CRITICBENCH: Benchmarking LLMs for Critique-Correct Reasoning

Anonymous ACL submission

Abstract

The ability of Large Language Models (LLMs) to critique and refine their reasoning is crucial for their application in evaluation, feedback provision, and self-improvement. This paper introduces CRITICBENCH, a comprehensive benchmark designed to assess LLMs' abilities to critique and rectify their reasoning across a variety of tasks. CRITICBENCH encompasses five reasoning domains: mathematical, commonsense, symbolic, coding, and algorithmic. It compiles 15 datasets and incorporates responses from three LLM families. Utilizing CRITICBENCH, we evaluate and dissect the performance of 17 LLMs in generation, critique, and correction reasoning, i.e., GQC reasoning, and analyze the key factors affecting LLM critical reasoning. Our findings reveal: (1) a linear relationship in GQC capabilities, with critique-focused training markedly enhancing performance; (2) a task-dependent variation in critique and correction effectiveness, with logic-oriented tasks being more amenable to correction; (3) GQC knowledge inconsistencies that decrease as model size increases; and (4) an intriguing inter-model critiquing pattern, where stronger models are better at critiquing weaker ones, while weaker models can surprisingly surpass stronger ones in their self-critique. We hope these insights into the nuanced critique-correct reasoning of LLMs will foster further research in LLM critique and self-improvement[1].

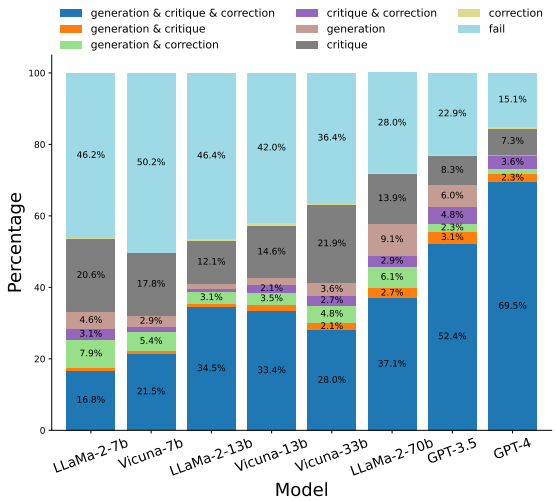

Figure 1: The distribution of the correct sets and intersection for model generation (G), critique (Q), and correction (C) on CRITICBENCH, with "&" representing intersection.

1 Introduction

The advent of large language models (LLMs) has revolutionized artificial intelligence, showcasing remarkable proficiency across diverse tasks (Brown et al., 2020; Ouyang et al., 2022; Achiam et al., 2023; Chowdhery et al., 2023; Touvron et al., 2023; Team et al., 2023). Their potential for self-evaluation and improvement is particularly fascinating, with studies suggesting that LLMs can effectively assess model outputs (Liu et al., 2023; Fu et al., 2023a; Chiang et al., 2023), and even engage in self-reflection and correction (Madaan et al., 2023; Chen et al., 2023b; Gou et al., 2023a). This capability rests on the LLMs' *critical reasoning* skills, which involves (1) *critique* - identifying issues in provided responses, and (2) *correct* - proposing suitable modifications.

However, a comprehensive understanding of LLMs' critical reasoning abilities remains elusive. Prior research (Lightman et al., 2023; Li et al., 2023b; Luo et al., 2023) has focused on a narrow range of models and datasets and has yielded inconsistent findings (Madaan et al., 2023; Huang et al., 2023), underscoring the need for a thorough investigation. It is imperative to systematically gauge LLMs' proficiency in critiquing and correcting provided answers.

To address these challenges, we present CRIT-

[1]Code and data will be released.

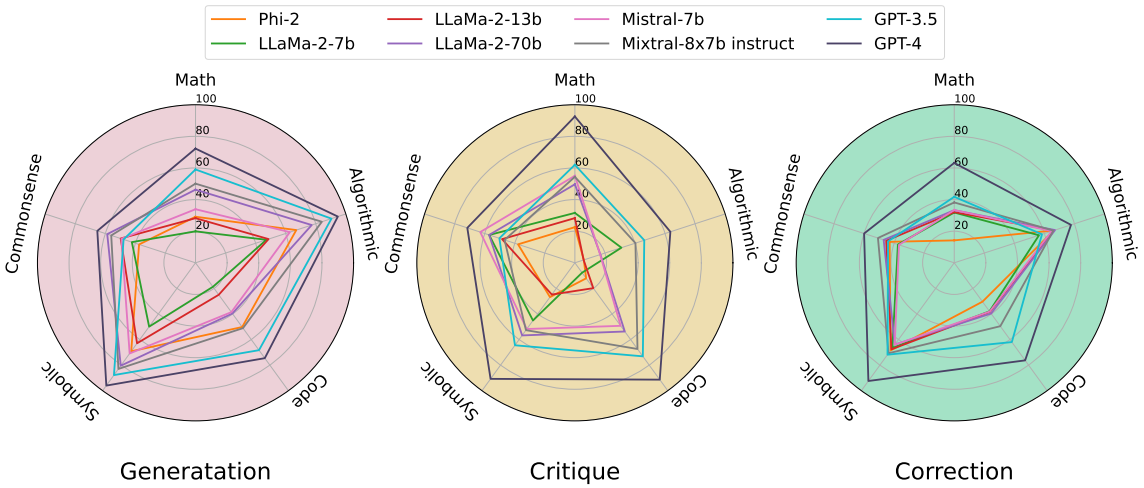

Figure 2: Models' performance on CRITICBENCH.

ICBENCH, a comprehensive benchmark designed to evaluate the critique and correction skills of LLMs. CRITICBENCH encompasses 15 datasets spanning five task categories: mathematical, commonsense, symbolic, coding, and algorithmic. We leverage eight models from the LLaMA, Vicuna, and GPT series to create the responses that are to be critiqued and corrected. Moreover, we include GPT-4 and undertake manual data reviews to ensure data quality, culminating in 3.8K data instances (see Table 2 for detailed data collection information). We conduct extensive experiments on CRITICBENCH with 17 LLMs, including closed-source models GPT-3.5 and GPT-4, open-source models like Phi-2 (Javaheripi et al., 2023), the LLaMA family (Touvron et al., 2023), the Vicuna family (Chiang et al., 2023), and the Mistral family (Jiang et al., 2023, 2024), as well as two models specifically trained for critiquing, namely Auto-J (Li et al., 2023b) and UltraCM (Cui et al., 2023). Partial results are shown in Figure 2.

Our contributions are summarized as follows:

- We present CRITICBENCH, a benchmark comprising five different domains to systematically assess critique and correction reasoning in various LLMs. Utilizing CRITICBENCH, we investigate the impact of base models, training strategies, prompt strategies, and oracle feedback on the critique-correct reasoning performance of LLMs.

- We reveal that LLMs exhibit a linear relationship in their generation, critique, and correction (GQC) capabilities, despite limited training on critique tasks.

- The type of task has a significant impact on correction performance. LLMs struggle more with incorrect answers in detail-oriented tasks like algorithmic tasks compared to logic-centric tasks like code generation.

- By comparing the sets of questions where models correctly generate, critique, and correct, we find that a model's knowledge is inconsistent across these three tasks, with stronger models showing more consistency in GQC capabilities.

- We observe that although stronger models are better at critiquing, models with weaker generative abilities can still accurately evaluate responses from stronger models, sometimes even outperforming the latter in self-critique.

2 Related Work

LLM Reasoning The advent of few-shot prompting (Brown et al., 2020), has significantly improved the performance of LLMs in executing natural language reasoning tasks. The proposal of Chain of Thought (CoT) further enhanced the reasoning capabilities of LLMs (Wei et al., 2022). Studies such as (Wang et al., 2022; Zhou et al., 2022; Fu et al., 2022; Zheng et al., 2023a; Yoran et al., 2023; Diao et al., 2023) have enhanced this process in various ways, achieving remarkable results. Additionally, research by (Nakano et al., 2021; Chen et al., 2022; Gao et al., 2023; Li et al., 2023a; Gou et al., 2023b) proposed the use of external tools such as search engines and Python interpreters to augment and assist model reasoning. These methods, from vari-

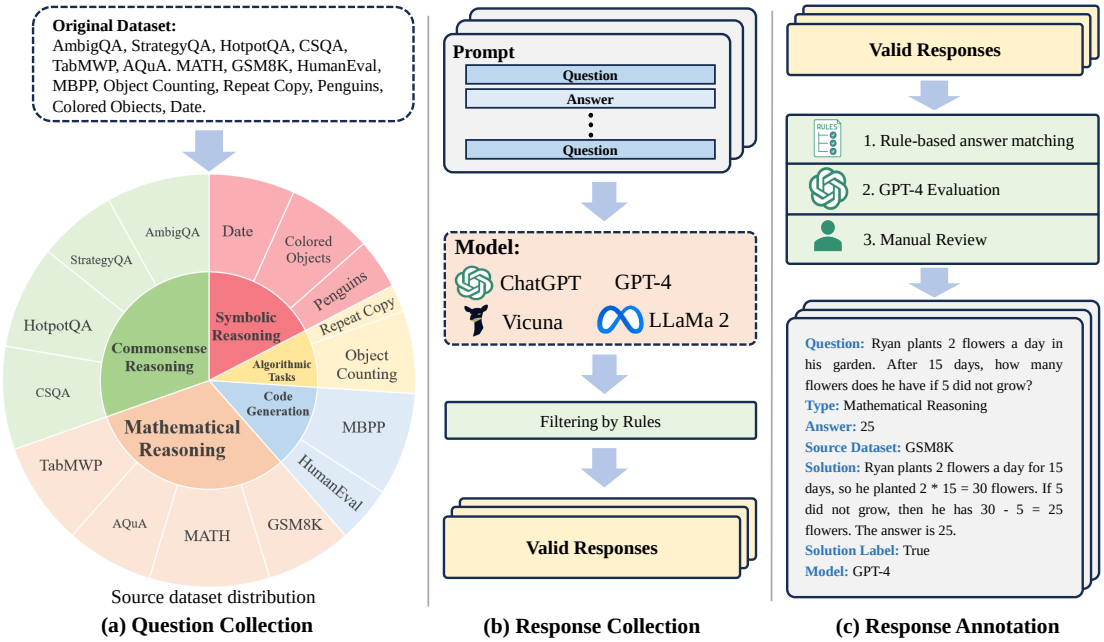

Figure 3: An overview for the CRITICBENCH construction.

ous angles, continuously push the development of model reasoning ability.

LLM Critiquing & Correction As LLMs continue to evolve in their capabilities across various domains, many studies seek to enhance performance by requiring LLMs to provide critiques of the generated responses in various forms, including utilizing internal feedback (Madaan et al., 2023; Dhuliawala et al., 2023; Miao et al., 2023; Welleck et al., 2022; Zheng et al., 2023b), leveraging external feedback (Kim et al., 2023; Shinn et al., 2023; Gou et al., 2023a; Chen et al., 2023b), and employing multiple models for critiquing and debating answers (Liang et al., 2023; Du et al., 2023; Chen et al., 2023a; Yin et al., 2023). These studies all demonstrate the potential of LLMs in Critique-Correcting Reasoning. Additionally, some researchers have trained critique models to provide supervisory signals for generative models (Ke et al., 2023; Li et al., 2023b; Ye et al., 2023; Wang et al., 2023; Cui et al., 2023). However, these works focus on specific methods without providing a comprehensive evaluation and analysis of Critique-Correcting Reasoning, which also limits the development of this field.

Critiquing Tasks Many researchers have promoted related research by constructing critiquing datasets, including text generation (Stiennon et al., 2020; Matiana et al., 2021), semantic understanding (Pougué-Biyong et al., 2021), factuality detection (Thorne et al., 2018), mathematics (Lightman et al., 2023; Luo et al., 2023), and alignment (Li et al., 2023b). However, these datasets all focus on the critique itself and do not comprehensively integrate existing benchmarks to facilitate comparative analysis of generation, critique, and correction. In contrast, we construct CRITICBENCH to delve deeper into the models' performance on GQC.

3 The CRITICBENCH

3.1 Overview of CRITICBENCH

CRITICBENCH is designed to assess the two key aspects of LLMs' critical reasoning: critique and correction. By combining these two aspects, LLMs can critique a given response and apply corrective reasoning to produce an updated answer. In this section, we detail the principles and processes involved in the construction of CRITICBENCH; the construction process is listed in Figure 3.

CRITICBENCH is designed to follow these collection principles: (1) it encompasses multiple task types, aimed at comprehensively showcasing the model's abilities; (2) it incorporates diverse models for response generation, promoting response variety; (3) it employs acknowledged datasets, enabling straightforward comparisons with the models' generation capabilities; and (4) it ensures data quality through both GPT-4 and manual review.

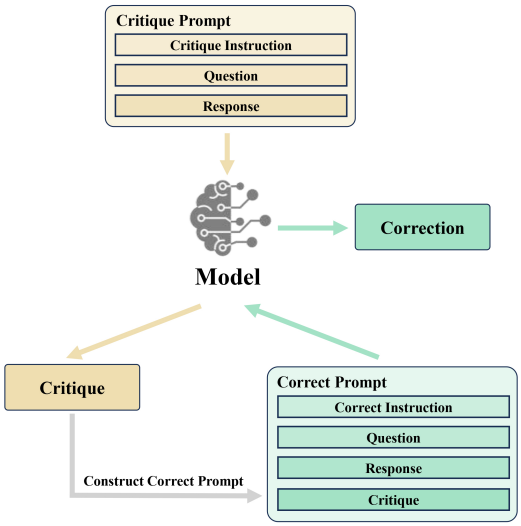

Figure 4: Evaluation process on CRITICBENCH.

3.2 Question Collection

This section describes the data collection methodology, following defined principles. A specific quantity of data is extracted from an existing dataset, utilizing any relevant subset if available, or alternatively, selecting randomly from the dataset.

Mathematical Reasoning We selected GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), AQuA (Ling et al., 2017), and TabMWP (Lu et al., 2023) for mathematical reasoning. We utilized the existing subsets from Shi et al. (2022) and Lightman et al. (2023) respectively, randomly sampled 300 questions from TabMWP, and incorporated all available questions from AQuA.

Commonsense Reasoning To thoroughly evaluate the GQC ability of LLMs in commonsense reasoning tasks, we employ four datasets: CSQA (Talmor et al., 2019), AmbigNQ (Min et al., 2020), StrategyQA (Geva et al., 2021), and HotpotQA (Yang et al., 2018). We randomly sample 300 questions from CSQA, HotpotQA, and AmbigNQ, and include all questions in StrategyQA.

Symbolic Reasoning To enrich the variety of symbolic reasoning question types, we utilized three datasets from BIG Bench (Srivastava et al., 2023), namely Penguins, which requires understanding of tabular data; Colored Object, which involves analyzing the relative positions, absolute positions, and colors of various objects; and Date, which involves understanding and calculating dates. For these three datasets, we used their complete sets.

Code Generation For code generation, we selected MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021) to construct our dataset. For MBPP, we randomly sampled 300 questions, while for HumanEval, we used its complete set.

Algorithmic Tasks We utilized Object Counting and Repeat Copy, sourced from BIG-Bench (Srivastava et al., 2023) to evaluate the model's ability to manage details. Object Counting involves enumerating items presented within questions, while Repeat Copy requires the generation of word sequences based on instructions. The complete sets of these tasks are employed to form the detail-oriented algorithmic task of CRITICBENCH. By utilizing the aforementioned datasets, we ensure that the questions in CRITICBENCH cover a diverse range of examination angles. The five types of tasks correspond to a variety of knowledge domains, with the detail-focused algorithmic tasks, the detail and logic-encompassing mathematical reasoning, and the logic-focused code generation corresponding to different styles of reasoning processes. Specific data statistics can be seen in Appendix A

3.3 Response Collection

Following the collection of benchmark questions, we employ various LLMs, including GPT-3.5, GPT-4, LLaMa2 (7B, 13B, and 70B variants), and vicuna (7B, 13B, and 33B variants) to generate response for each question, using greedy decoding. The details of the prompts used are available in Table D.

Next, we filter out the responses that did not provide valid reasoning. We then apply a random sampling strategy to maintain a consistent number of model-generated responses across each dataset.

3.4 Response Annotation

Response correctness is initially determined by rule-based matching, followed by a more detailed evaluation using GPT-4 to assess response precision. This includes flagging mathematically correct answers with incorrect reasoning and recognizing near-correct commonsense responses. Discrepancies between GPT-4 evaluations and initial annotations are resolved through manual review. During this review, we identified questions in the Date dataset that lack correct options, with detailed examples provided in Figure 6. Examples of annotations are provided in Appendix B.

3.5 Evalution

Evaluation Process The evaluation process on CRITICBENCH is illustrated in Figure 4. First, a critique prompt is constructed using the response

Model	Type	Generation	Critiquing			Correction		
			ZS-AO	ZS-CoT	FS	ZS-CoT	FS	FS (oracle)
Phi-2	SIFT	43.74	40.45(-10.35)	29.31(-21.49)	27.92(-22.88)	17.41(-30.96)	37.67(-10.70)	50.04(+1.67)
LLaMa-2-7b	BASE	34.48	-	-	41.72(-9.08)	-	41.91(-6.46)	50.98(+2.61)
LLaMa-2-7b chat	RLHF	34.22	60.47(+9.67)	47.36(-3.44)	42.50(-8.30)	10.33(-38.04)	38.22(-10.15)	51.79(+3.42)
Vicuna-7b	SIFT	32.31	6.82(-43.98)	13.85(-36.95)	40.87(-9.93)	15.32(-33.05)	41.23(-7.14)	51.48(+3.11)
Mistral-7b	BASE	47.37	-	-	56.09(+5.29)	-	42.46(-5.91)	53.18(+4.81)
LLaMa-2-13b	BASE	42.68	-	-	33.11(-17.69)	-	45.25(-3.12)	50.85(+2.48)
LLaMa-2-13b chat	RLHF	41.67	58.84(+8.04)	43.54(-7.26)	51.10(+0.30)	12.13(-36.24)	41.33(-7.04)	52.18(+3.81)
Vicuna-13b	SIFT	40.12	41.94(-8.86)	12.97(-37.83)	47.17(-3.63)	10.77(-37.60)	42.22(-6.15)	51.71(+3.34)
Vicuna-33b	SIFT	45.18	24.04(-26.67)	45.94(-4.86)	53.41(+2.61)	37.62(-10.75)	42.43(-5.94)	52.24(+3.87)
LLaMa-2-70b	BASE	57.61	-	-	53.39(+2.59)	-	45.75(-2.62)	55.14(+6.77)
LLaMa-2-70b chat	RLHF	51.53	68.00(+17.20)	54.82(+4.02)	64.27(+13.47)	31.92(-16.45)	43.95(-4.42)	55.53(+7.16)
Mixtral-8×7b	BASE	58.43	-	-	64.21(+13.41)	-	49.31(+0.94)	56.13(+7.76)
Mixtral-8×7b instruct	SIFT	60.03	33.56(-17.24)	46.54(-4.26)	54.06(+3.26)	41.12(-7.25)	50.82(+2.45)	56.39(+8.02)
GPT-3.5	RLHF	64.21	69.96(+19.16)	51.51(+0.71)	60.17(+9.37)	46.89(-1.48)	51.24(+2.87)	61.03(+12.66)
GPT-4	RLHF	**76.15**	**81.65**(+30.85)	**78.89**(+28.09)	**86.19**(+35.39)	**56.42**(+8.05)	**69.83**(+21.46)	**74.98**(+26.61)
Average score	-	48.65	48.57(-2.23)	42.47(-8.33)	51.75(+0.95)	27.99(-20.38)	45.57(-2.8)	54.91(+6.54)
Auto-J	CT	-	-	65.29(+14.49)	-	-	-	-
UltraCM	CT	-	-	61.11(+10.31)	-	-	-	-

Table 1: Average performance on CRITICBENCH. In the table, the values in parentheses under the "Critiquing" column show comparisons to the baseline critique score of random guessing (50.80). Similarly, in the "Correction" column, the parentheses display changes relative to the model's original accuracy (48.37). Blue highlights indicate improvement, while orange marks decline. Type: BASE refers to the pretrained model, SIFT means its enhancement via Supervised Instruction Finetuning, RLHF denotes further training with Reinforcement Learning from Human Feedback, and CT represents Critique Training.

within CRITICBENCH, prompting the model to perform a critique. Subsequently, a correct prompt is built incorporating the critique to obtain the model's correction results.

Generation and Correction Metrics. We use the accuracy metric S_a to assess models' generation and correction capability as follows:

$$S_a = \frac{c}{N} \tag{1}$$

where c is the number of correct predictions, N is the total number of questions.

Critique Metrics. By prompting models to discriminate the correctness of a response, we evaluate their ability to critique. To mitigate issues of class imbalance where models misclassify most predictions as correct or incorrect, we adopt the F1 score as a more robust metric for error identification:

$$S_p = \frac{\sum_{i=1}^{m} q_i}{m}, \tag{2}$$

$$S_r = \frac{\sum_{i=1}^{n} q_i}{n}, \tag{3}$$

$$q_i = \begin{cases} 1, & \text{if correctly classified as wrong,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

$$S_f = 2 \times \frac{S_p \times S_r}{S_p + S_r}, \tag{5}$$

where S_p is the precision score, S_r is the recall score, m is the number of classified as wrong, n is the number of actual wrong, and q_i indicates the correct discrimination of a response as wrong.

4 Experiments

4.1 Experimental Setup

To conduct a comprehensive assessment, we have selected the following models: Phi-2, the LLaMa family, the Vicuna family, the Mistral family, and the GPT family. We evaluate those models on CRITICBENCH under three phases: (1) *Generation*: Models are tasked with answering questions through the utilization of CoT (Wei et al., 2022). (2) *Critique*: This phase evaluates provided response correctness. After testing three prompts from Huang et al. (2023), we selected the most effective for zero-shot use. By modifying the original prompt, we experimented with two prompts: a zero-shot answer only (ZS-AO) and a zero-shot chain of thought (ZS-COT) requesting analysis. For pre-trained models with weaker instruction adherence (labeled BASE), we focused on few-shot (FS) setting. Additionally, to assess critique training effects, we evaluated the 13B model Auto-J (Li et al., 2023b) and UltraCM (Cui et al., 2023) by extracting their discrimination on the response using a rule-based method. (3) *Correction*: After the

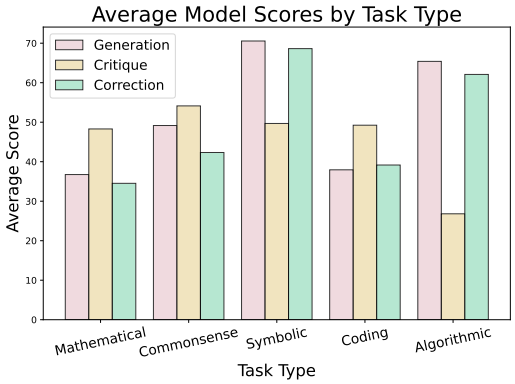

Figure 5: Average Score on different types of tasks.

critique, this phase refines responses by addressing identified inaccuracies. Besides ZS-COT and FS, we evaluated FS (oracle), applying corrections solely to inaccurately generated responses.

The prompts for above phases are detailed in D. For all experimental setups, we set the temperature to 0 during the three phases.

4.2 Results and Analysis

Table 1 showcases the performance of LLMs on CRITICBENCH. Specifically, we are interested in exploring the following research questions: **RQ1:** What factors influence the model's generation, critique, and correction? **RQ2:** What's the interrelationship between a model's capabilities in generation, critique, and correction? **RQ3:** How do critique-correct reasoning capabilities differ across various task types? **RQ4:** Is the model's knowledge consistent in generation, critique, and correction? **RQ5:** How does the inter-model critiquing patterns manifest among models of varying capability? In the following sections, we will discuss these research questions in turn.

4.2.1 RQ1: Key Factors in LLM Critical Reasoning

Base Model Observations reveal that Phi-2 (2.7B), despite excelling in generation tasks, exhibits weaker performance in critique and correction tasks compared to models with similar generation performance (*e.g.*, LLaMa-2-13b, Vicuna-33b). This suggests Phi-2's training focus leans heavily towards generation, lacking proficiency in critique and correction tasks. This underscores the necessity of evaluating generation, critique, and correction collectively to achieve a comprehensive assessment of a model's mastery of knowledge. Additionally, Mistral-7b stands out as the top performer among models of similar size, even outperforming Vicuna-33b. However, in Figure 2, GPT-4 consistently maintains a significant lead in GQC of all types of tasks. Despite this, other models like LLaMA-70b and Mixtral-8×7b demonstrate competitiveness against GPT-3.5.

Training Strategy By comparing the results of different models under LLaMa family, it is observed that the alignment tax has limited the RLHF's generation performance. However, for critique and correction, RLHF consistently outperforms BASE, suggesting that RLHF might suppress the expression of knowledge in generation. Simultaneously, in critique, CT demonstrates results surpassing those of GPT-3.5 with a smaller parameter size (13B), proving the effectiveness of CT.

Prompt Strategy The critique results from zero-shot settings show sensitivity to prompts. For instance, Vicuna-13b's ZS-AO flagged 22.04% of responses as incorrect, compared to only 4.8% in ZS-CoT, against an actual error rate of 51.63%. This inconsistency likely stems from the model's insufficient training on critique tasks, making it struggle without clear examples. Meanwhile, in correction, few-shot also significantly outperforms zero-shot. Therefore, FS is always the better choice in both critique and correction. For subsequent analysis, we will primarily focus on FS results.

Orcale Feedback We also explored an oracle setting for corrections by modifying only incorrect responses in CRITICBENCH. Results from FS (oracle) outperformed those without an oracle, showing that reliable external feedback can significantly enhance correction efficiency. However, for more advanced models (from LLaMa-2-70b to GPT-4), corrections in the oracle setting still fell short of direct generation, indicating they are still influenced by incorrect responses from other models.

4.2.2 RQ2: Correlations of GQC Abilities

The capabilities of generation, critique, and correction exhibit a positive correlation. Figures 6 illustrates the interconnectedness among three capabilities. It is observed that there is a positive linear relationship between generating and judging. The improvement rates of generation and critique are nearly identical, even though the model primarily focuses on learning tasks related to generation during training. However, the linear correlation between the generation and correction capacities is not prominent. Weaker models tend to exhibit diminished correctness upon correction, as compared

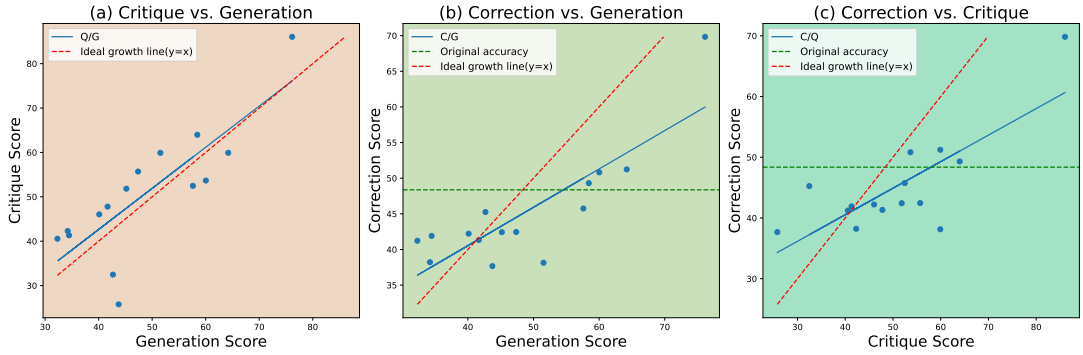

Figure 6: Interrelationship between a model's capabilities in generation, critique, and correction. Each point on the graph represents a model, with coordinates indicating its performance in Generation (G), Critique (Q), and Correction (C). The graph features fitted lines for the scatter plots, denoted by blue lines for Q/G, C/G, and C/Q, while a red dashed line represents the ideal growth line (y=x). The green dashed line marks the original accuracy of responses from CRITICBENCH.

to the initial benchmark responses. This observation indicates that a model's limited capability to generate precise answers impacts its ability to correct those responses. Similarly, the relationship between critique and correction reveals that, even if the model can discriminate between correct and incorrect responses, it is not able to rectify them.

4.2.3 RQ3: Impact of Task Type

The model's critique and correction capability depends on whether the task focuses on details or logic. In Figure 7, we illustrate the variability in critique and correction capabilities across various task types. Figure 7 (a), (b), and (c) demonstrate the varying relationships between Q and G across different types of tasks. Specifically, the models exhibit weaker critique performance in detail-oriented algorithmic tasks compared to their generation abilities, indicated by dots below the ideal growth line y=x. In contrast, for mathematical reasoning and code generation tasks, their critique capabilities surpass generation capabilities. The ability to correct errors in algorithmic tasks is also limited, even when the model answers correctly. For mathematical tasks requiring detailed and logical reasoning, significantly higher accuracy in generation is needed for effective corrections, explaining the lack of improvement of math in the self-refine (Madaan et al., 2023). Interestingly, For the logic-focused Code Generation tasks, improvements are realized as long as the model's generation performance surpasses that of the original responses' performance. This result indicates that models, when performing critique and correction, are easily disrupted by incorrect answers in tasks that focus on details, but not in those that emphasize logic.

Additionally, Figure 5 displays the model's average GQC score on different types of tasks. It can be observed that, similar to algorithmic tasks, the model struggles to effectively critique in detail-oriented symbolic reasoning, where its performance is significantly lower than that in generation and correction. This further demonstrates the model's lack of critique ability on such tasks.

4.2.4 RQ4: Consistency of GQC Knowledge

GQC knowledge inconsistencies persist across all models. The relationship between human abilities of generation, critique, and correction suggests that generation falls under critique, with correction closely linked to generation. This implies humans can identify errors without necessarily knowing the correct answer, whereas generating the right answer also implies the ability to critique and correct it. However, the knowledge acquired by LLMs is not entirely consistent across generation, critique, and correction tasks.

Figure 1 delineates the overlap and distinctiveness of GQC, highlighting their inconsistencies of knowledge. As a model's parameter size increases, its knowledge coherence across GQC, and instances of complete task failure (where G, Q, and C all fail) decrease. Notably, it can be noted that questions correctly critiqued alone always occupy a significant portion, indicating that the model possesses a considerable amount of knowledge that is not expressed through generation or correction.

4.2.5 RQ5: Patterns of Inter-Model Critique

Figure 8 presents a visualization illustrating the inter-model critiquing result. Overall, it can be

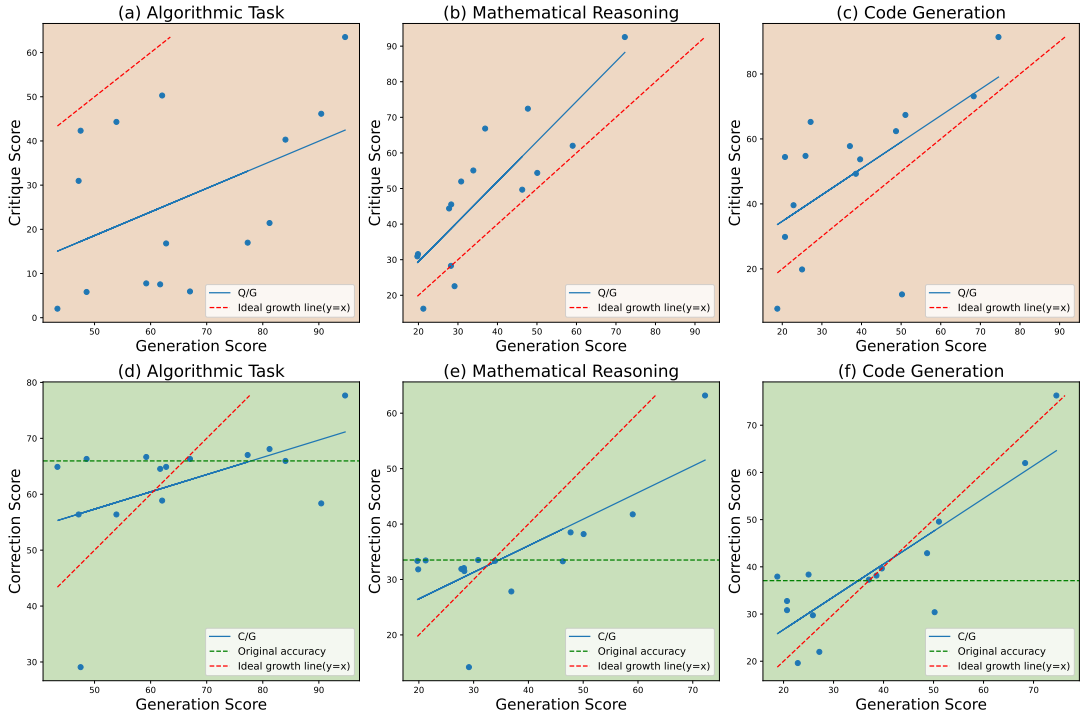

Figure 7: Critique/Generation (Q/G) and Correction/Generation (C/G) line in Algorithmic Task, Mathematical Reasoning, and Code Generation.

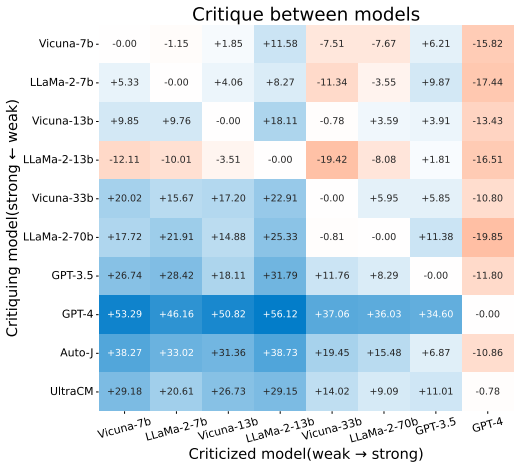

Figure 8: The results of model critiques are depicted in the graph, where the self-critique scores of models are set to 0 as a baseline for comparison, and models are arranged from weakest to strongest in order of generation accuracy.

observed that strong models consistently have a superior ability to critique than weak models, and the responses of weak models are more easily critiqued accurately, possibly because the errors made by weak models are more evident. Interestingly, certain weaker models approach or even exceed the self-critique scores of stronger models, suggesting that their critique capacity against stronger models might surpass the latter's self-critique. Post-critique training, models such as Auto-J and UltraCM show enhanced ability to assess response correctness across different models, with UltraCM's performance nearing its self-critique level against GPT-4, underscoring the value of critique training.

5 Conclusion

In conclusion, our investigation through CRITICBENCH has illuminated the capacities and limitations of Large Language Models (LLMs) in GQC. Our focused exploration using CRITICBENCH on the relationship among models in GQC revealed a linear correlation and subtle inconsistencies between GQC, while our analysis across different task types found that models perform better in Q and C for tasks focused on logic compared to those requiring attention to detail. Additionally, by examining the outcomes of models critiquing each other, we discovered that weaker models could sometimes correct the outputs of stronger models more effectively than those models could self-correct. These findings underscore the effectiveness of CRITICBENCH in evaluating and analyzing the GQC capabilities of models.

Limitations

Measuring the ability of model critique effectively has always been a challenge. In this paper, we use discrimination results as a valid indicator to measure its critique ability, mainly for the following reasons: (1) Fine-grained indicators that provide scores based on evaluation principles are only suitable for specific tasks and lack generality. Moreover, different tasks have different focuses, and evaluation principles valued by humans may dynamically change. It is somewhat idealistic to exhaustively list all evaluation principles once and for all. (2) Scores based on evaluation principles rely on human annotations or results from GPT-4 for validation. However, reliable human annotations incur high costs, and GPT-4 may contain errors and biases. Using GPT-4 results also makes it impossible to evaluate its critique ability. Additionally, in reasoning tasks, the most important aspect of critique and correction is to judge whether there are errors in the reasoning process and its results. Therefore, considering the above considerations, we use a binary metric to measure the results.

Future work should address these challenges by exploring alternative evaluation methodologies that mitigate reliance on costly human annotations. Additionally, there is a need to develop more nuanced critique metrics that can effectively capture the diverse aspects of model performance across various tasks and evaluation scenarios.

Ethics Statement

We constructed the model based on existing public datasets and models, as detailed in Section 3, and annotated the results using GPT-4 and human evaluation. We acknowledge that, despite employing rule-based filtering, GPT-4 review, and human review, unpredictable errors may still exist in the responses generated by different models.

Additionally, when using the critique ability of Large Language Models (LLMs), it's also important to be aware of the risks involved, such as potential biases. When the GQC capability of an LLM is inconsistent, and its critique ability surpasses the other two aspects, it is necessary to carefully discern whether its discriminate results contain harmful biases.

References

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

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. 2023a. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. *arXiv preprint arXiv:2309.13007*.

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.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

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

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul

Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*.

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*.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023a. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Yao Fu, Litu Ou, Mingyu Chen, Yuhao Wan, Hao Peng, and Tushar Khot. 2023b. Chain-of-thought hub: A continuous effort to measure large language models' reasoning performance. *arXiv preprint arXiv:2305.17306*.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies. *Transactions of the Association for Computational Linguistics (TACL)*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023a. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yujiu Yang, Minlie Huang, Nan Duan, Weizhu Chen, et al. 2023b. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, et al. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. 2023. Phi-2: The surprising power of small language models.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, et al. 2023. Critiquellm: Scaling llm-as-critic for effective and explainable evaluation of large language model generation. *arXiv preprint arXiv:2311.18702*.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. 2023. Language models can solve computer tasks. *arXiv preprint arXiv:2303.17491*.

Chengshu Li, Jacky Liang, Andy Zeng, Xinyun Chen, Karol Hausman, Dorsa Sadigh, Sergey Levine, Li Fei-Fei, Fei Xia, et al. 2023a. Chain of code: Reasoning with a language model-augmented code emulator. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. 2023b. Generative judge for evaluating alignment. *arXiv preprint arXiv:2310.05470*.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *International Conference on Learning Representations (ICLR)*.

Liangchen Luo, Zi Lin, Yinxiao Liu, Lei Shu, Yun Zhu, Jingbo Shang, and Lei Meng. 2023. Critique ability of large language models. *arXiv preprint arXiv:2310.04815*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

Shahbuland Matiana, JR Smith, Ryan Teehan, Louis Castricato, Stella Biderman, Leo Gao, and Spencer Frazier. 2021. Cut the carp: Fishing for zero-shot story evaluation. *arXiv preprint arXiv:2110.03111*.

Ning Miao, Yee Whye Teh, and Tom Rainforth. 2023. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *arXiv preprint arXiv:2308.00436*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Bhargavi Paranjape, Scott Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Marco Tulio Ribeiro. 2023. Art: Automatic multi-step reasoning and tool-use for large language models. *arXiv preprint arXiv:2303.09014*.

John Pougué-Biyong, Valentina Semenova, Alexandre Matton, Rachel Han, Aerin Kim, Renaud Lambiotte, and Doyne Farmer. 2021. Debagreement: A comment-reply dataset for (dis) agreement detection in online debates. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. 2022. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tianlu Wang, Ping Yu, Xiaoqing Ellen Tan, Sean O'Brien, Ramakanth Pasunuru, Jane Dwivedi-Yu, Olga Golovneva, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. Shepherd: A critic for language model generation. *arXiv preprint arXiv:2308.04592*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. 2022. Generating sequences by learning to self-correct. In *The Eleventh International Conference on Learning Representations*.

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. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, and Minjoon Seo. 2023. Selfee: Iterative self-revising llm empowered by self-feedback generation. *Blog post, May*, 3.

Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuan-Jing Huang, and Xipeng Qiu. 2023. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15135–15153.

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. 2023. Answering questions by meta-reasoning over multiple chains of thought. *arXiv preprint arXiv:2304.13007*.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. 2023a. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023b. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

A Statistics of CriticBench

Type	Dataset	Samples	License
Math	GSM8K	250	mit
	MATH	500	mit
	AQuA	254	apache-2.0
	TabMWP	300	cc-by-nc-sa
Commonsense	CSQA	300	mit
	AmbigNQ	300	cc-by-sa-3.0
	StrategyQA	229	mit
	HotpotQA	300	cc-by-sa-4.0
Symbolic	Penguins	146	apache-2.0
	Colored Object	250	apache-2.0
	Date	250	apache-2.0
Coding	MBPP	500	cc-by-4.0
	HumanEval	164	mit
Algorithmic	Object Counting	250	apache-2.0
	Repeat Copy	32	apache-2.0
All	-	3825	-

Table 2: Statistics of CriticBench.

B Annotation Example

Question: Machine A puts out a yo-yo every 6 minutes. Machine B puts out a yo-yo every 9 minutes. After how many minutes will they have produced 10 yo-yos? Answer Choices: A)24 minutes B)32 minutes C)36 minutes D)64 minutes E)72 minutes
Solution: Machine A puts out a yo-yo every 6 minutes. Machine B puts out a yo-yo every 9 minutes. So Machine A will produce 10 yo-yos in 60 minutes. Machine B will produce 10 yo-yos in 90 minutes. So Machine A will produce 10 yo-yos in 60 minutes and Machine B will produce 10 yo-yos in 90 minutes. So after 120 minutes, they will have produced 10 yo-yos. The answer is C) 120.
Answer: C
Rule-based answer matching: Correct
GPT-4 evaluation: Wrong
Manual review: Wrong
Final label: Wrong

Table 3: Inconsistencies between rule-based matching and GPT-4 evaluations (AQuA).

Question:

```python
def words_in_sentence(sentence):
    """
    You are given a string representing a sentence,
    the sentence contains some words separated by a space,
    and you have to return a string that contains the words from the original sentence,
    whose lengths are prime numbers,
    the order of the words in the new string should be the same as the original one.

    Example 1:
        Input: sentence = "This is a test"
        Output: "is"

    Example 2:
        Input: sentence = "lets go for swimming"
        Output: "go for"

    Constraints:
        * 1 <= len(sentence) <= 100
        * sentence contains only letters
    """
```

Solution:

```python
def words_in_sentence(sentence):
    words = sentence.split()
    prime_words = [word for word in words if len(word) in [2, 3, 5, 7]]
    return ' '.join(prime_words)
```

Check function:

```python
def check(candidate):
    # Check some simple cases
    assert candidate("This is a test") == "is"
    assert candidate("lets go for swimming") == "go for"
    assert candidate("there is no place available here") == "there is no place"
    assert candidate("Hi I am Hussein") == "Hi am Hussein"
    assert candidate("go for it") == "go for it"
    # Check some edge cases that are easy to work out by hand.
    assert candidate("here") == ""
    assert candidate("here is") == "is"
check(words_in_sentence)
```

Check result: Correct

GPT-4 evaluation: Wrong

Manual review: Wrong

Final label: Wrong

Table 4: Inconsistencies between rule-based matching and GPT-4 evaluations (HumanEval).

Question: Lake Wales Ridge National Wildlife Refuge is on the US route that runs between a city in what Florida county?
Solution: Lake Wales Ridge National Wildlife Refuge is on the US route that runs between a city in Polk County. The answer is Polk County.
Answer: polk
Rule-based answer matching: Wrong
GPT-4 evaluation: Correct
Manual review: Correct
Final label: Correct

Table 5: Inconsistencies between rule-based matching and GPT-4 evaluations (HotpotQA).

C Error in Date

Upon examination, we discovered that all questions in the dataset related to "Jane and John married on Jan 2, 1958." have issues with their options.

Question: Jane and John married on Jan 2, 1958. It is their 5-year anniversary today. What is the date today in MM/DD/YYYY? Options: (A) 01/02/1894 (B) 01/02/1961 (C) 12/23/1960 (D) 01/02/1960 (E) 01/03/1961 (F) 01/02/1990
Provided Answer: B
Explanation: If Jane and John married on Jan 2, 1958, the date of Jane and John's 5-year anniversary should be Jan 2, 1963.

Table 6: Incorrect Question in Date (no correct options provided).

D Prompt

D.1 Generation prompt

In Table 7, we detailed all the prompts for generation.

Dataset	Prompt Source
GSM8K	CoT (Wei et al., 2022)
MATH	CoT Hub (Fu et al., 2023b)
AQuA	Self-Consistency (Wang et al., 2022)
CSQA	CoT (Wei et al., 2022)
AmbigNQ	CRITIC (Gou et al., 2023a)
StrategyQA	CoT (Wei et al., 2022)
HotpotQA	Self-Consistency (Wang et al., 2022)
Penguins	BIG-Bench Hard (Suzgun et al., 2022)
Colored Object	BIG-Bench Hard (Suzgun et al., 2022)
Date	BIG-Bench Hard (Suzgun et al., 2022)
MBPP	DeepSeek-Coder (Guo et al., 2024)
Object Counting	BIG-Bench Hard (Suzgun et al., 2022)
Repeat Copy	ART (Paranjape et al., 2023)
TabMWP	Written by Authors
HumanEval	Written by Authors

Table 7: Generation Prompts and Sources

D.2 Critique prompt

Question: {question} Answer: {response} Review your previous answer and find problems with your answer. Finally, conclude with either '[[correct]]' if the above answer is correct or '[[wrong]]' if it is incorrect. Verdict:

Table 8: Zero-shot Answer Only Prompt for Critique

Question: {question} Answer: {response} Review your previous answer and find problems with your answer. Provide your analysis and verdict in JSON format as follows: {"analysis": "your analysis", "verdict": "correct or wrong"}. Analysis and verdict:

Table 9: Zero-shot Chain of Thought Prompt for Critique

Review the above answer and determine whether it's correct. If wrong, find the problems with it.
Finally, conclude with either '[[correct]]' if the above answer is correct or '[[wrong]]' if it is incorrect.
Here are some examples:
—

Question: To locate a choker not located in a jewelry box or boutique where would you go? Answer Choices: (a) jewelry store (b) neck (c) jewlery box (d) jewelry box (e) boutique
Answer: If you want to locate a choker that is not in a jewelry box or boutique, you would most likely go to your neck (b) since a choker is a type of necklace that is worn around the neck. So the answer is (b).
Verdict: The question asks where you would find a choker if it's not located in a jewelry box or boutique. The given answer suggests that the choker would be found on your neck, which is a logical place for a choker when it is being worn. However, the question seems to be seeking a location where one might find a choker for purchase or retrieval other than a jewelry box or boutique. The options provided are (a) jewelry store, (b) neck, (c) jewelry box, (d) jewelry box, and (e) boutique.
Considering the context, option (a) jewelry store is the most appropriate answer. A jewelry store is a common place to find chokers for purchase or display, apart from a jewelry box or boutique. The given answer assumes the choker is being worn, which is not necessarily implied by the question. So the verdict is: [[wrong]]

—
.....
—

Question: Stanley had a dream that was very vivid and scary. He had trouble telling it from what? Answer Choices: (a) reality (b) imagination (c) dreamworker (d) nightmare (e) awake
Answer: The answer should be something that Stanley had trouble distinguishing his dream from. In this case, the answer is (a) reality. Stanley had trouble telling his vivid and scary dream from reality. So the answer is (a).
Verdict: The question asks what Stanley had trouble distinguishing his vivid and scary dream from. The given answer states that the correct choice is (a) reality, indicating that Stanley had difficulty telling his dream apart from real life. This interpretation is consistent with the typical use of the phrase "had trouble telling it from" in the context of dreams. It suggests that the dream felt so real to Stanley that he struggled to differentiate it from actual events or experiences. So the verdict is: [[correct]]

—

Please consider the example provided and think it step by step.
Question: {question}
Answer: {response}
Verdict:

Table 10: Few-shot Prompt (4-shot) of CSQA for Critique

Review the above answer and determine whether it's correct. If wrong, find the problems with it. Finally, conclude with either '[[correct]]' if the above answer is correct or '[[wrong]]' if it is incorrect. Here are some examples:
—

Question: Based on when the project was being worked on, when did the Manhattan project begin and end?
Answer: The Manhattan Project began in 1939 and ended in 1946. It was a research and development project during World War II that produced the first nuclear weapons. So the answer is: Began 1939, end 1946
Verdict: The Manhattan Project actually began in 1942, not 1939, and was completed in 1946. It was indeed a research and development project during World War II that led to the creation of the first nuclear weapons. The incorrect start date in the answer makes it inaccurate. So the verdict is: [[wrong]]

—

Question: Who helped lewis and clark on their journey?
Answer: Lewis and Clark were helped on their journey by the Corps of Discovery, which included members like Sacagawea, who served as an interpreter and guide. So the answer is: Sacagawea
Verdict: The answer to the question ""Who helped Lewis and Clark on their journey?"" is correct. Lewis and Clark were assisted by the Corps of Discovery, which included Sacagawea, an important member who served as an interpreter and guide. Therefore, the answer identifying Sacagawea as a key helper is accurate. So the verdict is: [[correct]]

—

Question: Who does the voice of cortana in halo video games?
Answer: The voice of Cortana in the Halo video games is provided by actress Jen Taylor. So the answer is: Jen Taylor
Verdict: The answer provided for the voice of Cortana in the Halo video games is correct. Jen Taylor is indeed the actress who voices Cortana in the series. So the verdict is: [[correct]]

—

Question: Who is lin's father in legend of korra?
Answer: In the animated series ""The Legend of Korra,"" Lin Beifong's father is Toph Beifong, who is a prominent character in the Avatar universe and one of the original members of Team Avatar from the preceding series, ""Avatar: The Last Airbender."" So the answer is: Toph Beifong
Verdict: In ""The Legend of Korra,"" Lin Beifong's father is not Toph Beifong. Toph Beifong is Lin's mother. Lin's father is revealed to be a man named Kanto. This information is disclosed in season 4 of the series, making the original answer incorrect. So the verdict is: [[wrong]]

—

Please consider the example provided and think it step by step.
Question: {question}
Answer: {response}
Verdict:

Table 11: Few-shot Prompt (4-shot) of AmbigNQ for Critique

Review the above answer and determine whether it's correct. If wrong, find the problems with it.
Finally, conclude with either '[[correct]]' if the above answer is correct or '[[wrong]]' if it is incorrect.
Here are some examples:
—

Question: Herman likes to feed the birds in December, January and February. He feeds them 1/2 cup in the morning and 1/2 cup in the afternoon. How many cups of food will he need for all three months?
Answer: Herman feeds the birds 1/2 cup in the morning and 1/2 cup in the afternoon, so each day he uses 1/2 + 1/2 = 1 cup of food. There are 3 months in total, so he will need 3 * 1 = 3 cups of food for all three months. The answer is 3.
Verdict: The answer calculates Herman feeding the birds 1 cup of food per day (1/2 cup in the morning and 1/2 cup in the afternoon). However, it incorrectly multiplies this daily amount by 3, assuming there are only 3 days in three months, resulting in 3 cups of food for all three months. This is a significant error in understanding the duration of three months.
So the verdict is: [[wrong]]

—
.....
—

Question: Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?
Answer: Weng earns $12 per hour for babysitting. Yesterday, she babysat for 50 minutes, which is equivalent to 50/60 = 5/6 of an hour. To calculate her earnings, multiply her hourly rate by the fraction of an hour she worked: $12 * 5/6 =$10. Weng earned $10 for her 50 minutes of babysitting. The answer is 10.
Verdict: The given answer accurately calculates Weng's earnings for babysitting for 50 minutes. It correctly identifies that 50 minutes is 5/6 of an hour and then multiplies this fraction by her hourly rate of $12. The calculation of $12 * 5/6 equals $10, which is the correct amount Weng earned for her 50 minutes of babysitting.
So the verdict is: [[correct]]
—

Please consider the example provided and think it step by step.
Question: {question}
Answer: {response}
Verdict:

Table 12: Few-shot Prompt (4-shot) of GSM8K for Critique

Review the above answer and determine whether it's correct. If wrong, find the problems with it.
Finally, conclude with either '[[correct]]' if the above answer is correct or '[[wrong]]' if it is incorrect.
Here are some examples:
—
.....
—

Table:
[TITLE]: Ferry fares
Ferry | Car | Bicycle
Seattle-Bremerton | $12 | $8
Mukilteu-Clinton | $7 | $5
Southport-Fort Fisher | $5 | $2
Fauntleroy-Vashon | $15 | $5
Ocracoke | $15 | $3
Question: For an economics project, Layla determined the cost of ferry rides for bicycles and cars. How much does it cost to take a bicycle on the Southport-Fort Fisher ferry? Answer: To find the cost of taking a bicycle on the Southport-Fort Fisher ferry, you can refer to the table:
Southport-Fort Fisher | $5 | $2
The cost for a bicycle on the Southport-Fort Fisher ferry is $2.
So, it costs $2 to take a bicycle on this ferry. The answer is 2.
Verdict:
Upon reviewing the provided answer:
The question asks for the cost of taking a bicycle on the Southport-Fort Fisher ferry. The table provides the cost of ferry rides for both cars and bicycles.
The entry for the Southport-Fort Fisher ferry shows the cost for a bicycle as $2.
The answer correctly identifies this value from the table and states that it costs $2 to take a bicycle on the Southport-Fort Fisher ferry. No calculations are needed as the information is directly available in the table.
The answer accurately extracts the relevant data from the table and correctly identifies the cost of taking a bicycle on the Southport-Fort Fisher ferry as $2.
So the verdict is: [[correct]]

—

Please consider the example provided and think it step by step.
Question: {question}
Answer: {response}
Verdict:

Table 13: Few-shot Prompt (4-shot) of TabMWP for Critique

Review the provided code and determine whether it's correct. If wrong, find the problems with it. Finally, conclude with either '[[correct]]' if the provided answer is correct or '[[wrong]]' if it is incorrect. Here are some examples:

—

Code:

```
def last_digit_sum(a: int, b: int) -> int:
    """
    Complete the function that takes two integers and returns the sum of their last digits.
    Assume the input is always valid.

    Examples:
    last_digit_sum(123, 453) should return 6 (3 + 3).
    last_digit_sum(789, 10) should return 9 (9 + 0).
    last_digit_sum(100, 27) should return 7 (0 + 7).
    last_digit_sum(25, -31) should return 6 (5 + 1).
    """
    return abs(a \% 100) + abs(b \% 100)
```

Verdict: The provided function last_digit_sum is supposed to return the sum of the last digits of two integers, but the current implementation returns the sum of the absolute values of the last two digits of each integer. The examples in the problem statement also indicate that the function should return the actual sum of the last digits, not their absolute values.
So the verdict is: [[wrong]]

—

...

—

Code:

```
from typing import List

def has_similar_word_lengths(words: List[str], threshold: int) -> bool:
    """
    Check if in the given list of words, are any two words whose length difference is less than or equal to
    the given threshold.
    >>> has_similar_word_lengths(["hello", "world", "hi", "python"], 2)
    True
    >>> has_similar_word_lengths(["chat", "learning", "ai"], 1)
    False
    """
    for i in range(len(words)):
        for j in range(i + 1, len(words)):
            if abs(len(words[i]) - len(words[j])) <= threshold:
                return True
    return False
```

Verdict: The provided code iterates over all unique pairs of words by using two nested loops, compares the lengths of the words, and returns True if it finds a pair that satisfies the condition (length difference is less than or equal to the threshold). If no such pair is found, it returns False.
So the verdict is: [[correct]]

—

Please consider the example provided and think it step by step.
Code:
{code}
Verdict:

Table 14: Few-shot Prompt (4-shot) of HumanEval for Critique

Review the provided answer and determine whether it's correct. If wrong, find the problems with it. Finally, conclude with either '[[correct]]' if the provided answer is correct or '[[wrong]]' if it is incorrect. Here are some examples:
—

Question: Say pickup a pound of green beans twice, replacing a pound with a bunch for even times and a handful for odd
Answer: pickup a handful of green beans pickup a bunch of green beans
Verdict: Let's break down the instruction:
The instruction is to "pick up a pound of green beans twice".
For even times, replace "a pound" with "a bunch".
For odd times, replace "a pound" with "a handful".
Following these instructions:
The first time (odd), "pick up a pound of green beans" becomes "pick up a handful of green beans".
The second time (even), "pick up a pound of green beans" becomes "pick up a bunch of green beans".
Therefore, the provided answer of "pick up a handful of green beans pick up a bunch of green beans" matches the instructions given.
So the verdict is: [[correct]]

—
...
—

Question: Say stack ten bricks three times, replacing ten with twenty for even times and five for odd times.
Answer: stack five bricks stack twenty bricks stack ten bricks.
Verdict: According to the instruction:
Say "stack ten bricks" three times.
Replace "ten" with "twenty" for even times.
Replace "ten" with "five" for odd times.
Applying these rules:
First time (odd): Replace "ten" with "five", resulting in "stack five bricks".
Second time (even): Replace "ten" with "twenty", resulting in "stack twenty bricks".
Third time (odd again): Replace "ten" with "five", resulting in "stack five bricks".
The provided answer incorrectly says "stack ten bricks" for the third time instead of replacing "ten" with "five". So the verdict is: [[wrong]]

—

Please consider the example provided and think it step by step.
Question: {question}
Answer: {solution}
Verdict:

Table 15: Few-shot Prompt (4-shot) of Repeat Copy for Critique

D.3 Correct prompt

Question: {question} Answer: {answer} Analysis and verdict: {critique} Based on the problems you found, improve your answer. Please reiterate your solution, with your final answer as either a single numerical number, a single word, or a short phrase, in the JSON format as follows: {"solution":"improved solution", "answer":"your answer"}. Improved solution and answer:

Table 16: Zero-shot Chain of Thought Prompt for Correction

Based on the problems you found, improve your answer. You must choose only one option from A to E. Please reiterate your answer, with your final answer a single letter from A to E, in the form [[answer]]. Here are some examples:
—

Question: To locate a choker not located in a jewelry box or boutique where would you go? Answer Choices: (a) jewelry store (b) neck (c) jewlery box (d) jewelry box (e) boutique
Answer: If you want to locate a choker that is not in a jewelry box or boutique, you would most likely go to your neck (b) since a choker is a type of necklace that is worn around the neck. So the answer is (b).
Verdict: The question asks where you would find a choker if it's not located in a jewelry box or boutique. The given answer suggests that the choker would be found on your neck, which is a logical place for a choker when it is being worn.
However, the question seems to be seeking a location where one might find a choker for purchase or retrieval other than a jewelry box or boutique. The options provided are (a) jewelry store, (b) neck, (c) jewelry box, (d) jewelry box, and (e) boutique.
Considering the context, option (a) jewelry store is the most appropriate answer. A jewelry store is a common place to find chokers for purchase or display, apart from a jewelry box or boutique. The given answer assumes the choker is being worn, which is not necessarily implied by the question.
So the verdict is: [[wrong]]
Improved Answer: The improved answer is: [[a]] jewelry store.

—
.....
—

Question: Stanley had a dream that was very vivid and scary. He had trouble telling it from what? Answer Choices: (a) reality (b) imagination (c) dreamworker (d) nightmare (e) awake
Answer: The answer should be something that Stanley had trouble distinguishing his dream from. In this case, the answer is (a) reality. Stanley had trouble telling his vivid and scary dream from reality. So the answer is (a).
Verdict: The question asks what Stanley had trouble distinguishing his vivid and scary dream from. The given answer states that the correct choice is (a) reality, indicating that Stanley had difficulty telling his dream apart from real life. This interpretation is consistent with the typical use of the phrase "had trouble telling it from" in the context of dreams. It suggests that the dream felt so real to Stanley that he struggled to differentiate it from actual events or experiences. So the verdict is: [[correct]]
Improved Answer: The answer is [[a]]

—

Question: {question}
Answer: {answer}
Verdict: {critique}
Improved Answer:

Table 17: Few-shot Prompt (4-shot) of CSQA for Correction

Based on the problems you found, improve your answer. Please reiterate your answer, with your final answer as either a single numerical number, a single word, or a short phrase, in the form [[answer]].
Here are some examples:
—

Question: Based on when the project was being worked on, when did the Manhattan project begin and end?
Answer: The Manhattan Project began in 1939 and ended in 1946. It was a research and development project during World War II that produced the first nuclear weapons. So the answer is: Began 1939, end 1946
Verdict: The Manhattan Project actually began in 1942, not 1939, and was completed in 1946. It was indeed a research and development project during World War II that led to the creation of the first nuclear weapons. The incorrect start date in the answer makes it inaccurate. So the verdict is: [[wrong]]
Improved Answer: The Manhattan Project [[began in 1942 and ended in 1946]].

—

Question: Who helped lewis and clark on their journey?
Answer: Lewis and Clark were helped on their journey by the Corps of Discovery, which included members like Sacagawea, who served as an interpreter and guide. So the answer is: Sacagawea
Verdict: The answer to the question "Who helped Lewis and Clark on their journey?" is correct. Lewis and Clark were assisted by the Corps of Discovery, which included Sacagawea, an important member who served as an interpreter and guide. Therefore, the answer identifying Sacagawea as a key helper is accurate. So the verdict is: [[correct]]
Improved Answer: [[Sacagawea]] helped Lewis and Clark on their journey.

—

Question: Who does the voice of cortana in halo video games?
Answer: The voice of Cortana in the Halo video games is provided by actress Jen Taylor. So the answer is: Jen Taylor
Verdict: The answer provided for the voice of Cortana in the Halo video games is correct. Jen Taylor is indeed the actress who voices Cortana in the series. So the verdict is: [[correct]]
Improved Answer: [[Jen Taylor]]

—

Question: Who is lin's father in legend of korra?
Answer: In the animated series "The Legend of Korra," Lin Beifong's father is Toph Beifong, who is a prominent character in the Avatar universe and one of the original members of Team Avatar from the preceding series, "Avatar: The Last Airbender." So the answer is: Toph Beifong
Verdict: In "The Legend of Korra," Lin Beifong's father is not Toph Beifong. Toph Beifong is Lin's mother. Lin's father is revealed to be a man named Kanto. This information is disclosed in season 4 of the series, making the original answer incorrect. So the verdict is: [[wrong]] Improved Answer: Lin Beifong's father in "The Legend of Korra" is [[Kanto]].

—

Question: {question}
Answer: {answer}
Verdict: {critique}
Improved Answer:

Table 18: Few-shot Prompt (4-shot) of AmbigNQ for Correction

Based on the problems you found, improve your answer. Please reiterate your answer, with your final answer a single numerical number, in the form [[answer]].
Here are some examples:
—

Question: Herman likes to feed the birds in December, January and February. He feeds them 1/2 cup in the morning and 1/2 cup in the afternoon. How many cups of food will he need for all three months?
Answer: Herman feeds the birds 1/2 cup in the morning and 1/2 cup in the afternoon, so each day he uses 1/2 + 1/2 = 1 cup of food. There are 3 months in total, so he will need 3 * 1 = 3 cups of food for all three months. The answer is 3.
Verdict: The answer calculates Herman feeding the birds 1 cup of food per day (1/2 cup in the morning and 1/2 cup in the afternoon). However, it incorrectly multiplies this daily amount by 3, assuming there are only 3 days in three months, resulting in 3 cups of food for all three months. This is a significant error in understanding the duration of three months. So the verdict is: [[wrong]]
Improved Answer: To calculate the amount of food Herman will need for all three months, we need to consider the number of days in December, January, and February. December and January typically have 31 days each, while February can have 28 or 29 days, depending on whether it's a leap year. Assuming a non-leap year, February has 28 days. Therefore, the total number of days is 31 (December) + 31 (January) + 28 (February) = 90 days. Since Herman feeds the birds 1 cup of food each day (1/2 cup in the morning and 1/2 cup in the afternoon), the total amount of food he will need for all three months is 90 cups. The answer is [[90]].

—
.....
—

Question: Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?
Answer: Weng earns $12 per hour for babysitting. Yesterday, she babysat for 50 minutes, which is equivalent to 50/60 = 5/6 of an hour. To calculate her earnings, multiply her hourly rate by the fraction of an hour she worked: $12 * 5/6 = $10. Weng earned $10 for her 50 minutes of babysitting. The answer is 10.
Verdict: The given answer accurately calculates Weng's earnings for babysitting for 50 minutes. It correctly identifies that 50 minutes is 5/6 of an hour and then multiplies this fraction by her hourly rate of $12. The calculation of $12 * 5/6 equals $10, which is the correct amount Weng earned for her 50 minutes of babysitting.
So the verdict is: [[correct]]
Improved Answer: The answer is [[10]].

—

Question: {question}
Answer: {answer}
Verdict: {critique}
Improved Answer:

Table 19: Few-shot Prompt (4-shot) of GSM8K for Correction

Based on the problems you found, improve your answer. Please reiterate your answer, with your final answer as either a single numerical number or one option from A to D, in the form [[answer]].
Here are some examples:
—
.....
—

Table:
[TITLE]: Ferry fares
Ferry | Car | Bicycle
Seattle-Bremerton | $12 | $8
Mukilteu-Clinton | $7 | $5
Southport-Fort Fisher | $5 | $2
Fauntleroy-Vashon | $15 | $5
Ocracoke | $15 | $3
Question: For an economics project, Layla determined the cost of ferry rides for bicycles and cars. How much does it cost to take a bicycle on the Southport-Fort Fisher ferry? Answer: To find the cost of taking a bicycle on the Southport-Fort Fisher ferry, you can refer to the table:
Southport-Fort Fisher | $5 | $2
The cost for a bicycle on the Southport-Fort Fisher ferry is $2.
So, it costs $2 to take a bicycle on this ferry. The answer is 2.
Verdict:
Upon reviewing the provided answer:
The question asks for the cost of taking a bicycle on the Southport-Fort Fisher ferry. The table provides the cost of ferry rides for both cars and bicycles.
The entry for the Southport-Fort Fisher ferry shows the cost for a bicycle as $2.
The answer correctly identifies this value from the table and states that it costs $2 to take a bicycle on the Southport-Fort Fisher ferry. No calculations are needed as the information is directly available in the table.
The answer accurately extracts the relevant data from the table and correctly identifies the cost of taking a bicycle on the Southport-Fort Fisher ferry as $2.
So the verdict is: [[correct]]
Improved Answer: The answer is [[2]].

—

Question: {question}
Answer: {answer}
Verdict: {critique}
Improved Answer:

Table 20: Few-shot Prompt (4-shot) of TabMWP for Correction

Based on the problems you found, improve your code in the form [BEGIN]\nimproved code\n[DONE]
Here are some examples:
—
.....
—
Code:

```
from typing import List

def has_similar_word_lengths(words: List[str], threshold: int) -> bool:
    """
    Check if in the given list of words, are any two words whose length difference is less than or equal to
    the given threshold.
    >>> has_similar_word_lengths(["hello", "world", "hi", "python"], 2)
    True
    >>> has_similar_word_lengths(["chat", "learning", "ai"], 1)
    False
    """
    for i in range(len(words)):
        for j in range(i + 1, len(words)):
            if abs(len(words[i]) - len(words[j])) <= threshold:
                return True
    return False
```

Verdict: The provided code iterates over all unique pairs of words by using two nested loops, compares the lengths of the words, and returns True if it finds a pair that satisfies the condition (length difference is less than or equal to the threshold). If no such pair is found, it returns False.
So the verdict is: [[correct]]
Improved Code:
[BEGIN]

```
from typing import List

def has_similar_word_lengths(words: List[str], threshold: int) -> bool:
    """
    Check if in the given list of words, are any two words whose length difference is less than or equal to
    the given threshold.
    >>> has_similar_word_lengths(["hello", "world", "hi", "python"], 2)
    True
    >>> has_similar_word_lengths(["chat", "learning", "ai"], 1)
    False
    """
    for i in range(len(words)):
        for j in range(i + 1, len(words)):
            if abs(len(words[i]) - len(words[j])) <= threshold:
                return True
    return False
```

[DONE]

—

Code:
{code}
Verdict: {critique}
Improved Code:

Table 21: Few-shot Prompt (4-shot) of HumanEval for Correction

Based on the problems you found, improve your answer. Please reiterate your answer, with your final answer in the form [[answer]].
Here are some examples:
—

Question: Say pickup a pound of green beans twice, replacing a pound with a bunch for even times and a handful for odd
Answer: pickup a handful of green beans pickup a bunch of green beans Verdict: Let's break down the instruction:
The instruction is to "pick up a pound of green beans twice".
For even times, replace "a pound" with "a bunch".
For odd times, replace "a pound" with "a handful".
Following these instructions:
The first time (odd), "pick up a pound of green beans" becomes "pick up a handful of green beans".
The second time (even), "pick up a pound of green beans" becomes "pick up a bunch of green beans".
Therefore, the provided answer of "pick up a handful of green beans pick up a bunch of green beans" matches the instructions given. So the verdict is: [[correct]]
Improved Answer: [[pickup a handful of green beans pickup a bunch of green beans]]

—
.....
—

Question: Say stack ten bricks three times, replacing ten with twenty for even times and five for odd times.
Answer: stack five bricks stack twenty bricks stack ten bricks.
Verdict: According to the instruction:
Say "stack ten bricks" three times.
Replace "ten" with "twenty" for even times.
Replace "ten" with "five" for odd times.
Applying these rules:
First time (odd): Replace "ten" with "five", resulting in "stack five bricks".
Second time (even): Replace "ten" with "twenty", resulting in "stack twenty bricks".
Third time (odd again): Replace "ten" with "five", resulting in "stack five bricks".
The provided answer incorrectly says "stack ten bricks" for the third time instead of replacing "ten" with "five".
So the verdict is: [[wrong]]
Improved Answer: [[stack five bricks stack twenty bricks stack five bricks]]

—

Question: {question}
Answer: {answer}
Verdict: {critique}
Improved Answer:

Table 22: Few-shot Prompt (4-shot) of Repeat Copy for Correction