

Confidence Matters: Revisiting Intrinsic Self-Correction Capabilities of Large Language Models

Anonymous COLING 2025 submission

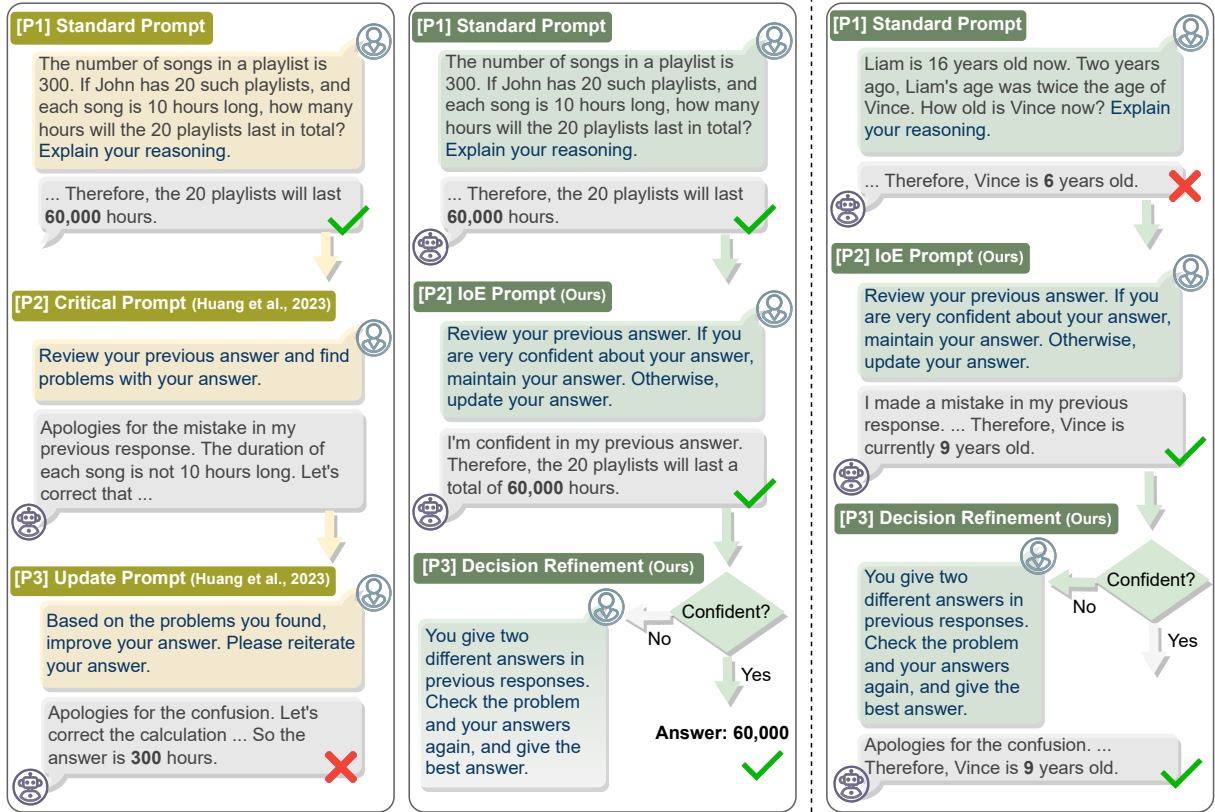


Figure 1: **Comparisons between our IoE-based Prompt and Critical Prompt.** Left: Critical Prompt (Huang et al., 2023) as baseline. Middle/Right: the proposed IoE-based Prompt. Regarding our prompts, when the answers of [P1] standard question and [P2] IoE prompt match, the final answer will be directly output, as shown in the middle. Otherwise, the decision prompt for final decision-making will execute, as the example shown in the right. All examples are generated from GSM8K (Cobbe et al., 2021) and evaluated by gpt-3.5-turbo-0613 model.

Abstract

The recent success of Large Language Models (LLMs) has catalyzed an increasing interest in their self-correction capabilities. This paper presents a comprehensive investigation into the intrinsic self-correction of LLMs, attempting to address the ongoing debate about its feasibility. Our research has identified an important latent factor, the confidence of LLMs, during the self-correction process. Overlooking this factor may cause the models to over-criticize themselves, resulting in unreliable conclusions regarding the efficacy of self-correction. We have experimentally observed that LLMs possess the

capability to initially understand the confidence in their own responses. It motivates us to develop an “If-or-Else” (IoE) prompting framework, designed to guide LLMs in assessing their own confidence, facilitating intrinsic self-corrections. We conduct extensive experiments and demonstrate that our IoE-based Prompt can achieve a consistent improvement regarding the accuracy of self-corrected responses over the initial answers. Our study not only sheds light on the underlying factors affecting self-correction in LLMs, but also introduces a practical framework that utilizes the IoE prompting principle to efficiently improve self-correction capabilities with confidence.

	One-pass Correction	Zero-shot	Task-agnostic	Confidence-based
Self-Refine (Madaan et al., 2023)	×	×	×	×
IterRefinement (Chen et al., 2023b)	×	×	×	×
SelFee (Ye et al., 2023)	×	✓	×	×
Self-Verification (Gero et al., 2023)	✓	×	×	×
Self-Defense (Helbling et al., 2023)	✓	✓	×	×
SelfCheckGPT (Manakul et al., 2023)	✓	✓	×	×
Critical Prompt (Huang et al., 2023)	✓	✓	✓	×
IoE-based Prompt (Ours)	✓	✓	✓	✓

Table 1: **The summary of related work on intrinsic self-correction.** This table categorizes intrinsic self-correction based on four properties including One-pass Correction (self-correction achieved in a single attempt, regardless of the number of prompts, and no multiple attempts required), Zero-shot (not requiring few-shot examples), Task-agnostic (applicable across various tasks), and Confidence-based (reliant on LLM confidence levels). Please refer to Appendix A for more discussions.

1 Introduction

Large language models (LLMs) trained with massive data and parameters showcase promising performance on human instruction understanding (Jin et al., 2023; Weld et al., 2022; Wu et al., 2022) and text generation (Fernandes et al., 2023; Qian et al., 2022). However, the exploration of their inherent reasoning abilities and iterative refinement capabilities is ongoing.

Among the most promising areas of exploration is the capability for “self-correction” (Pan et al., 2023b; Shinn et al., 2023; Yao et al., 2022; Madaan et al., 2023; Bai et al., 2022b; Ganguli et al., 2023; Chen et al., 2023b; Kim et al., 2023), which refers to whether LLMs can refine their responses based on their previous outputs, leveraging either external or internal feedback. Numerous studies have demonstrated effective self-correction performance through the integration of external feedback generated by the learned critic models (Paul et al., 2023; Akyürek et al., 2023; Welleck et al., 2022; Peng et al., 2023; Gao et al., 2023) or interacting with human or tools (Gou et al., 2023; Chern et al., 2023; Olausson et al., 2023b). Despite effectiveness, training extra models or interacting with the real world may result in extra costs. Thus, recent studies (Madaan et al., 2023; Chen et al., 2023a; Manakul et al., 2023; Huang et al., 2023) are beginning to explore *intrinsic self-correction*, noted for its cost efficiency. Table 1 summarizes and compares these related studies, and more discussions are in Appendix A. The practicality of *intrinsic self-correction* remains a topic of debate. For example, Huang et al. (Huang et al., 2023) argue that LLMs are not yet capable of self-correcting reasoning, observing a decrease in model performance when prompted to find problems and revise answers by LLMs themselves.

In this paper, we revisit the scenario of *intrinsic self-correction* of LLMs, identifying the critical role of LLMs’ confidence¹ in this process. Ignoring the confidence, directly using negative prompts like “find your problem” in (Huang et al., 2023) (we named this specific prompt as Critical Prompt) may lead LLMs to doubt their initially correct responses, resulting in the alteration of these responses to incorrect ones. This phenomenon is not unexpected, as strong criticism can undermine the model’s confidence in its answers, similar to a child facing challenges from a strict teacher.

Before demonstrating how confidence aids self-correction, we need to address how to estimate the confidence of LLMs’ responses (Lin et al., 2022; Tian et al., 2023; Xiong et al., 2023; Diao et al., 2023). Widely used methods include verbalized confidence (Lin et al., 2022) and sampling-based confidence (Xiong et al., 2023). Verbalized confidence uses prompts to elicit LLMs to self-estimate their confidence, while sampling-based confidence generates multiple responses and then aggregates them to obtain confidence. However, as evaluated in (Xiong et al., 2023), verbalized confidence often results in overconfidence, and sampling-based confidence requires multiple runs. In this paper, we propose a more efficient method that estimates confidence without explicitly verbalizing it, by determining whether the LLM’s answer remains the same after reconsideration. It uses the following instruction prompt: “If you are confident about your answer, maintain your answer. Otherwise, update your answer.” The model is considered confident when the answers before and after self-assessment remain the same. We found that this self-assessment method aligns with sampling-

¹Confidence refers to the level of certainty that LLMs assign to their own responses, indicating how likely the model believes its output is correct.

based confidence on deterministic tasks and demonstrates improved performance on open-ended tasks, even though only two runs are needed.

With the estimated confidence, we further yield two insightful observations: 1) The impact of self-correction prompts varies across different confidence levels. 2) Guiding self-correction using assessed confidence levels notably enhances performance. They motivate us to introduce the If-or-Else (IoE) prompting to guide LLMs in assessing their own confidence and then improve self-correction. Figure 1 shows an example of If-or-Else question. If the LLM believes itself confident, the response remains unchanged. Otherwise, if the LLM lacks confidence, it will revise the answer.

Our findings demonstrate that incorporating confidence significantly enhances LLMs’ self-correction capabilities, as compared to (Huang et al., 2023). Additionally, we conducted a detailed investigation into mechanisms behind the effective performance of IoE-based Prompt in facilitating self-correction. In summary, our main contributions are as follows: 1) We identify that understanding confidence can benefit LLMs’ intrinsic self-correction capability. 2) Given that LLMs have the inherent ability to assess their confidence, we propose an IoE prompting principle to facilitate self-correction. 3) We conduct extensive experiments on five LLMs across six benchmarks to validate our IoE-based Prompt.

2 Observations

In this section, we present experimental observations to answer two questions: how to assess confidence efficiently and how does confidence affect intrinsic self-correction?

2.1 How to Assess Confidence Efficiently?

Different from verbalized (Lin et al., 2022) and sampling-based (Xiong et al., 2023) confidence estimation methods, we propose a self-assessment method to estimate confidence by checking whether the answer would be updated. Compared to verbalized confidence, our method is not necessary to explicitly verbalize confidence. Different from sampling-based methods which require multiple runs, our method only needs two runs. As shown in paper (Xiong et al., 2023), verbalized confidence often results in overconfidence and sampling-based methods exhibit better performance. Here, we mainly compare our method with

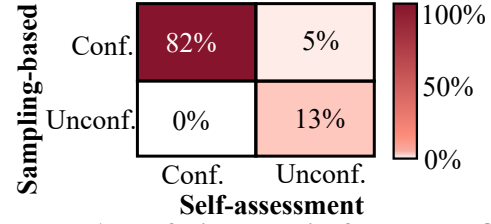


Figure 2: A confusion matrix for the confident and unconfident sets by sampling-based and self-assessment confidence estimation methods.

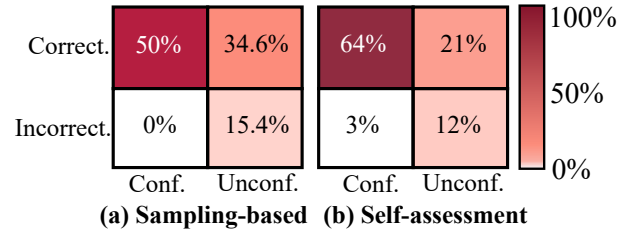


Figure 3: Confusion matrices comparing confident/unconfident and correct/incorrect classifications.

the sampling-based method, such as consistency checking, which runs the inference multiple times and checks whether those results are the same. Our self-assessment method evaluates the model with prompts like “If you are confident about your answer, maintain your answer. Otherwise, update your answer.” We conduct the experiments in deterministic and open tasks. The deterministic tasks have one unique solution, while the open tasks can possibly have multiple solutions.

Assessing confidence for deterministic tasks. We employ GPT-3.5-0613 model with the temperature set to 0.² Experiments are conducted on GSM8K dataset (Cobbe et al., 2021). As shown in the confusion matrix in Figure 2, we compared the confident and unconfident question sets assessed by the self-assessment and the sampling-based methods. The observed significant overlap in the classification of confident and unconfident questions indicates that the self-assessment can efficiently evaluate confidence levels for the deterministic tasks.

Assessing confidence for open tasks. To test the effectiveness of confidence assessment methods on open tasks, we created a task with multiple correct answers. An example question is: *Can you provide a year that is not a leap year and is divisible by 5?* The details of datasets can be found in Appendix C.1. We assess GPT-4 on the created open task due to its better consistency and the

²According to the OpenAI document, lower values of temperature will make it more focused and deterministic. However, even though we set it to 0, there are still variations.

results are presented in Figure 3. In these tasks, the non-deterministic answers lead the sampling-based method towards classifying answers as low confidence. However, it is noteworthy that LLMs demonstrated the capability to assess meaningful confidence levels in their responses. This observation is particularly significant as it highlights the inherent ability of LLMs to understand and quantify the confidence of their responses, even in the absence of clear and deterministic answers.

Takeaway #1: Self-assessment can efficiently assess the confidence of LLMs’ responses.

2.2 How does Confidence Affect Intrinsic Self-Correction?

We delve deeper into the role of confidence in the intrinsic self-correction. Initially, we evaluate the impact of self-correction methods at varying levels of confidence. Subsequently, our observations find that incorporating the estimated confidence enables LLMs to enhance their self-correction capabilities by retaining the initial answer with high confidence and reviewing more for the ones with low confidence. The experiments are based on GSM8K (Cobbe et al., 2021) and GPT-3.5-0613.

Self-correction at varying levels of confidence.

First, we applied the sampling-based method to separate the responses into two subgroups: confident and unconfident ones. Then, we compared the results using the Standard Prompt against those using the further self-corrective Critical Prompt (Huang et al., 2023). For simplicity, we combined the second and third stages of Critical Prompt shown in the left part of Figure 1, phrasing it as “*There are some problems in your previous answer. Find problems with your answer and improve your answer.*” We report the accuracy of different confidence subgroups given different prompts. The accuracy is averaged out of 5 runs. The results are presented in Table 2. According to the results, for confidence level, we found 1167 confident and 152 unconfident questions out of 100 in total. We observed that across all 1319 questions, LLM failed in intrinsic self-correction, as evidenced by a decrease in accuracy from 74.9% to 74.0% after executing the critical prompt. Intriguingly, for the subgroup of 152 unconfident questions, the accuracy increased from 25.6% to 36.1%. Despite the relatively low accuracy, LLM successfully performed self-correction in the unconfident set. Additionally, the same self-correction prompt exhibits varying performance depending on different levels of confidence. These

Prompts	Standard	Standard + Critical
Confident (1167/1319)	81.4%	76.7%
Unconfident (152/1319)	25.6%	36.1%
Total (1319/1319)	74.9%	74.0%

Table 2: **Evaluation of critical prompt’s impact across different confidence levels.** For the unconfident questions, LLM achieved better self-correction.

Repetition	Prompt	Accuracy
1 time	Standard Prompt	74.9%
	+ Self-assessment	77.1%
	+ Decision Refinement	78.5%
10 times	Standard Prompt	75.0%
	+ Sampling-based	76.1%
	+ Decision Refinement	77.2%

Table 3: **Comparison of different methods: Self-assessment vs. Sampling-based methods.**

results offer empirical evidence that confidence matters in the self-correction.

Self-correction using self-assessed confidence.

We evaluate the effectiveness of two confidence assessment methods for enhancing LLMs’ self-correction: self-assessment and sampling-based methods. For both methods, we apply the Standard Prompt first. Then, for the self-assessment method, we apply the prompt shown in Figure 1 to guide LLMs in simultaneously assessing confidence and performing self-correction. For the sampling-based method, we first assess the confidence level by consistency checking, then apply the Critical Prompt to those unconfident answers for self-correction.

Table 3 exhibits the performance comparison between two confidence assessment methods. The results show that both methods effectively facilitate self-correction, suggesting the importance of confidence in this process. Notably, our self-assessment method not only demonstrates improved efficiency but also outperforms the sampling-based method in effectiveness, showing a superior ability to assess confidence. With decision refinement (demonstrated in Section 3), the results can be improved.

Takeaway #2: Understanding confidence enhances self-correction, making this process more adaptive and preventing over-criticism.

3 Methodology

In this section, we elaborate on If-or-Else (IoE) prompting principle, which utilizes prompts to guide the LLMs in simultaneously assessing confidence levels and correcting unconfident answers.

IoE Prompt. We propose a hypothesis that LLMs

have the inherent ability to understand and assess their confidence. Contrasting with Critical Prompt (Huang et al., 2023), which directly instructs to “*find problems with your answer*”, our IoE-based prompt emphasizes the significance of confidence through the If-or-Else principle. If the LLM considers itself confident, the response should remain. Otherwise, if the LLM is unconfident, it should double-check the possible issues and revise the response. Specifically, as illustrated in Figure 1, we implement the prompt as “*If you are very confident about your answer, maintain it. Otherwise, update your answer.*” We name this prompt as *IoE Prompt*. Please note that, in the following contents, we use *IoE-based Prompt* to generally represent our proposed prompting method, while *IoE Prompt* refers to the single prompt for self-correction. Following the IoE Prompt, we trigger the model to output the final answer by saying: “*Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.*”.

It’s important to note that our single IoE prompt effectively combines the processes of confidence assessment, response reviewing, and potential updates. This method requires only a single inference, making it more efficient than the two-stage update process described in (Huang et al., 2023), where a *critical prompt* firstly identifies problems, and then an *update prompt* follows to revise the answers.

Decision Refinement. Furthermore, we introduce a decision refinement stage to further enhance results when the answer after IoE Prompt differs from the initial response. This stage shares the insight that answers reflecting low confidence require additional evaluation and refinement. The discrepancy in answers suggests low confidence, indicating the need for further decision refinement. Specifically, the decision refinement is conducted using the prompt: “*You have provided two different answers in previous responses. Review the problem and your answers again, and provide the best answer.*” This process is illustrated on the right side of Figure 1. The LLM, when uncertain about an initially correct answer, might change its answer to an incorrect one. In such cases, the third prompt becomes crucial, allowing for a reevaluation of the responses to yield a more reliable final answer.

4 Experiments

In this section, we evaluate the effectiveness of our method. First, we introduce the utilized mod-

Benchmark	Reasoning Type	# Questions
GSM8K	Arithmetic Reasoning	1319
SVAMP	Arithmetic Reasoning	1000
HotpotQA	Multi-hop Reasoning	100
Sports	Commonsense Reasoning	1000
LLC	Symbolic Reasoning	150
Domestic Robot	Multi-modal Reasoning	100

Table 4: Statistics of various benchmarks: types and number of questions.

els and benchmarks in Section 4.1. Then, we conduct a thorough comparison between our IoE-based Prompt and the baseline prompts in Section 4.2. Lastly, we experimentally explore the underlying mechanics of IoE Prompt in Section 4.3. More implementation details are shown in Appendix C.

4.1 Models and Benchmarks

Throughout our paper, we considered two sets of LLM families, GPT-based and Mistral-based models. We set the temperature to 0 for all models, aiming at more robust evaluations.

Models. GPT family (Achiam et al., 2023) is one of the most popular LLMs. The used models include gpt-3.5-turbo-0613, gpt-3.5-turbo-1106, and gpt-4-0613. Particularly, we employ gpt-4-vision-preview for multi-modal reasoning. For simplicity, we refer to these GPT-based models as GPT-3.5-0613, GPT-3.5-1106, GPT-4, and GPT-4V, respectively. Mistral (Jiang et al., 2024a) and Qwen (Bai et al., 2023) models are also used. We apply Mistral-Medium and Qwen1.5-72B-Chat for the main experiments. For scalability analysis, we also consider Mistral-Tiny and Mistral-Small. Refer to Appendix C for more details.

Benchmarks. We consider 6 benchmarks for different reasoning, as summarized in Table 4. **GSM8K** (Cobbe et al., 2021) is a dataset of high-quality linguistically diverse grade school math word problems created by human problem writers. We use the test set with 1319 problems. **SVAMP** (Patel et al., 2021) is another dataset for elementary-level math word problems with 1000 questions. **HotpotQA** (Yang et al., 2018) is a question-answering dataset featuring multi-hop reasoning. We follow (Shinn et al., 2023), which considers 100 questions with context. **Sports** is from BIG-Bench (bench authors, 2023) to test a general understanding of sports. The answer format is either Yes or No. This dataset contains 1000 questions. **Last Letter Concatenation (LLC)** was initially designed by (Wei et al., 2022) to concate-

		GSM8K	SVAMP	HotpotQA	Sports	LLC	Average
GPT-3.5-0613	Standard Prompt	74.9 _{0.1}	82.2 _{0.1}	51.0 _{0.4}	75.8 _{0.2}	68.0 _{0.2}	70.4 _{0.2}
	+ Self-Refine (Madaan et al., 2023)	75.0 _{0.1}	82.3 _{0.1}	51.0 _{0.4}	75.9 _{0.2}	69.3 _{0.2}	70.7 _{0.2}
	+ Critical Prompt (Huang et al., 2023)	74.1 _{0.3}	80.0 _{0.1}	47.0 _{0.4}	53.6 _{0.3}	76.0 _{0.2}	66.1 _{0.3}
	+ IoE Prompt (Ours)	77.1 _{0.3}	81.9 _{0.0}	55.0 _{0.4}	77.1 _{0.3}	74.0 _{0.2}	73.0 _{0.2}
	+ IoE Prompt + Decision (Ours)	78.5 _{0.0}	83.3 _{0.1}	53.0 _{0.6}	76.5 _{0.2}	77.3 _{0.2}	73.7 _{0.2}
GPT-3.5-1106	Standard Prompt	80.3 _{0.3}	82.9 _{0.2}	61.0 _{0.4}	74.2 _{0.2}	41.7 _{0.4}	67.9 _{0.3}
	+ Self-Refine (Madaan et al., 2023)	80.1 _{0.3}	82.9 _{0.2}	61.0 _{0.4}	74.1 _{0.2}	41.3 _{0.4}	67.9 _{0.3}
	+ Critical Prompt (Huang et al., 2023)	77.3 _{0.3}	81.5 _{0.2}	54.0 _{0.4}	68.4 _{0.2}	40.7 _{0.4}	64.4 _{0.1}
	+ IoE Prompt (Ours)	80.9 _{0.2}	83.2 _{0.0}	62.0 _{0.4}	75.7 _{0.2}	38.7 _{0.2}	68.1 _{0.3}
	+ IoE Prompt + Decision (Ours)	82.3 _{0.0}	84.2 _{0.2}	63.0 _{0.4}	74.7 _{0.1}	44.7 _{0.4}	69.8 _{0.2}
GPT-4	Standard Prompt	92.5 _{0.1}	92.8 _{0.0}	68.0 _{0.0}	80.7 _{0.1}	91.3 _{0.4}	85.1 _{0.1}
	+ Self-Refine (Madaan et al., 2023)	92.7 _{0.1}	93.1 _{0.0}	68.0 _{0.0}	81.2 _{0.1}	92.0 _{0.4}	85.2 _{0.1}
	+ Critical Prompt (Huang et al., 2023)	88.4 _{0.1}	89.5 _{0.1}	62.0 _{0.0}	82.9 _{0.1}	89.9 _{0.6}	82.5 _{0.2}
	+ IoE Prompt (Ours)	93.4 _{0.0}	93.2 _{0.1}	70.0 _{0.0}	83.1 _{0.0}	93.3 _{0.6}	86.6 _{0.1}
	+ IoE Prompt + Decision (Ours)	93.6 _{0.0}	93.1 _{0.0}	70.0 _{0.0}	83.3 _{0.0}	94.7 _{0.4}	86.9 _{0.1}
Mistral-Medium	Standard Prompt	84.8 _{0.1}	85.7 _{0.1}	67.0 _{0.0}	75.6 _{0.1}	60.7 _{0.2}	74.8 _{0.1}
	+ Self-Refine (Madaan et al., 2023)	84.5 _{0.1}	85.3 _{0.1}	67.0 _{0.0}	75.8 _{0.1}	60.9 _{0.2}	74.7 _{0.1}
	+ Critical Prompt (Huang et al., 2023)	62.5 _{0.2}	74.5 _{0.1}	65.0 _{0.0}	51.0 _{0.2}	35.4 _{0.4}	57.7 _{0.2}
	+ IoE Prompt (Ours)	85.4 _{0.1}	85.7 _{0.1}	69.0 _{0.0}	75.6 _{0.1}	61.3 _{0.2}	75.2 _{0.1}
	+ IoE Prompt + Decision (Ours)	85.6 _{0.1}	85.8 _{0.0}	69.0 _{0.0}	75.9 _{0.1}	61.3 _{0.2}	75.3 _{0.1}
Qwen1.5-72B	Standard Prompt	84.6 _{0.0}	86.3 _{0.0}	55.3 _{0.3}	81.8 _{0.0}	42.0 _{0.0}	70.0 _{0.1}
	+ Self-Refine (Madaan et al., 2023)	84.3 _{0.0}	86.7 _{0.0}	55.3 _{0.3}	82.0 _{0.0}	42.3 _{0.0}	70.0 _{0.1}
	+ Critical Prompt (Huang et al., 2023)	83.0 _{0.0}	85.4 _{0.1}	57.0 _{0.0}	47.3 _{0.1}	18.2 _{0.6}	58.2 _{0.2}
	+ IoE Prompt (Ours)	85.7 _{0.0}	87.8 _{0.1}	53.0 _{0.0}	81.3 _{0.0}	40.2 _{0.6}	69.6 _{0.1}
	+ IoE Prompt + Decision (Ours)	85.8 _{0.0}	88.1 _{0.0}	58.0 _{0.0}	83.2 _{0.6}	44.7 _{0.4}	71.9 _{0.2}

Table 5: **The accuracy (%) comparisons between our IoE-based Prompt and the baseline methods.** The mean and standard deviation (in smaller font, the same for the following tables) of 3 trials are reported on 5 different benchmarks by 5 large models. IoE Prompt + Decision denotes further using the decision refinement stage.

nate the last letters of words in a name (e.g., "Taylor Swift" → "rt"). This dataset contains 150 questions. **Domestic Robot** simulates a housekeeper robot performing tasks within a household setting with 100 questions by BenchLMM (Cai et al., 2023).

4.2 Comparison with Baseline Methods

In this section, we mainly compare our method with the Self-Refine method (Madaan et al., 2023) and Critical Prompt (Huang et al., 2023). Although Self-Refine requires few shot examples and multiple iterations, it can be generalized to multiple tasks. Critical Prompt considers almost the same setting as us, just a prompt template applied to various tasks. By comparing with them, we can directly observe the importance of confidence for LLMs. We performed comprehensive evaluations using five different LLMs across six benchmark datasets. Some successful and failed examples with the complete prompts and responses are given in Appendix D.1 and Appendix D.2, respectively.

Language Understanding and Reasoning. The main results related to language-based understanding and reasoning are summarized in Table 5. Specifically, we evaluated five models: GPT-3.5-0613, GPT-3.5-1106, GPT-4, Mistral-Medium, and

Qwen1.5-72B. The proposed IoE-based Prompt consistently demonstrates improvements across all benchmarks compared to the standard prompt, Critical Prompt, and Self-Refine for all models. Although self-refine involves multi-round refinements, it only improves the performance by a small margin. Self-Refine usually performs well in open tasks (Madaan et al., 2023). For those benchmarks with deterministic answers, the overall improvement by self-refine is quite limited. We provided the average performance across all five datasets for a straightforward comparison. Using the GPT-3.5-0613 model as an example, our IoE-based Prompt obtains an average accuracy of 73.7%, representing a 3.3% improvement over the standard prompt and a 7.6% increase compared to the Critical Prompt. Across different tasks, we found that IoE-based Prompt works better on math and LLC tasks. Besides, we observed that the magnitude of improvement from our IoE-based Prompt decreases as the model’s capability increases. We found that the additional decision refinement consistently enhances performance across all models. This consistent improvement supports our insight that confidence matters in intrinsic self-correction, and responses with low confidence require more examination.

	Domestic Robot	GSM8K		
	GPT-4V	Mistral-Tiny	Mistral-Small	Mistral-Medium
Standard Prompt	40.0 _{0.0}	38.4 _{0.2}	63.5 _{0.2}	84.8 _{0.1}
+ Self-Refine (Madaan et al., 2023)	40.0 _{0.0}	38.6 _{0.2}	63.8 _{0.2}	85.0 _{0.1}
+ Critical Prompt (Huang et al., 2023)	34.0 _{0.0}	31.1 _{0.3}	53.5 _{0.2}	62.5 _{0.2}
+ IoE Prompt (Ours)	42.0 _{0.0}	40.2 _{0.2}	68.6 _{0.2}	85.4 _{0.1}
+ IoE Prompt + Decision (Ours)	42.0 _{0.0}	40.8 _{0.2}	68.9 _{0.2}	85.6 _{0.1}

Table 6: **The accuracy (%) comparisons between IoE-based Prompt and baseline methods. Left:** Evaluation on Domestic Robot benchmark by GPT-4V model. **Right:** Evaluation on GSM8K by different Mistral models.

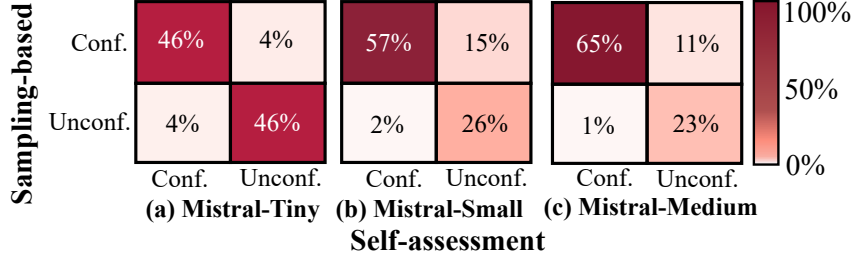


Figure 4: **Confusion matrices for the confident and unconfident sets by sampling-based and self-assessment confidence assessment methods in different mistral models.** We evaluate the models on GSM8K.

Multi-Modal Reasoning. For the evaluation of our IoE-based Prompt in multi-modal reasoning tasks, we utilized GPT-4V as the base model, conducting benchmarks on the Domestic Robot dataset. The results, detailed in Table 6, demonstrate that our method outperforms the standard prompt and the Critical Prompt baseline by +2% and +8% in accuracy, respectively. This suggests the effectiveness of the IoE principle in improving self-correction within multi-modal scenarios.

4.3 Detailed Analysis

In this section, we delve into the mechanism underlying our IoE-based Prompt through a series of experiments: 1) To analyze the scalability properties, We evaluate our IoE prompting strategy on a series of mistral models with different sizes. 2) To analyze the sensitivity and robustness of our prompt, we modify the IoE prompt in various ways. 3) We also investigate how our IoE-based Prompt can incorporate existing prompting techniques, such as the Chain-of-Thought (CoT) prompt (Wei et al., 2022) and the Rephrase-and-Respond (RaR) prompt (Deng et al., 2023). Refer to Appendix B and C for more results and analysis.

4.3.1 Scalability Analysis

Setup. To evaluate the performance of our method across models of varying sizes, we choose Mistral models and evaluate IoE strategy on a 7B model (Mistral-tiny), an 8x7B MoE model (Mistral-small), and an even larger model (Mistral-medium). We

use GSM8K for evaluation.

Results and Analysis. Figure 4 shows that the capability for self-assessment of confidence levels in LLMs is robust across a wide range of model sizes. Table 6 (Right) presents accuracy results for three models of different sizes. The Standard Prompt yields progressively higher accuracy as model size increases. However, introducing the Critical Prompt results in a significant drop in accuracy across all models. Conversely, our IoE Prompt enhances accuracy for all model sizes compared to the Standard Prompt, with the Medium model achieving a slight increase by only 0.6% (from 84.8% to 85.4%), while the Small model achieving the most significant increase by 5.1% (from 63.5% to 68.6%). When a model is robust, its accuracy with the standard prompt is already high, making further self-correction less likely. Further improvement is observed with the IoE Prompt + Decision, reaching the highest accuracy for each model size, with the Medium model attaining 85.6%. Overall, the IoE Prompt and its combination with the Decision component consistently improve performance. The results indicate that the benefits of IoE prompting are consistent across different model sizes and architectures, presenting great robustness.

4.3.2 Sensitivity and Robustness Analysis

Setup. To examine the sensitivity and robustness of the critical prompts and our prompts, we tried different rephrased critical prompts and our prompts. The two rephrased critical prompts include: “As-

sume that this answer could be either correct or incorrect. Review the answer carefully and report any serious problems you find” and “Examine your prior response and identify any issues within it”. The two rephrased IoE prompts include: “If you are very certain about your answer, maintain your answer. Otherwise, update your answer” and “If you feel strongly confident about your answer, stick with it. Otherwise, consider revising your response”. Meanwhile, we also evaluate the impacts of the degree adverb “very” and politeness “please”, and how the harsh tone “find your problem” can differ from the gentle tone “update your answer”. We evaluate on GSM8K by GPT-3.5-0613. Refer to Appendix C.4 for complete prompts.

Results and Analysis. The experimental results are presented in Table 7. The critical prompt, together with the two rephrased prompts, got worse results than the standard prompt. As for our IoE prompt, when the word “confidence” is replaced with “certain”, the accuracy remains nearly unchanged; When the phrase is substantially rephrased, the accuracy slightly increases from 77.1% to 77.6%. We can see that our IoE method is robust against all modifications, maintaining high accuracy. The expressions “confident” and “very confident” lead to similar accuracy. The use of “please” before verbs slightly improves accuracy, indicating that politeness could be helpful for self-correction. A notable decrease in accuracy was observed upon adding the phrase “find your problems” before “update your answer”. This indicates that specific terms such as “problem” or “error” might instigate doubt within the LLMs regarding its initial response, potentially leading to less reliable outcomes. Conversely, the more neutral term “update your answer” seems to be mild, consequently benefiting self-correction processes.

4.3.3 Integration with CoT and RaR

Setup. We explored the possible integration of our IoE-based Prompt with established prompting methods for self-correction, specifically for CoT and RaR. The CoT method prompts the model to process information step-by-step, yielding promising performance in many reasoning tasks. Meanwhile, RaR encourages LLMs to ask themselves better questions by rephrasing and responding.

Results and Analysis. The results are summarized in Table 8. Using CoT in the standard prompt resulted in a higher accuracy rate of 74.9%, compared to 73.4% without CoT. Intriguingly, once

Prompt	Accuracy(%)
Standard Prompt	74.9 0.1
+ Critical Prompt	74.1 0.3
+ Critical Prompt, Rephrased (1)	69.9 0.5
+ Critical Prompt, Rephrased (2)	73.2 0.3
+ IoE Prompt	77.1 0.3
+ IoE Prompt, Rephrased (1)	77.0 0.2
+ IoE Prompt, Rephrased (2)	77.6 0.3
+ IoE Prompt w. ‘very’	77.1 0.3
+ IoE Prompt w.o. ‘very’	76.8 0.2
+ IoE Prompt w. ‘please’	77.9 0.2
+ IoE Prompt w.o. ‘please’	77.1 0.3
+ IoE Prompt w. “find your problems”	75.9 0.2
+ IoE Prompt w.o. “find your problems”	77.1 0.3

Table 7: **Sensitivity and robustness analysis** on GSM8K by GPT-3.5-0613.

Prompt	Accuracy(%)
Standard Prompt w. CoT	74.9 0.1
+ IoE Prompt w. CoT	76.6 0.3
+ IoE Prompt w.o. CoT	77.1 0.3
+ IoE Prompt w. RaR	68.4 0.4
+ IoE Prompt w.o. RaR	77.1 0.3
Standard Prompt w.o. CoT	73.4 0.1
+ IoE Prompt w. CoT	77.1 0.3
+ IoE Prompt w.o. CoT	75.5 0.2
+ IoE Prompt w. RaR	65.1 0.3
+ IoE Prompt w.o. RaR	75.5 0.3

Table 8: **Integrating CoT and RaR into IoE-based Prompt** on GSM8K by GPT-3.5-0613.

the standard prompt is with CoT, IoE without CoT yielded better accuracy. This observation suggests that the redundant application of CoT may not be beneficial. Additionally, employing RaR in the IoE prompt led to a significant decrease in accuracy no matter whether we use CoT or not in the standard prompt, indicating that RaR does not contribute positively to the IoE-based Prompt.

5 Conclusion

In this paper, we conducted a comprehensive exploration of the intrinsic self-correction abilities of LLMs. Our research highlights the critical importance of LLMs’ confidence in the self-correction process. We thus introduce the If-or-Else (IoE) prompting principle, designed to guide LLMs in evaluating the confidence of their responses, encouraging further reviewing when confidence is low. Our extensive experimental analyses validate the effectiveness of the IoE Prompt, demonstrating its capability to accurately assess confidence levels and significantly enhance self-correction.

Limitations and Potential Risks

Our work has certain limitations that should be taken into account. Firstly, the experiments outlined in this paper were conducted using a restricted set of benchmarks and models. Specifically, the majority of language models utilized in our study are commercial, denoted as GPT-3.5-0613, GPT-3.5-1106, and GPT-4. Unfortunately, comprehensive documentation regarding these models, including details such as pretraining corpus, model dimensions, and inherent biases, is lacking in existing literature. Moreover, access to these models is not freely available, necessitating financial resources for research purposes. Secondly, our experimentation is exclusively confined to English datasets. Consequently, the efficacy of the current models may not translate seamlessly to other languages, thereby limiting the language generalizability of our findings.

Turning to the potential risks associated with our prompting methodology, there exists a plausible concern regarding its susceptibility to exploitation by malicious attackers. Specifically, the prompting techniques employed could potentially be utilized to manipulate the model into generating text that is toxic or harmful. Regrettably, our approach does not incorporate explicit safeguards against such misuse.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Afra Feyza Akyürek, Ekin Akyürek, Aman Madaan, Ashwin Kalyan, Peter Clark, Derry Wijaya, and Niket Tandon. 2023. RL4f: Generating natural language feedback with reinforcement learning for repairing model outputs. *arXiv preprint arXiv:2305.08844*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones,

Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022b. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Rizhao Cai, Zirui Song, Dayan Guan, Zhenhao Chen, Xing Luo, Chenyu Yi, and Alex Kot. 2023. BenchLMM: Benchmarking cross-style visual capability of large multimodal models. *arXiv preprint arXiv:2312.02896*.

Pinzhen Chen, Zhicheng Guo, Barry Haddow, and Kenneth Heafield. 2023a. Iterative translation refinement with large language models. *arXiv preprint arXiv:2306.03856*.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023b. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*.

I Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. 2023. FacTool: Factuality detection in generative ai—a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Matthew T Dearing, Yiheng Tao, Xingfu Wu, Zhiling Lan, and Valerie Taylor. 2024. Lassi: An llm-based automated self-correcting pipeline for translating parallel scientific codes. *arXiv preprint arXiv:2407.01638*.

Yihe Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. 2023. Rephrase and respond: Let large language models ask better questions for themselves. *arXiv preprint arXiv:2311.04205*.

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*.

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

Zhaopeng Feng, Yan Zhang, Hao Li, Wenqiang Liu, Jun Lang, Yang Feng, Jian Wu, and Zuozhu Liu. 2024. Improving llm-based machine translation with systematic self-correction. *arXiv preprint arXiv:2402.16379*.

642	Patrick Fernandes, Aman Madaan, Emmy Liu, António	Jie Huang, Xinyun Chen, Swaroop Mishra,	696
643	Farinhas, Pedro Henrique Martins, Amanda Bertsch,	Huaixiu Steven Zheng, Adams Wei Yu, Xiny-	697
644	José GC de Souza, Shuyan Zhou, Tongshuang Wu,	ing Song, and Denny Zhou. 2023. Large language	698
645	Graham Neubig, et al. 2023. Bridging the gap: A sur-	models cannot self-correct reasoning yet. <i>arXiv</i>	699
646	vey on integrating (human) feedback for natural lan-	<i>preprint arXiv:2310.01798</i> .	700
647	guage generation. <i>arXiv preprint arXiv:2305.00955</i> .		
648	Emily First, Markus Rabe, Talia Ringer, and Yuriy Brun.	Albert Q Jiang, Alexandre Sablayrolles, Antoine	701
649	2023. Baldur: Whole-proof generation and repair	Roux, Arthur Mensch, Blanche Savary, Chris Bam-	702
650	with large language models. In <i>Proceedings of the</i>	ford, Devendra Singh Chaplot, Diego de las Casas,	703
651	<i>31st ACM Joint European Software Engineering Con-</i>	Emma Bou Hanna, Florian Bressand, et al. 2024a.	704
652	<i>ference and Symposium on the Foundations of Soft-</i>	Mixtral of experts. <i>arXiv preprint arXiv:2401.04088</i> .	705
653	<i>ware Engineering</i> , pages 1229–1241.		
654	Yao Fu, Hao Peng, Tushar Khot, and Mirella Lapata.	Dongwei Jiang, Jingyu Zhang, Orion Weller, Nathaniel	706
655	2023. Improving language model negotiation with	Weir, Benjamin Van Durme, and Daniel Khoshabi.	707
656	self-play and in-context learning from ai feedback.	2024b. Self-[in] correct: Llms struggle with re-	708
657	<i>arXiv preprint arXiv:2305.10142</i> .	fining self-generated responses. <i>arXiv preprint</i>	709
		<i>arXiv:2404.04298</i> .	710
658	Deep Ganguli, Amanda Askell, Nicholas Schiefer,	Shuyang Jiang, Yuhao Wang, and Yu Wang. 2023. Self-	711
659	Thomas Liao, Kamilė Lukošiušė, Anna Chen, Anna	evolve: A code evolution framework via large lan-	712
660	Goldie, Azalia Mirhoseini, Catherine Olsson, Danny	guage models. <i>arXiv preprint arXiv:2306.02907</i> .	713
661	Hernandez, et al. 2023. The capacity for moral self-		
662	correction in large language models. <i>arXiv preprint</i>	Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing	714
663	<i>arXiv:2302.07459</i> .	Zong. 2023. Instance-aware prompt learning for lan-	715
		guage understanding and generation. <i>ACM Transac-</i>	716
664	Luyu Gao, Zhu Yun Dai, Panupong Pasupat, Anthony	<i>tions on Asian and Low-Resource Language Informa-</i>	717
665	Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent	<i>tion Processing</i> .	718
666	Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al.	Saurav Kadavath, Tom Conerly, Amanda Askell, Tom	719
667	2023. Rarr: Researching and revising what language	Henighan, Dawn Drain, Ethan Perez, Nicholas	720
668	models say, using language models. In <i>Proceedings</i>	Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli	721
669	<i>of the 61st Annual Meeting of the Association for</i>	Tran-Johnson, et al. 2022. Language models	722
670	<i>Computational Linguistics (Volume 1: Long Papers)</i> ,	(mostly) know what they know. <i>arXiv preprint</i>	723
671	pages 16477–16508.	<i>arXiv:2207.05221</i> .	724
672	Zelalem Gero, Chandan Singh, Hao Cheng, Tristan	Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han,	725
673	Naumann, Michel Galley, Jianfeng Gao, and Hoi-	and Rui Zhang. 2024. When can llms actually cor-	726
674	fung Poon. 2023. Self-verification improves few-	rect their own mistakes? a critical survey of self-	727
675	shot clinical information extraction. <i>arXiv preprint</i>	correction of llms. <i>arXiv preprint arXiv:2406.01297</i> .	728
676	<i>arXiv:2306.00024</i> .		
677	Amelia Glaese, Nat McAleese, Maja Trębacz, John	Geunwoo Kim, Pierre Baldi, and Stephen McAleer.	729
678	Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh,	2023. Language models can solve computer tasks.	730
679	Laura Weidinger, Martin Chadwick, Phoebe Thacker,	<i>arXiv preprint arXiv:2303.17491</i> .	731
680	et al. 2022. Improving alignment of dialogue agents	Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio	732
681	via targeted human judgements. <i>arXiv preprint</i>	Savarese, and Steven Chu Hong Hoi. 2022. Coderl:	733
682	<i>arXiv:2209.14375</i> .	Mastering code generation through pretrained models	734
		and deep reinforcement learning. <i>Advances in Neural</i>	735
683	Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong	<i>Information Processing Systems</i> , 35:21314–21328.	736
684	Shen, Yujia Yang, Nan Duan, and Weizhu Chen.	Miaoran Li, Baolin Peng, and Zhu Zhang. 2023a. Self-	737
685	2023. Critic: Large language models can self-correct	checker: Plug-and-play modules for fact-checking	738
686	with tool-interactive critiquing. <i>arXiv preprint</i>	with large language models. <i>arXiv preprint</i>	739
687	<i>arXiv:2305.11738</i> .	<i>arXiv:2305.14623</i> .	740
688	Alec Helbling, Mansi Phute, Matthew Hull, and	Ruosun Li, Teerth Patel, and Xinya Du. 2023b.	741
689	Duen Horng Chau. 2023. Llm self defense: By self	Prd: Peer rank and discussion improve large lan-	742
690	examination, llms know they are being tricked. <i>arXiv</i>	guage model based evaluations. <i>arXiv preprint</i>	743
691	<i>preprint arXiv:2308.07308</i> .	<i>arXiv:2307.02762</i> .	744
692	Hanxian Huang, Zhenghan Lin, Zixuan Wang, Xin	Stephanie Lin, Jacob Hilton, and Owain Evans. 2022.	745
693	Chen, Ke Ding, and Jishen Zhao. 2024. Towards llm-	Teaching models to express their uncertainty in	746
694	powered verilog rtl assistant: Self-verification and	words. <i>arXiv preprint arXiv:2205.14334</i> .	747
695	self-correction. <i>arXiv preprint arXiv:2406.00115</i> .		

748	Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler	Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu	802
749	Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon,	Chen. 2022. Controllable natural language gen-	803
750	Nouha Dziri, Shrimai Prabhunoye, Yiming Yang,	eration with contrastive prefixes. <i>arXiv preprint</i>	804
751	et al. 2023. Self-refine: Iterative refinement with	<i>arXiv:2202.13257</i> .	805
752	self-feedback. <i>arXiv preprint arXiv:2303.17651</i> .		
753	Potsawee Manakul, Adian Liusie, and Mark JF Gales.	David E Rumelhart, Geoffrey E Hinton, and Ronald J	806
754	2023. Selfcheckgpt: Zero-resource black-box hal-	Williams. 1986. Learning representations by back-	807
755	lucination detection for generative large language	propagating errors. <i>nature</i> , 323(6088):533–536.	808
756	models. <i>arXiv preprint arXiv:2303.08896</i> .		
757	Ninareh Mehrabi, Palash Goyal, Christophe Dupuy,	William Saunders, Catherine Yeh, Jeff Wu, Steven Bills,	809
758	Qian Hu, Shalini Ghosh, Richard Zemel, Kai-Wei	Long Ouyang, Jonathan Ward, and Jan Leike. 2022.	810
759	Chang, Aram Galstyan, and Rahul Gupta. 2023. Flirt:	Self-critiquing models for assisting human evaluators.	811
760	Feedback loop in-context red teaming. <i>arXiv preprint</i>	<i>arXiv preprint arXiv:2206.05802</i> .	812
761	<i>arXiv:2308.04265</i> .		
762	Theo X Olausson, Jeevana Priya Inala, Chenglong	Noah Shinn, Federico Cassano, Ashwin Gopinath,	813
763	Wang, Jianfeng Gao, and Armando Solar-Lezama.	Karthik R Narasimhan, and Shunyu Yao. 2023. Re-	814
764	2023a. Demystifying gpt self-repair for code genera-	flexion: Language agents with verbal reinforcement	815
765	tion. <i>arXiv preprint arXiv:2306.09896</i> .	learning. In <i>Thirty-seventh Conference on Neural</i>	816
766	Theo X Olausson, Jeevana Priya Inala, Chenglong	<i>Information Processing Systems</i> .	817
767	Wang, Jianfeng Gao, and Armando Solar-Lezama.		
768	2023b. Is self-repair a silver bullet for code genera-	Kaya Stechly, Matthew Marquez, and Subbarao Kamb-	818
769	tion. In <i>arXiv preprint arXiv:2306.09896</i> .	hampati. 2023. GPT-4 doesn’t know it’s wrong: An	819
770	Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida,	analysis of iterative prompting for reasoning prob-	820
771	Carroll Wainwright, Pamela Mishkin, Chong Zhang,	lems. <i>arXiv preprint arXiv:2310.12397</i> .	821
772	Sandhini Agarwal, Katarina Slama, Alex Ray, et al.		
773	2022. Training language models to follow instruc-	Katherine Tian, Eric Mitchell, Allan Zhou, Archit	822
774	tions with human feedback. <i>Advances in Neural</i>	Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn,	823
775	<i>Information Processing Systems</i> , 35:27730–27744.	and Christopher D Manning. 2023. Just ask for cali-	824
776	Liangming Pan, Alon Albalak, Xinyi Wang, and	bration: Strategies for eliciting calibrated confidence	825
777	William Yang Wang. 2023a. Logic-lm: Empow-	scores from language models fine-tuned with human	826
778	ering large language models with symbolic solvers	feedback. <i>arXiv preprint arXiv:2305.14975</i> .	827
779	for faithful logical reasoning. <i>arXiv preprint</i>		
780	<i>arXiv:2305.12295</i> .	Gladys Tyen, Hassan Mansoor, Peter Chen, Tony Mak,	828
781	Liangming Pan, Michael Saxon, Wenda Xu, Deepak	and Victor Cărbune. 2023. Llms cannot find rea-	829
782	Nathani, Xinyi Wang, and William Yang Wang.	soning errors, but can correct them! <i>arXiv preprint</i>	830
783	2023b. Automatically correcting large lan-	<i>arXiv:2311.08516</i> .	831
784	guage models: Surveying the landscape of di-		
785	verse self-correction strategies. <i>arXiv preprint</i>	Yifei Wang, Yuyang Wu, Zeming Wei, Stefanie Jegelka,	832
786	<i>arXiv:2308.03188</i> .	and Yisen Wang. 2024. A theoretical understand-	833
787	Arkil Patel, Satwik Bhattamishra, and Navin Goyal.	ing of self-correction through in-context alignment.	834
788	2021. Are nlp models really able to solve	<i>arXiv preprint arXiv:2405.18634</i> .	835
789	simple math word problems? <i>arXiv preprint</i>		
790	<i>arXiv:2103.07191</i> .	Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten	836
791	Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beat-	Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou,	837
792	riz Borges, Antoine Bosselut, Robert West, and	et al. 2022. Chain-of-thought prompting elicits rea-	838
793	Boi Faltings. 2023. Refiner: Reasoning feedback	soning in large language models. <i>Advances in Neural</i>	839
794	on intermediate representations. <i>arXiv preprint</i>	<i>Information Processing Systems</i> , 35:24824–24837.	840
795	<i>arXiv:2304.01904</i> .		
796	Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng,	Henry Weld, Xiaoqi Huang, Siqu Long, Josiah Poon,	841
797	Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou	and Soyeon Caren Han. 2022. A survey of joint intent	842
798	Yu, Weizhu Chen, et al. 2023. Check your facts and	detection and slot filling models in natural language	843
799	try again: Improving large language models with	understanding. <i>ACM Computing Surveys</i> , 55(8):1–	844
800	external knowledge and automated feedback. <i>arXiv</i>	38.	845
801	<i>preprint arXiv:2302.12813</i> .	Sean Welleck, Ximing Lu, Peter West, Faeze Brah-	846
		man, Tianxiao Shen, Daniel Khoshabi, and Yejin	847
		Choi. 2022. Generating sequences by learning to	848
		self-correct. <i>arXiv preprint arXiv:2211.00053</i> .	849
		Chengke Wu, Xiao Li, Yuanjun Guo, Jun Wang, Zengle	850
		Ren, Meng Wang, and Zhile Yang. 2022. Natural	851
		language processing for smart construction: Current	852
		status and future directions. <i>Automation in Construc-</i>	853
		<i>tion</i> , 134:104059.	854

855	Haoze Wu, Zihan Qiu, Zili Wang, Hang Zhao, and Jie Fu. 2024a. Gw-moe: Resolving uncertainty in moe router with global workspace theory. <i>arXiv preprint arXiv:2406.12375</i> .	Kexun Zhang, Danqing Wang, Jingtao Xia, William Yang Wang, and Lei Li. 2023c. Algo: Synthesizing algorithmic programs with generated oracle verifiers. <i>arXiv preprint arXiv:2305.14591</i> .	910
856			911
857			912
858			913
859	Yifan Wu, Siyu Yu, and Ying Li. 2024b. Log parsing with self-generated in-context learning and self-correction. <i>arXiv preprint arXiv:2406.03376</i> .		
860			
861			
862	Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-grained human feedback gives better rewards for language model training. <i>arXiv preprint arXiv:2306.01693</i> .		
863			
864			
865			
866			
867			
868	Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2023. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. <i>arXiv preprint arXiv:2306.13063</i> .		
869			
870			
871			
872			
873	Hao Yan, Saurabh Srivastava, Yintao Tai, Sida I Wang, Wen-tau Yih, and Ziyu Yao. 2023. Learning to simulate natural language feedback for interactive semantic parsing. <i>arXiv preprint arXiv:2305.08195</i> .		
874			
875			
876			
877	Kevin Yang, Yuandong Tian, Nanyun Peng, and Dan Klein. 2022. Re3: Generating longer stories with recursive reprompting and revision. <i>arXiv preprint arXiv:2210.06774</i> .		
878			
879			
880			
881	Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. <i>arXiv preprint arXiv:1809.09600</i> .		
882			
883			
884			
885			
886	Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. <i>arXiv preprint arXiv:2210.03629</i> .		
887			
888			
889			
890	Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, and Minjoon Seo. 2023. Selfee: Iterative self-revising llm empowered by self-feedback generation. <i>Blog post</i> .		
891			
892			
893			
894	Wenhao Yu, Zhihan Zhang, Zhenwen Liang, Meng Jiang, and Ashish Sabharwal. 2023a. Improving language models via plug-and-play retrieval feedback. <i>arXiv preprint arXiv:2305.14002</i> .		
895			
896			
897			
898	Xiao Yu, Baolin Peng, Michel Galley, Jianfeng Gao, and Zhou Yu. 2023b. Teaching language models to self-improve through interactive demonstrations. <i>arXiv preprint arXiv:2310.13522</i> .		
899			
900			
901			
902	Hanning Zhang, Shizhe Diao, Yong Lin, Yi R Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. 2023a. R-tuning: Teaching large language models to refuse unknown questions. <i>arXiv preprint arXiv:2311.09677</i> .		
903			
904			
905			
906			
907	Kechi Zhang, Zhuo Li, Jia Li, Ge Li, and Zhi Jin. 2023b. Self-edit: Fault-aware code editor for code generation. <i>arXiv preprint arXiv:2305.04087</i> .		
908			
909			

Part I

Appendix

Table of Contents

A Related Work	13
B More Experimental Results	15
B.1 Correction v.s. Misleading . . .	15
B.2 One-Stage v.s. Two-Stage . . .	15
C More Implementation Details	16
C.1 Data Generation for Open Tasks	16
C.2 Model Details	16
C.3 Normalization Details	16
C.4 Complete Prompts	17
D Example Visualization	18
D.1 Successful Examples	18
D.2 Failed Examples	18

A Related Work

The field of LLMs has seen significant interest in understanding and enhancing their self-correction abilities (Pan et al., 2023b; Feng et al., 2024; Jiang et al., 2024b; Kamoi et al., 2024; Huang et al., 2024; Dearing et al., 2024; Wang et al., 2024; Wu et al., 2024b). Self-correction refers to the process by which an individual identifies and rectifies their own errors or mistakes using feedback. The concept of *self-correction* finds its roots in the fundamental principles of machine learning, where neural networks can update their parameters based on prediction errors and iterative refinement (Rumelhart et al., 1986). In recent years, this area of research has been primarily divided into two categories based on the source of feedback: external and intrinsic feedback (Pan et al., 2023b).

Self-Correction from External Feedback. Numerous studies have consistently highlighted the efficacy of self-correction mechanisms facilitated by the incorporation of external feedback, which can either originate from learned critic models or interactions with the external environment. In the realm of learned critic models, diverse approaches have been explored for model training, including supervised learning or in-context learning, as evidenced by the works (Paul et al., 2023; Yu et al., 2023b; Mehrabi et al., 2023; Yan et al., 2023; First

et al., 2023; Li et al., 2023a; Yu et al., 2023a; Yang et al., 2022). Alternatively, reinforcement learning strategies have been employed for model refinement, as demonstrated in the studies by (Akyürek et al., 2023; Peng et al., 2023; Le et al., 2022; Bai et al., 2022b).

Furthermore, external feedback can be sourced from various entities within the environment, including humans, agents, tools, program executors, other language models, or oracle verifiers. Human feedback, as investigated by (Ouyang et al., 2022; Wu et al., 2023; Bai et al., 2022a; Ganguli et al., 2023; Glaese et al., 2022), has shown considerable effectiveness in enhancing model performance. Some agents (Kim et al., 2023; Shinn et al., 2023) have been proven to be helpful for self-correction. Similarly, tools such as (Gou et al., 2023; Chern et al., 2023; Pan et al., 2023a) have been leveraged to provide valuable insights for model improvement. Moreover, interactions with program executors have been explored by (Zhang et al., 2023b; Chen et al., 2023b; Jiang et al., 2023; Olausson et al., 2023a), showcasing how executable code can serve as a source of corrective feedback. Additionally, collaborations with other language models, as demonstrated by (Du et al., 2023; Li et al., 2023b; Fu et al., 2023; Saunders et al., 2022; Welleck et al., 2022), have proven beneficial in refining language generation processes. Lastly, the utilization of oracle verifiers, as illustrated by (Zhang et al., 2023c), has contributed to enhancing the accuracy and robustness of language models.

To sum up, the integration of external feedback, whether from learned critic models or interactions with the external world, has been consistently shown to significantly enhance the performance and capabilities of large language models across various tasks. Despite effectiveness, training extra models or interacting with the real world may lead to extra costs. Thus, recent studies have placed significant emphasis on investigating intrinsic self-correction mechanisms, renowned for their cost efficiency.

Self-Correction from Intrinsic Feedback. In contrast to extrinsic feedback, which relies on external sources for evaluation, intrinsic feedback draws solely from the internal knowledge and parameters of the LLM to reassess its output. A notable example of this approach is Self-Refine, introduced by (Madaan et al., 2023), which employs a scalar value as a termination criterion to iteratively refine

the model’s output, guided exclusively by supervision from a set of few-shot examples. Similarly, research by (Chen et al., 2023a) delves into the realm of iterative self-feedback processes, with a primary focus on enhancing qualitative and stylistic aspects of the output.

SelfFee (Ye et al., 2023) a LLaMA-based instruction-following model that has been fine-tuned to continuously revise its own answer until it achieves a high-quality response in a single inference. Furthermore, one work (Gero et al., 2023) explored a comprehensive mitigation framework for clinical information extraction, utilizing self-verification mechanisms. This approach harnesses the LLM’s ability to provide provenance for its own extraction process and validate its outputs, thereby enhancing the reliability of the information extracted.

LLM Self-Defense (Helbling et al., 2023) was devised as a proactive defense mechanism against adversarial prompt attacks, where the LLM autonomously evaluates induced responses to discern and filter out potentially misleading or harmful inputs. On the other hand, SelfCheckGPT (Manakul et al., 2023) presents a straightforward yet effective approach grounded in sampling methods. This technique facilitates the fact-checking of responses generated by black-box models without the need for external databases, thereby offering a resource-efficient solution to ensure the accuracy and reliability of model outputs.

Critical Prompt (Huang et al., 2023) shed light on the intrinsic limitations of current LLMs regarding self-correction capabilities. Their findings underscored significant performance improvements when employing Oracle feedback, which involves external validation of the model’s responses. However, when intrinsic self-correction mechanisms were evaluated, the results often revealed degradation in performance. This discrepancy led the authors to conclude that existing LLMs still lack the inherent capacity to rectify errors autonomously.

In contrast to Self-Refine (Madaan et al., 2023) and IterRefinement (Chen et al., 2023a), our prompting method does not necessitate multiple iterations or the use of few-shot examples. Unlike versatile applications, SelfFee (Ye et al., 2023) is designed explicitly for dialogue generation, while Self-Verification (Gero et al., 2023) is tailored for clinical information extraction with the demand of few-shot examples, Self-Defense (Helbling et al., 2023) focuses on rectifying harmful text, and Self-

CheckGPT (Manakul et al., 2023) specializes in detecting hallucinations. In our study, we adopt a similar framework to Critical Prompt (Huang et al., 2023); however, the key distinction lies in the essence of the prompts. Our prompt relies on the confidence levels of LLMs, whereas the critical prompt does not. Table 1 has summarized these related studies on intrinsic self-correction.

Model Confidence/Uncertainty. Recent work has discussed the confidence or uncertainty of large language models (LLMs) (Lin et al., 2022; Tian et al., 2023; Tyen et al., 2023; Stechly et al., 2023; Kadavath et al., 2022; Diao et al., 2023; Zhang et al., 2023a; Wu et al., 2024a). For example, Lin et al. (Lin et al., 2022) showed that GPT-3 can be fine-tuned to express calibrated uncertainty using words. Diao et al. (Diao et al., 2023) leveraged LLMs’ intrinsic uncertainty to select the most uncertain questions for annotation by chain-of-thought prompting. Furthermore, Tian et al. (Tian et al., 2023) demonstrated that for language models fine-tuned with human feedback, verbalized probabilities are better calibrated than conditional probabilities. Recently, Xiong et al. (Xiong et al., 2023) conducted an empirical evaluation of confidence elicitation in LLMs and showed that 1) direct verbalization of confidence by LLMs tends to be overconfident, and 2) sampling-based confidence, which computes consistency among multiple responses, can help mitigate this overconfidence. However, sampling-based confidence requires multiple runs, which is cost-expensive. In this paper, we propose a more efficient self-assessment confidence estimation method. Specifically, we assess the confidence by the prompts: "If you are confident about your answer, maintain your answer. Otherwise, update your answer." The model is considered confident when the answers before and after self-assessment are consistent.

We want to highlight that the focus of our paper is not about how to assess confidence, instead it mainly focuses on how to leverage confidence to improve self-correction. We acknowledge that this efficient confidence estimation method motivates our IoE-based prompt for self-correction well.

B More Experimental Results

In this section, we delve into the mechanism underlying our IoE-based Prompt through additional studies: 1) We visualize the changes in LLMs’ responses during the self-correction process. 2) We

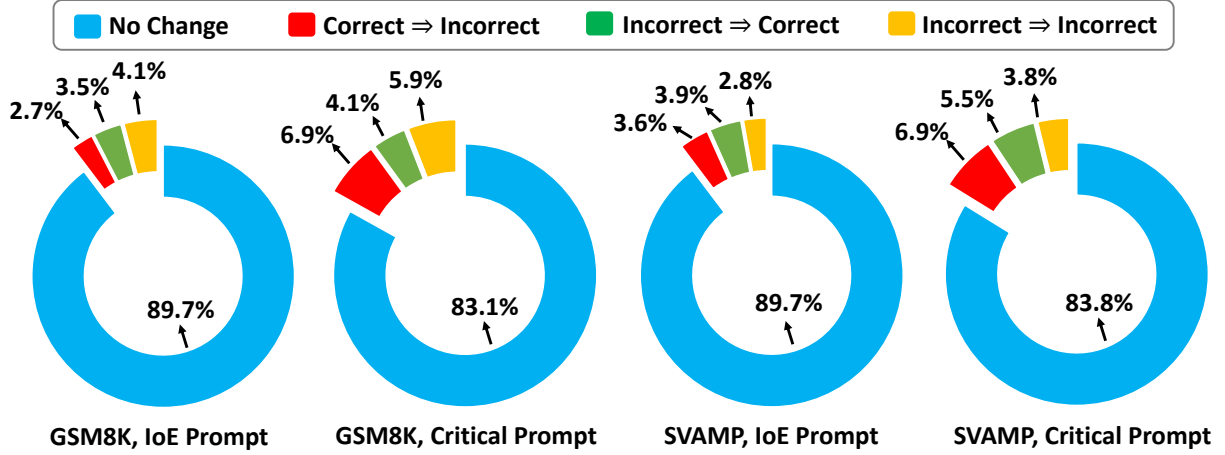


Figure 5: **Evaluation on the changes of answers after self-correction.** We compare the results of our IoE-based Prompt and Critical Prompt (Huang et al., 2023) on GSM8K and SVAMP by GPT-3.5-1106 model. *No Change*: The answer remains unchanged. *Correct ⇒ Incorrect*: A correct answer is altered to an incorrect answer. *Incorrect ⇒ Correct*: An incorrect answer is changed to a correct answer. *Incorrect ⇒ Incorrect*: An incorrect answer is changed to another incorrect answer.

Prompt	Accuracy
Standard Prompt	74.9% 0.1
+ Critical Prompt (One-Stage)	73.7% 0.3
+ Critical Prompt (Two-Stage)	74.1% 0.2
+ IoE Prompt (One-Stage)	77.1% 0.3
+ IoE Prompt (Two-Stage)	77.5% 0.3

Table 9: **Comparison between one-stage and two-stage prompts** on GSM8K by GPT-3.5-0613.

evaluate the self-correction effectiveness by contrasting one-stage prompts with two-stage prompts.

B.1 Correction v.s. Misleading

Setup. We compared the answer changes with the Critical Prompt and our IoE-based Prompt. The GPT-3.5-1106 model was employed, with GSM8K and SVAMP as benchmarks. We statistically analyzed the responses, categorizing them based on how they change from one to another.

Results and Analysis. The results are illustrated in Figure 5. A key observation when comparing the IoE-based Prompt with the Critical Prompt is a notable decrease in the correct-to-incorrect transitions using our method. This suggests that our approach effectively reduces the risk of being misled by excessive criticism. Furthermore, self-correction is successful when the correct-to-incorrect probability is lower than the incorrect-to-correct probability. This observation provides insights into why the Critical Prompt baseline fails in self-correction, whereas our method achieves success.

B.2 One-Stage v.s. Two-Stage

Setup. To enhance inference efficiency, we employ a single prompt that simultaneously handles feedback and updates. It contrasts with the two-stage process utilized in Critical Prompt: one prompt for finding problems and another for updating answers. This section aims to analyze and compare the effectiveness of one-stage prompts against two-stage prompts for self-correction.

Results and Analysis. Table 9 presents a comparative analysis of one-stage and two-stage strategies applied to both IoE-based and Critical Prompt. The results indicate that their performance is comparable across both methods, with the two-stage strategy demonstrating a slight advantage. Nonetheless, considering the trade-off between incremental improvement and additional inference overhead, we have adopted the one-stage strategy in our IoE-based Prompt for other experiments.

C More Implementation Details

C.1 Data Generation for Open Tasks

In the observational experiment discussed in section 2, we designed 100 reasoning questions. Those questions follow the patterns:

- *Can you provide a year which is not a leap year and can be divided by β ?*

Here β is an integer randomly (uniformly) selected from the interval $[2, 10]$.

C.2 Model Details

GPT-4 (Achiam et al., 2023) is one of state-of-the-art LLMs known for its advanced reasoning abilities. We utilize the gpt-4-0613 version of this model. GPT-4 also supports multi-modal reasoning, thus for those tasks, we employ GPT-4V, specifically the gpt-4-vision-preview version. For ease of reference, we will refer to these models as GPT-4 and GPT-4V, respectively.

GPT-3.5 exhibits proficiency in generating natural language and code. The standout model within the GPT-3.5 family is gpt-3.5-turbo, renowned for its exceptional capabilities and cost-effectiveness. Here we opt for two models, i.e., gpt-3.5-turbo-0613 and gpt-3.5-turbo-1106. For simplicity, we call them GPT-3.5-0613 and GPT-3.5-1106, respectively.

Mixtral (Jiang et al., 2024a) is the first open-source mixture-of-experts LLM to reach the state-of-the-art performance. The tiny model, named mistral-tiny, refers to the 7B model. The small model, named mistral-small, refers to the standard Mixtral 8x7B model. Here, we apply Mistral-Medium for the main experiments, which have a larger structure and better performance than the small one. We also use all three models to analyze the scalability properties.

Qwen 1.5 (Bai et al., 2023) is the improved version of Qwen, the large language model series developed by Alibaba Cloud. Qwen 1.5 provides a series of variations with model sizes ranging from 0.5B to 110B. In this paper, we choose the 72B version, labeled as Qwen-1.5-72B-chat (we refer to it as Qwen-1.5 72B for simplicity) in the main experiments.

C.3 Normalization Details

In our experiments, we use an API interface based on the personal laptop to evaluate the different models.

we separately run our prompts and baseline prompts, and the accuracy of the standard prompt by baseline may differ a bit from that of ours due to the variations and randomness in the results, even though we have set the temperature to 0. For fair comparison in all the tables of experimental results, we *normalize* the baseline results so that the accuracy of standard prompts by baseline and ours are equal.

How to Normalize. Assume the accuracy rates of the standard prompt and the IoE-based prompt by

our proposed method are p_1 and p_3 , respectively. Meanwhile, assume the accuracy rates of the standard prompt and the update prompt by the baseline method are \hat{p}_1 and \hat{p}_3 , respectively. In order to maintain the accuracy of the standard prompt being consistent and remove the effect of randomness, then we will normalize the accuracy of the update prompt by the baseline to $\hat{p}_3 * p_1 / \hat{p}_1$. In the end, we will record the results as follows.

Prompt	Accuracy
Standard Prompt	p_1
+ Critical Prompt	$\hat{p}_3 * p_1 / \hat{p}_1$
+ IoE Prompt	p_3

C.4 Complete Prompts

Since the standard prompt is the same for all settings, we skip that here. In this section, we will list the prompts for self-correction.

In the following prompts, some keywords are marked in red. For the corresponding prompt without the keyword, you can obtain the prompt by just removing the keywords.

C.4.1 Prompts for Sensitivity Analysis (Table 7)

Critical Prompt:

P2: *Review your previous answer and find problem with your answer.*

Critical Prompt, Rephrased (1):

P2: *Assume that this answer could be either correct or incorrect. Review the answer carefully and report any serious problems you find.*

Critical Prompt, Rephrased (2):

P2: *Examine your prior response and identify any issues within it.*

IoE Prompt:

P2: *Review your previous answer. If you are very confident about your answer, maintain your answer. Otherwise, update your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.*

IoE Prompt, Rephrased (1):

1250	P2: Review your previous answer. If you are very	C.4.2 Prompts for Ablation Study on using	1291
1251	<i>certain</i> about your answer, maintain your an-	CoT and RaR (Table 8)	1292
1252	swer. Otherwise, update your answer. Your	Standard Prompt w. CoT:	1293
1253	final answer should be put between two ##	P1: <Question> ... <i>Explain your reasoning step-</i>	1294
1254	symbols, like ## ANSWER ##, at the end of	<i>by-step</i> . Your final answer should be put be-	1295
1255	your response.	tween two ## symbols, like ## ANSWER ##,	1296
		at the end of your response.	1297
1256	IoE Prompt, Rephrased (2):	IoE Prompt w. CoT:	1298
1257	P2: Review your previous answer. <i>If you feel</i>	P2: Review your previous answer. If you are very	1299
1258	<i>strongly confident about your answer, stick</i>	confident about your answer, maintain your	1300
1259	<i>with it. Otherwise, consider revising your</i>	answer. Otherwise, update your answer. <i>Ex-</i>	1301
1260	<i>response</i> . Your final answer should be put be-	<i>plain your reasoning step-by-step</i> . Your final	1302
1261	tween two ## symbols, like ## ANSWER ##,	answer should be put between two ## sym-	1303
1262	at the end of your response.	bols, like ## ANSWER ##, at the end of your	1304
		response.	1305
1263	IoE Prompt w. ‘very’:	IoE Prompt w. RaR:	1306
1264	P2: Review your previous answer. If you are <i>very</i>	P2: (Original) "Review your previous answer. If	1307
1265	confident about your answer, maintain your	you are very confident about your answer,	1308
1266	answer. Otherwise, update your answer. Your	maintain your answer. Otherwise, update	1309
1267	final answer should be put between two ##	your answer. Your final answer should be put	1310
1268	symbols, like ## ANSWER ##, at the end of	between two ## symbols, like ## ANSWER ##,	1311
1269	your response.	at the end of your response." <i>Rephrase and</i>	1312
		<i>expand the question, and respond</i> .	1313
1270	IoE Prompt w. ‘please’:	P2: (Rephrased) Critically assess your earlier re-	1314
1271	P2: <i>Please</i> review your previous answer. If you	sponse, identifying any deficiencies or inac-	1315
1272	are very confident about your answer, <i>please</i>	curacies present in your answer. Elaborate	1316
1273	maintain your answer. Otherwise, <i>please</i> up-	on the process of self-evaluation, emphasizing	1317
1274	date your answer. Your final answer should be	the significance of recognizing and addressing	1318
1275	put between two ## symbols, like ## ANSWER	potential issues to enhance the overall qual-	1319
1276	##, at the end of your response.	ity and reliability of the information provided.	1320
		Your final answer should be put between two	1321
1277	IoE Prompt w. “find your problems”:	## symbols, like ## ANSWER ##, at the end	1322
1278	P2: Review your previous answer. If you are very	of your response.	1323
1279	confident about your answer, maintain your	C.4.3 Prompts for Ablation Study on the	1324
1280	answer. Otherwise, <i>find your problems and</i>	Number of Stages (Table 9)	1325
1281	<i>update your answer</i> . Your final answer should	Critical Prompt (One-Stage):	1326
1282	be put between two ## symbols, like ## AN-	P2: Review your previous answer and find prob-	1327
1283	SWER ##, at the end of your response.	lems with your answer. Based on the problems	1328
		you found, improve your answer. Please reiter-	1329
1284	IoE Prompt w.o. “find your problems”:	ate your answer. Your final answer should be	1330
1285	P2: Review your previous answer. If you are very	put between two ## symbols, like ## ANSWER	1331
1286	confident about your answer, maintain your	##, at the end of your response.	1332
1287	answer. Otherwise, <i>update your answer</i> . Your	Critical Prompt (Two-Stage):	1333
1288	final answer should be put between two ##	P2: Review your previous answer and find prob-	1334
1289	symbols, like ## ANSWER ##, at the end of	lems with your answer.	1335
1290	your response.		

P3: *Based on the problems you found, improve your answer. Please reiterate your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.*

IoE Prompt (One-Stage):

P2: *Review your previous answer. If you are very confident about your answer, maintain your answer. Otherwise, update your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.*

IoE Prompt (Two-Stage):

P2: *Review your previous answer. If you are very confident about your answer, maintain your answer. Otherwise, update your answer.*

P3: *Based on the problems you found if any, update your answer. Please reiterate your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.*

D Example Visualization

D.1 Successful Examples

In the main paper, we only provide a simplified example of GSM8K. Here, we provide complete evaluation examples for each benchmark where our IoE-based Prompt leads to correct answers, including the prompts and the full responses by the LLMs. Specifically, we provide the following examples:

- Figure 6: an example on GSM8K (Cobbe et al., 2021).
- Figure 7: an example on SVAMP (Patel et al., 2021).
- Figure 8: an example on Sports Understanding (Cai et al., 2023).
- Figure 9: an example on LLC (Wei et al., 2022).
- Figure 10: an example on HotpotQA (Yang et al., 2018).
- Figure 11: an example on Domestic Robot (Cai et al., 2023).

D.2 Failed Examples

In this section, we present some examples where our IoE-based Prompt failed to self-correct the reasoning. Specifically, we have the following examples:

- Figure 12: an example on GSM8K (Cobbe et al., 2021), where the model is confident.
- Figure 13: another example on GSM8K (Cobbe et al., 2021), where the model is unconfident.

Failure Analysis. We would like to analyze why our prompting method failed to achieve self-correction.

In this Figure 12, the initial answer is incorrect, when applying our IoE prompt, the answer is still incorrect and the model is confident in its incorrect answer. The reason for this failure can be that the model is not capable enough to provide a correct answer given the current knowledge.

In this Figure 13, the initial answer is correct, however, after applying our IoE-based Prompt, the model becomes unconfident and the answer is changed to an incorrect one. To dive into the initial question (“Josh decides to try flipping a house. He buys a house for \$80,000 and then puts in \$50,000 in repairs. This increased the value of the house by 150%. How much profit did he make?”), we find that: this question can be understood from multiple aspects, and some can be easily misunderstood. For example, according to “This increased the value of the house by 150%”, we can see that the value of the house has become $\$80,000 \times (1 + 150\%) = \$200,000$. However, it is very likely that one may mistake that “This increased the value of the house to its 150%”, therefore, the value of the house is $\$80,000 \times 150\% = \$120,000$, which is wrong.

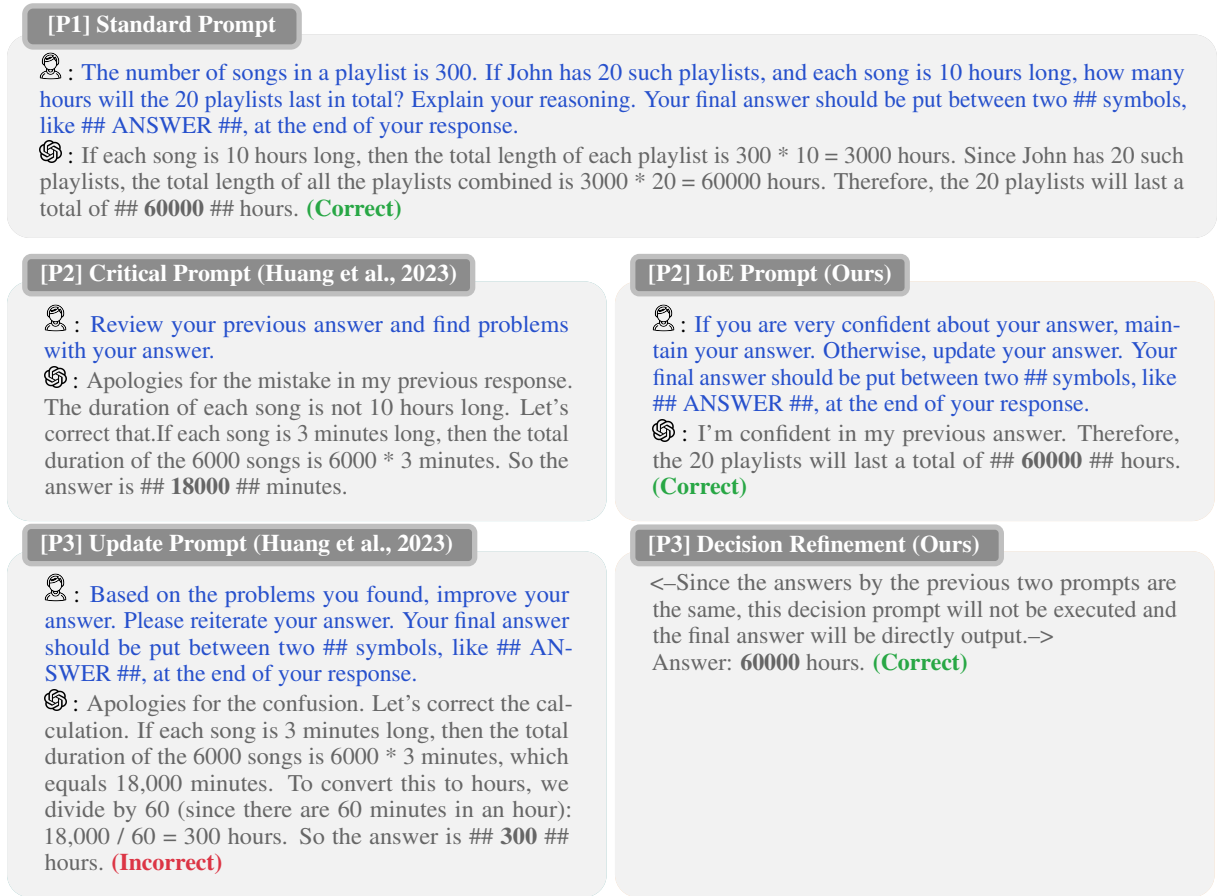





Figure 6: One example on GSM8K benchmark (Cobbe et al., 2021) evaluated with gpt-3.5-turbo-1106 model. The [P1] standard prompt is the same for all methods. **Left:** Critical Prompt as the baseline. **Right:** our proposed IoE-based Prompt.


[P1] Standard Prompt

 : The Razorback shop makes \$86 dollars off each Tshirt and \$232 off each jersey. During the Arkansas and Texas tech game they sold 254 t-shirts and 43 jerseys. How much more does a jersey cost than a t-shirt? Explain your reasoning. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.


 : To find out how much more a jersey costs than a t-shirt, we can first calculate the total revenue from t-shirts and jerseys separately. Total revenue from t-shirts = $\$86 * 254 = \$21,844$. Total revenue from jerseys = $\$232 * 43 = \$9,976$. Now we can find the difference in revenue between jerseys and t-shirts: $\$9,976 - \$21,844 = -\$11,868$. Since the revenue from t-shirts is higher than the revenue from jerseys, we can conclude that a jersey costs \$11,868 less than a t-shirt. Therefore, a jersey costs \$11,868 less than a t-shirt. The final answer is ## -11868 ##. **(Incorrect)**


[P2] Critical Prompt (Huang et al., 2023)

 : Review your previous answer and find problems with your answer.


 : Apologies for the mistake in my previous response. Let's recalculate the difference in revenue between the sales of t-shirts and jerseys. Total revenue from t-shirts = $254 \text{ t-shirts} * \$86 \text{ per t-shirt} = \$21,844$. Total revenue from jerseys = $43 \text{ jerseys} * \$232 \text{ per jersey} = \$9,976$. To find the difference in revenue, we should subtract the total revenue from t-shirts from the total revenue from jerseys: $\$9,976 - \$21,844 = -\$11,868$. Upon reviewing the calculation, I realize that I made an error in the subtraction. The correct calculation should be: $\$21,844 - \$9,976 = \$11,868$. So the correct answer is \$11,868. This means that a jersey costs ## 11,868 ## more than a t-shirt. Thank you for bringing this to my attention.


[P2] IoE Prompt (Ours)

 : If you are very confident about your answer, maintain your answer. Otherwise, update your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.

 : Apologies for the confusion. Let's recalculate. The difference in revenue between jerseys and t-shirts is: $\$232 - \$86 = \$146$. So, a jersey costs \$146 more than a t-shirt. Therefore, the final answer is ## 146 ##. **(Correct)**


[P3] Update Prompt (Huang et al., 2023)


 : Based on the problems you found, improve your answer. Please reiterate your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.

 : Apologies for the confusion in my previous responses. Let's recalculate the difference in revenue between the sales of t-shirts and jerseys. Total revenue from t-shirts = $254 \text{ t-shirts} * \$86 \text{ per t-shirt} = \$21,844$. Total revenue from jerseys = $43 \text{ jerseys} * \$232 \text{ per jersey} = \$9,976$. To find the difference in revenue, we should subtract the total revenue from jerseys from the total revenue from t-shirts: $\$21,844 - \$9,976 = \$11,868$. So the answer is \$11,868. This means that a jersey costs ## 11,868 ## more than a t-shirt. Thank you for your patience, and I appreciate your understanding.

(Incorrect)

[P3] Decision Refinement (Ours)

 : You give two different answers in previous responses. Check the problem and your answers again, and give the best answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.

 : Total revenue from t-shirts = $\$86 * 254 = \$21,844$. Total revenue from jerseys = $\$232 * 43 = \$9,976$. The difference in revenue between jerseys and t-shirts is: $\$232 - \$86 = \$146$. So, a jersey costs \$146 more than a t-shirt. Therefore, the final answer is ## 146 ##.

(Correct)

Figure 7: One example on SVAMP benchmark (Patel et al., 2021) evaluated with gpt-3.5-turbo-1106 model. The [P1] standard prompt is the same for all methods. **Left:** Critical Prompt as the baseline. **Right:** our proposed IoE-based Prompt.

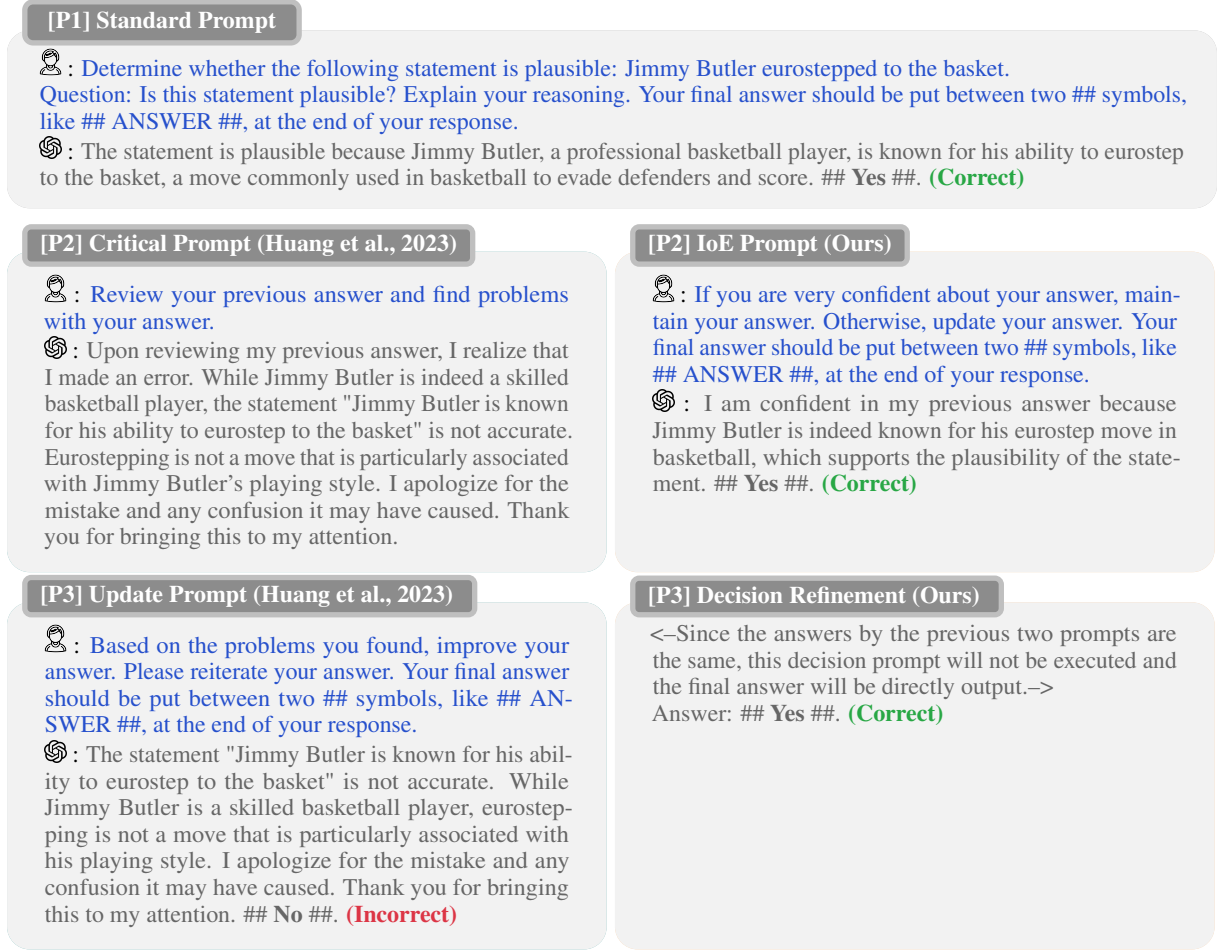


Figure 8: One example on Sports Understanding from BIG-Bench (Cai et al., 2023) evaluated with gpt-3.5-turbo-1106 model. The [P1] standard prompt is the same for all methods. **Left:** Critical Prompt as the baseline. **Right:** our proposed IoE-based Prompt.

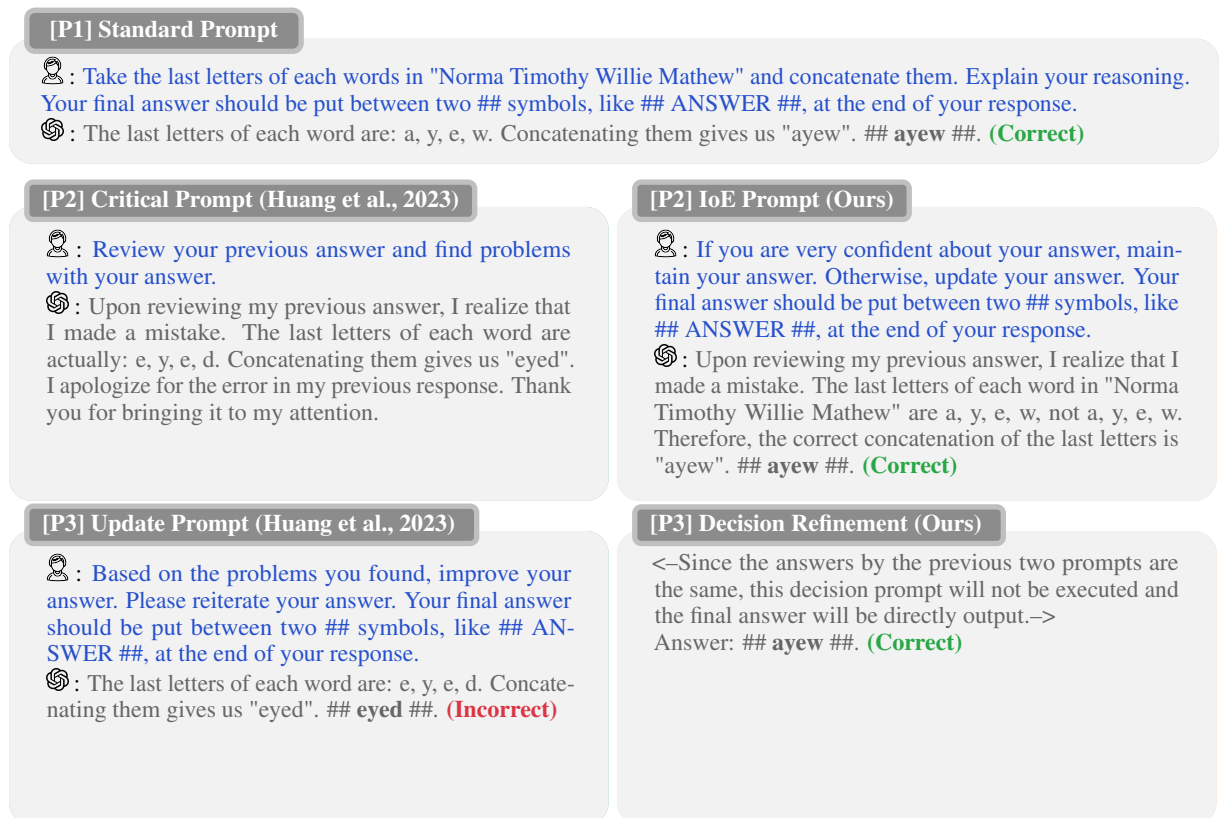



Figure 9: One example on Last Letter Concatenation (LLC) benchmark (Wei et al., 2022) evaluated with gpt-3.5-turbo-1106 model. The [P1] standard prompt is the same for all methods. **Left:** Critical Prompt as the baseline. **Right:** our proposed IoE-based Prompt.



[P1] Standard Prompt

 : **Context:** Peppes Pizza is a Norwegian pizza chain that serves American style and Italian style pizza. Peppes is the largest pizza chain in Scandinavia. The restaurant was founded by two Americans, Louis Jordan and his wife Anne from Hartford, Connecticut. The restaurant chain is part of Umoe Catering As which consists of restaurants such as Burger King, TGI Fridays, La Baguette and Cafe Opus. Peppes Pizza is one of the first restaurants that brought foreign food to Norway. 9 million pizzas are served by Peppes each year with deliveries in 11 cities in Norway. Their menu was first put online in March 1995. The servings have been described as enough for two people and that the pizza chain is a cut above the rest. Gino's East is a Chicago-based restaurant chain, notable for its deep-dish pizza (sometimes called Chicago-style pizza), and for its interior walls, which patrons have covered in graffiti and etchings. The restaurant features deep-dish pizza baked in cast-iron pans, as well as sandwiches, soups and salads. Big Mama's & Papa's Pizzeria is a pizza restaurant chain primarily located in Southern California. The chain is notable for its extremely large Giant Sicilian pizza, which is claimed to be the largest deliverable pizza in the world. Additionally, the chain gained notoriety when, during the 2014 Academy Awards, host Ellen Degeneres had Big Mama's pizzas delivered onstage. Papa John's Pizza is an American restaurant franchise company. It runs the third largest take-out and pizza delivery restaurant chain in the United States, with headquarters in Jeffersontown, Kentucky, a suburb of Louisville. Pizza 73 is a Canadian restaurant chain that offers a number of different styles of pizza, along with chicken wings. It has been operated by Pizza Pizza since 2007. Toronto-based Pizza Pizza had acquired the restaurant for a total of \$CAN70.2 million. There are 89 locations throughout Western Canada, which include the provinces of British Columbia, Alberta, and Saskatchewan. The restaurant's name originates from its original phone number: 473 . Founded by David Tougas and Guy Goodwin in 1985, Pizza 73 is headquartered in Edmonton, Alberta, Canada. Papa Gino's, Inc. is a restaurant chain based in Dedham, Massachusetts specializing in American-style pizza along with pasta, subs, salads, and a variety of appetizers. There are over 150 Papa Gino's locations in Connecticut, Maine, Massachusetts, New Hampshire, and Rhode Island. Pie Five Pizza Co. is a fast casual restaurant chain specializing in handcrafted personal pizza made in less than 5 minutes. The brand is owned by Rave Restaurant Group, which also owns Pizza Inn. As of December 2016, Pie Five has 98 restaurants in the following locations: Alabama, Arkansas, Florida, Georgia, Illinois, Indiana, Kansas, Maryland, Minnesota, Missouri, Nebraska, North Carolina, Oklahoma, Tennessee, Texas, Utah, Kentucky, Virginia and Washington, D.C. with more than 400 additional company-owned and franchise units anticipated. Gino's Pizza and Spaghetti is a restaurant chain with 40 locations, most of them within the U.S. state of West Virginia. The company was founded by Kenney Grant in 1961. Many locations are shared with Tudor's Biscuit World although the Gino's brand is exclusive to West Virginia. There is one located in Ohio, while there are stand alone Tudor's locations in eastern Kentucky, southern Ohio and southwest Virginia. Gino's serves pizza, spaghetti, sandwiches, and more. Company headquarters are located in Huntington, West Virginia and Nitro, West Virginia. Pizza Inn is an American restaurant chain and international food franchise, specializing in American-style pan pizza and side dishes. The company is based in the Dallas suburb of The Colony, Texas. Tudor's Biscuit World is a restaurant chain based in Huntington, West Virginia, most commonly found in West Virginia. Many West Virginia locations share a building with Gino's Pizza and Spaghetti, although the chain is more extensive than Gino's (which is exclusive to West Virginia), having locations in southern Ohio, eastern Kentucky, and southwestern Virginia. In 2016 a franchise was opened in Panama City, Florida. Tudor's serves biscuits, biscuit sandwiches, homestyle breakfasts and dinners, muffins, and several side dishes. The chain was originally based in Charleston, West Virginia and many of the biscuit sandwiches are named for sports teams of interest in that area, including teams at Marshall University, West Virginia University, and The University of Charleston.



Question: Which restaurant chain's headquarters is further north, Pizza Inn or Papa Gino's? Explain your reasoning. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.

 : Pizza Inn's headquarters is further north than Papa Gino's. Pizza Inn is based in The Colony, Texas, which is further north than Dedham, Massachusetts, where Papa Gino's is headquartered. The Colony, Texas is located at a latitude of approximately 33.0801 N, while Dedham, Massachusetts is located at a latitude of approximately 42.2418 N. Therefore, Pizza Inn's headquarters is further north. ## Pizza Inn ##. **(Incorrect)**



[P2] Critical Prompt (Huang et al., 2023)

 : Review your previous answer and find problems with your answer.
 : Apologies for the confusion. Upon reviewing the information, I realize that I made an error in my previous response. The correct answer is that Papa Gino's headquarters in Dedham, Massachusetts is further north than Pizza Inn's headquarters in The Colony, Texas. Therefore, the correct answer is Papa Gino's. Thank you for bringing this to my attention.

[P2] IoE Prompt (Ours)

 : If you are very confident about your answer, maintain your answer. Otherwise, update your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.
 : Upon reviewing my previous answer, I realize that I made an error. Dedham, Massachusetts, where Papa Gino's is headquartered, is further north than The Colony, Texas, where Pizza Inn is based. Dedham, Massachusetts is located at a latitude of approximately 42.2418 N, while The Colony, Texas is located at a latitude of approximately 33.0801 N. Therefore, Papa Gino's headquarters is further north. ## Papa Gino's ##. **(Correct)**

[P3] Update Prompt (Huang et al., 2023)

 : Based on the problems you found, improve your answer. Please reiterate your answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.
 : Upon reviewing the information, I realize that I made an error in my previous response. The correct answer is that Papa Gino's headquarters in Dedham, Massachusetts is further north than Pizza Inn's headquarters in The Colony, Texas. Therefore, the correct answer is ## Papa Gino's ##. **(Correct)**

[P3] Decision Refinement (Ours)



 : You give two different answers in previous responses. Check the problem and your answers again, and give the best answer. Your final answer should be put between two ## symbols, like ## ANSWER ##, at the end of your response.
 : Upon re-evaluating the locations of the headquarters for Pizza Inn and Papa Gino's, I have found that The Colony, Texas, where Pizza Inn is based, is further south than Dedham, Massachusetts, where Papa Gino's is headquartered. The Colony, Texas is located at a latitude of approximately 33.0801 N, while Dedham, Massachusetts is located at a latitude of approximately 42.2418 N. Therefore, Papa Gino's headquarters is further north. ## Papa Gino's ##. **(Correct)**

Figure 10: One example on HotpotQA benchmark with context (Yang et al., 2018) evaluated by gpt-3.5-turbo-1106 model. The [P1] standard prompt is the same for all methods. **Left:** Critical Prompt as the baseline. **Right:** our proposed IoE-based Prompt.

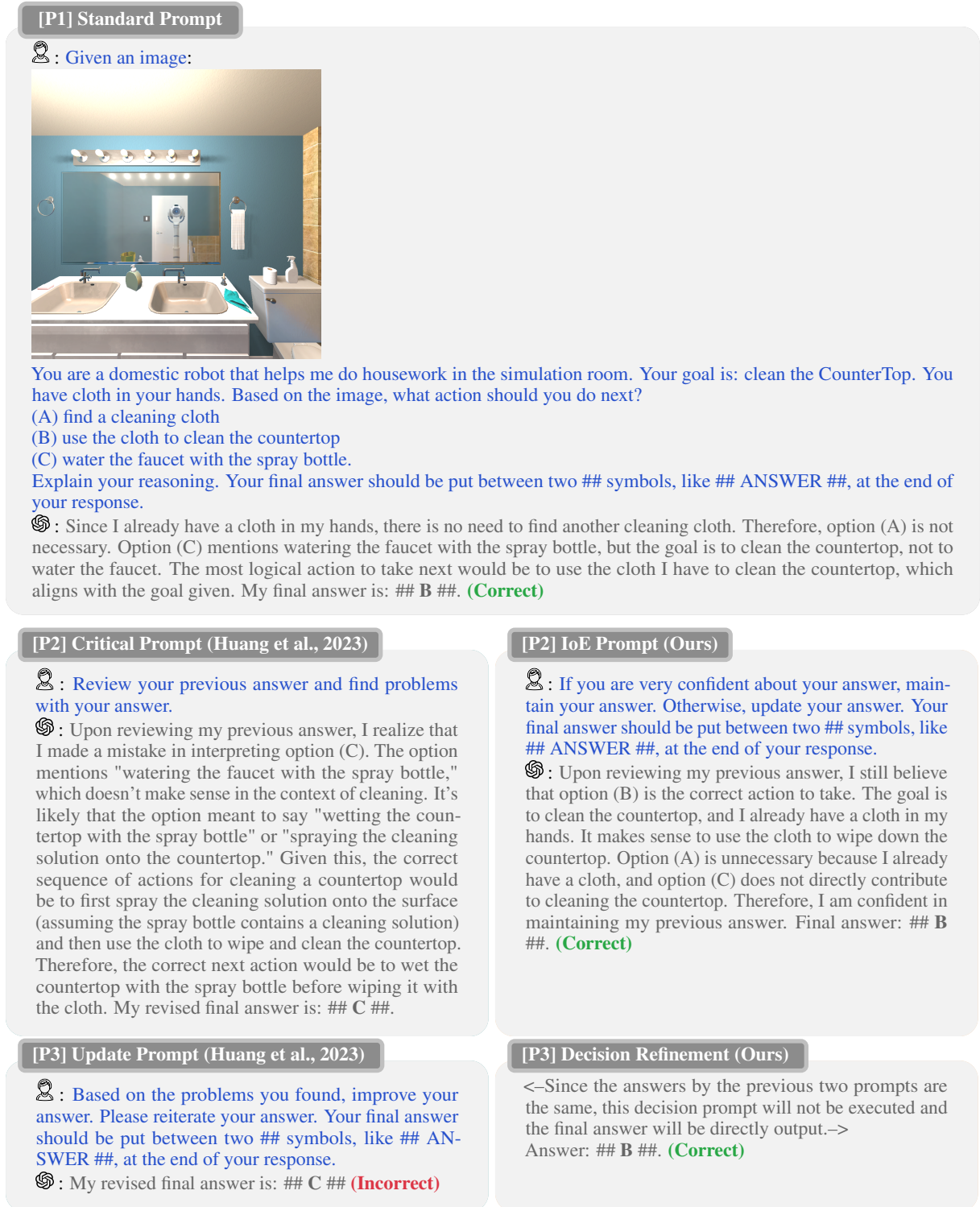


Figure 11: One example on Domestic Robot benchmark (Cai et al., 2023) evaluated with gpt-4-vision-preview model. The [P1] standard prompt is the same for all methods. **Left:** Critical Prompt as the baseline. **Right:** our proposed IoE-based Prompt.

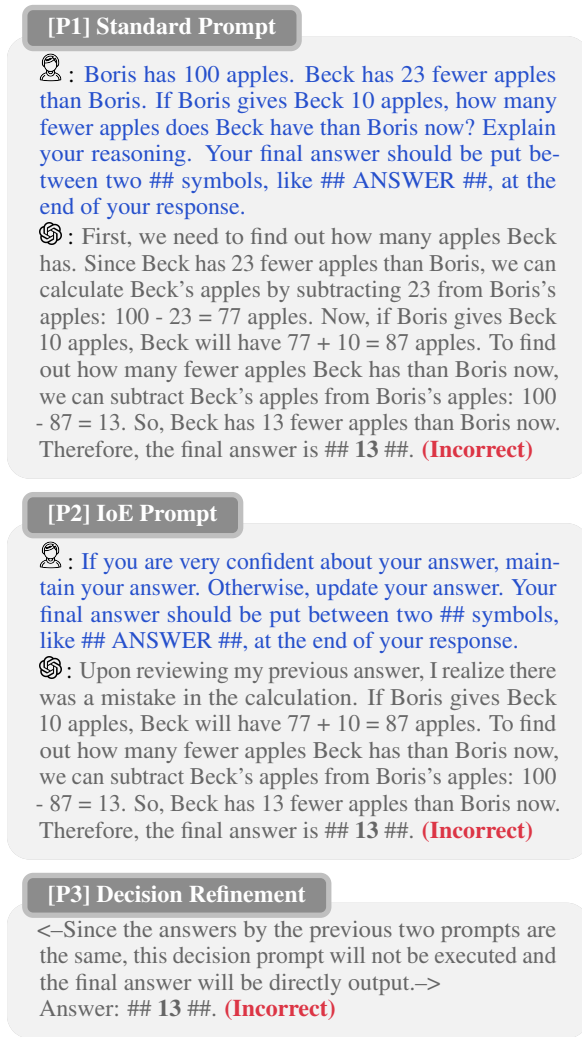


Figure 12: One example on GSM8K (Cobbe et al., 2021) with gpt-3.5-turbo-1106 model, where our IoE-based prompt failed to achieve self-correction.

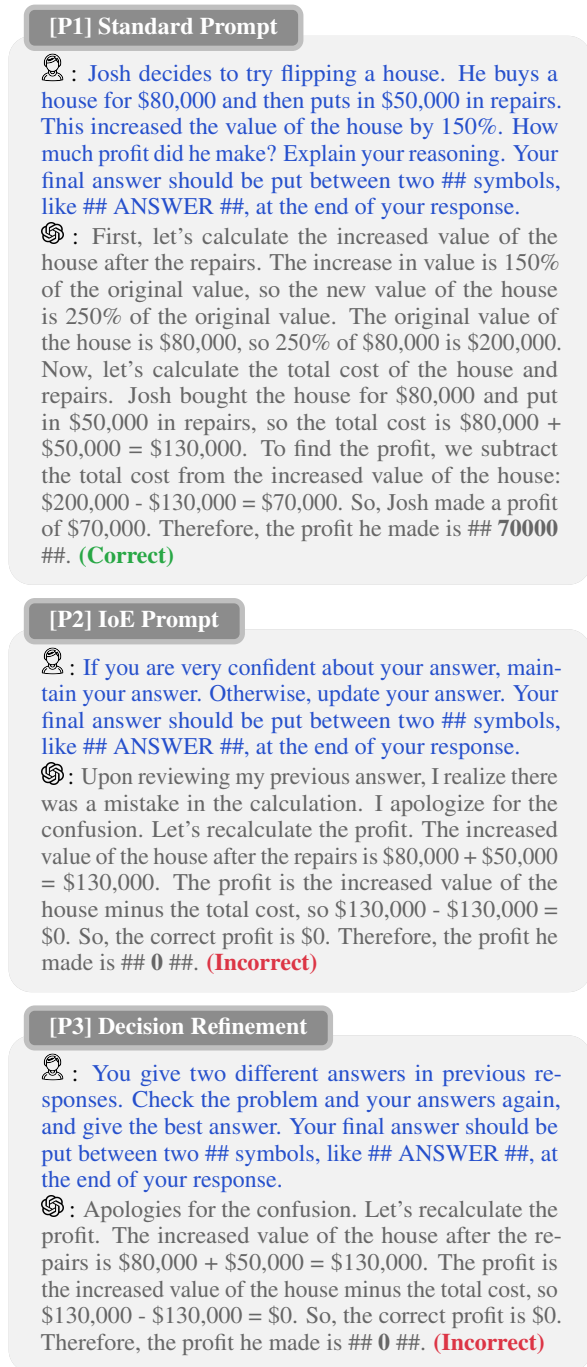


Figure 13: Another example on GSM8K (Cobbe et al., 2021) with gpt-3.5-turbo-1106 model, where our IoE-based prompt failed to achieve self-correction.