أكاديمية كاوست
KAUST ACADEMY

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

When Should an LLM Listen to Another LLM? Predicting Beneficial Critique Acceptance in Multi-LLM Collaboration

Lama Alshammari, Farah Alshammari, Abdullah Alasiri,
Ghassan Alqahtani, Dr. Eman Alnabati

lamaalham7i@gmail.com, farah.alshammari@icloud.com, abdullah.alasiri@kaust.edu.sa,
ghassan.alqahtani@kaust.edu.sa, emantn@mit.edu

Abstract

Large language models (LLMs) can benefit from external critique, but such feedback may also cause correct answers to become incorrect. This work investigates whether the usefulness of critique can be predicted before revision, enabling selective rather than unconditional critique acceptance.

We evaluate a Solver–Critic protocol on GSM8K using Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct in both model directions. The Critic first solves each problem independently, then evaluates only the Solver's final answer. A lightweight logistic regression classifier uses pre-revision features derived from answer disagreement, critique properties, and Solver–Critic direction to predict whether revision will be beneficial.

Across 2,000 development interactions, revision benefit was strongly asymmetric: revisions triggered by Qwen critiques of Llama were beneficial 29.1% of the time versus 11.2% in the reverse direction ($p = 0.00013$), and this asymmetry appears driven primarily by differences in correction effectiveness rather than false-alarm frequency. Solver–critic direction carried stronger and more consistent predictive signal than the critique-content features examined. On a frozen held-out test set, a lightweight gating policy was directionally consistent with these findings, though its improvement over naive baselines was not statistically meaningful at that sample size. These results suggest that in multi-LLM collaboration, who critiques whom matters more than what the critique says.

Date: August 24, 2026
Code: https://github.com/ghassanbinhadi/ProjectTPD3

Contents

1 Introduction **2**

2 Related Work **2**

2.1 Test-Time Critique and Iterative Refinement . . . 2

2.2 Process Supervision and Step-Level Verifiers . . . 2

2.3 Critique Reliability and Helpfulness Prediction . . . 3

2.4 Positioning of Our Work . . . 3

3 Methodology **3**

3.1 Method Details . . . 3

3.2 Evaluation Procedure . . . 5

4 Results **6**

4.1 Pipeline Characterization . . . 6

4.2 Four-Transition Characterization . . . 7

4.3 Descriptive Rates . . . 7

4.4 Statistical Comparisons . . . 7

4.5 Predictor Performance . . . 8

4.6 Three-Way Ablation . . . 9

4.7 Frozen Test-Set Policy Comparison . . . 9

4.8 Demo Cases . . . 10

5 Discussion **11**

5.1 What the classifier is actually learning . . . 11

5.2 What the policy comparison tells us . . . 11

5.3 Sensitivity to dataset size . . . 11

5.4 Critic reliability across directions . . . 11

6 Limitations and Future Work **12**

7 Conclusion **12**

References **13**

1 Introduction

Large Language Models (LLMs) can perform complex reasoning tasks, but they may still produce incorrect answers. One approach to improving reliability is to use a second LLM as a Critic. However, critique is not always helpful: it may correct an error, but it may also cause a correct answer to become incorrect.

This work investigates the question: **when should an LLM listen to another LLM?** Specifically, given a Solver and a Critic, the goal is to predict before revision whether using the critique will improve the Solver's correctness.

The study uses GSM8K mathematical reasoning problems with automatically verifiable numeric answers. Revision is considered only when the Critic disagrees with the Solver, so the study focuses on selective revision under disagreement.

The proposed system uses two LLMs in alternating Solver–Critic roles. The Critic first solves the problem independently, then evaluates the Solver's final answer. When disagreement occurs, the Solver may revise its answer. Interactions are labeled as HELPED, HURT, STABLE-CORRECT, or STABLE-WRONG, and pre-revision features are used to train a lightweight classifier that predicts when revision is likely to be beneficial.

2 Related Work

Recent efforts to enhance the mathematical reasoning capabilities of Large Language Models (LLMs) have shifted from pure generation toward post-generation refinement, process-level verification, and dynamic feedback selection. This section reviews prior work across these dimensions and highlights the gap addressed by our study.

2.1 Test-Time Critique and Iterative Refinement

Early formulations of automated refinement demonstrated that an LLM could generate feedback on its own initial outputs to iteratively correct errors without modifying model weights [1]. This paradigm has since evolved into broader frameworks of automated correction spanning self-feedback and external-feedback mechanisms evaluated across generation and post-hoc stages [2].

However, relying solely on self-refinement often suffers from confirmation bias, prompting recent architectures to decouple the generation and evaluation steps into distinct *solver–critic* setups [3, 4]. In mathematical problem-solving, step-wise critic frameworks such as Critic-CoT [5] provide structured critique chains to filter or revise invalid reasoning steps on standard benchmarks like GSM8K and MATH. While these systems demonstrate that allocating additional inference-time compute to critique improves task accuracy, they fundamentally assume that critique is uniformly beneficial, treating the revision step as an unconditionally safe operation.

2.2 Process Supervision and Step-Level Verifiers

To improve the precision of feedback in complex mathematical reasoning, recent literature emphasizes process-level verification over outcome-only supervision [6–8]. Outcome supervision evaluates only the terminal answer, often failing to detect intermediate reasoning flaws in false-positive rollouts. In contrast, Process Reward Models (PRMs) and generative step-level verifiers—such as Math-Shepherd [7] and GenRM [9]—provide fine-grained step rewards to guide search and re-ranking algorithms. Similarly, hybrid methods such as GLoRe [10] identify both *when* and *where* to trigger localized refinements.

Despite their empirical gains, these approaches typically rely on extensive step-level synthetic labels, specialized reward training, or computationally demanding Best-of-N inference passes. Table 1 summarizes representative critique and verification approaches alongside their limitations.

Table 1 Comparison of representative critique, verification, and refinement frameworks in mathematical reasoning.

Approach	Core Role	Reported Math Benefit	Key Limitation
Self-Refine [1]	Self-feedback revision	Broad improvements in multi-task reasoning	No guarantee of reliable self-correction; degrades valid steps
CRITIC [11]	Tool-grounded critique	Consistent gains on QA, math, and code	Performance strictly bounded by external tool reliability
Math-Shepherd [7]	Step-level PRM supervision	GSM8K: 77.9% → 84.1% (89.1% w/ verification)	High computational cost for dense process-level supervision
GLoRe [10]	Localized step refinement	LLaMA-2 13B on GSM8K: 53% → 65%	Requires synthetic step labeling and heavy search reranking
GenRM [9]	Generative verifier reranking	GSM8K: 73.0% → 93.4% via Best-of-N	Dependent on complex verifier prompting and sampling budget
Ours	**Pre-revision utility prediction**	**Selective refinement on GSM8K**	**Lightweight feature-based decision before revision**

2.3 Critique Reliability and Helpfulness Prediction

A critical yet frequently overlooked phenomenon in multi-agent critique is that critique generation does not guarantee error correction [12]. Comprehensive empirical evaluations indicate that ungrounded self-correction frequently hallucinates errors in correct solutions or fails to rectify genuine mistakes, with model performance degrading unless external verifiers or superior critique models are employed [12, 13]. Furthermore, recent investigations into multi-agent review dynamics reveal that critique can be actively harmful: imperfect critics often convince a correct solver to alter its valid derivation into an erroneous one [4, 14].

While prior studies have identified factors correlating with critique quality—such as problem difficulty, error localization depth, and solver-critic capability disparity [3, 10, 14]—they primarily treat these insights qualitatively or incorporate them into heavy retraining pipelines.

2.4 Positioning of Our Work

Our work bridges the gap between critique-generation systems and heavy process verifiers. Rather than training larger reward models or assuming that the solver should blindly trust the critic, we investigate a more selective and computationally efficient question: *Can we predict, prior to triggering revision, whether a proposed critique will help or harm the reasoning trajectory?*

By extracting lightweight, interaction-level features (such as numerical disagreement indicators, answer divergence, and step characteristics) from solver-critic exchanges on GSM8K, our approach enables an informed gating mechanism. This allows the system to selectively accept revisions only when feedback is genuinely predictive of improvement, thereby mitigating the regression risks inherent in current multi-agent reasoning loops.

3 Methodology

3.1 Method Details

3.1.1 System Overview

The proposed system studies whether feedback from an independent LLM can improve another LLM's answer. For each GSM8K problem, a Solver first generates an answer, while a Critic independently solves the same problem without observing the Solver's response. Only after completing its independent solve does the Critic receive the Solver's final answer for evaluation. If the Critic returns `INCORRECT`, the Solver is allowed to revise its answer; otherwise, the original answer is retained. The protocol is executed in both directions: Qwen as Solver with Llama as Critic, and Llama as Solver with Qwen as Critic. Figure 1 summarizes the interaction protocol.

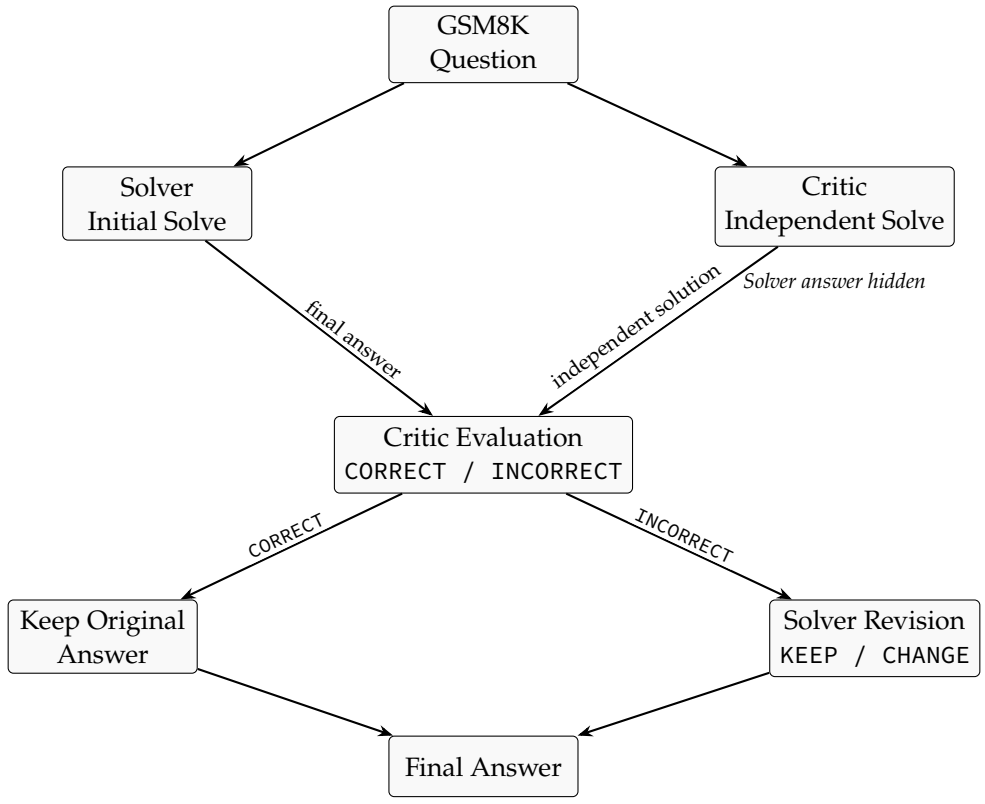


Figure 1 Solver–Critic interaction protocol. The Critic independently solves each problem before observing the Solver's final answer. Revision is triggered only when the Critic returns an `INCORRECT` verdict.

3.1.2 Solver–Critic Interaction and Selective Revision

For each GSM8K problem, the Solver first generates a step-by-step solution followed by a final numeric answer. The Critic then solves the same problem independently using the same problem-solving prompt, without access to the Solver's response.

After completing its independent solve, the Critic enters the evaluation stage. The critique prompt provides the original problem, the Critic's own reasoning and final answer, and only the Solver's final answer; the Solver's reasoning is not exposed. The Critic returns a structured response containing a `CORRECT` or `INCORRECT` verdict, an error description, and a proposed answer.

Revision is triggered only when the Critic returns `INCORRECT`. In this stage, the Solver receives its original answer together with the Critic's error description and proposed answer. It is instructed to reconsider the problem and explicitly decide whether to `KEEP` or `CHANGE` its answer, rather than

changing solely because it was challenged. The revision response contains a brief explanation, the `KEEP/CHANGE` decision, and a final numeric answer. When the Critic returns `CORRECT`, no revision is performed and the original Solver answer is retained.

The pipeline is executed stage-wise, with one model loaded at a time using 4-bit quantization. Solver generations are completed first, followed by the Critic's independent solutions and critique judgments. The Solver is then reloaded only for interactions requiring revision.

The first generation attempt uses greedy decoding (`do_sample=False`) with `max_new_tokens=1024`. If a response fails the required structured parser, exactly one retry is allowed using the same prompt with sampling enabled at temperature `0.7`. Checkpointing and resume logic preserve completed interactions and prevent previously completed `(question_id, direction)` pairs from being regenerated after an interrupted run.

3.1.3 Feature Construction

Features are constructed only from information available before the revision decision so that the gating model does not use post-revision outcomes or ground-truth answers. Each interaction is aligned using its question identifier and Solver–Critic direction, and Critic text is recovered from the stored critique response when required.

Candidate features capture three types of pre-revision information: (i) numerical disagreement between the Solver and Critic answers, (ii) properties of the Critic's feedback, and (iii) the Solver–Critic direction. Numerical disagreement is represented using normalized, log-compressed, and signed log-compressed answer distances. Text-derived features include the length of the error description and the frequency of hedging terms. Direction is represented using one-hot indicators.

The default modeling set therefore contains the three answer-distance features, hedging count, error-description length, and two directional indicators. Candidate features derived from ground-truth labels, revision outputs, post-revision correctness, reasoning generated during revision, or other post-decision information are excluded. Near-constant features on the disagreement population are also removed using the predefined minority-value threshold of 0.05.

3.1.4 Benefit Prediction and Decision Policy

The benefit predictor operates only on valid disagreement interactions for which the Critic returned `INCORRECT` and a revision was generated. This matches the decision of interest: once disagreement occurs, should the Solver's revised answer be adopted?

A logistic regression classifier with balanced class weights is trained to predict the binary `beneficial` label. Validation uses 5-fold `StratifiedGroupKFold` grouped by `question_id`, ensuring that the two directions of the same problem cannot appear across training and validation folds.

Three feature configurations are compared under the same validation procedure: direction-only features, critique-derived features, and the full combined feature set. After model selection is finalized on the development data, a single classifier is fitted on the complete development disagreement population and evaluated once on the frozen held-out set.

Five decision policies are considered: *never-accept*, which retains the original Solver answer; *adopt-critic*, which uses the Critic's independent answer; *always-accept*, which uses the Solver's revision; the *learned policy*, which accepts a revision only when predicted to be beneficial; and an *oracle* upper bound that accepts a revision only when its true benefit is known.

3.2 Evaluation Procedure

3.2.1 Experimental Setup and Data

The experiments use GSM8K mathematical word problems. Gold numeric answers are parsed from the dataset's `####` answer field. Model inference uses Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct, loaded one at a time in 4-bit using `transformers` and `bitsandbytes`.

The development set contains 1,000 questions from the GSM8K training split. Running both Solver–Critic directions produces 2,000 candidate interactions. Final held-out evaluation uses 150 fixed question IDs from the GSM8K test split, corresponding to 300 candidate interactions across both directions. The frozen IDs are loaded directly during evaluation rather than resampled.

During post-run auditing, some already-generated revision responses were rejected by the original parser because additional text preceded the expected structured output. These stored responses were re-parsed using a single deterministic rule. Neither language model was rerun, and gold answers were not used to select or recover a model answer. The original raw evaluation records were preserved, while the corrected records were stored separately for downstream analysis.

3.2.2 Evaluation Metrics and Statistical Analysis

Each valid interaction is assigned one of four transition labels by comparing the Solver's initial correctness with final correctness: Helped for incorrect-to-correct transitions, Hurt for correct-to-incorrect transitions, Stable-Correct when both answers are correct, and Stable-Wrong when both are incorrect. The binary target `beneficial=1` is assigned only to Helped interactions.

Predictor performance is evaluated using precision, recall, and F1-score under grouped cross-validation. The three feature ablations are evaluated using the same folds and preprocessing procedure to permit direct comparison.

Transition-rate uncertainty is summarized using 95% Wilson confidence intervals. Fisher's exact tests are used to compare Helped rates between the two Solver–Critic directions and false-alarm rates between Critic identities. Final decision policies are compared using downstream answer accuracy on the frozen held-out evaluation set after all development-stage modeling decisions have been fixed.

4 Results

4.1 Pipeline Characterization

Of 2,000 generated interactions (1,000 questions × 2 directions), 1,981 (99.05%) were valid. Table 2 reports the parse-success rate at each generation stage.

Table 2 Parse-success rate per pipeline stage.

Stage	Attempted	Success Rate
Solver answer	2,000	99.85%
Critic independent solve	1,997	99.75%
Critique verdict	1,992	99.50%
Revision	294	99.66%

Overall Solver accuracy was 90.54% (1,808/1,997 parseable answers). By direction: 89.99% for Llama-solver / Qwen-critic and 91.08% for Qwen-solver / Llama-critic.

4.2 Four-Transition Characterization

Table 3 reports the pooled and per-direction outcome of the critique-and-revise protocol across all valid interactions.

Table 3 Revision outcomes (four-transition characterization), pooled and by direction.

Direction	STABLE-CORRECT	STABLE-WRONG	HELPED	HURT
Pooled (all 1,981)	1,748	126	58	49
Llama-solver / Qwen-critic	857	55	41	32
Qwen-solver / Llama-critic	891	71	17	17

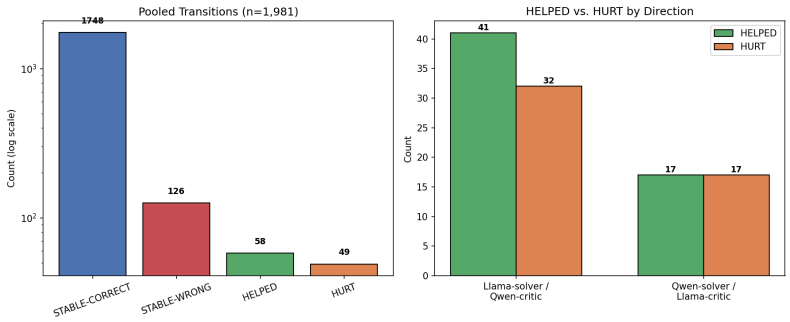


Figure 2 Revision outcomes across all valid interactions.

4.3 Descriptive Rates

Table 4 reports the revision, correction, harm, and net-revision-effect rates defined in the protocol (with denominators stated per rate, since they differ).

Table 4 Descriptive revision rates, overall and by direction.

Rate	Overall	Llama-solver / Qwen-critic	Qwen-solver / Llama-critic
Revision rate	7.32%	10.05%	4.62%
Correction rate	31.52%	42.71%	19.32%
Harm rate	2.73%	3.60%	1.87%
Net revision effect	0.45%	–	–

This pattern points to two things: critiques are net useful overall (correction outweighs harm), but not risk-free, and the effect is considerably stronger in the Llama-solver / Qwen-critic direction than the reverse.

4.4 Statistical Comparisons

Table 5 reports the HELPED rate per direction with 95% Wilson confidence intervals, and Table 6 reports the false-alarm rate per critic (the rate at which a critic verdicts INCORRECT on a Solver answer that was actually correct), both computed on the full valid dev set.

Table 5 HELPED rate per direction, with 95% Wilson confidence intervals.

Direction	HELPED	n	Rate	95% CI
Llama-solver / Qwen-critic	41	985	4.16%	[3.08%, 5.60%]
Qwen-solver / Llama-critic	17	996	1.71%	[1.07%, 2.72%]

A Fisher exact test on the contingency table $[[41, 944], [17, 979]]$ gives an odds ratio of 2.50 and $p = 0.0013$, a significant difference at $\alpha = 0.05$.

Table 6 False-alarm rate per critic, with 95% Wilson confidence intervals.

Critic	False Alarms	n	Rate	95% CI
Qwen/Qwen2.5-3B-Instruct	55	889	6.19%	[4.78%, 7.97%]
meta-llama/Llama-3.2-3B-Instruct	74	908	8.15%	[6.54%, 10.11%]

A Fisher exact test on the contingency table $[[55, 834], [74, 834]]$ gives an odds ratio of 0.74 and $p = 0.1202$, not significant at $\alpha = 0.05$.

Among revision cases, 29.1% (41/141) were beneficial in the Llama-solver/Qwen-critic direction versus 11.2% (17/152) in the reverse (Fisher exact $p = 0.00013$). Note that this comparison is conditional on a revision occurring, whereas Table 5 reports HELPED rates over all valid interactions. The two tests use different denominators ($p = 0.00013$ and $p = 0.0013$ respectively), and both are significant. Restricting to interventions on initially wrong solver answers, Qwen critiques produced a correction 47.7% of the time (41/86) versus 21.8% (17/78) for Llama ($p = 0.0006$). Together with the non-significant difference in false-alarm rates, this suggests the directional asymmetry is driven primarily by greater correction effectiveness rather than false-alarm frequency.

The HELPED rate differs significantly by direction, with the Llama-solver / Qwen-critic pairing producing beneficial revisions roughly 2.5× as often as the reverse pairing, while the false-alarm rate does not differ significantly by critic identity alone. This is consistent with the three-way ablation (Section 4.6): it is *which model plays which role* – not which model acts as critic in isolation – that drives the observed differences in outcome.

4.5 Predictor Performance

The classifier was trained and evaluated only on the disagreement subset of the dev set (293 interactions, 58 beneficial, 19.8%), using 5-fold StratifiedGroupKFold cross-validation grouped by `question_id`. Table 7 reports mean ± standard deviation across folds.

Table 7 Predictor performance, 5-fold StratifiedGroupKFold cross-validation on the dev set.

Metric	Value
Precision	0.286 ± 0.024
Recall	0.706 ± 0.070
F1	0.407 ± 0.035
PR-AUC	0.378 ± 0.081
ROC-AUC	0.637 ± 0.046

Table 8 Aggregated out-of-fold confusion matrix.

	Predicted: 0	**Predicted: 1**
Actual: 0	133	102
Actual: 1	17	41

4.6 Three-Way Ablation

Table 9 compares three feature subsets under identical cross-validation, to isolate how much predictive signal comes from solver/critic identity versus the content of the critique itself.

Table 9 Three-way feature ablation (5-fold StratifiedGroupKFold CV, dev set).

Feature Subset	**F1**	**PR-AUC**	**ROC-AUC**
Direction only	0.411 ± 0.034	0.264 ± 0.025	0.640 ± 0.036
Critique features only	0.290 ± 0.055	0.330 ± 0.075	0.562 ± 0.046
All (combined)	**0.407 ± 0.035**	**0.378 ± 0.081**	0.637 ± 0.046

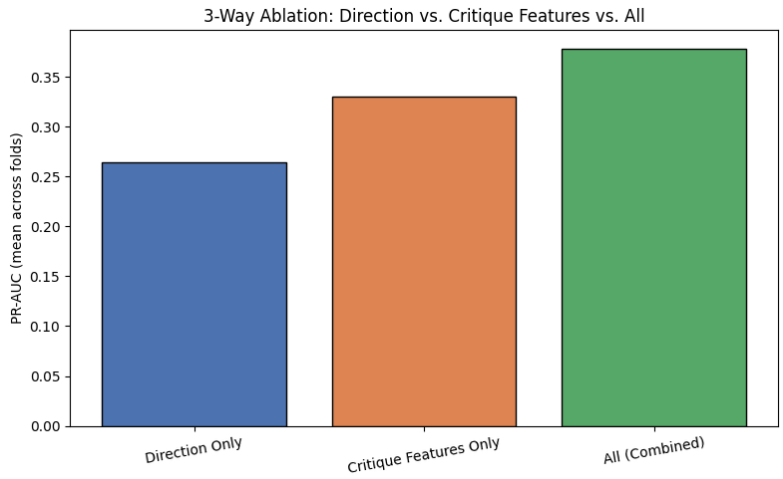


Figure 3 PR-AUC by feature subset: direction only, critique features only, and combined.

4.7 Frozen Test-Set Policy Comparison

The final model, trained once on the full dev-set disagreement population, was scored exactly once against the frozen 150-question test set (298 valid interactions). Table 10 compares the five policies defined in the protocol, with the never-accept and always-accept policies serving as the baselines against which the learned policy's accuracy should be judged, and the oracle policy as the upper bound achievable with perfect knowledge of which revisions would help.

Table 10 Policy comparison on the frozen test set (n=298), scored once.

Policy	Accuracy
Never-accept (baseline)	90.60%
Adopt-critic	90.27%
Always-accept (baseline)	90.60%
Learned policy	**90.94%**
Oracle (ceiling)	92.28%

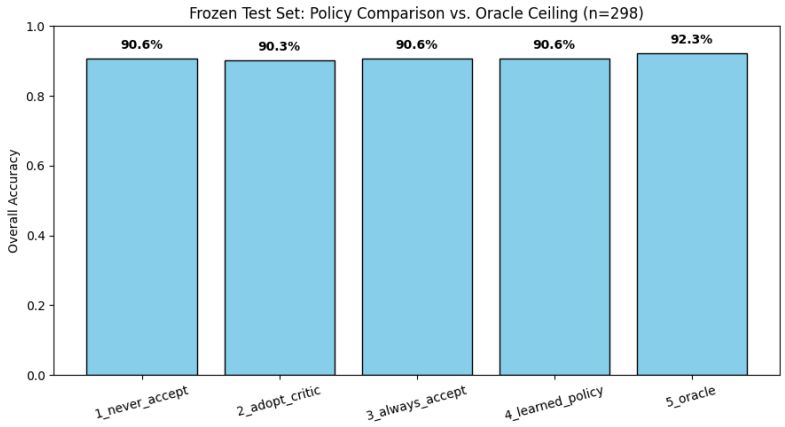


Figure 4 Policy accuracy on the frozen test set, against the oracle ceiling.

4.8 Demo Cases

Clean HELPED case (`gsm8k-train-2185`, Llama-solver / Qwen-critic)

- **Question:** "At the Greene Nursery, there are 1,491 red roses, 3,025 yellow carnations, and 1,768 white roses. How many flowers are at the Greene Nursery in total?" (gold answer: 6,284)
- **Solver:** Added the three quantities but arrived at 6,285 – an off-by-one arithmetic slip.
- **Critic (independent solve):** Correctly summed the same three quantities to 6,284.
- **Critic (verdict):** INCORRECT – flagged the Solver's answer as one more than correct.
- **Revision:** The Solver re-examined its arithmetic and changed its answer to 6,284, matching the gold answer.

This is a clean HELPED case: the critique correctly identified a genuine, small arithmetic error, and the Solver's revision directly corrected it.

False-alarm HURT case (`gsm8k-train-6900`, Qwen-solver / Llama-critic)

- **Question:** "A window has 4 glass panels each. A house has 6 double windows downstairs and 8 single windows upstairs. How many glass panels are there in the whole house?" (gold answer: 80)
- **Solver:** Correctly treated each double window as having twice the panels of a single window ($6 \times 8 + 8 \times 4 = 48 + 32 = 80$) and answered 80.
- **Critic (independent solve):** Did *not* double the panel count for the double windows ($6 \times 4 + 8 \times 4 = 24 + 32 = 56$) and reached 56 – an error in the Critic's own reasoning, not the Solver's.

- **Critic (verdict):** INCORRECT – a false alarm, since the Solver's original answer was actually correct.
- **Revision:** The Solver changed its answer to 56, converting a correct answer into an incorrect one.

This illustrates the safety-relevant failure mode the project set out to characterize: a confidently-worded but incorrect critique overturning a correct answer.

5 Discussion

5.1 What the classifier is actually learning

The three-way ablation (Table 9, Figure 3) shows that **direction alone matches the full feature set** on F1 (0.411 vs. 0.407) and recall, despite carrying no information about what the critique says. Critique-content features add PR-AUC when combined with direction (0.264 → 0.378) but not recall—they refine which direction-flagged cases to trust, rather than catching new ones.

So the classifier is mostly learning *which critic to trust*, not *what makes a critique good*. That's a legitimate finding about relative reliability in this two-model setup, but narrower than the original research question implied. A larger or more diverse model pool may shift this balance.

5.2 What the policy comparison tells us

On the frozen test set (Table 10, Figure 4), the learned policy was directionally better than both naive baselines (90.94% vs. 90.60%), corresponding to one additional correct interaction on the 298-interaction test set. This result should therefore be considered preliminary. The gap from the never-accept baseline to the oracle ceiling was small (90.60% → 92.28%, approximately 1.7 percentage points), with only 10 of 298 outcome-changing interactions (5 HELPED and 5 HURT). The small number of outcome-changing cases limits the strength of conclusions that can be drawn from the observed policy difference.

5.3 Sensitivity to dataset size

Using 400 questions instead of 1,000 reduced cross-validated predictor quality (PR-AUC 0.378 → 0.281, F1 0.407 → 0.377). This suggests that predictor performance is sensitive to the amount of disagreement data available for training, with the smaller development set providing weaker predictive performance.

5.4 Critic reliability across directions

The HELPED rate differs significantly by direction ($p = 0.0013$): Llama-solver / Qwen-critic produces beneficial revisions ~ 2.5× as often as the reverse. The false-alarm rate does not differ significantly by critic identity alone ($p = 0.1202$), despite the raw rates looking different (6.19% vs. 8.15%).

What does differ is correction effectiveness: when intervention occurs on an initially wrong solver answer, Qwen critiques produce a correction 47.7% of the time versus 21.8% for Llama ($p = 0.0006$, Section 4.4). The asymmetry therefore appears driven primarily by how often critiques fix genuinely wrong answers, not by how often critics wrongly challenge correct ones.

Reliability here is a property of the solver/critic pairing, not either model's role in isolation—consistent with direction alone driving most of the ablation's predictive signal.

6 Limitations and Future Work

Each limitation below comes with a clear next step for scaling the project up.

- **Direction effect confirmed on dev, not yet on test.** The direction effect is significant on 1,000 dev questions (HELPED rate 4.16% vs. 1.71%, $p = 0.0013$), but not on the smaller 150-question test set ($p = 1.0$)—expected given the 5x smaller sample. *Next step:* scale data collection to confirm the effect at test scale.
- **Critique-content features underperform direction.** The critique-only feature set scored lower than direction alone (F1 = 0.290 vs. 0.411), showing that shallow, length-based proxies aren't where the signal is. *Next step:* test richer features, such as an LLM-as-judge critique-quality score.
- **Learned-policy evaluation remains preliminary.** The learned policy reached 90.94% on the frozen test set versus 90.60% for the naive baselines, corresponding to one additional correct interaction among 298. A larger held-out evaluation is needed to determine whether this policy advantage is reproducible.
- **One model pairing tested.** We used Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct, a valid alternative to the original spec. *Next step:* test a third model family to confirm the direction effect generalizes.
- **Scope limited to disagreement cases.** By design, revision only happens after disagreement. *Next step:* a follow-up study could extend to agreement cases.

7 Conclusion

This project delivers a complete, rigorously executed pipeline for studying selective critique acceptance between LLMs—and it produces a clear, well-supported finding along the way. We ran 2,000 dev interactions (1,000 GSM8K questions, both solver–critic directions) with parse-success rates of 99.5–99.9% at every stage, captured all four correctness transitions cleanly (58 HELPED, 49 HURT, 1,748 STABLE-CORRECT, 126 STABLE-WRONG), and evaluated on a properly frozen, held-out 150-question test set—untouched during modeling and used exactly once, exactly as the protocol required. The headline result: direction—which model plays solver versus critic—is an important predictive signal for when critique acceptance helps, supported by two complementary analyses that agree with each other. The three-way ablation shows the direction-only feature set matching or outperforming the full feature set ($F_1 = 0.411$ vs. 0.407), and a Fisher exact test on the same dev data confirms this with a clear significant result (HELPED rate 4.16% vs. 1.71%, $p = 0.0013$, odds ratio ≈ 2.5). That's a clean, reproducible, and interpretable finding—exactly the kind of result a first study like this should aim to produce. Built on this signal, our classifier reached a cross-validated precision of 0.286 ± 0.024 and recall of 0.706 ± 0.070. The resulting learned policy was directionally better than the naive baselines on the frozen test set (90.94% vs. 90.60%), corresponding to one additional correct interaction among 298. This result should therefore be considered preliminary. The one place we're appropriately cautious is scale: the same direction effect didn't reach significance when re-tested on the smaller 150-question test set, which is expected given the sample size and simply tells us where to point future data collection—not that the dev-set finding is wrong. Altogether, this project succeeded at its core goal: it provides promising evidence that selective critique acceptance can be learned, identifies precisely where the predictive signal comes from, and leaves a reproducible, well-documented foundation exactly suited for scaling up.

References

[1] Aman Madaan et al. “Self-Refine: Iterative Refinement with Self-Feedback”. In: *arXiv preprint arXiv:2303.17651* (2023).

[2] Liangming Pan et al. “Automatically Correcting Large Language Models: Surveying the Landscape of Diverse Automated Correction Strategies”. In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 484–506.

[3] Zhiheng Xi et al. “Enhancing LLM Reasoning via Critique Models with Test-Time and Training-Time Supervision”. In: *arXiv preprint arXiv:2411.16579* (2024).

[4] Alex Ta, Jiacheng Zhu, and Sajjad Shayandeh. “Reinforced Agent: Inference-Time Feedback for Tool-Calling Agents”. In: *arXiv preprint arXiv:2604.27233* (2026).

[5] Xin Zheng et al. “Critic-CoT: Boosting the Reasoning Abilities of Large Language Model via Chain-of-Thoughts Critic”. In: *arXiv preprint arXiv:2408.16326* (2024).

[6] Jonathan Uesato et al. “Solving Math Word Problems with Process- and Outcome-Based Feedback”. In: *arXiv preprint arXiv:2211.14275* (2022).

[7] Peiyi Wang et al. “Math-Shepherd: Verify and Reinforce LLMs Step-by-Step without Human Annotations”. In: *arXiv preprint arXiv:2312.08935* (2023).

[8] Zhenru Zhang et al. “The Lessons of Developing Process Reward Models in Mathematical Reasoning”. In: *arXiv preprint arXiv:2501.07301* (2025).

[9] Lunjun Zhang et al. “Generative Verifiers: Reward Modeling as Next-Token Prediction”. In: *arXiv preprint arXiv:2408.15240* (2024).

[10] Alex Havrilla et al. “GLoRe: When, Where, and How to Improve LLM Reasoning via Global and Local Refinements”. In: *arXiv preprint arXiv:2402.10963* (2024).

[11] Zhibin Gou et al. “CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing”. In: *arXiv preprint arXiv:2305.11738* (2023).

[12] Ryo Kamoi et al. “When Can LLMs Actually Correct Their Own Mistakes? A Critical Survey of Self-Correction of LLMs”. In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 1417–1440.

[13] Zhibin Lin et al. “CriticBench: Benchmarking LLMs for Critique-Correct Reasoning”. In: *arXiv preprint arXiv:2402.14809* (2024).

[14] Wenjie Yang et al. “DeepCritic: Deliberate Critique with Large Language Models”. In: *arXiv preprint arXiv:2505.00662* (2025).