

Improving Language Model Self-Correction with Meta-Feedback

Anonymous ACL submission

Abstract

Large language models (LLMs) are capable of self-correct their responses by generating feedback and refining the initial output. However, their performance may sometimes decline following self-correction, either because the feedback contains errors or because they unnecessarily attempt to refine an already accurate response. To address these limitations, we investigate whether LLMs can generate *meta-feedback* that pinpoints errors in the feedback rather than the response. While the ability of LLMs to generate self-feedback has been well-researched, their potential to provide constructive meta-feedback remains under-explored. We design a novel self-correction prompting framework, **Feedback-on-Feedback (FoF)**, which leverages meta-feedback to improve the feedback before refining the response. Our framework first samples multiple feedbacks for the initial response, and prompts the LLM to generate a meta-feedback that analyze the inconsistency between these feedbacks. Based on the meta-feedback, the LLM generates a refined feedback that subsequently guides the revision of the response. Our FoF framework uniformly outperforms competitive baselines across two base models in different sizes and three datasets spanning arithmetic reasoning, machine translation and programming, with an improvement of up to 1.68% in GSM8K task by LLaMA3-8B model.

1 Introduction

LLMs have revolutionized the field of natural language processing, demonstrating exceptional performance across various tasks such as language generation, translation, and question answering (OpenAI et al., 2024). Despite their remarkable capabilities, LLMs often struggle with producing consistently accurate, coherent, and contextually relevant responses (Madaan et al., 2023; Chen et al., 2023; Welleck et al., 2022). One critical area of improvement is the intrinsic ability of LLMs

to identify and correct errors in their own outputs. Self-correction is the ability of language models to identify and rectify errors, inconsistencies, or shortcomings in their own generated outputs without external intervention (Pan et al., 2023; Madaan et al., 2023; Chen et al., 2023). This process typically involves the model first generating a critique that identifies the limitations of its initial response, followed by revising the response based on the self-generated critique. The critique-revise process can be iterated multiple times to progressively refine the model’s output, allowing for a more thorough and comprehensive self-correction (Madaan et al., 2023).

Previous research on self-correction in LLMs has explored various approaches, including iterative refinement based on self-feedback (Madaan et al., 2023), which is error correction through self-critique (Shinn et al., 2023; Kim et al., 2023), and improving consistency and confidence of generated answers (Li et al., 2024a,b). However, these methods often rely on external feedback, oracle labels which are often unavailable during inference. Another line of research dives into the intrinsic self-correction ability (Huang et al., 2024) of LLMs to refine the answer without access to external information and oracle label. However, (Huang et al., 2024) suggest that intrinsic self-correction harms the model performance since the model refine a correct answer into an incorrect one more frequently than the other way round. Another limitation of previous method besides replying on oracle labels—the quality of LLM-generated feedbacks can be arbitrarily bad without proper guidance or selection, leading to inferior performance (Shridhar et al., 2023; Liang et al., 2023b). This naturally leads to why we want to use meta-feedback to improve the quality of the feedback before applying them to refine the answer.

Particularly, their capacity to provide constructive feedback on their own self-feedback, known

043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083

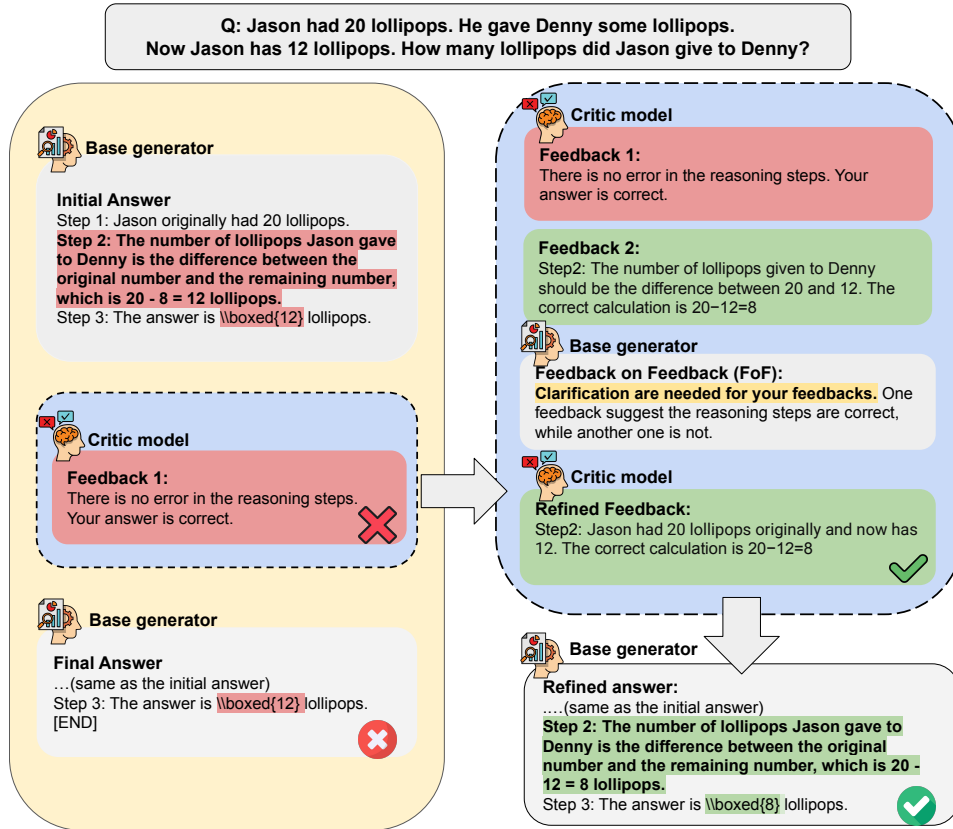


Figure 1: An illustrative example of FoF compared to the Self-refine setting (Madaan et al., 2023). In the Self-refine setting (left), the base generator generates a initial answer to the given problem, and the critic model provides feedback on the initial response. However, since the feedback is wrong, the answer model remains with the wrong answer. In the FoF setting (right), the critic model samples two feedback on the initial response simultaneously. The base generator identifies the need for clarification between the conflicting feedbacks, which requires the critic model to correct the feedback. Based on the refined feedback, the answer model refines the wrong answer and provides the correct answer. The question is from GSM8K (Cobbe et al., 2021) and all answers and feedbacks are generated by gpt-3.5-turbo-0515.

as meta-feedback, remains less explored. This vast capability raises an intriguing question: **Can the meta-feedback improve the quality of feedback generated by LLMs, and subsequently enhance the final output?**

To address these limitations, we propose a **Feedback-on-Feedback (FoF)** method. FoF addresses the limitations of existing methods by leveraging meta-feedback to improve the quality of feedback and, consequently, the accuracy of the refined answers. Unlike methods that rely on external feedback or oracle labels, the FoF framework 1) identifies the inconsistency between multiple LLM-self-generated feedbacks based on their semantic similarities, 2) generates an additional meta-feedback to analyze their inconsistency, 3) refines the feedback with meta-feedback and 4) revise the answers with the refined feedback. An example of potential

failure in intrinsic self-correction is demonstrated in Figure 1. When the first feedback indicates the initial answer is correct and the second feedback shows there is still an error in the answer, combining different stances of feedback and the clarification from meta-feedback together provides a more accurate feedback. This approach enables FoF to operate effectively in zero-shot scenarios without demonstrations, highlighting its generalizability across various tasks.

We conduct experiments on three datasets: GSM8K (arithmetic reasoning) (Cobbe et al., 2021), CSMT (machine translation) (He et al., 2020), and MBPP (programming problem-solving) (Austin et al., 2021). Our FoF method outperforms the Self-Refine (Madaan et al., 2023) baseline and standard prompt (without self-corrective feedback and answer) across all tasks and two models includ-

ing one close source model: GPT-3.5-0515 (Brown et al., 2020) and a open source model: LLaMA3-8B (Touvron et al., 2023) in zero-shot setting. For example, FoF achieves up to a 1.68% improvement in the GSM8K task using LLaMA-3-8B ompared to self-refine, a 6.37% improvement in the CSMT task using LLaMA3-8B over the standard prompt, and a 2.59% improvement in the MBPP task using LLaMA-8B over self-refine. These results demonstrate the effectiveness of the FoF method in improving the self-correction ability of LLMs across various tasks and model sizes.

Our contributions are threefold:

1. We introduce the FoF prompting method, which shows meta-feedback could examine the quality of multiple feedbacks, then improve the accuracy of answer.
2. We demonstrate improvements across multiple datasets, including GSM8K, CSMT and MBPP along GPT-3.5-0515 and LLama3-8B, demonstrating up to 1.68% improvements in the GSM8K task over self-refine.
3. We highlight the importance of selecting and integrating multiple feedback to improve answer accuracy. By addressing inconsistencies between feedback, our approach ensures more accurate and consistent self-correction.

2 Related works

Natural language feedback. The ability of LLMs to self-correct has garnered significant attention, with various approaches proposed to enhance this capability. Recent advancements leverage model natural language feedback and iterative refinement techniques (Ye et al., 2023; Madaan et al., 2023; Shinn et al., 2023; Kim et al., 2023). Approaches include evaluating alignment to guide iterative refinement (Madaan et al., 2023; Gou et al., 2024; Ye et al., 2023; Aggarwal et al., 2023; Akyurek et al., 2023; Shinn et al., 2023; Paul et al., 2024; Zheng et al., 2023; Kim et al., 2023), integrating diverse prompts and internal or external verifiers to score reasoning paths (Gero et al., 2023; Li et al., 2023c; Zelikman et al., 2022), and using multi-agent debate where LLMs interact to reach consensus (Du et al., 2023; Cohen et al., 2023; Li et al., 2023a; Liang et al., 2023a), which is orthogonal to the method FoF that includes extra agents for self-feedback.

However, some of the methods (Shinn et al., 2023; Madaan et al., 2023; Kim et al., 2023) depend on oracle labels or external feedback to determine when to stop the self-correction process. Multi-agent debate settings have also been found to be less efficient than self-consistency approaches. (Huang et al., 2024). These issues and limitations raise questions about the true intrinsic self-correcting capabilities of LLMs (Huang et al., 2024). In contrast with those methods, our approach do not involve oracle label and feedback from external verifier. Our approach completely depends on model’s intrinsic self-feedback ability.

Consistency in reasoning steps. Numerous researches showcase that the accuracy of final answer is influenced by both consistency and continuity of reasoning steps (Wang et al., 2023d; Li et al., 2023c). Consistency-relevant approaches focus on improving the consistency and confidence of LLM-generated answers. Consistency work often involves sampling then selection framework Works (Shridhar et al., 2023), self-consistency samples the reasoning steps 40 times (Wang et al., 2023d), Adaptive Consistency which reduces sampling to 7.9 times with an early stop criterion, and SCREWS (Shridhar et al., 2023) which integrates multiple selection methods like majority-voting and machine-selection. Confidence Matters (Li et al., 2024a) and Think Twice (Li et al., 2024b) sample answers, prompting the model to generate a new answer if conflicts arise between the initial responses. With all current works focus on the consistency on the reasoning steps, our method is crafted to focus on the consistency between feedback.

Feedback Quality Evaluation Recent studies have focused on evaluating the quality of feedback to enhance the self-correction ability of LLMs (Sun et al., 2024). Alignment evaluation ensures LLMs’ outputs align with human values and ethical standards by assessing biases, toxicity, and truthfulness (Hendrycks et al., 2023; Huang et al., 2023). LLMs (OpenAI et al., 2024; Fu et al., 2023; Liu et al., 2023; Ke et al., 2023; Li et al., 2023b) and humans (Saunders et al., 2022; Wang et al., 2023c) have been used as critics or annotators to evaluate and improve generated outputs. CriticBench (Lan et al., 2024) introduces a benchmark for assessing feedback and meta-feedback capabilities using both subjective GPT-4 evaluation and objective human alignment (Figure 6). Unlike CriticBench,

which relies on costly human consumption and long prompt inputs, we use a lightweight evaluation method with gpt3.5-0515 as a judge to provide explanations and scores for feedback/meta-feedback (Figure 10).

3 Method

In this section, we introduce the **Feedback-on-Feedback (FoF)** prompting method, which consists of a base generator, a critic model, and a three-step feedback refinement process that includes feedback generation, meta-feedback generation, and feedback refinement. A detailed FoF algorithm could be found in E.

3.1 Base Generator

The base generator is a large language model (LLM) that takes the question Q as input and generates an initial answer R_0 . The initial answer is generated using zero-shot chain-of-thought prompting (Brown et al., 2020; Wei et al., 2023). Given an input question Q , a generation prompt p_{gen} (Madaan et al., 2023) (see Figure 7 in the appendix), and a base generator BG , the initial answer R_0 is generated as follows:

$$R_0 = BG(p_{gen} \parallel Q) \quad (1)$$

3.2 Critic Model

The critic model is another LLM that takes the CoT which contains initial answer R_0 and the question Q as input and provides feedback on the quality of the answer. To generate the feedback, we prompt the critic model with the prompt p_{fb} shown in Figure 8 in the appendix.

The critic model samples feedback with a temperature of 0 to generate F_1 and F_2 based on its training data and the given prompt (see Figure ??):

$$F_1, F_2 = CM(p_{fb} \parallel Q, R_0) \quad (2)$$

3.3 Feedback Refinement

The feedback refinement process aims to improve the quality of the feedback and generate a refined answer. It consists of the following steps:

3.3.1 Feedback Similarity Check

We compute the semantic similarity S between the two feedback samples F_1 and F_2 using a similarity function:

$$S = \text{SemanticSimilarity}(F_1, F_2) \quad (3)$$

To determine different categories of feedback similarity (agree/disagree/need clarification), we define the thresholds θ_1 and θ_2 to categorize the feedback similarity levels. In Section 4, we will discuss our specific settings for these thresholds.

3.3.2 Feedback-on-Feedback (FoF) Generation

If the feedback samples F_1 and F_2 have low similarity, we generate FoF using the base generator BG and the prompt p_{fof} shown in Figure 2:

$$FoF = BG(p_{fof} \parallel F_1, F_2) \quad (4)$$

Need Clarification: Clarifications are needed from the sampling feedbacks, try to clarify the feedbacks

Disagree: Critic model is giving two different feedbacks, check the feedbacks and give the best feedback

Here are the two sampling feedbacks from the critic model on your previously generated reasoning step:
 {Feedback Sample 1} + {Feedback Sample 2}
 + **Need Clarification/Disagree**

Figure 2: The prompt used for generating FoF

3.3.3 Refined Feedback Generation

The refined feedback RF is generated by the critic model CM using all the history contexts including the question Q , the initial answer R_0 , the FoF, and the feedback samples F_1 and F_2 , and the prompt p_{rf} (see Figure 3):

The critic model may need clarification or disagree with you:{fof_original}
 Please give only one **refined feedback** based on the fof from the critic model.

Figure 3: The prompt used for feedback refinement

$$RF = CM(p_{rf} \parallel Q, R_0, FoF, F_1, F_2) \quad (5)$$

3.3.4 Final Answer Refinement

The final refined answer R_f is generated by the base generator BG using the question Q , the initial answer R_0 , and the refined feedback RF , along with the refined answer prompt p_{ra} (see Figure 9 in appendix):

$$R_f = BG(p_{ra} \parallel Q, R_0, RF) \quad (6)$$

The refined answer R_f is the final output of the FoF prompting method, which incorporates the feedback and refinement process to improve the accuracy and reliability of the generated answer.

4 Experiments

4.1 Models

We utilize state-of-the-art language models as the base generator and critic in our FoF framework:

We use the GPT-3.5-turbo model and LLaMA3-8B, one is open-source model and another is proprietary model. They demonstrated strong performance in various natural language tasks, including question answering and reasoning. We utilize the LLaMA3-8B model, which balances advanced capabilities with computational efficiency. Since GPT4 is considered as a strong model due to its performance on various benchmarks and its large parameter 1.76 trillion (OpenAI et al., 2024), We show the usage of GPT-4 as critic model which showcase that higher quality feedback from strong model would enhance the model accuracy,

4.2 Benchmarks

We evaluate the performance of our FoF approach on two diverse benchmarks:

Math Reasoning: We use the GSM8K dataset (Cobbe et al., 2021), comprising 8.5K grade school math word problems to assess multi-step reasoning and numerical accuracy. For our evaluation, we specifically utilize the test set from GSM8K, which contains 1,319 examples.

Machine Translation: We employ the Common-sense Machine Translation (CSMT) dataset (He et al., 2020) to evaluate translation quality, using automatic metrics BLEURT (Sellam et al., 2020) and COMET (Stewart et al., 2020). Bleurt (Sellam et al., 2020) is a learned evaluation metric based on BERT, which takes source text as reference. While COMET is a neural framework, which takes source text along with the gold answer translation as references. We take the test set from CSMT, which contains 200 examples.

Programming Problem Solving: We use the MBPP (Multiple Benchmark Programming Problems) dataset (Austin et al., 2021), featuring 974 Python problems to test the model’s ability to generate correct and efficient code. We perform experiments on the test set of MBPP, which contains

500 python problems, where each problem has 3 unit tests. We follow prior work in including the first unit test in the prompt as part of the problem description (Chen et al., 2023, 2021), and keep the remaining 2 unit tests hidden for full evaluation. We evaluate MBPP based on the pass@k metric, pass@k indicates the proportion of problems for which at least one out of k generated solutions is correct (Chen et al., 2021). We use pass@1 to evaluate FoF’s performance on MBPP.

Note that we only ran these experiments once. To ensure fairness between the Self-Refine and FoF settings, we designed FoF to take the initial answer and first round feedback from Self-Refine as part of the sampling feedback in the first round.

4.3 Baselines

CoT-Prompting: Chain-of-Thought (CoT) prompting (Wei et al., 2023) is a technique that elicits reasoning in large language models by encouraging them to generate intermediate reasoning steps before arriving at the final answer. This method enhances the model’s ability to solve complex problems by breaking down the problem-solving process into smaller steps, where the prompt contains "let’s think step by step".

Self-refine Prompting: The primary baseline method in this study is the Self-Refine method (Madaan et al., 2023). Self-refine prompting is an iterative refinement method that enables the model to generate self-feedback and use it to improve its initial outputs. Self-refine Prompting is also referred to as critical prompting by Huang et al. (Huang et al., 2024), which contains the guided sentence like "find the error in your reasoning step".

4.4 Feedback Sampling

In our experiments, we sample 2 feedback responses from the critic model with a temperature of 0.7. This temperature value ensures that the generated feedback samples are diverse (Renze and Guven, 2024; Wang et al., 2020, 2023a), allowing us to test the core idea of generating meta-feedback effectively.

4.5 Stop Condition

We follow the setup by Self-Refine (Madaan et al., 2023), where the feedback refinement process stops when it reaches the feedback round limit or when the feedback contains the phrase "there is no error".

4.6 Semantic Similarity Threshold

The semantic similarity threshold θ_1 and θ_2 , set at 0.5 and 0.8 respectively. These thresholds were chosen based on manual inspection of a few examples from the validation set. Optimizing these thresholds is resource-consuming, as it would require extensive hyperparameter tuning.

- If $0 \leq S < 0.5$, the feedback samples are considered to disagree with each other.
- If $0.5 \leq S \leq 0.8$, the feedback samples need clarification, examples could be find in 5.3.
- If $S > 0.8$, the feedback samples are considered to agree with each other.

5 Result

5.1 Main result

By comparing our approach with standard prompting method and self-refine prompting method, we can directly observe impact of meta-feedback on enhancing the feedback quality, which results in better final answer accuracy. We perform evaluations using two different large-scale models across three benchmark datasets. These evaluations span multiple types of tasks, covering arithmetic reasoning, commonsense reasoning, and programming problem solving.

As summarized in Table 1, the FoF method consistently demonstrates improvements across all benchmarks compared to the standard prompt and Self-Refine. For instance, using GPT-3.5-0515, our FoF method achieved an average accuracy of 78.71% on GSM8K, representing a 0.79% improvement over the standard prompt and a slightly increase compared to Self-Refine. Notably, for the LLaMA3-8B model, the FoF method achieve 45.17% accuracy, marking a 3.58% improvement over the standard prompt and a 1.68% increase compared to Self-Refine. The improvements from our method tend to decrease as the model capability increases, yet the decision refinement stage consistently enhances performance across all models. It is notable that the performance of GPT-3.5-0515 on GSM8K decrease after applying Self-Refine, this is aligned with the finding of Huang et al (Huang et al., 2024). In the MBPP task, we assessed the effectiveness of the FoF method using the GPT-3.5-0515 and LLaMA3-8B models. As shown in Table 1, the FoF method achieved an accuracy of

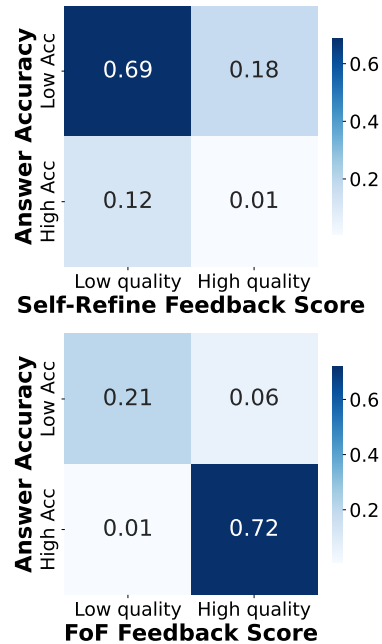


Figure 4: Heatmaps comparing feedback score correlation with answer accuracy for Self-Refine (top) and FoF (bottom) approaches. Low accuracy (0-0.5) and high accuracy (0.5-1) are shown along with low (0-5) and high (5-10) feedback quality. Self-Refine shows weak correlation, with 42% of data in the low feedback and low accuracy quadrant. FoF shows positive correlation, with 72% of data in the high feedback and high accuracy quadrant. Note the imbalance: 228/500 examples have Self-Refine feedback and 118/500 have FoF feedback, indicating fewer examples for FoF but still demonstrating that higher feedback quality leads to higher accuracy in the MBPP task.

75.27% with GPT-3.5-0515, reflecting a 3.77% improvement over the standard prompt and a 1.19% increase compared to the Self-Refine method.

In the Machine Translation Tasks, we evaluate the performance using the BLEU and COMET metrics. Our FoF approach achieves significant improvements in both BLEU and COMET scores after 4 rounds of iterative refinement. The BLEU score increases from 63.77 to 67.37, while the COMET score improves from 71.5 to 75.27. These results demonstrate the effectiveness of the FoF mechanism in enhancing the quality of the generated translations via iterative feedback and refinement rounds.

Higher Feedback Quality Leads to Better Answer Figure 4 visualizes the correlation between feedback scores and answer accuracies for both methods. The FoF heatmap reveals a strong positive correlation, with 72% of data points falling

		GSM8K		CSMT		MBPP
		Acc	Acc with Oracle Label	BLEURT	COMET	Acc
GPT-3.5-0515	+ Initial Answer	77.92	77.92	63.77	71.50	71.50
	+ Self-refine	77.42	78.79	66.07	74.08	74.08
	+ FoF	78.71	80.08	67.37	75.27	75.27
Llama-3-8B	+ Initial Answer	41.59	41.59	60.30	62.45	45.30
	+ Self-refine	43.49	44.02	63.07	66.03	49.08
	+ FoF	45.17	45.7	66.29	68.03	51.67

Table 1: The performance comparison between our FoF method and the self-refine method

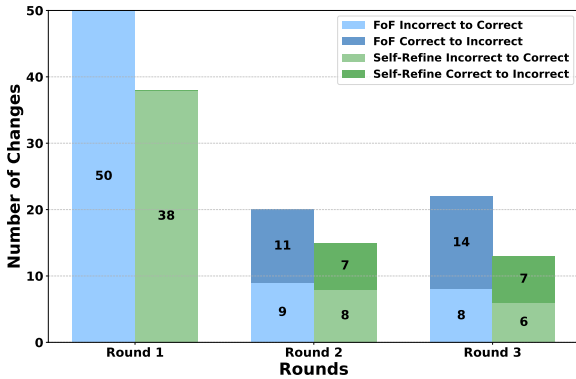


Figure 5: Comparison of FoF and Self-Refine methods (Madaan et al., 2023) on GSM8K using the GPT-3.5-turbo-0515 model. **Wrong** \rightarrow **Correct**: Number of initially incorrect answers changed to correct across 3 answer rounds. **Correct** \rightarrow **Wrong**: Number of initially correct answers changed to incorrect across 3 answer rounds.

into the high feedback score and high answer accuracy quadrant. In contrast, the Self-Refine heatmap shows a weaker correlation, with data points distributed across all quadrants.

Our analysis suggests that refining feedback through an iterative process improves the correlation between feedback scores and answer accuracy. Our analysis indicates that iterative feedback refinement enhances the correlation between feedback scores and answer accuracies. This finding is consistent with CriticBench (Lan et al., 2024), which states that higher feedback quality leads to improved accuracy in question answering.

FoF Changes More Answers Than Self-Refine

We further evaluate the changes in the answers after applying self-correction with the FoF method. The results on the GSM8K datasets by GPT-3.5-0515 model are illustrated in Figure 3. Our FoF method significantly increases the rate of Incorrect

Base Model	Critic Model	Prompt Type	# of Feedback Samples	GSM8K Accuracy
GPT-3.5	GPT-3.5	+ Standard Prompt	0	77.27
		+ Self-refine	0	79.26
		+ Self-refine	2	77.78
		+ FoF	2	79.79
GPT-3.5	GPT-4	+ Standard Prompt	0	78.24
		+ Self-refine	0	85.88
		+ Self-refine	2	85.48
		+ FoF	2	86.05

Table 2: Ablation study on the impact of critic model quality on final accuracy. Results are shown for the GSM8K dataset with GPT-3.5 as the base generator and using GPT-3.5, GPT-4 as the critic model.

\rightarrow Correct changes, demonstrating its effectiveness in enhancing answer accuracy. The pie charts in Figures 4 and 5 provide a clearer comparison of the two approaches after 4 rounds of refinement. While both methods have similar percentages of wrong-to-wrong transitions (22.5% for FoF and 22.4% for Self-Refine), Figure 5 demonstrate that FoF outperforms Self-Refine in the wrong-to-correct category (3.2% vs. 3.5%) in each round. FoF exhibits greater diversity compared to Self-Refine in terms of answer generation. This aligns with Huang et al. (Huang et al., 2024), who notes that mischanges result in self-correction failures. The improvements of FoF across tasks are due to fewer mischanges in feedback and answer rounds.

5.2 Ablation Studies

We conducted an ablation study to investigate the impact of the critic model’s quality on the final performance of our FoF approach. We compared two critic models, GPT-3.5 and GPT-4, while keeping the base generator fixed as GPT-3.5. Table 2 presents the results on the GSM8K dataset. The findings highlight the importance of the critic

model’s quality in the FoF framework. By employing a more advanced language model as the critic, the system can generate higher-quality critiques, which in turn guide the base generator to produce more accurate corrections. This finding aligns with the results presented in CriticBench (Lan et al., 2024), which evaluates the critique capabilities of various LLMs and emphasizes the importance of high-quality critiques for model improvement.

We also included a self-refine with two sampling variants to ensure a comparison using the same amount of API calls and a similar number of tokens. We sampled two feedbacks to not exceed the token limit of 4098. In this setting, self-refine generates two sampling feedbacks, and the base generator selects the one it has the most confidence in. The results indicate that LLMs lack the ability to choose the best feedback without external verification mechanisms, such as consistency checks.

5.3 Case Study

5.3.1 Failed Examples and Analysis

While our FoF method demonstrates significant improvements in self-correction capabilities, it is important to acknowledge and analyze the instances where the method did not perform as expected. In this example, the initial answer is correct but includes incorrect intermediate steps. The two feedback samples provide incorrect guidance, leading FoF to agree with the inaccurate feedback and produce a refined response that reinforces the wrong answer. This case illustrates how multiple inaccurate feedback samples can compound errors, distracting FoF from identifying the correct solution. Consequently, FoF fails to recognize the initial answer as correct and follows the misleading feedback, resulting in an incorrect final answer. More failed and successful examples are available in B.1 and B.1.4.

Gold Answer: (...detailed reasoning steps)
 $7*2=14$
Initial Answer: (...detailed reasoning steps)
 $5*(14/5) = 5*2 = 10$ (Correct initial answer with wrong middle steps)
Feedback sample 1: The solution should round 14/5 (which equals 2.8) up to 3. (Wrong feedback which focuses on fractions.)
Feedback sample 2: The reasoning steps deduce the total number of old records brought in by the 5 people as 10. (Wrong feedback which leads to the incorrect answer) Clarifications are needed.
FoF: The first feedback seems not directly related to the problem, while the second feedback claims the reasoning steps are correct. Based on the reasoning steps, I also think it is correct. (FoF agrees with the incorrect feedback)

6 Conclusion

In this study, we investigated the FoF approach for enhancing the intrinsic self-correction (Huang et al., 2024) capability of LLMs by applying meta-feedback (Lan et al., 2024) in the feedback refinement process. By applying our FoF method, LLMs achieve higher accuracy in math reasoning tasks and better quality in generation tasks and machine translation tasks. The study also highlights the importance of critic model quality in generating high-quality feedback, which has an impact on the base generator’s performance. The FoF method exhibits a positive correlation between feedback scores and answer accuracy, indicating that higher feedback quality leads to improved performance. The ablation study emphasizes the significance of the critic model’s quality in the FoF framework, with more advanced language models as critics leading to more accurate corrections by the base generator. Future work could explore the integration of various feedback sources, including human-in-the-loop (Cai et al., 2023; Mosqueira-Rey et al., 2022) with more sampling involved. With the rapid development of LLM research, there are more advanced metrics to evaluate semantic similarity, such as semantic entropy (Kuhn et al., 2023) and SimGpt (Wang et al., 2023b). In this work, we propose FoF with the basic semantic similarity metric, cosine similarity, to show some primary results in this new area.

7 Limitations

Despite the promising results of the FoF approach, our study has several limitations. Our findings contribute to the growing body of research on LLM self-correction and highlight the importance of developing structured approaches to guide LLMs in refining their outputs. However, it is crucial to acknowledge the biases and limitations of using LLMs as critics in math reasoning tasks, as discussed by Zheng et al. (Zheng et al., 2023). The limited capability of LLMs in grading math and reasoning questions could impact the effectiveness of FoF and Self-refine on datasets like GSM8K.

To mitigate these limitations and enhance the robustness of our FoF method, future research should explore incorporating techniques such as swapping positions, using few-shot examples, penalizing unnecessarily lengthy feedback, and employing different LLMs for generating initial responses and feedback. Additionally, integrating chain-of-thought prompting, human alignment, or reference-guided grading (Wei et al., 2023; Ouyang et al., 2022; Dong et al., 2023; Yoon, 2023; Hasanbeig et al., 2023) could improve the LLM’s ability to generate accurate feedback for math and reasoning questions.

References

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. 2023. [Let’s sample step by step: Adaptive-consistency for efficient reasoning and coding with llms](#). *Preprint*, arXiv:2305.11860.

Afra Feyza Akyurek, Ekin Akyurek, Ashwin Kalyan, Peter Clark, Derry Tanti Wijaya, and Niket Tandon. 2023. [RL4F: Generating natural language feedback with reinforcement learning for repairing model outputs](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7716–7733, Toronto, Canada. Association for Computational Linguistics.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. [Program synthesis with large language models](#). *Preprint*, arXiv:2108.07732.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin

Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. [Language models are few-shot learners](#). *Preprint*, arXiv:2005.14165.

Zefan Cai, Baobao Chang, and Wenjuan Han. 2023. [Human-in-the-loop through chain-of-thought](#). *Preprint*, arXiv:2306.07932.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. [Evaluating large language models trained on code](#). *Preprint*, arXiv:2107.03374.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023. [Teaching large language models to self-debug](#). *Preprint*, arXiv:2304.05128.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.

Roi Cohen, May Hamri, Mor Geva, and Amir Globerson. 2023. [Lm vs lm: Detecting factual errors via cross examination](#). *Preprint*, arXiv:2305.13281.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. [Raft: Reward ranked finetuning for generative foundation model alignment](#). *Preprint*, arXiv:2304.06767.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. [Improving factuality and reasoning in language models through multiagent debate](#). *Preprint*, arXiv:2305.14325.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. [Gptscore: Evaluate as you desire](#). *Preprint*, arXiv:2302.04166.

Zelalem Gero, Chandan Singh, Hao Cheng, Tristan Naumann, Michel Galley, Jianfeng Gao, and Hoifung Poon. 2023. [Self-verification improves few-shot clinical information extraction](#). *Preprint*, arXiv:2306.00024.

669	Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. Critic: Large language models can self-correct with tool-interactive critiquing . <i>Preprint</i> , arXiv:2305.11738.	Ruosun Li, Teerth Patel, and Xinya Du. 2023a. Prd: Peer rank and discussion improve large language model based evaluations . <i>Preprint</i> , arXiv:2307.02762.	725 726 727 728
674	Hosein Hasanbeig, Hiteshi Sharma, Leo Betthausser, Felipe Vieira Frujeri, and Ida Momennejad. 2023. Al-lure: Auditing and improving llm-based evaluation of text using iterative in-context-learning . <i>Preprint</i> , arXiv:2309.13701.	Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. AlpacaEval: An automatic evaluator of instruction-following models . https://github.com/tatsu-lab/alpaca_eval .	729 730 731 732 733
679	Jie He, Tao Wang, Deyi Xiong, and Qun Liu. 2020. The box is in the pen: Evaluating commonsense reasoning in neural machine translation . In <i>Findings of the Association for Computational Linguistics: EMNLP 2020</i> , pages 3662–3672, Online. Association for Computational Linguistics.	Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023c. Making large language models better reasoners with step-aware verifier . <i>Preprint</i> , arXiv:2206.02336.	734 735 736 737
685	Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2023. Aligning ai with shared human values . <i>Preprint</i> , arXiv:2008.02275.	Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023a. Encouraging divergent thinking in large language models through multi-agent debate . <i>Preprint</i> , arXiv:2305.19118.	738 739 740 741 742
689	Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large language models cannot self-correct reasoning yet . <i>Preprint</i> , arXiv:2310.01798.	Weixin Liang, Yuhui Zhang, Hancheng Cao, Binglu Wang, Daisy Ding, Xinyu Yang, Kailas Vodrahalli, Siyu He, Daniel Smith, Yian Yin, Daniel McFarland, and James Zou. 2023b. Can large language models provide useful feedback on research papers? a large-scale empirical analysis . <i>Preprint</i> , arXiv:2310.01783.	743 744 745 746 747 748 749
694	Yue Huang, Qihui Zhang, Philip S. Y. and Lichao Sun. 2023. Trustgpt: A benchmark for trustworthy and responsible large language models . <i>Preprint</i> , arXiv:2306.11507.	Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment . In <i>Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing</i> , pages 2511–2522, Singapore. Association for Computational Linguistics.	750 751 752 753 754 755 756
698	Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. 2023. CritiqueLLM: Scaling llm-as-critic for effective and explainable evaluation of large language model generation . <i>Preprint</i> , arXiv:2311.18702.	Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback . <i>Preprint</i> , arXiv:2303.17651.	757 758 759 760 761 762 763 764
704	Geunwoo Kim, Pierre Baldi, and Stephen McAleer. 2023. Language models can solve computer tasks . <i>Preprint</i> , arXiv:2303.17491.	Eduardo Mosqueira-Rey, Elena Hernández-Pereira, David Alonso-Ríos, José Bobes-Bascarán, and Ángel Fernández-Leal. 2022. Human-in-the-loop machine learning: a state of the art . <i>Artificial Intelligence Review</i> , 56.	765 766 767 768 769
707	Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation . <i>Preprint</i> , arXiv:2302.09664.	OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke	770 771 772 773 774 775 776 777 778 779 780 781
711	Tian Lan, Wenwei Zhang, Chen Xu, Heyan Huang, Dahua Lin, Kai Chen, and Xian ling Mao. 2024. Crit-bench: Evaluating large language models as critic . <i>Preprint</i> , arXiv:2402.13764.		
715	Loka Li, Guangyi Chen, Yusheng Su, Zhenhao Chen, Yixuan Zhang, Eric Xing, and Kun Zhang. 2024a. Confidence matters: Revisiting intrinsic self-correction capabilities of large language models . <i>Preprint</i> , arXiv:2402.12563.		
720	Moxin Li, Wenjie Wang, Fuli Feng, Fengbin Zhu, Qifan Wang, and Tat-Seng Chua. 2024b. Think twice before assure: Confidence estimation for large language models through reflection on multiple answers . <i>Preprint</i> , arXiv:2403.09972.		

782	Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully	Chelsea Voss, Carroll Wainwright, Justin Jay Wang,	846
783	Chen, Ruby Chen, Jason Chen, Mark Chen, Ben	Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei,	847
784	Chess, Chester Cho, Casey Chu, Hyung Won Chung,	CJ Weinmann, Akila Welihinda, Peter Welinder, Ji-	848
785	Dave Cummings, Jeremiah Currier, Yunxing Dai,	ayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner,	849
786	Cory Decareaux, Thomas Degry, Noah Deutsch,	Clemens Winter, Samuel Wolrich, Hannah Wong,	850
787	Damien Deville, Arka Dhar, David Dohan, Steve	Lauren Workman, Sherwin Wu, Jeff Wu, Michael	851
788	Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti,	Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qim-	852
789	Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix,	ing Yuan, Wojciech Zaremba, Rowan Zellers, Chong	853
790	Simón Posada Fishman, Juston Forte, Isabella Ful-	Zhang, Marvin Zhang, Shengjia Zhao, Tianhao	854
791	ford, Leo Gao, Elie Georges, Christian Gibson, Vik	Zheng, Juntang Zhuang, William Zhuk, and Bar-	855
792	Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-	ret Zoph. 2024. Gpt-4 technical report . <i>Preprint</i> ,	856
793	Lopes, Jonathan Gordon, Morgan Grafstein, Scott	arXiv:2303.08774.	857
794	Gray, Ryan Greene, Joshua Gross, Shixiang Shane		
795	Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris,	Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Car-	858
796	Yuchen He, Mike Heaton, Johannes Heidecke, Chris	roll L. Wainwright, Pamela Mishkin, Chong Zhang,	859
797	Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele,	Sandhini Agarwal, Katarina Slama, Alex Ray, John	860
798	Brandon Houghton, Kenny Hsu, Shengli Hu, Xin	Schulman, Jacob Hilton, Fraser Kelton, Luke Miller,	861
799	Hu, Joost Huizinga, Shantanu Jain, Shawn Jain,	Maddie Simens, Amanda Askell, Peter Welinder,	862
800	Joanne Jang, Angela Jiang, Roger Jiang, Haozhun	Paul Christiano, Jan Leike, and Ryan Lowe. 2022.	863
801	Jin, Denny Jin, Shino Jomoto, Billie Jonn, Hee-	Training language models to follow instructions with	864
802	woo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Ka-	human feedback . <i>Preprint</i> , arXiv:2203.02155.	865
803	mali, Ingmar Kanitscheider, Nitish Shirish Keskar,		
804	Tabarak Khan, Logan Kilpatrick, Jong Wook Kim,	Liangming Pan, Michael Saxon, Wenda Xu, Deepak	866
805	Christina Kim, Yongjik Kim, Jan Hendrik Kirchner,	Nathani, Xinyi Wang, and William Yang Wang. 2023.	867
806	Jamie Kiros, Matt Knight, Daniel Kokotajlo,	Automatically correcting large language models: Sur-	868
807	Łukasz Kondraciuk, Andrew Kondrich, Aris Kon-	veying the landscape of diverse self-correction strate-	869
808	stantinidis, Kyle Kosic, Gretchen Krueger, Vishal	gies . <i>Preprint</i> , arXiv:2308.03188.	870
809	Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan		
810	Leike, Jade Leung, Daniel Levy, Chak Ming Li,	Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beat-	871
811	Rachel Lim, Molly Lin, Stephanie Lin, Mateusz	riz Borges, Antoine Bosselut, Robert West, and	872
812	Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue,	Boi Faltings. 2024. Refiner: Reasoning feed-	873
813	Anna Makanju, Kim Malfacini, Sam Manning, Todor	back on intermediate representations . <i>Preprint</i> ,	874
814	Markov, Yaniv Markovski, Bianca Martin, Katie	arXiv:2304.01904.	875
815	Mayer, Andrew Mayne, Bob McGrew, Scott Mayer		
816	McKinney, Christine McLeavey, Paul McMillan,	Matthew Renze and Erhan Guven. 2024. The effect of	876
817	Jake McNeil, David Medina, Aalok Mehta, Jacob	sampling temperature on problem solving in large	877
818	Menick, Luke Metz, Andrey Mishchenko, Pamela	language models . <i>Preprint</i> , arXiv:2402.05201.	878
819	Mishkin, Vinnie Monaco, Evan Morikawa, Daniel		
820	Mossing, Tong Mu, Mira Murati, Oleg Murk, David	William Saunders, Catherine Yeh, Jeff Wu, Steven Bills,	879
821	Mély, Ashvin Nair, Reiichiro Nakano, Rajeesh Nayak,	Long Ouyang, Jonathan Ward, and Jan Leike. 2022.	880
822	Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh,	Self-critiquing models for assisting human evaluators .	881
823	Long Ouyang, Cullen O’Keefe, Jakub Pachocki, Alex	<i>Preprint</i> , arXiv:2206.05802.	882
824	Paino, Joe Palermo, Ashley Pantuliano, Giambat-		
825	tista Parascandolo, Joel Parish, Emy Parparita, Alex	Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020.	883
826	Passos, Mikhail Pavlov, Andrew Peng, Adam Perel-	BLEURT: Learning robust metrics for text genera-	884
827	man, Filipe de Avila Belbute Peres, Michael Petrov,	tion . In <i>Proceedings of the 58th Annual Meeting of</i>	885
828	Henrique Ponde de Oliveira Pinto, Michael, Poko-	<i>the Association for Computational Linguistics</i> , pages	886
829	rny, Michelle Pokrass, Vitchyr H. Pong, Tolly Pow-	7881–7892, Online. Association for Computational	887
830	ell, Alethea Power, Boris Power, Elizabeth Proehl,	Linguistics.	888
831	Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh,		
832	Cameron Raymond, Francis Real, Kendra Rimbach,	Noah Shinn, Federico Cassano, Edward Berman, Ash-	889
833	Carl Ross, Bob Rotsted, Henri Roussez, Nick Ry-	win Gopinath, Karthik Narasimhan, and Shunyu Yao.	890
834	der, Mario Saltarelli, Ted Sanders, Shibani Santurkar,	2023. Reflexion: Language agents with verbal rein-	891
835	Girish Sastry, Heather Schmidt, David Schnurr, John	forcement learning . <i>Preprint</i> , arXiv:2303.11366.	892
836	Schulman, Daniel Selsam, Kyla Sheppard, Toki		
837	Sherbakov, Jessica Shieh, Sarah Shoker, Pranav	Kumar Shridhar, Harsh Jhamtani, Hao Fang, Ben-	893
838	Shyam, Szymon Sidor, Eric Sigler, Maddie Simens,	jamin Van Durme, Jason Eisner, and Patrick Xia.	894
839	Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin	2023. Screws: A modular framework for reasoning	895
840	Sokolowsky, Yang Song, Natalie Staudacher, Fe-	with revisions . <i>Preprint</i> , arXiv:2309.13075.	896
841	lipe Petroski Such, Natalie Summers, Ilya Sutskever,		
842	Jie Tang, Nikolas Tezak, Madeleine B. Thompson,	Craig Stewart, Ricardo Rei, Catarina Farinha, and Alon	897
843	Phil Tillet, Amin Tootoonchian, Elizabeth Tseng,	Lavie. 2020. COMET - deploying a new state-of-	898
844	Preston Tuggle, Nick Turley, Jerry Tworek, Juan Fe-	the-art MT evaluation metric in production . In <i>Pro-</i>	899
845	lipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya,	<i>ceedings of the 14th Conference of the Association</i>	900
		<i>for Machine Translation in the Americas (Volume 2:</i>	901
		<i>User Track)</i> , pages 78–109, Virtual. Association for	902
		Machine Translation in the Americas.	903

904	Shichao Sun, Junlong Li, Weizhe Yuan, Ruifeng Yuan,	Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan	956
905	Wenjie Li, and Pengfei Liu. 2024. The critique of	Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin,	957
906	critique . <i>Preprint</i> , arXiv:2401.04518.	Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang,	958
907	Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier	Joseph E. Gonzalez, and Ion Stoica. 2023. Judg-	959
908	Martinet, Marie-Anne Lachaux, Timothée Lacroix,	ing llm-as-a-judge with mt-bench and chatbot arena.	960
909	Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal	<i>Preprint</i> , arXiv:2306.05685.	961
910	Azhar, Aurelien Rodriguez, Armand Joulin, Edouard		
911	Grave, and Guillaume Lample. 2023. Llama: Open		
912	and efficient foundation language models . <i>Preprint</i> ,		
913	arXiv:2302.13971.		
914	Chi Wang, Susan Xueqing Liu, and Ahmed H. Awadal-		
915	lah. 2023a. Cost-effective hyperparameter optimiza-		
916	tion for large language model generation inference .		
917	<i>Preprint</i> , arXiv:2303.04673.		
918	Pei-Hsin Wang, Sheng-Iou Hsieh, Shih-Chieh Chang,		
919	Yu-Ting Chen, Jia-Yu Pan, Wei Wei, and Da-Chang		
920	Juan. 2020. Contextual temperature for language		
921	modeling . <i>Preprint</i> , arXiv:2012.13575.		
922	Shuhe Wang, Beiming Cao, Shengyu Zhang, Xiaoya Li,		
923	Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy.		
924	2023b. Sim-gpt: Text similarity via gpt annotated		
925	data . <i>Preprint</i> , arXiv:2312.05603.		
926	Tianlu Wang, Ping Yu, Xiaoqing Ellen Tan, Sean		
927	O'Brien, Ramakanth Pasunuru, Jane Dwivedi-Yu,		
928	Olga Golovneva, Luke Zettlemoyer, Maryam Fazel-		
929	Zarandi, and Asli Celikyilmaz. 2023c. Shepherd:		
930	A critic for language model generation . <i>Preprint</i> ,		
931	arXiv:2308.04592.		
932	Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc		
933	Le, Ed Chi, Sharan Narang, Aakanksha Chowdh-		
934	ery, and Denny Zhou. 2023d. Self-consistency im-		
935	proves chain of thought reasoning in language mod-		
936	els . <i>Preprint</i> , arXiv:2203.11171.		
937	Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten		
938	Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and		
939	Denny Zhou. 2023. Chain-of-thought prompting elic-		
940	its reasoning in large language models . <i>Preprint</i> ,		
941	arXiv:2201.11903.		
942	Sean Welleck, Ximing Lu, Peter West, Faeze Brah-		
943	man, Tianxiao Shen, Daniel Khashabi, and Yejin		
944	Choi. 2022. Generating sequences by learning to		
945	self-correct . <i>Preprint</i> , arXiv:2211.00053.		
946	Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong		
947	Kim, Hyeonbin Hwang, and Minjoon Seo. 2023.		
948	Selfee: Iterative self-revising llm empowered by self-		
949	feedback generation . Blog post.		
950	Su-Youn Yoon. 2023. Short answer grading using one-		
951	shot prompting and text similarity scoring model .		
952	<i>Preprint</i> , arXiv:2305.18638.		
953	Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D.		
954	Goodman. 2022. Star: Bootstrapping reasoning with		
955	reasoning . <i>Preprint</i> , arXiv:2203.14465.		

A Experiment Prompts with demonstration

Here are the shortened versions of the experimental prompts, including instructions and demonstrations. For clarity, only 1-2 demonstrations are provided per prompt. Each demonstrated result string is highlighted for readability, though this highlighting is not part of the actual prompt. Following each prompt, a test question and a cue (e.g., "Answer:") are provided to indicate where the result string should be placed.

A.1 Initial Answer Generation

FoF on GSM8K with the Initial Answer Round Shown Below

===== system =====

Can you answer this question by writing reasoning steps and the answer:

On Monday, Walt walked 4 miles. Tuesday, he walked 6 times as many miles as he walked on Monday. His total mileage Monday through Wednesday was 41 miles. How many miles did he walk on Wednesday?

Please ensure your reasoning steps only consists of the essential steps and the final answer should be a single numerical number, in the form boxedanswer, at the end of your response.

===== assistant =====

Reasoning steps: 1. On Monday, Walt walked 4 miles. 2. On Tuesday, he walked 6 times as many miles as he walked on Monday. Therefore, he walked $4 * 6 = 24$ miles on Tuesday. 3. His total mileage Monday through Wednesday was 41 miles. So, he walked a total of $4 + 24 + x = 41$ miles, where x represents the number of miles he walked on Wednesday. 4. Solving the equation, we get $x = 41 - 4 - 24 = 13$. Therefore, Walt walked 13 miles on Wednesday. Answer: 13.

A.2 Feedback Generation

FoF on GSM8K with 2 Sampled Feedback Shown Below

===== assistant =====

There is an error in the code above because of lack of understanding of the question. What is the error? To find the error, go through semantically complete reasoning steps, and check if everything looks good

===== user =====

Here are the two sampling feedback from the critic model on your previously generated reasoning step:

The reasoning steps provided by the programmer model are incorrect. The mistake lies in step 4 where the equation is solved incorrectly. The correct equation should be $4 + 24 + x = 41$, not $4 + 24 + x = 41 - 4 - 24$. The reasoning steps provided by the programmer model are correct and logical. There are no logical issues with the steps to solve the math word problem.

Clarifications are needed from the sampling feedback, try to clarify the feedback.

A.3 FoF Generation

FoF on GSM8K with Refined Feedback Shown Below

===== assistant =====

Based on the two sampling feedback, there seems to be a disagreement about the correctness of the reasoning steps provided by the programmer model. One feedback suggests that there is a mistake in step 4, while the other feedback states that the reasoning steps are correct and logical.

To clarify the feedback, it would be helpful to understand the specific issue pointed out in step 4 by the first feedback. Additionally, it would be beneficial to know whether there are any other concerns or suggestions related to the reasoning steps provided.

===== user =====

Here is the refined feedback from the critic model.

The reasoning steps provided by the programmer model are correct and logical. There is no mistake in step 4 where the equation is solved.

A.4 Final Answer Generation

FoF on GSM8K with the Final Answer Round Shown Below

===== user =====

Can you improve the reasoning steps in natural languages according to the feedback? Please ensure your final answer should be a single numerical number, in the form boxedanswer, at the end of your response.

===== assistant =====

Reasoning steps: 1. On Monday, Walt walked 4 miles. 2. On Tuesday, he walked 6 times as many miles as he walked on Monday. Therefore, he walked $4 * 6 = 24$ miles on Tuesday. 3. His total mileage Monday through Wednesday was 41 miles. So, the combined distance he walked on Monday, Tuesday, and Wednesday is $4 + 24 + x = 41$ miles, where x represents the number of miles he walked on Wednesday. 4. Solving the equation, we get $x = 41 - 4 - 24 = 13$.

Therefore, Walt walked 13 miles on Wednesday. Answer: boxed13.

B Comparison between FoF and other existing works

B.1 Failed Examples and Analysis

While our Feedback-on-Feedback (FoF) method demonstrates significant improvements in self-correction capabilities, it is important to acknowledge and analyze the instances where the method did not perform as expected. Understanding these failures can provide valuable insights into the limitations of our approach and highlight areas for future improvement.

B.1.1 Failed Example 1: Arithmetic Reasoning

In one instance from the GSM8K dataset, the initial answer model generated the following response to a multi-step arithmetic problem:

Here are feedbacks from the critic model on your previously generated reasoning steps:

Feedback 1: The error in the reasoning steps is the assumption that the total mileage for Monday through Wednesday is given as 41 miles. The prompt does not provide any information about the total mileage for Monday through Wednesday. Therefore, we cannot assume that the equation $7x + x + \text{Wednesday's mileage} = 41$ is correct. The correct approach would be to set up the equation $7x + \text{Wednesday's mileage} = 41$, but we would need additional information or a different equation to solve for Wednesday's mileage.

Despite iterative refinement, the final answer remained incorrect due to the conflicting nature of the feedback provided. This indicates that our method struggles when presented with contradictory critiques, especially in arithmetic reasoning tasks where precision is crucial.

962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979

980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

	Iterative Answer	Automated Critique	Zero-shot	Consistency on Answer	Consistency on Feedback
Self-Refine (Madaan et al., 2023)	✓	✓			
Self-Verification (Gero et al., 2023)		✓			
Reflexion (Shinn et al., 2023)	✓	✓			
Self-Debug (Chen et al., 2023)		✓			
SelFee (Ye et al., 2023)	✓		✓		
Critical Prompt (Huang et al., 2024)	✓	✓	✓		
Think Twice (Li et al., 2024b)	✓	✓	✓	✓	
Confidence Matters (Li et al., 2024a)	✓	✓	✓	✓	
FoF (Ours)	✓	✓	✓	✓	✓

Table 3: Summary of related work on intrinsic self-correction (Huang et al., 2024) This table categorizes intrinsic self-correction methods based on five properties: Iterative Answer (the use of multiple rounds of answer generation), Automated Critique (whether the critique is generated automatically), Zero-shot (whether the method can perform without additional examples), Consistency on Answer (consistency in generating the correct answer), and Consistency on Feedback (consistency in the feedback provided).

1000	<p>Let’s assume that Walt walked x miles on Monday. On Tuesday, he walked 6 times as many miles as he walked on Monday, which is $6x$ miles. The total mileage for Monday through Wednesday is given as 41 miles. Let’s represent Wednesday’s mileage as y miles. So, the equation representing the total mileage would be $x + 6x + y = 41$. Simplifying the equation, we have $7x + y = 41$. Since we don’t have any other information, we cannot determine the exact values of x and y individually. Therefore, we cannot determine the specific number of miles Walt walked on Wednesday without additional information. Answer: Unknown.</p>	<p>This example highlights the difficulty in handling idiomatic expressions and the need for a more nuanced understanding of context during the feedback generation process.</p>	1033
1001			1034
1002			1035
1003			1036
1004			
1005			
1006			
1007			
1008			
1009			
1010			
1011			
1012			
1013			
1014			
1015			
1016	B.1.2 Failed Example 2: Machine Translation		
1017	<p>For the machine translation task using the CSMT dataset, the model was asked to translate the Chinese sentence "他想拉同村的干部一起下水去贩毒。" into English. The initial translation was as follows:</p> <p>Translation: "He wants to recruit cadres from the same village to go underwater together and engage in drug trafficking."</p> <p>The critic model’s feedback included one response indicating that the translation was accurate, while another suggested an alternative translation to better capture the idiomatic meaning. The gold answer was:</p> <p>Gold Answer: "He wants to take the cadres of the same village to sell drugs with him."</p>	<p>B.1.3 Failed Example 3: Programming Problem Solving</p> <p>In the MBPP dataset, the model was tasked with solving a programming problem that required generating a function to find the volume of a triangular prism. The initial response was:</p> <pre>def find_Volume(base, height, length): return base * height * length</pre> <p>The critic model provided the following feedback:</p> <p>Feedback: There is no error in the code provided. The function ‘find_Volume’ correctly calculates the volume by multiplying the base, height, and length together.</p> <p>However, the gold answer revealed that the function should calculate the volume of a triangular prism, which requires dividing the product by 2:</p> <p>Gold Answer: <code>def find_Volume(l,b,h):</code> <code>return ((l * b * h) / 2)</code></p> <p>The final solution did not incorporate the correct formula for calculating the volume of a triangular prism, resulting in an incorrect function. This failure illustrates the challenge of correctly interpreting and implementing mathematical formulas in code generation tasks.</p>	1037
1018			1038
1019			1039
1020			1040
1021			1041
1022			1042
1023			1043
1024			1044
1025			1045
1026			1046
1027			1047
1028			1048
1029			1049
1030			1050
1031			1051
1032	1052		

1063 B.1.4 Successful Examples and Analysis

1064 Despite multiple inaccurate feedback samples, a
 1065 robust FoF process can identify and clarify the er-
 1066 rors, leading to a correct refined feedback and final
 1067 answer.

1068 In this example, the initial answer is incorrect,
 1069 and both feedback samples provide wrong sugges-
 1070 tions. However, FoF successfully identifies the
 1071 errors in both feedback samples, then provides a
 1072 clear explanation of the correct formula, demon-
 1073 strating its ability to identify and clarify errors in
 1074 the feedback samples. This leads to a refined feed-
 1075 back that provides the correct answer, showcasing
 1076 the robustness of FoF in handling inaccurate feed-
 1077 back and guiding the model towards the correct
 1078 solution.

Successful example due to correct FoF clarifi-
 cation

Gold Answer:

```
def find_rect_num(n):
    return n * (n + 1)
```

Initial Answer:

```
def find_rect_num(n):
    return n * (n + 1) * (2 * n + 1) // 6
```

Feedback sample 1: The answer is using wrong formula, the correct formula should be $(n * (n + 1) // 2) ** 2$. (*Wrong feedback*)

Feedback sample 2: The given answer complicates the problem; the answer should be $n ** 2$. (*Wrong feedback*)

Clarifications are needed from the sampling feedback, try to clarify the feedback.

FoF: The refined answers provided in both feedback are incorrect. Feedback sample 1 uses wrong formula. Feedback sample 2 oversimplifies the problem by suggesting $n ** 2$. The correct formula for the number of rectangles that can be formed using n points on a line is $n * (n + 1)$. (*FoF correctly identifies the errors in the feedback samples*)

Refined Feedback: The correct formula for the question should be $n * (n + 1)$ (*Correct Refined feedback*)

1079 C Criticbench evaluation method

1080 D Prompts

1081 E FoF algorithm

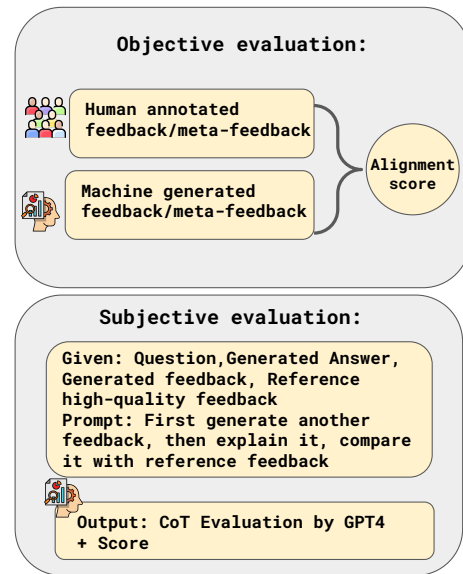


Figure 6: Overview of two evaluation methods in Criticbench

Can you answer this question by writing reasoning steps and the answer:{question}

Please ensure your **reasoning steps** only consist of the essential steps and the final answer should be a single numerical number, in the form `\\boxed{answer}`, at the end of your response.

Figure 7: The initial prompt used for generating initial answer

Below, you'll find the reasoning steps from the programmer model:{Reasoning Steps}

Assume the reasoning steps could be correct or wrong.

Please go through semantically and logically complete reasoning steps, check if everything looks good and **give your feedback**.

Figure 8: The feedback prompt used for generating F_1 and F_2

Here are the feedbacks from the critic model on your previously generated reasoning :{feedback}

Can you **improve the reasoning steps** in natural languages according to the feedbacks?

Please ensure your final answer should be a single numerical number, in the form `\\boxed{{answer}}`, at the end of your response.

Figure 9: The prompt used for feedback refinement

Algorithm 1 FoF Algorithm

Require: Question Q , Base Generator BG , Critic Model CM , Semantic Similarity Thresholds θ_1, θ_2 , Feedback Rounds

Ensure: Final Answer R_f

```
1:  $R_0 \leftarrow BG(p_{gen} \parallel Q)$  ▷ Initial generation (Eqn. 1)
2: while Round < Feedback Rounds do ▷ Iterative refinement loop
3:    $F_1, F_2 \leftarrow CM(p_{fb} \parallel Q, R_0)$  ▷ Feedback generation (Eqn. 2)
4:    $S \leftarrow \text{SemanticSimilarity}(F_1, F_2)$ 
5:   if  $S < \theta_1$  or  $\theta_1 < S < \theta_2$  then ▷ If feedback 1 and 2 disagree with each other or clarification needed
6:      $FoF \leftarrow BG(p_{fof} \parallel F_1, F_2)$  ▷ FoF generation (Eqn. 4)
7:      $RF \leftarrow CM(p_{rf} \parallel Q, R_0, FoF, F_1, F_2)$  ▷ Refine feedback (Eqn. 5)
8:   else
9:      $RF \leftarrow F_1$  ▷ Use first feedback
10:  end if
11:   $R_f \leftarrow BG(p_{fof} \parallel Q, R_0, RF)$  ▷ Refine initial answer (Eqn. 6)
12:  if RF contains "this answer is correct" then ▷ Check for stop condition
13:    return  $R_f$ 
14:  end if
15:   $R_0 \leftarrow R_f$  ▷ Update initial answer for the next iteration
16:  Round  $\leftarrow$  Round + 1 ▷ Increment round counter
17: end while
18: return  $R_f$  ▷ Return final answer after maximum rounds
```

You are an experienced code reviewer responsible for evaluating the feedback provided on a code snippet. Please adhere to the following guidelines during your review:

1. Begin by analyzing the accuracy and helpfulness of the feedback in relation to the given code. Provide constructive comments, and then affirm the feedback's quality with a score from 1 to 10, where 1 denotes the lowest quality and 10 denotes the highest quality.
2. You have access to the code snippet as well as the feedback provided. However, you will not have access to the improved code (if any) based on the feedback.
3. The feedback should correctly identify any errors or areas of improvement in the code. If the feedback is incorrect or not constructive, it should be given a low score.

Figure 10: The prompt used for scoring feedback quality