Fixed 1.0	10.16 ±2.65	7.44 ±2.23	2.72 ±4.81
FOLIO / Keep All	16.33 ±0.87	31.21 ±2.90	−14.89 ±3.70

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD, percentage points. Gain: PARO-only successes; Loss: comparator-only successes. Net = Gain − Loss in each paired seed. The SD of Net is computed from paired differences, not by subtracting SDs. Fixed weights are selected by mean Development OS, then CP, then smaller weight.

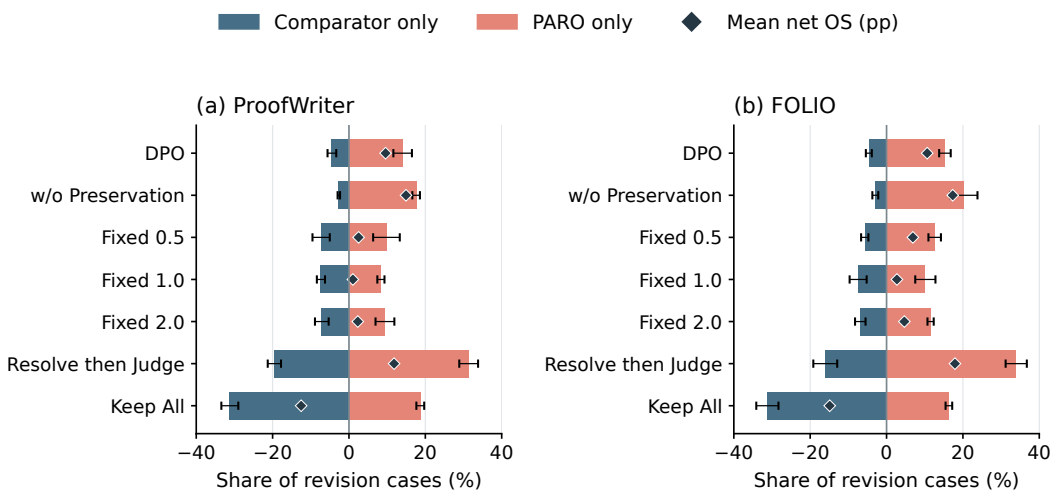

Figure 9: Paired Overall Success changes. Right bars are mean PARO-only successes; left bars are mean comparator-only successes shown with a negative sign. Error bars show one SD, and diamonds show the mean net change in percentage points (pp). All revision cases remain in each denominator. These transitions are distinct from item-level preservation costs.

latter cell, rejecting an incorrect binary target proposal consistently requires the correct final target value, even though these cases are outside Target Correction's eligible set. Other therefore cannot be interpreted as no repair needed.

Tables 31 and 32 expose all eight groups defined by initial target correctness and feedback without changing any scores. Correct targets may still coexist with pre-existing dependent or unrelated errors; retention success is scoped rather than full-answer correctness. Means and SDs for each row retain variation within each feedback scenario, while the complete per-seed scores retain any reversals between methods. These are descriptive breakdowns of the same evaluated model responses, not independent evidence of learned specialization.

Table 31: ProofWriter: Overall Success by initial target and feedback.

Initial / feedback	Cases	Keep All	DPO	Fixed 1.0	PARO
Incorrect / CT	100	0.00 ±0.00	47.40 ±4.04	<u>56.40</u> ±5.13	**57.40** ±5.32
Incorrect / IT	100	0.00 ±0.00	47.40 ±7.86	<u>60.80</u> ±5.40	**63.20** ±5.59
Incorrect / CT+IU	100	0.00 ±0.00	38.40 ±1.82	<u>47.20</u> ±2.59	**49.00** ±3.08
Incorrect / CT+CU	100	0.00 ±0.00	46.40 ±3.78	<u>56.40</u> ±2.88	**56.80** ±3.77
Correct / CT	203	**100.00** ±0.00	47.39 ±3.06	53.50 ±3.28	<u>56.06</u> ±3.20
Correct / IT	203	**100.00** ±0.00	44.43 ±1.45	54.29 ±4.63	<u>54.68</u> ±3.54
Correct / CT+IU	203	**100.00** ±0.00	41.28 ±4.55	49.06 ±3.22	<u>49.66</u> ±5.02
Correct / CT+CU	203	**100.00** ±0.00	46.40 ±3.64	<u>53.50</u> ±3.60	53.30 ±2.84

Fixed λ: w/o Adaptation with penalty weight λ. Cases is the per-seed denominator; performance cells are mean ± SD percentages. CT/IT: correct/incorrect target proposal; IU/CU: incorrect/correct unrelated proposal. The eight rows partition each Test panel. Bold/underline rank means within each row.

F.9 Integer Counts and Dependency Coverage

All displayed mean percentages follow the exact aggregate integers in the integer-count CSV exports, rounded only for display. These counts sum five repeated panels; per-seed denominators remain those in Table 12. SDs use the five separate rates, not a binomial approximation from their sum. Overall Success counts come from conjunctions on complete states, not products of marginal scores. A component with no eligible opportunities is N/A, not zero. Malformed model outputs remain failures in every applicable denominator under the specified parsing rules. These aggregate scores do not separately report parsing success rates.

Table 32: FOLIO: Overall Success by initial target and feedback.

Initial / feedback	Cases	Keep All	DPO	Fixed 1.0	PARO
Incorrect / CT	91	0.00 ±0.00	45.27 ±2.95	52.09 ±6.67	**54.51** ±2.41
Incorrect / IT	91	0.00 ±0.00	47.03 ±4.49	57.36 ±6.75	**59.78** ±6.76
Incorrect / CT+IU	91	0.00 ±0.00	39.78 ±3.76	49.67 ±8.56	**50.11** ±3.85
Incorrect / CT+CU	91	0.00 ±0.00	45.93 ±4.35	52.53 ±7.79	**57.36** ±5.24
Correct / CT	218	**100.00** ±0.00	47.71 ±2.13	53.67 ±5.63	56.88 ±5.34
Correct / IT	218	**100.00** ±0.00	46.97 ±1.85	56.97 ±5.96	60.18 ±3.73
Correct / CT+IU	218	**100.00** ±0.00	41.56 ±1.40	49.08 ±6.61	50.28 ±4.37
Correct / CT+CU	218	**100.00** ±0.00	44.68 ±2.84	52.11 ±5.43	55.69 ±4.08

Fixed λ: w/o Adaptation with penalty weight λ. Cases is the per-seed denominator; performance cells are mean ± SD percentages. CT/IT: correct/incorrect target proposal; IU/CU: incorrect/correct unrelated proposal. The eight rows partition each Test panel. Bold/underline rank means within each row.

Table 33 applies additional dependency coverage to the same saved model responses. Whenever an initial target is wrong, initially incorrect dependents must finish correct, including under an incorrect target proposal. The additional check changes neither responses nor the primary denominator. Overall Success is unchanged for the displayed trained methods and Keep All, but decreases for Resolve then Judge in both domains. These results describe the same saved responses under expanded scoring, rather than robustness to arbitrary dependency structures.

Table 33: Expanded dependency coverage on the same states.

Panel / method	Primary OS	Extended OS	Lost cases
ProofWriter / DPO	44.88 ±2.57	44.88 ±2.57	0.0
ProofWriter / TD-DPO	44.72 ±3.00	44.72 ±3.00	0.0
ProofWriter / Fixed 0.5	51.95 ±2.84	51.95 ±2.84	0.0
ProofWriter / Fixed 1.0	53.45 ±2.58	53.45 ±2.58	0.0
ProofWriter / Fixed 2.0	52.16 ±2.05	52.16 ±2.05	0.0
ProofWriter / PARO	54.47 ±2.98	54.47 ±2.98	0.0
ProofWriter / Keep All	67.00 ±0.00	67.00 ±0.00	0.0
ProofWriter / Resolve then Judge	42.66 ±2.95	41.25 ±3.03	17.0
FOLIO / DPO	45.02 ±1.69	45.02 ±1.69	0.0
FOLIO / TD-DPO	42.69 ±3.95	42.69 ±3.95	0.0
FOLIO / Fixed 0.5	48.77 ±5.25	48.77 ±5.25	0.0
FOLIO / Fixed 1.0	52.94 ±5.90	52.94 ±5.90	0.0
FOLIO / Fixed 2.0	51.00 ±3.84	51.00 ±3.84	0.0
FOLIO / PARO	55.66 ±3.70	55.66 ±3.70	0.0
FOLIO / Keep All	70.55 ±0.00	70.55 ±0.00	0.0
FOLIO / Resolve then Judge	37.75 ±8.48	36.57 ±8.52	14.6

Fixed λ: w/o Adaptation with penalty weight λ. OS is mean ± SD, percent; Lost cases is the mean count per seed. Extended coverage also corrects initially incorrect dependents when the target is initially incorrect under incorrect feedback. No denominator or answer is changed.

F.10 Overall Success Across Training Runs

The CSV supplement retains all five Test values, including deterministic controls in the Qwen-family panels. Their means and sample SDs reproduce every Overall Success summary. The PDF provides aggregate counts and summary rates; recomputing every SD requires those separate per-seed values. Variability is conditional on the fixed initial answers; it does not include changes in the admitted population.

G Additional Backbone Comparisons

Figure 10 compares separately trained and evaluated dataset/backbone panels. The panels share candidate families, method definitions, eligibility criteria, and fixed-weight controls, but use backbone-

specific initial-answer admission. Table 12 and the result tables give their denominators. Within-panel method comparisons remain distinct from comparisons across different admitted populations.

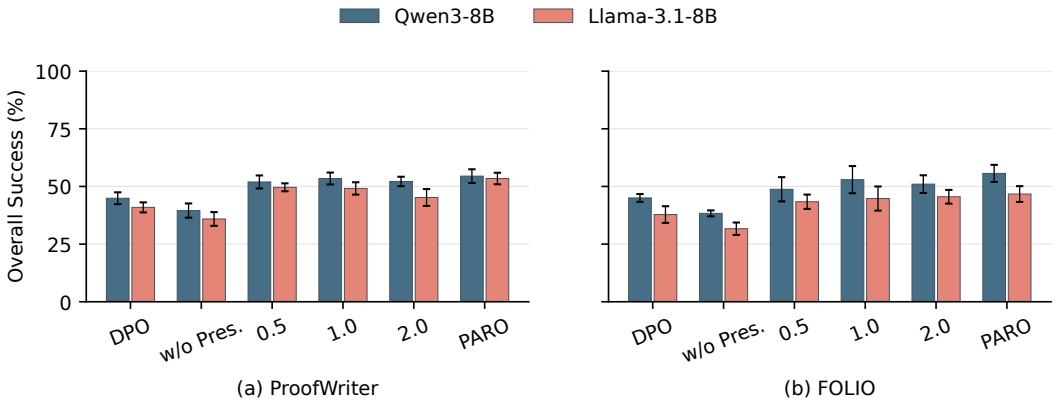

Figure 10: Mean Overall Success across backbone regimes and datasets, with one-SD error bars. Each backbone uses its own admitted initial answers; paired method comparisons are within, not across, those populations.

Table 34: ProofWriter: Llama-family panel.

Method	TC ↑	PR ↑	DP ↑	CP ↑	OS ↑
Plan-SFT	74.90 ±4.00	80.66 ±2.72	54.00 ±4.35	86.43 ±1.45	33.61 ±1.32
DPO	<u>83.03</u> ±1.94	72.68 ±2.44	60.86 ±2.70	91.71 ±0.92	40.93 ±2.17
w/o Preservation	**89.08** ±2.01	73.22 ±4.78	**73.53** ±3.39	86.90 ±1.72	35.89 ±2.98
Fixed 0.5	75.52 ±1.84	93.55 ±1.79	63.93 ±3.54	93.96 ±0.42	<u>49.62</u> ±1.73
Fixed 1.0	77.03 ±1.96	<u>94.64</u> ±0.81	63.55 ±2.44	93.96 ±0.48	49.14 ±2.68
Fixed 2.0	70.20 ±4.55	**95.96** ±0.49	59.95 ±2.98	**94.58** ±1.06	45.22 ±3.66
PARO	81.51 ±1.78	91.48 ±2.21	<u>68.78</u> ±2.67	<u>94.41</u> ±0.48	**53.44** ±2.50

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

Table 35: FOLIO: Llama-family panel.

Method	TC ↑	PR ↑	DP ↑	CP ↑	OS ↑
Plan-SFT	70.44 ±2.84	79.80 ±3.09	48.07 ±3.51	84.59 ±1.46	29.04 ±1.73
DPO	<u>78.82</u> ±2.43	73.00 ±4.78	55.31 ±3.78	87.94 ±2.71	37.81 ±3.61
w/o Preservation	**83.72** ±4.88	73.00 ±3.81	**68.12** ±6.65	84.12 ±2.26	31.68 ±2.69
Fixed 0.5	69.56 ±2.72	91.50 ±3.39	55.12 ±6.46	<u>91.24</u> ±1.54	43.37 ±3.10
Fixed 1.0	65.31 ±4.38	<u>91.80</u> ±1.89	55.85 ±5.28	90.29 ±1.57	44.76 ±5.23
Fixed 2.0	66.67 ±3.24	**96.50** ±1.12	55.02 ±2.40	**92.72** ±0.65	<u>45.53</u> ±2.97
PARO	72.80 ±2.73	89.40 ±2.10	<u>59.66</u> ±3.82	90.70 ±1.24	**46.71** ±3.43

Fixed λ: w/o Adaptation with penalty weight λ. Mean ± SD across five fixed seeds; percentages. Bold/underline mark the best/second distinct displayed means.

H Additional Mechanism Analyses

This section examines feedback cues, judgment postprocessing, reward equivalence, and risk response. The balanced rule check uses 200 synthetic problems from 100 paired families per split. Postprocessing reuses the Qwen-family panels of Table 12. Budget and matched-risk analyses instead use 400 synthetic problems from 200 paired families per split under four feedback scenarios

and seeds 11, 23, 37, 53, and 71. These populations remain separate from the main experimental comparison and from one another.

For the risk simulations, the probability of repairing an incorrect item is $\sigma(a - b\lambda + z)$, and the base damage probability is $h_0(\lambda) = 0.005 + h\exp(-k\lambda)$. Test damage uses $\sigma(\text{logit}\, h_0 + 0.35z)$. The difficulty term is $z = z_f + z_i + z_r$, with independent normal SDs 0.5, 0.25, and 0.15 for family, problem, and realization effects; the family effect is shared by its two variants. These effects have zero mean. Proposal-judgment correctness has probability $\sigma(\text{logit}(0.95) + z)$. Reference labels are independent binary draws with probability 0.5. The initial target is incorrect with probability 0.45, the first dependent is incorrect, and the remaining items are correct. Each problem repeats under four feedback scenarios. All fixed weights and the Adaptive proxy share initial states and random uniforms within each environment and seed.

Table 36: Simulation parameters for budget and matched-risk checks.

Environment	a	b	h	k	Budget (%)
Low risk	1.55	0.22	0.09	1.10	7.5
Moderate risk	1.55	0.25	0.25	0.90	6.5
High risk	1.55	0.32	0.42	0.80	5.0
Tight budget	1.55	0.55	0.32	0.70	2.5

Parameters define the response surface, not fitted model behavior. The matched-risk check uses the Moderate risk surface with budgets of 3, 5, 7.5, and 10 percent.

Calibration starts at $\lambda = 1$ and performs 120 updates. Each update draws 128 Bernoulli trials with probability $h_0(\lambda)$ and applies $\lambda' = \text{clip}_{[0,3]}\{\lambda + 3(\widehat{h} - \epsilon)\}$. Its terminal weight is frozen for Test. Calibration omits the Test difficulty term z. The budget comparison selects among $\{0, 0.25, \ldots, 3\}$ on Development, considering only weights within budget and breaking ties by Overall Success, Content Preservation, then smaller weight. The matched-risk check instead uses $\{0, 0.1, \ldots, 3\}$ and the Test oracle described below. Neither calculation executes policy-gradient training. Additional order, transfer-profile, and tree-stress simulations remain in the supporting analysis, separate from the results reported here.

H.1 Balanced Feedback Design

The balanced simulation varies proposal truth, initial correctness, and order together. Each split contains 100 separate abstract families with two variants each. Crossing the four target/supplemental truth combinations (TT, TF, FT, FF) with initially correct or incorrect target and supplemental items gives 3,200 dual-proposal cases. Development and Test randomize proposal order independently; the paired rule check then reverses each list. Original-order model scores are not reused for these inputs.

The supplemental item is a separate editable item. The seven-item design includes one target, two dependents, the supplemental item, and three initially correct unrelated items. Answer-scope Success must check the target, applicable dependents, supplemental final value, and preservation of the remaining three items. Correct proposal judgments and consistency then determine Overall Success. These expanded criteria are not applied retrospectively to the original six-item result tables.

Table 37 compares rules under the balanced design. Position × Initial-agreement combines visible proposal position with agreement with the addressed initial value. Its four feature cells admit sixteen binary mappings. Development selects the largest Proposal Judgment count, with fixed lexicographic tie-breaking, and Test retains that mapping. Predictions use no reference labels. Balancing removes the original deterministic agreement advantage; this rule analysis neither regenerates model answers nor establishes PARO's performance after the input change.

Table 37: Simulation: balanced dual-proposal rule check.

Rule	Original	Reversed
Position × Initial-agreement	3164/6400 (49.44%)	3236/6400 (50.56%)
Initial-agreement	3200/6400 (50.00%)	3200/6400 (50.00%)

Proposal Judgment only. No revised answers or model robustness scores are assigned. Development selects one of 16 mappings; Test never selects it.

H.2 JUDGMENT POSTPROCESSING

Table 38 compares original and postprocessed Overall Success for every method in both Qwen-family panels. The postprocessing analysis fixes each evaluated model's final answer and assigns ACCEPT exactly when it agrees with the proposed value, without using reference labels for prediction. Answers, Final Accuracy counts, and scoring rules remain unchanged. Computing paired changes within each training run retains the distinct gains of each method. This is postprocessing of saved outputs, without new training or answer generation. Postprocessing applies to parsed responses. Original parse failures remain failures in every applicable denominator, and missing answers are not inferred.

Table 38: Judgment postprocessing of saved outputs, with final answers unchanged.

Domain / method	Original OS	Processed OS	Difference
FOLIO / Accept All	35.28 ± 0.00	35.28 ± 0.00	0.00 ± 0.00
FOLIO / DPO	45.02 ± 1.69	49.14 ± 1.81	4.13 ± 0.82
FOLIO / Fixed 0.5	48.77 ± 5.25	54.09 ± 5.40	5.32 ± 0.97
FOLIO / Fixed 1.0	52.94 ± 5.90	57.20 ± 5.12	4.26 ± 1.49
FOLIO / Fixed 2.0	51.00 ± 3.84	54.69 ± 4.10	3.69 ± 1.26
FOLIO / IPO	44.13 ± 4.69	49.14 ± 5.18	5.02 ± 0.64
FOLIO / Keep All	70.55 ± 0.00	70.55 ± 0.00	0.00 ± 0.00
FOLIO / PARO	55.66 ± 3.70	60.15 ± 3.97	4.48 ± 0.64
FOLIO / Plan-SFT	35.89 ± 4.58	41.49 ± 4.62	5.60 ± 0.73
FOLIO / Resolve then Judge	37.75 ± 8.48	37.75 ± 8.48	0.00 ± 0.00
FOLIO / w/o Preservation	38.33 ± 1.29	51.18 ± 1.67	12.85 ± 1.38
FOLIO / Separate Penalties	48.77 ± 2.89	54.16 ± 2.92	5.39 ± 1.17
FOLIO / SimPO	43.04 ± 5.38	47.91 ± 5.80	4.87 ± 0.91
FOLIO / TD-DPO	42.69 ± 3.95	48.66 ± 4.46	5.97 ± 0.76
FOLIO / TDPO1	43.93 ± 2.41	49.69 ± 1.90	5.76 ± 0.53
ProofWriter / Accept All	33.50 ± 0.00	33.50 ± 0.00	0.00 ± 0.00
ProofWriter / DPO	44.88 ± 2.57	52.87 ± 3.39	7.99 ± 1.98
ProofWriter / Fixed 0.5	51.95 ± 2.84	58.80 ± 2.71	6.85 ± 1.42
ProofWriter / Fixed 1.0	53.45 ± 2.58	61.06 ± 3.30	7.61 ± 1.17
ProofWriter / Fixed 2.0	52.16 ± 2.05	58.63 ± 2.10	6.47 ± 0.46
ProofWriter / IPO	46.49 ± 2.73	54.26 ± 1.89	7.77 ± 1.23
ProofWriter / Keep All	67.00 ± 0.00	67.00 ± 0.00	0.00 ± 0.00
ProofWriter / PARO	54.47 ± 2.98	63.10 ± 2.24	8.63 ± 0.85
ProofWriter / Plan-SFT	40.20 ± 3.76	47.82 ± 2.66	7.62 ± 1.53
ProofWriter / Resolve then Judge	42.66 ± 2.95	42.66 ± 2.95	0.00 ± 0.00
ProofWriter / w/o Preservation	39.54 ± 3.09	55.28 ± 3.31	15.74 ± 0.65
ProofWriter / Separate Penalties	53.18 ± 3.03	61.40 ± 3.99	8.22 ± 1.16
ProofWriter / SimPO	45.35 ± 2.58	53.86 ± 1.93	8.51 ± 1.47
ProofWriter / TD-DPO	44.72 ± 3.00	53.09 ± 3.38	8.37 ± 0.61
ProofWriter / TDPO1	46.22 ± 2.79	55.91 ± 2.41	9.69 ± 1.80

Fixed λ: w/o Adaptation with penalty weight λ. Percentages, mean $\pm$ sample SD. All final answers and Final Accuracy counts remain identical. Original parse failures remain failures; postprocessing does not infer missing fields.

H.3 Dense Correctness and Damage

For one initial answer, let A_0, A_f count initially and finally correct items, C repaired items, and H newly damaged items. Then

$$A_f = A_0 + C - H, \qquad C - \lambda H = \lambda A_f + (1 - \lambda)C - \lambda A_0.$$

At $\lambda = 1$, $C - H$ and dense final correctness rank the same candidates. This algebra does not compare trained policies. Dense-reward RL therefore requires a matched training control: retain initialization, success and judgment rewards, reference regularization, samples and optimizer; replace the correction/damage component with scaled itemwise final correctness. Development alone selects its coefficient. The reported results contain no matched Dense-reward RL comparison, so this analysis establishes candidate-ordering equivalence at the stated weight rather than a trained-policy advantage over dense correctness reward.

H.4 Budget and Realized Risk

Table 39: Simulation: a shared nominal budget with separate Development selection.

Environment / proxy	Budget	OS	TC	Damage	Exceed
Low risk / Fixed	7.5	70.78 ± 2.06	71.00 ± 4.53	0.98 ± 0.22	0/5
Low risk / Adaptive	7.5	54.36 ± 1.93	79.37 ± 2.86	7.49 ± 0.33	2/5
Moderate risk / Fixed	6.5	66.96 ± 1.57	71.97 ± 2.04	2.37 ± 0.20	0/5
Moderate risk / Adaptive	6.5	56.34 ± 2.62	77.82 ± 2.05	6.83 ± 0.81	3/5
High risk / Fixed	5.0	55.50 ± 3.06	62.38 ± 3.23	4.45 ± 0.34	0/5
High risk / Adaptive	5.0	53.56 ± 2.23	64.31 ± 3.71	5.32 ± 0.55	4/5
Tight budget / Fixed	2.5	–	–	–	N/A
Tight budget / Adaptive	2.5	50.20 ± 2.53	44.53 ± 3.78	4.36 ± 0.50	5/5

Mean $\pm$ sample SD across five Monte Carlo realizations, in percentages. Correction is Target Correction. Fixed selection maximizes Development OS within the 13-weight grid subject to Development damage at or below the budget. Exceed counts Test realizations above that budget. No feasible Development weight yields –, not zero performance. Adaptive uses the existing calibration rule. Equal budgets do not imply matched realized risk or calibration cost.

The budget comparison reports OS, TC, damage to all initially correct items, and budget exceedances together. Equal nominal budgets can yield different Test damage: calibration omits latent Test difficulty, allowing the Adaptive proxy to exceed its budget. The analysis separates a response to observed risk from optimal revision quality.

H.5 Fixed Penalties at Matched Risk

Table 40: Simulation: Adaptive minus a dense-grid mixture at matched expected damage.

Budget	Adaptive damage	Δ TC	Δ OS	Matched
3	3.25 ± 0.35	-0.27 ± 0.12	-0.34 ± 0.21	5/5
5	5.53 ± 0.44	-0.10 ± 0.14	-1.32 ± 0.43	5/5
7.5	7.66 ± 1.06	-0.22 ± 0.33	-2.45 ± 0.70	5/5
10	11.17 ± 1.41	-0.39 ± 0.26	-4.19 ± 0.79	5/5

Five Monte Carlo realizations; mean $\pm$ sample SD in percentage points. The comparator maximizes each metric separately over mixtures of two fixed-grid policies at the Adaptive proxy's realized Test damage. It is a Test oracle, not a deployable selector. Mixture performance is an expectation, not new observations. Unmatched realizations are excluded and counted explicitly; no extrapolation is used.

The comparison sweeps 31 fixed weights from zero to three in steps of 0.1. Hindsight mixtures match the Adaptive proxy's realized Test risk while maximizing correction or Overall Success separately. These mixtures are diagnostic oracles rather than Development-selected comparators. The Adaptive proxy does not exceed their matched-risk frontier. The simulator thus illustrates budget response, not a trained-model advantage or a new attainable policy frontier. Monte Carlo variability remains distinct from uncertainty grouped by source and from matched training comparisons.

I Detailed Related Work

This section expands Section 2 by research topic. Table 1 compares evaluation targets rather than model quality.

I.1 Feedback and Self-correction

Self-Refine alternates feedback and revision (Madaan et al., 2023); Reflexion guides later attempts through verbal reflections (Shinn et al., 2023). Learned correctors and reinforcement learning train revision (Kumar et al., 2025; Welleck et al., 2023). CRITIC uses tool feedback (Gou et al., 2024), while Self-Debugging uses code explanations and execution results (Chen et al., 2024). Correction without external feedback can fail (Huang et al., 2024); surveys distinguish this setting from reliable guidance or additional training (Kamoi et al., 2024). Error location and correction also pose distinct difficulties (Tyen et al., 2024). ClaimSift fixes evidence and initial answers while varying proposal correctness within and across messages.

I.2 Unreliable Feedback and Evidence Conflicts

Unreliable feedback raises a different question from correction under reliable guidance. Models can favor agreement with users over correctness (Sharma et al., 2024), and contextual evidence can conflict with parametric knowledge (Xie et al., 2024). Sycophancy evaluations distinguish resistance to user pressure from correction supported by evidence (Ma et al., 2026). ClaimSift evaluates these decisions together with judgment–revision consistency, dependency propagation, and preservation. The supplied context provides the evidence; revision changes the answer rather than stored model knowledge.

I.3 Fine-grained Evaluation and Revision

Evaluation can separate answer correctness from the decisions that produce it. TruthfulQA measures truthful answers (Lin et al., 2022), while FActScore evaluates atomic facts (Min et al., 2023). EntailmentBank supplies entailment trees (Dalvi et al., 2021), EX-FEVER provides multi-hop verification evidence (Ma et al., 2024), and process supervision evaluates intermediate reasoning (Lightman et al., 2024). Revision frameworks also distinguish requirements: RARR preserves original content while revising unsupported claims (Gao et al., 2023), RefineBench uses checklists (Lee et al., 2026), and ReFeed considers multiple summarization dimensions (Yun et al., 2025). ClaimSift defines eligible correction and preservation opportunities and checks their joint satisfaction with proposal judgments.

I.4 Repair Without Regressions

Repair evaluation distinguishes fixing an error from preserving existing behavior. SWE-bench checks newly passing tests together with previously passing tests (Jimenez et al., 2024). ClaimSift applies the corresponding distinction to answer revision and adds correct and incorrect proposals, explicit judgments, dependency requirements, and judgment–revision consistency. Answer-item checks and software tests remain different evaluation objects without a shared performance scale.

I.5 Reasoning Sources and Preference Optimization

ProofWriter supplies rule-based derivations (Tafjord et al., 2021); FOLIO provides natural-language contexts with first-order annotations (Han et al., 2024). FOLIO with dependency annotations additionally requires interventions. MQuAKE evaluates multi-hop knowledge edits (Zhong et al., 2023), whereas ClaimSift revises answers against fixed evidence. Preference learning motivates response comparisons (Christiano et al., 2017; Ouyang et al., 2022). DPO, IPO, TDPO, and SimPO differ in their objectives (Gheshlaghi Azar et al., 2024; Meng et al., 2024; Rafailov et al., 2023; Zeng et al., 2024); SEPO estimates token rewards (Yang et al., 2025), and TD-DPO emphasizes differing tokens (Lai et al., 2026). PARO combines rewards for complete revisions, a leave-one-out baseline (Ahmadian et al., 2024), and adaptive penalties (Achiam et al., 2017; Moskovitz et al., 2024). Preservation cost defines new damage separately from its weighting.