

539 A Prompts and evaluation details for problem solving and feedback 540 generation

541 A.1 Prompts used for the solver model

542 We developed different prompting strategies for the solver model to incorporate feedback and generate
543 new solutions across iterations. The system prompts in Figure 8 were used for the solver model
544 across different tasks:

```
Math-500 and AIME 2024:
You are a smart assistant that solves math problems. Please think step by step to solve the problem. If you think you're ready
to output the answer, you can wrap your answer with \boxed{}. Please follow this format

TriviaQA:
You are a smart assistant that solves trivia questions. If you think you're ready to output the answer, you can just output an
answer.

MMLU and MMLU Pro:
The following are multiple-choice questions about {category}. Let's think step by step. Please explain your reasoning clearly as
you work toward the answer. When you're ready, conclude your answer with the phrase: \"The answer is (X)\" where X is the
correct letter choice. Make sure to always include parentheses around the letter.

GPQA:
The following are multiple-choice questions. Let's think step by step. Please explain your reasoning clearly as you work toward
the answer. When you're ready, please finish your answer with \"The answer is (X)\" where X is the correct letter choice. Make
sure to always include parentheses around the letter.

PopQA:
You are a smart assistant that answers fact-based questions. If you think you're ready to output the answer, you can just output
an answer.

5-Digit Multiplication:
You are a smart assistant in solving multiplication questions. Please think step by step. If you think you're ready to output
the answer, you can wrap your answer with \boxed{}. Please follow this format.

Hexadecimal 5-Digit Multiplication:
You are a smart assistant in solving hexadecimal multiplication questions. Please think step by step. If you think you're ready
to output the answer, you can wrap your answer with \boxed{}. Please follow this format.
```

Figure 8: System prompts used for the solver model across all tasks.

545 At the initial generation round, we directly use the question as the prompt for the solver model.
546 For multiple-choice questions, we format the question by concatenating the question and answer
547 choices with their corresponding labels (i.e., A to D for MMLU and GPQA, A to J for MMLU
548 Pro). In subsequent rounds, we provide the solver model with its complete previous history and
549 the corresponding feedback from the feedback generator model. We clearly label each iteration so
550 the solver model can track all its previous attempts. The general template for the iterative prompt
551 structure is provided in Figure 9.

```
Question:
[Original Question]

Prompt for Generation after the Initial Round:
You are given the full history of your previous attempts and the feedback provided for each attempt.
History:

Attempt at (iteration 1) and the corresponding feedback:
Answer: [answer1] Feedback: [Feedback 1 Depends on Feedback Strategy].

Attempt at (iteration 2) and the corresponding feedback:
Answer: [answer2] Feedback: [Feedback 2 Depends on Feedback Strategy].

Question: [Original Question]. Please answer the question again.

* [answer1] and [answer2] corresponds to the model's first two failed attempts. [Original Question] refers to the initial question the
model attempts to answer.
```

Figure 9: Prompt used for iterative self-improvement

552 A.2 Evaluation details for problem solving

553 We employ few-shot prompting across all tasks to provide consistent context for the solver model. For
554 TriviaQA and PopQA, we randomly sample 5 questions without replacement as few-shot examples.

For MMLU and MMLU Pro, we similarly sample 5 questions from the corresponding question category to ensure domain-relevant examples.

For PopQA, we employ an LLM-as-a-judge approach [64] to assess answer correctness. This is necessary because PopQA provides limited answer aliases (extensive alternative phrasings for exact string matching) compared to TriviaQA. Without this approach, models would be penalized for minor formatting differences rather than genuine comprehension errors, leading to an underestimation of their true problem-solving capabilities. For other tasks, we follow the same evaluation metrics provided by lm-eval-harness.

A.3 Prompts used for feedback generation

We implement three distinct feedback generation strategies as described in §2.2. For Binary Correctness Feedback (F_1), we provide minimal information: “Your answer was incorrect. Please answer the question again.”

For Self-Generated Reflective Feedback (F_2) and Strong-Model Reflective Feedback (F_3), we employ identical prompt templates that differ only in the model used for generation. The feedback generator receives the complete interaction history, including all previous solver attempts and corresponding feedback. When available, we provide the feedback model with detailed solution explanations that justify the correct answer; for datasets lacking such explanations, we provide only the ground truth answer. This approach ensures the feedback model has sufficient context to generate targeted, informative guidance while maintaining consistency across feedback types, and its template is shown in Figure 10.

```
Prompt for Generating the Feedback:
There was a mistake in answering the following question:

[Original Question]

You are provided with the full history of previous attempts made by a separate model, along with corresponding feedback.
History:
Attempt at (iteration 1) and the corresponding feedback:

Answer: [answer1] Feedback: [Feedback 1 Depends on Feedback Strategy].

Attempt at (iteration 2) and the corresponding feedback:

Answer: [answer2] Feedback: [Feedback 2 Depends on Feedback Strategy].

Most Recent Answer: [answer3]

The correct final answer is: [Ground Truth Answer]
The correct reasoning process that leads to this answer is: [Solution with Process for this Question]

Please provide feedback identifying which step(s) were incorrect or how to get to the correct answer WITHOUT revealing
the correct final answer or the content of the correct option.

* [answer1] [answer2] [answer3] refer to all previous attempts the model had. [answer3] is the most recent attempt.
[Ground Truth Answer] is the answer for this question. [Solution with Process for this Question] is the detailed solution
provided for the question in the dataset.
```

Figure 10: Prompt used for generating the feedback.

A.4 Answer masking in feedback

To ensure fair evaluation, we implement comprehensive answer masking to prevent feedback from directly revealing ground truth solutions while preserving feedback quality. Our approach allows feedback to contain detailed solution steps and guidance but strictly prohibits explicit disclosure of final answers. We use “[masked]” as the replacement token for filtered content.

Multiple-choice questions. We mask all possible representations of the correct choice letter. For example, if the correct answer is A, we filter variants including (A), \boxed{A} , **A**, etc.

Open-ended questions. For TriviaQA, we filter all terms matching the words in “aliases” and “normalized aliases” answer fields. For PopQA, we mask entries from the “possible answers” answer field. For mathematical tasks (5-digit multiplication and MATH-500), we mask standalone numerical

585 answers and those in `\boxed{}` notation. Hexadecimal multiplication follows similar patterns. For
 586 multiplication tasks, we additionally mask intermediate partial products to prevent reduction to simple
 587 addition problems (detailed in Appendix C).

588 A.5 Error Categorization Prompt

589 The prompt template used for categorizing persistent model errors after 9 iterations is shown in Figure 11.

```

System Prompt:
You are an error categorizer specialized in analyzing why Language Learning Models (LLMs) "
fail to self-improve when solving problems. When provided with an LLM's prediction trajectory and the feedback it receives,
you will categorize the errors into one of six categories:
1. Problem is Impossible to Solve
  - The problem itself is fundamentally flawed
  - External tools are required (e.g., calculator for complex calculations, search engine for obscure facts)
2. Problem is Too Complicated
  - The problem exceeds the model's knowledge scope
  - Example: A level 5 math problem beyond the model's training
3. Feedback is Wrong
  - The feedback generator model provides incorrect guidance
  - The feedback fails to identify actual mistakes in the model's response
  - The feedback is too vague or generic to be helpful
4. Model is Not Following Feedback
  - The model fails to incorporate feedback properly
5. Style or Formalization Issue
  - Logical correctness but formatting or notation problems
6. Unknown
  - Cannot categorize the failure into above categories
End your response with exactly: 'The error is: [category]' where category is one of:\n- impossible to solve\n- too
complicated\n- feedback is wrong\n- model is not following the feedback\n- style or formalization issue\n- unknown

User Prompt:
The question and the interaction between the agent model and the feedback model is: {question}
The process answer is: {process_answer}
Please categorize the error. End your response with:\n'The error is: [category]'

* {question} is the first 5 iterations of the answer-feedback pairs; {process_answer} is the best ground truth solution we can
provide from the dataset to the model.

```

Figure 11: Error Categorization

590

591 B Error categorization examples from iterative self-improvement

592 This section presents representative examples of persistent errors that prevent models from achieving
 593 correct solutions despite multiple feedback iterations. We illustrate the main error categories identified
 594 in our analysis: feedback resistance (where models fail to incorporate valid corrections, see Figure 12)
 595 and feedback quality issues (where the provided guidance is incorrect or misleading, see Figure 13).

596 C Synthetic digit multiplication task details

597 C.1 5-digit multiplication

598 We construct a controlled arithmetic dataset consisting of 450 5-digit multiplication problems follow-
 599 ing the template: "Calculate the following question: 19365×12534 ."

600 **Feedback generation** We employ a deterministic, human-designed template based on the distribu-
 601 tive property to generate ground truth solutions. This template systematically decomposes each
 602 multiplication into partial products, providing a clear step-by-step solution pathway. Our template-
 603 based approach serves two key purposes: (1) demonstrating structured problem decomposition
 604 strategies for complex arithmetic, and (2) ensuring feedback correctness and interoperability. In
 605 Figure 14, we illustrate an example template solution, which serves as the reference for feedback
 606 generation. The feedback model compares solver outputs against this structured breakdown to identify
 607 specific computational errors.

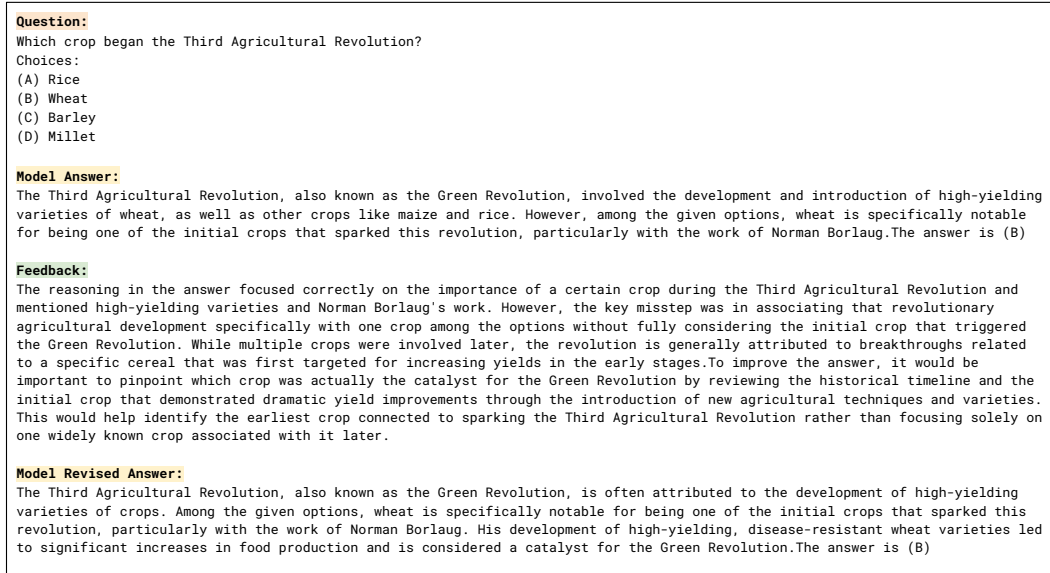


Figure 12: Llama-4-Scout resisting feedback from GPT-4.1 mini in MMLU

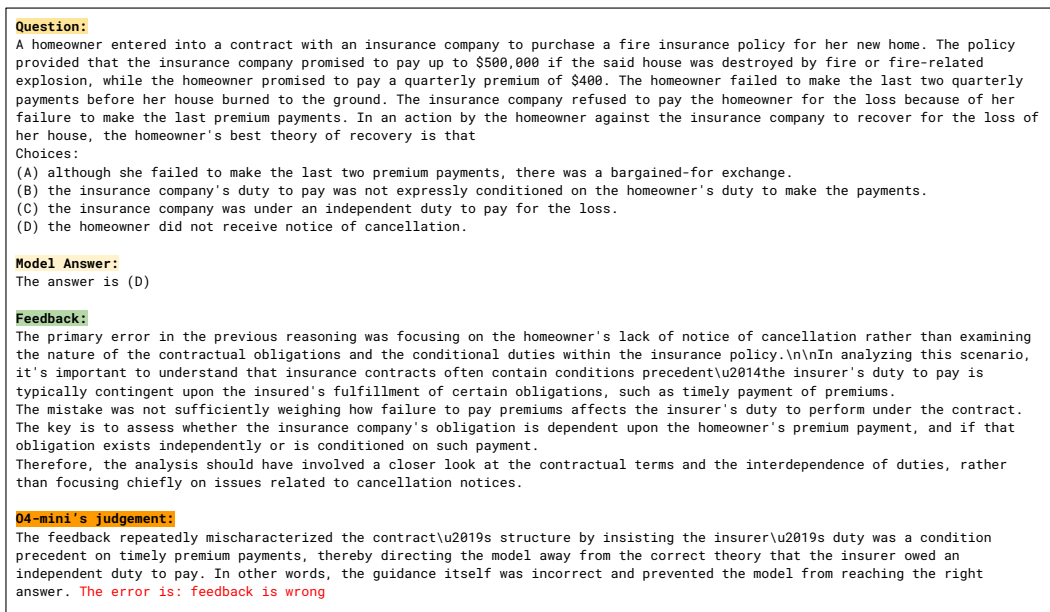


Figure 13: Wrong feedback provided by GPT-4.1 mini judged by o4-mini

608 **Answer masking strategy** To maintain task difficulty, we also mask intermediate partial answers
609 before providing feedback to the solver model. The final feedback combines the masked template
610 solution with model-generated guidance tailored to the specific errors observed.

611 C.2 Hexadecimal 5-Digit Multiplication

612 We also extend our synthetic arithmetic evaluation to hexadecimal multiplication, creating problems
613 that challenge models' ability to work with non-standard number systems. The question tem-
614 plate follows the format: "Calculate the following question, where each number is
615 represented in base 16: 69837×17635 ." All answers are expected in base-16 format.

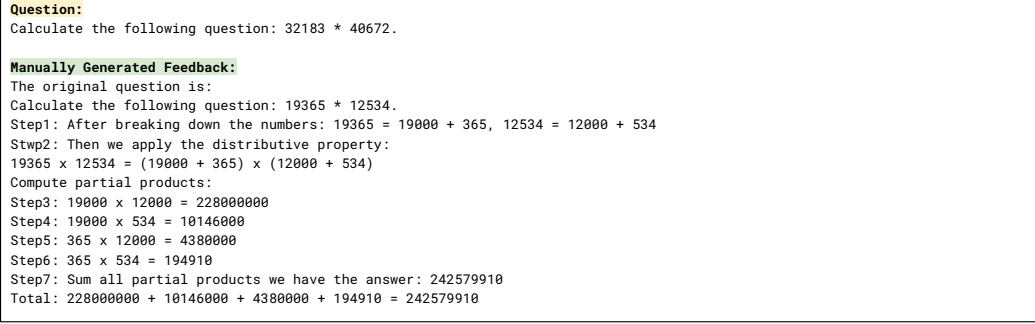


Figure 14: Templates for 5-digit multiplication solution.

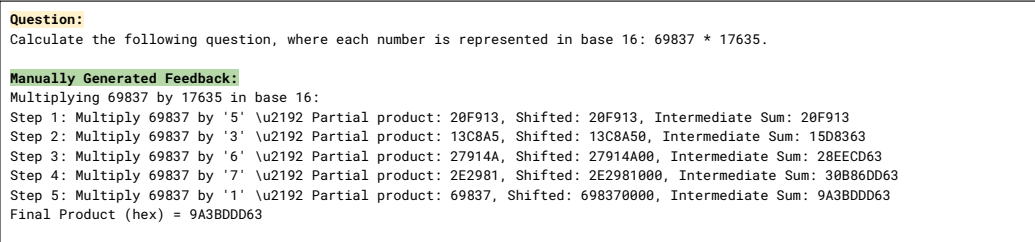


Figure 15: Hexadecimal 5-Digit Multiplication process solution.

Template-based feedback. Similar to decimal multiplication, we generate feedback using deterministic step-by-step templates. The multiplication process involves sequentially multiplying the first operand by each digit of the second operand (interpreting digits in base-16), then summing the resulting partial products with appropriate positional shifts. Figure 15 demonstrates an example template solution. We validate solution correctness by verifying that partial product summation matches results from standard base-16 calculators.

Masking strategy for hexadecimal multiplication. Given the increased computational complexity of hexadecimal arithmetic, we employ a more permissive masking approach. We mask only the final summation step while preserving intermediate partial products. This design balances providing sufficient guidance with maintaining the core challenge of hexadecimal computation.

D Analysis of model confidence and RIGID THINKING

Figure 16 presents confidence-accuracy relationships across four datasets: GPQA, MMLU, MMLU Pro, and TriviaQA. Each plot displays three metrics: initial accuracy at iteration 0, final accuracy after iterative feedback, and the improvement delta between them.

We define a model’s *initial confidence* in its generated answer as the exponential of the average log-probability per token in the answer sequence:

$$\text{Initial Confidence} = \exp \left(\frac{1}{T} \sum_{t=1}^T \log p(a_t \mid a_{<t}, q) \right)$$

where:

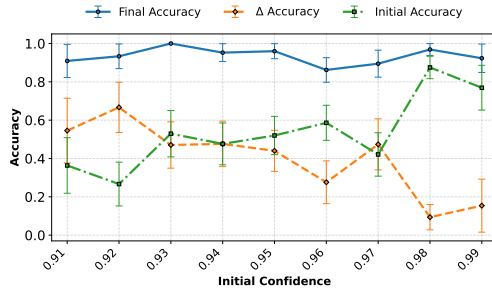
- T is the number of tokens in the generated answer (with EOS excluded),
- a_t is the t -th token in the answer,
- $a_{<t}$ denotes the prefix of the answer up to (but not including) position t ,
- q is the input question,
- $p(a_t \mid a_{<t}, q)$ is the model’s probability of generating token a_t given the previous tokens and the input.

This corresponds to the geometric mean of the per-token probabilities assigned by the model, and serves as a quantitative measure of the model’s confidence in its complete answer.

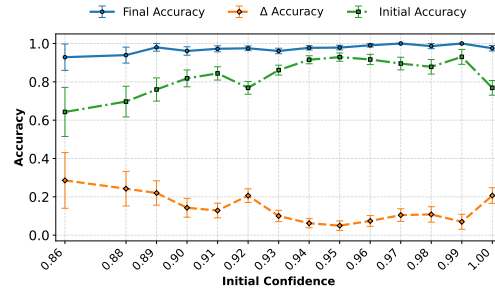
We find that **initial confidence strongly predicts initial accuracy** across all datasets. Higher confidence bins consistently correspond to higher initial performance (Figures 16a–16d), confirming that the model’s confidence is a good predictor of its initial accuracy.

However, **confidence poorly predicts improvement potential**. The relationship between initial confidence and accuracy gains varies substantially:

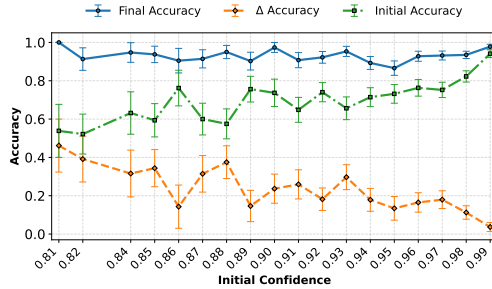
- GPQA shows peak improvements at moderate confidence levels, with diminishing returns at higher confidence
- MMLU and MMLU Pro exhibit relatively flat improvement patterns with considerable variance
- TriviaQA demonstrates erratic improvement fluctuations regardless of initial confidence



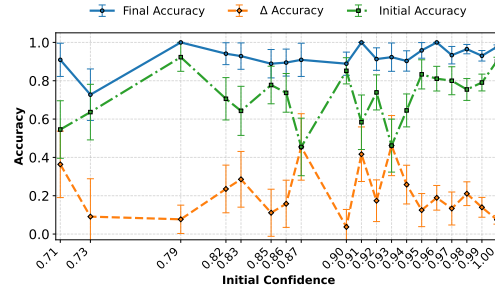
(a) GPQA confidence vs. accuracy



(b) MMLU confidence vs. accuracy



(c) MMLU Pro confidence vs. accuracy



(d) TriviaQA confidence vs. accuracy

Figure 16: Confidence vs. accuracy across different datasets using GPT-4.1 mini as feedback model and Llama-4-Scout as the solver.

E Analysis of data familiarity and RIGID THINKING

After analyzing data familiarity using answer frequency in the PopQA dataset, we found no clear correlation between model performance and frequency of answer words in training data. However, surface-level frequency may not fully capture a model’s true familiarity with content, as it fails to account for context quality, semantic relationships, and other factors affecting knowledge acquisition during pre-training.

To better capture actual familiarity, we examine a more direct behavioral signal called in-domain performance: the model’s accuracy in answering questions, measured using 100 generations per question with Llama-3.3. This behavioral familiarity metric reflects the cumulative effect of all factors contributing to the model’s internalized knowledge.

Figure 17 illustrates the in-domain performance of Llama-3.3 across four benchmarks—GPQA, TriviaQA, 5-digit multiplication, and MMLU Pro. We bucket questions based on the model’s initial accuracy and report both initial and final accuracy after iterative feedback. While the model shows improvement across all buckets, questions with higher behavioral familiarity (higher initial accuracy) consistently achieve higher final accuracy as well. This suggests that behavioral familiarity is a more informative predictor of both current performance and improvement potential than answer frequency alone.

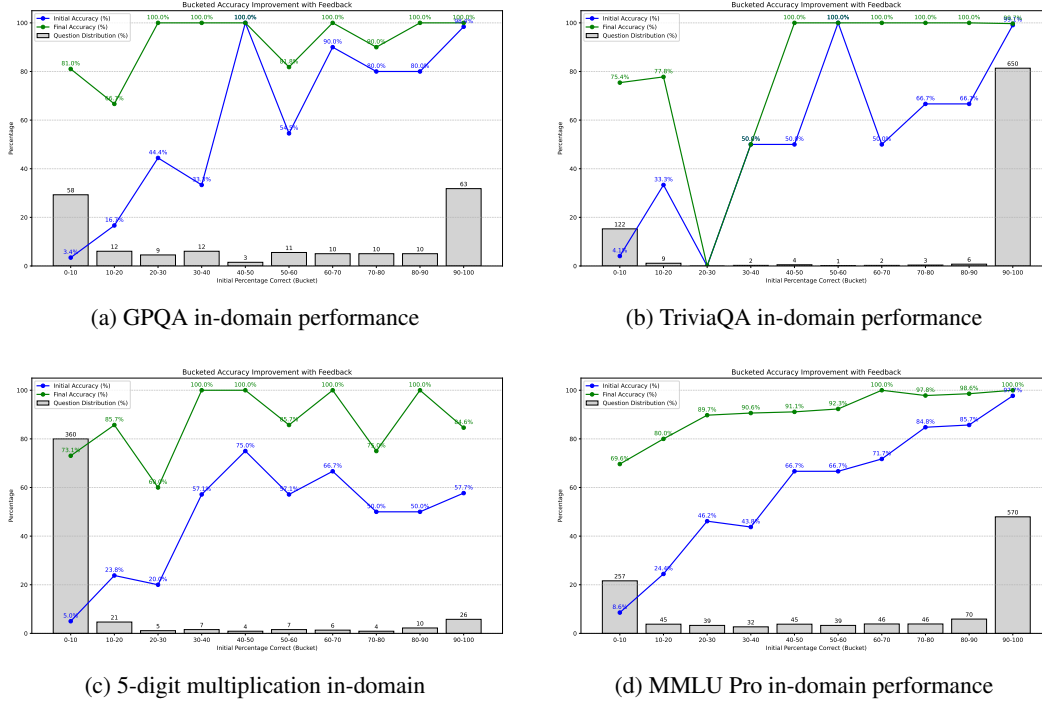


Figure 17: In-domain accuracy of Llama-3.3 across four benchmark tasks.

F Analysis of reasoning complexity and RIGID THINKING

To investigate whether the model’s improvement over iterations correlates with question difficulty or the reasoning complexity, we compare the performance of Llama-4-Scout on two synthesized multiplication tasks: 5-digit and 6-digit problems. Unlike prior datasets, which lack clear separability in difficulty levels, these tasks were manually constructed with 450 questions each to ensure a well-defined difference in complexity.

The initial accuracy of Llama-4-Scout is 2.2% on 5-digit multiplication and 0.889% on 6-digit multiplication. Interestingly, while the 6-digit task is objectively more difficult, we observe greater improvement across iterations compared to the 5-digit task. One possible explanation is that more difficult tasks offer more room for feedback-driven correction because solver model has less initial knowledge about how to solve them. However, we also observe cases where simpler questions yield higher final accuracy, suggesting that the relationship between task complexity and feedback effectiveness is non-monotonic and influenced by additional factors.

G Analysis of model type and RIGID THINKING

To better understand the overlap in failure cases among Llama-3.3, Llama-4-Scout, and Llama-4-Maverick, we compared their incorrect predictions across several benchmark datasets. Specifically, we report the number of shared mistakes between each pair of models, the number of questions all three models got wrong, the total number of unique mistakes (union), and the **Overlap Ratio**, defined as the proportion of all-three common errors to the total number of distinct errors.

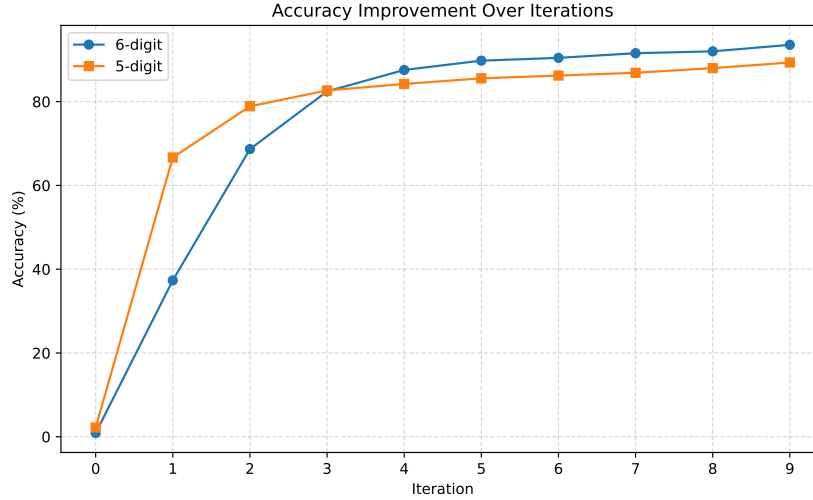


Figure 18: Comparison of the Improvement for 5-digit and 6-digit multiplication with GPT-4.1 mini as feedback model

687 The *Overlap Ratio* offers a normalized measure of agreement in model failures. Notably, **TriviaQA**
688 shows the highest overlap (35.7%), suggesting a subset of examples that all three models consistently
689 struggle with. Conversely, datasets such as **GPQA** and **5-digit Multiplication** exhibit minimal
690 overlap (6.9% and 0.7%, respectively), indicating that the models tend to fail on different questions.
691 These findings suggest that model failures are often idiosyncratic rather than being concentrated
692 around a universally difficult subset of examples. The relatively low overlap across datasets highlights
693 the challenge of achieving robust self-correction: errors are not easily attributable to a common set of
694 pitfalls, but rather reflect distinct weaknesses in each model’s reasoning and generalization.

Dataset	L3.3–Scout	L3.3–Maverick	Scout–Maverick	All-Three	Union	Overlap Ratio
TriviaQA	8	9	5	5	14	0.357
AIME	22	16	28	14	105	0.133
MATH-500	18	14	11	9	64	0.141
MMLU	15	12	19	11	55	0.200
MMLU Pro	49	40	43	30	163	0.184
GPQA	3	5	2	2	29	0.069
5-digit Mult.	21	3	1	1	141	0.007

Table 2: Pairwise and three-way common failure cases among Llama-3.3, Llama-4-Scout, and Llama-4-Maverick across datasets. Overlap Ratio is computed as the number of questions all three models failed on divided by the union of all distinct failures.