Self-Contrast: Better Reflection Through Inconsistent Solving Perspectives

Anonymous ACL submission

Abstract

The reflection capacity of Large Language Model (LLM) has garnered extensive attention. A post-hoc prompting strategy, e.g., reflexion and self-refine, refines LLM's response based on self-evaluated or external feedback. However, recent research indicates without external feedback, LLM's intrinsic reflection is unstable. Our investigation unveils that the key bottleneck is the quality of the self-evaluated feedback. We find LLMs often exhibit overconfidence or high randomness when self-evaluate, offering stubborn or inconsistent feedback, which causes poor reflection. To remedy this, we advocate **Self-Contrast**: It adaptively **explores** diverse solving perspectives tailored to the request, **contrasts** the differences, and **summarizes** these discrepancies into a checklist which could be used to re-examine and eliminate discrepancies. Our method endows LLM with diverse perspectives to alleviate stubborn biases. Moreover, their discrepancies indicate potential errors or inherent uncertainties that LLM often overlooks. Reflecting upon these can catalyze more accurate and stable reflection. Experiments conducted on a series of reasoning and translation tasks with different LLMs serve to underscore the effectiveness and generality of our strategy.

1 Introduction

Mastering reasoning and decision-making capabilities is of utmost importance to paving the way for artificial general intelligence. Recently, large language models (LLMs) (Brown et al., 2020; Chowdhery et al., 2022; Zhang et al., 2022a; Zeng et al., 2023; Touvron et al., 2023a; OpenAI, 2022, 2023; Touvron et al., 2023b) and applications built on them (Schick et al., 2023; Wu et al., 2023a; Shen et al., 2023; Zhang et al., 2023a) demonstrate impressive capabilities in various domains, especially combined with Chain-of-Thought (Wei et al., 2022; Kojima et al., 2022), ReAct (Yao et al., 2022), Tree-of-Thought (Yao et al., 2023) and other prompting

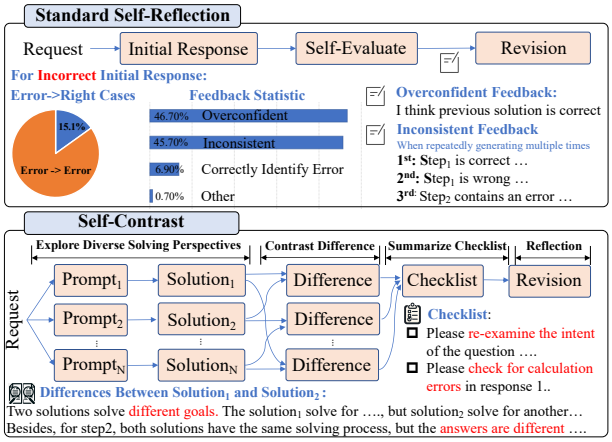

Figure 1: LLMs evaluate the initial response and provide feedback for revision. However, most erroneous responses remain uncorrected after reflection as the feedback is either overconfident (46.7%) or inconsistent (45.7%). Bottom: Self-Contrast explores multiple solving perspectives, and contrast their differences, and summarize them into insightful checklist for self-correction.

techniques (Gao et al., 2022; Wang et al., 2023d; Zhou et al., 2022; Besta et al., 2023).

Despite these advancements, LLMs are not entirely reliable (Zheng et al., 2023c; Frieder et al., 2023; Yuan et al., 2023b) since they frequently produce inaccuracies results, such as misunderstanding a key concept, overlooking some crucial details. A post-hoc prompting strategy, e.g., self-reflection, garnered considerable attention (Shinn et al., 2023; Madaan et al., 2023; Paul et al., 2023). It first generates an initial response (Initial Response), then gathers external feedback or self-evaluated feedback (Evaluation Phase) to refine prior response (Revision) (Welleck et al., 2022; Kadavath et al., 2022; Chen et al., 2023d; Liang et al., 2023; Kim et al., 2023; Zheng et al., 2023a; Du et al., 2023; Xi et al., 2023; Ganguli et al., 2023; Pan et al., 2023). Numerous studies proclaim this three-stage strategy (Initial Response→Evaluation→Revision), can endow LLMs with the potential to self-correct pre-

vious imperfect responses. For a time, this belief appeared to dominate the community.

However, recent studies (Huang et al., 2023b; Stechly et al., 2023; Liang et al., 2023; Valmeekam et al., 2023) have cast doubt on LLM's inherent reflection capability. Their research indicates that without external feedback, LLMs have difficulties in amending prior responses. It implies self-correction is unreliable when relying only on LLM itself and simple post-hoc prompting strategies.

We are also intrigued by LLM's internal reflection ability, as external feedback is not available in most scenarios. Our initial experiments (§ 2.1) indicate that intrinsic reflection has limited effect. Across various LLMs and tasks, the performance gains from reflection are not significant, and occasionally detrimental. In cases of incorrect initial responses, only 15.1% of incorrect responses are corrected through reflection. To ascertain the reasons for that, we further analyze the feedback generated by the self-evaluate process. As shown in Figure 1, LLMs often provide two unexpected feedback: 1) *Overconfidence* (46.7%): Stubbornly insisting that the previous solution is correct. 2) *Inconsistency* (45.7%): The feedback is highly inconsistent when self-evaluating the same response multiple times. These two feedbacks seriously undermine the effectiveness of reflection. It reveals that such a simple self-evaluate strategy faces difficulty in accurately identifying errors and consistently generating high-quality feedback for reflection.

As a remedy, we propose a contrastive strategy as an alternative to the direct evaluation: we examine the differences among multiple responses and draw inspiration to derive feedbacks from their disparities for reflection. The insight is that while generating accurate feedback directly may be challenging, identifying contrasts between various responses is often more feasible. More importantly, these discrepancies often indicate some potential errors, easily overlooked details or pitfalls. As shown in Figure 1, by contrasting two solutions, LLM finds they have different solving objectives, and suggests re-examining the intent of the original request in the checklist.

Embracing this philosophy, we advocate Self-Contrast, which steers LLM to autonomously create diverse solving perspectives by self-curated prompts and then select different results with significant discrepancies for comparison. Then LLM reflects on the reasons behind these discrepancies and generates multiple re-examining instructions, i.e., checklist, for reflection. Our experiments show that by creating diverse perspectives adaptively, Self-Contrast can mitigate biases introduced by specific prompts. Moreover, contrasting the discrepancies between perspectives inspires deeper reflection, thereby enhancing the likelihood of accurate self-correction.

Our contributions can be summarized as:

- Our comprehensive experiments unveil that the bottleneck for poor reflection performance lies in the LLM's inability to accurately evaluate prior responses. It often manifests as overconfident or inconsistent feedback, hindering the effectiveness of self-reflection.
- We advocate Self-Contrast: LLMs create multiple solving perspectives for diverse results, mitigating overconfident biases of a singular prompt. Then drawing inspiration from contrasting different perspectives, LLM summarizes more accurate checking instructions to resolve discrepancies and enhance reflection.
- Empirically, compared with vanilla reflection, Self-Contrast shows significant improvements and stability in both mathematical reasoning and challenging translation scenarios.

2 Evaluation of Intrinsic Reflection

We first comprehensively investigate the intrinsic reflection capability of LLMs, i.e., LLMs self-evaluate the initial response without external feedback and then refine it. Subsequently, we methodically investigate the factors influencing reflection.

2.1 Performance Pre- and Post-Reflection

We evaluate the reflection capabilities of multiple LLMs across a variety of benchmarks, including math reasoning and creative translation tasks. We report average accuracy for math reasoning and the BLEURT score between predicted sentences and references for the translation task (see § 4.1 for detail). Each result is evaluated multiple times on different prompts. Besides, we also report the significance level (one-tailed t-test) of the accuracy change pre- and post-reflection.

As shown in Table 1, we observe no significant accuracy changes before and after reflection. For instance, the performance of GPT-3.5 on GSM8K and SVAMP exhibit marginal changes of -0.8% and +0.7% after reflection respectively, both statistically insignificant. This negligible performance fluctuation can be validated across multiple LLMs and

	Math Reasoning		Translation
	GSM8K	SVAMP	CommonMT
GPT4	93.9⇒95.1	93.0⇒91.5	70.1⇒69.8
$t(\Delta>0)\uparrow$	0.91	-0.22	-0.11
GPT3.5	76.6⇒75.8	79.8⇒80.5	69.1⇒69.3
$t(\Delta>0)\uparrow$	-0.43	0.18	0.15
davinci-003	51.1⇒49.6	63⇒63.5	62.4⇒63.8
$t(\Delta>0)\uparrow$	-0.54	0.07	0.88
Llama2-70B	52.6⇒49.3	66⇒63.0	63.2⇒62.2
$t(\Delta>0)\uparrow$	-1.06	-1.86	-0.78
Llama2-13B	28.3⇒29.8	42.2⇒42.5	62.5⇒61.5
$t(\Delta>0)\uparrow$	0.3	0.7	0.08
Llama2-7B	19.8⇒17.0	37.5⇒36.1	53.7⇒48.8
$t(\Delta>0)\uparrow$	-1.94	-0.2	-0.7

Table 1: We calculate the average accuracy of the ten experiments for pre- and post-reflection: Pre Acc. ⇒ Post Acc. We also report the significance level of the accuracy change for ten trials, where $\Delta = Post - Pre$.

	#Invalid: 269 #Valid: 48 #Toxic: 52	Reflection Behavior		
		Invalid ✗⇒✗	Valid ✗⇒✓	Toxic ✓⇒✗
Feedback Type	I. Accurately identifies errors	0.4%	43.3%	0%
	II. Stubbornly offers erroneous feedback	0.8%	0%	31.1%
	III. Can not output consistent feedback	45.3%	**47.5%**	**65.4%**
	IV. Overconfidence No revision required	**53.5%**	9.2%	3.5%

Table 2: We consider three reflection behaviors based on the correctness of the pre- and post-reflection: Invalid, Valid, and Toxic. Besides, we summarize each sample's feedback into four categories when self-evaluation.

various benchmarks, far from expectations. Specifically, most reasoning cases suffer from a slight decrease, while the translation task shows little impact. Additionally, smaller LLMs (e.g., Llama2-7B) demonstrate poorer reflection ability, occasionally even exhibiting negative impacts. These experiments collectively suggest that LLMs appear to be incapable of self-correction through reflection.

2.2 Feedback Analysis

To investigate the reasons behind the failure of reflection, we further analyze the feedback generated during the self-evaluate process. We classify all samples in GSM8K into four categories based on their correctness of the pre- and post-reflection: 1) *Invalid Reflection* (✗⇒✗) means the results before and after reflection are both incorrect. 2) *Valid Reflection* (✗⇒✓) means a wrong solution is revised to correct through reflection. 3) *Toxic Reflection* (✓⇒✗) represents that an originally correct response is changed to incorrect after reflection. 4) *Others* counts the number of correct ⇒ correct.

Fail to Correct the Wrong Initial Response. As shown in Table 2, we observe the number of *Toxic Reflection* (✓⇒✗: 52) and *Valid Reflection* (✗⇒✓: 48) are nearly similar. This explains why there is no discernible difference in performance pre- and post-reflection. Besides, considering the scenario when the initial response is erroneous, we observe the number of *Invalid Reflection* (✗⇒✗: 269) is significantly larger than *Valid Reflection* (✗⇒✓: 48), which indicates LLM fails to correct errors in the initial responses for most cases.

Often Provide Overconfident or Inconsistent Feedback. We examine whether LLMs could generate feedback accurately and consistently. For each sample, we instruct LLM to evaluate its initial response multiple times and record multiple feedbacks. We manually assess the consistency and correctness of these feedbacks and then summarize each sample into 4 cases: I. *Accurately identifies errors*: In multiple repeated evaluations, LLM identifies errors and provides accurate and consistent feedback. II. *Stubbornly offers erroneous feedback*: The majority of evaluations provide incorrect feedback with specific errors. III. *Can not output consistent feedback*: Unable to assess consistently, as most feedback is different and quite random for a same initial response. V. *Overconfidence, no revision required*: LLM is overconfident and believes no revision is necessary.

As shown in Table 2, for the majority of Invalid Reflection, their feedback is either overconfident (53.5%) or highly inconsistent (45.3%), making it difficult to prompt reliable reflection. Similarly, in Toxic Reflection scenarios, 65.4% of the evaluation processes are highly inconsistent, leading to many correct answers being erroneously modified.

2.3 From Self-Evaluate to Self-Contrast

The aforementioned experiments indicate that feedback generated by the self-evaluate process is either highly random or excessively confident. This unstable self-evaluate may severely impact the reflection performance of LLMs.

As a remedy, we propose a contrastive strategy for reflection. Instead of directly evaluating a response, which can be challenging and inconsistent, we instruct LLM to initially contrast the differences between various solutions, and identify their discrepancies and the reasons behind them. As shown in Figure 1 (bottom), we sample Top-2 responses

Strategy	GSM8K	SVAMP	CommonMT
Self-Evaluate w/ top-1	-0.8	0.7	0.2
$t(\Delta>0)\uparrow$	-0.43	0.18	0.15
Self-Evaluate w/ top-2	0.12	0.8	0.16
$t(\Delta>0)\uparrow$	0.21	0.41	0.33
Self-Contrast w/ top-2	0.9	2.5	0.45
$t(\Delta>0)\uparrow$	**1.43**	**2.72**	**1.89**

Table 3: We report the accuracy change (Δ) between post- and pre-reflection for 3 settings and t-test value for Δ>0. Self-evaluate: Directly evaluate the initial response. Self-contrast: Contrast the difference between two responses and generate a checklist for reflection.

from LLM and then prompt LLM to contrast their differences in detail, rethink the reasons that caused the discrepancies, and summarize the checklist for re-examining and resolving the discrepancy. As shown in Table 3, we compare three scenarios: self-evaluate w/ top-1 response, self-evaluate w/ top-2 responses, and self-contrast w/ top-2. Our new strategy achieves a modest improvement over standard reflection using self-evaluate. Notably, it significantly enhances the significance levels (t-test: -0.43 to 1.43), suggesting it can greatly mitigate the uncertainty associated with self-evaluate process.

3 Self-Contrast

Prior sections illustrate the challenges LLMs encounter in accurately evaluating previous solutions, often resulting in overconfident or inconsistent feedback. Concurrently, we observe that leveraging the discrepancies between two different solutions can catalyze a more efficacious reflection, notably reducing the uncertainty during the reflection. Building upon this insight, we propose a more diverse inter-perspective Self-Contrast, facilitating more reliable self-reflection.

Self-Contrast consists of three procedures: Create Diverse Perspectives, Contrast Inter-Perspective Discrepancies, and Eliminate Discrepancies. In Create Diverse Perspectives (§ 3.1), we encourage LLMs to autonomously create a variety of prompts tailored to the user's request, each offering a unique perspective for problem-solving, e.g., different thinking styles, diverse identities, personalities, or preferences. These diverse perspectives prompt the LLM to generate different responses. In the second stage (§ 3.2), LLM contrasts the differences between each pair of responses. Lastly (§ 3.3), to eliminate discrepancies, we abstract these differences into a detailed checklist for re-examining. This checklist guides the LLM to meticulously examine the causes of discrepancies, including random errors or intrinsic biases, which result in inconsistent results among perspectives.

As shown in Figure 2, LLM designs five different prompts and their translation results based on the user's request ("这个计划被枪毙") . From a literal perspective, the phrase "被枪毙" is translated as "shot to death". This rigid translation fails to grasp the metaphor embedded in the military term. Conversely, from a liberal perspective, it is translated as "This plan was axed". After contrasting two different translations, LLMs believe they should scrutinize the source sentence for metaphors and ensure the translation aligns with the conventions of English expression.

3.1 Create Diverse Perspectives

Self-Curated Prompts First, it is imperative to define the concept of "solving perspective". It refers to deliberate prompting with a unique role, personality, thought style, etc., which prompts LLMs to solve user requests from a specific perspective. Diverse solving perspectives can endow LLMs with a broader range of thoughts for problem-solving, e.g., different angles and methodologies, thereby mitigating biases introduced by singular prompts.

To achieve this, we adopt a self-curated prompt strategy, where the LLM itself adaptively generates multiple different prompts for each request, each signifying a tailored perspective, then samples corresponding responses based on these prompts. It is noteworthy that the number of perspectives to be created, and the design of each perspective are entirely determined by LLMs, endowing them with more flexibility to address complex tasks. The details of the prompt are provided in Appendix D.1. In Figure 3, we present statistics on the number of prompts generated in self-curated prompt process.

3.2 Contrast Inter-Perspective Discrepancies

The LLM generates diverse responses based on self-curated prompts, each representing a specific perspective. Considering that some responses may be highly similar or even identical, we first filter these similar responses. Then, we select the responses with significant discrepancies for comparison.

Selecting To filter out similar responses, we employ the K-Medoids clustering algorithm based on their semantic similarity. We categorize all responses into k clusters, each encompassing a set of similar results. Then we select the centroids of each cluster as representative responses and discard

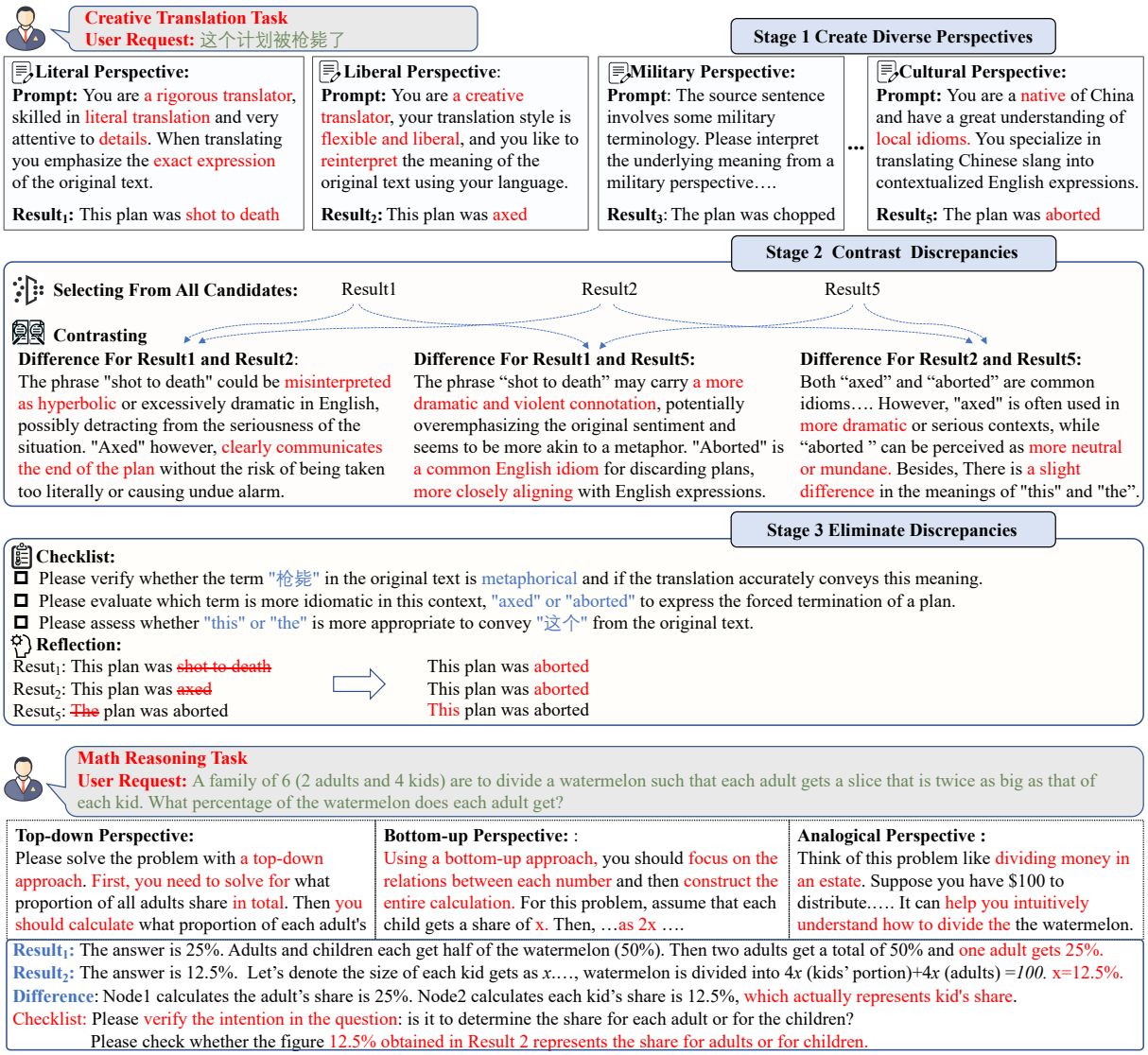

Figure 2: Self-Contrast designs diverse prompts for different solving perspectives and generates corresponding results. Then we filter out similar results and select those that are significantly different. To inspire reflection, we contrast the differences between selected results and prompt LLM to summarize a checklist. This checklist can be used to re-examine and eliminate discrepancies. Lastly, LLM revises each response to achieve a consistent result.

the remaining ones. It ensures the selected results exhibit substantial differences from each other.

Contrasting After selecting k responses from all candidates, we feed these responses concurrently into LLM and then instruct LLM to autonomously contrast the differences for each pair of responses in a single pass. When contrasting, LLMs need to explicitly answer these questions: **Whether** the two responses are different, **Where** the differences lie, and **Which factors** contributed to these inconsistent results. These questions guide the LLM to methodically explore the underlying reasons behind discrepancies, identifying potential errors and often overlooked details. As shown in Figure 2, for translation tasks, the LLM compares results 1, 2, and 5, and identifies that their primary differences lie in the use of different verbs to express "被枪毙". The detailed prompts are shown in Appendix D.2.

3.3 Eliminate Discrepancies

We abstract insightful checklists from these pairwise contrastive differences and then use them to resolve the inconsistencies across various perspectives for a consensus.

Summarizing Checklist To ascertain the truth and resolve discrepancies, the LLM is encouraged to summarize a detailed checklist for re-examining the user's request and candidate responses. This checklist contains multiple specialized checking instructions, such as verifying alignment with the

user's intent, identifying contradictions in LLM's responses, checking for miscalculations, etc. It explicitly points out some potential issues, e.g., previously overlooked details, logical pitfalls, or unreasonable steps, and compels LLM to re-examine them. Compared to conventional reflection instruction, e.g., *Please check your previous response*, our checklist is more precise and informative.

Reflection For Consensus Lastly, we employ the checklist and identified discrepancies to prompt reflection. LLM can revise the inconsistent perspectives and output k consistent responses.

Concretely, we use a JSON format for the revision prompt: *Request: {{request}}, Candidate: {{result1}, {result2}, {result3}..}, Discrepancy: {{difference1-2}, {difference1-3}..}, Checklist: {{instruction1},{instruction2},..}*. To eliminate discrepancies, we instruct LLM to revise the inconsistent steps of each candidate and output k revised responses with consistent answers. When revising, LLM should require careful and comprehensive consideration, as any minor modifications may lead to new discrepancies with others.

4 Experiments

4.1 Settings

Benchmarks We evaluate our method within two testbeds: mathematical reasoning and translation using GSM8K, SVAMP, and CommonMT benchmarks. Please see Appendix B.1 for details.

Evaluation Metrics For mathematical reasoning, we evaluate the precision of the final answer after their step-by-step reasoning, similar to the previous methodologies. For the translation task, we employ BLEURT[1] score as automatic metrics.

LLM Models and Prompts We conduct experiments using the GPT-3.5-Tubor-0613 and GPT-4-0613, alongside the Llama2-Chat model with three parameter scales (7B, 13B, and 70B). To make a fair comparison, we uniformly set the temperature to 0.2 for all experiments. For standard prompts and self-reflection baseline, we evaluate them 10 times using different prompts and average their results under zero-shot scenes. Prompts and other details can be found in Appendices B.2, C and D.

4.2 Baselines

We compare Self-Contrast with the following baselines: Standard CoT Prompt (Kojima et al., 2022). Self-Reflection (Shinn et al., 2023). Multi-Agent Debate (Du et al., 2023; Liang et al., 2023; He et al., 2020). ExpertPrompt (Xu et al., 2023a). Hint-Prompt (Zheng et al., 2023a). Math-Prompt (Imani et al., 2023). Moreover, for various task scenarios, we consider three forms of Self-Consistency (Wang et al., 2023d; Chen et al., 2023c): **SC-Vote**: The original Self-Consistency version, which samples K decoding results, followed by a voting process. **SC-Select**: Instead of voting, LLM also samples Top-K responses but then selects the most appropriate answer from K candidates by itself. **SC-Reflect**: After sampling Top-K responses, LLM reflects on these candidates and regenerates a new response as the final answer.

4.3 Main Results

In Tables 4 and 5, we report the accuracy and the average number of API/LLM calls (#Call), which serves as a proxy for the computational cost.

Consistent improvement over vanilla reflection. Compared to vanilla reflection, Self-Contrast brings significant and stable improvement. For mathematical reasoning, we achieve an average improvement of +7.2%. In contrast, the original self-reflection shows no clear improvement (-0.51%). A similar phenomenon is observed in creative translation, where Self-Contrast achieves a +0.95 improvement, whereas self-reflection results in a decrease of -1.6. Besides, compared to multi-agent and ensemble baselines, our improvement is also pronounced and consistent.

Better generality across different LLMs and tasks. From commercial LLMs (e.g., GPT4) to open-source models (Llama-2), and from reasoning to generative tasks, our strategy exhibits robust generalizability. Concretely, from the perspective of LLM, Self-Contrast achieves the best results on most models except Llama-2-7B. For instance, for GPT-3.5, the improvements are 7.8% on GSM8K and 9.2% on SVAMP, while for Llama-2-70B, the improvements are 11.6% and 9.3% respectively. As for Llama-2-7B, our performance is slightly lower than Self-consistency and Multi-Agent. This might be due to the weaker instruction-following capabilities of the Llama2-7B, making it challenging to contrast two inconsistent solutions. Besides task-wise, Self-Contrast applies to various task types, demonstrating high versatility. In contrast, Self-Consistency can not handle non-numerical tasks directly, e.g., translation, due to its voting mechanism (Table 5). Its variant strategies, SC-Select

[1] https://github.com/google-research/bleurt, BLEURT-20

	GSM8K					SVAMP					#Call Avg.
	GPT3.5	GPT4	L-7B	L-13B	L-70B	GPT3.5	GPT4	L-7b	L-13B	L-70B	
CoT Prompt	76.6	93.9	19.8	28.3	52.6	79.8	93.0	37.5	40.2	66	1
ExpertPrompt	77.3 ↑0.7	93.8 ↓0.1	21.6 ↑1.8	30.5 ↑2.2	53.1 ↑0.5	80.2 ↑0.4	93.3 ↑0.3	37.7 ↑0.2	41.9 ↑1.7	65.6 ↑0.4	2
Self-Reflection	75.8 ↓0.8	95.1 ↑1.2	17.0 ↓2.8	31.8 ↑3.5	49.3 ↓3.3	80.5 ↑0.7	91.5 ↓1.5	36.1 ↓1.4	42.5 ↑2.3	63.0 ↓3	3
Self-Consistency											
– SC-Vote	83.5 ↑6.9	94.2 ↑0.3	21.4 ↑1.6	37.6 ↑9.3	61.1 ↑8.5	84.6 ↑4.8	92.5 ↓0.5	**45.2 ↑7.7**	53.7 ↑13.5	72 ↑6	8
– SC-Select	76.3 ↓0.3	93.1 ↓0.8	16.2 ↓3.6	28.6 ↑0.3	54.6 ↑2.0	81.2 ↑1.4	93.2 ↑0.2	35.1 ↓2.4	38.9 ↓1.3	66.5 ↑0.5	9
– SC-Reflect	75.8 ↓0.8	93.3 ↓0.6	19.2 ↓0.6	29.1 ↓0.8	53.7 ↑1.1	81.1 ↑1.3	93.4 ↑0.4	32.5 ↓5	34.2 ↓6	67.5 ↑1.5	9
Multi-Agent	83.8 ↑7.2	93.5 ↓0.4	**23.8 ↑4**	34.9 ↑6.6	59.6 ↑7.0	84.1 ↑4.3	93.2 ↑0.2	42.5 ↑5	49.2 ↑9.0	70.1 ↑4.1	9
Hint-Prompt	78.8 ↑2.2	93.7 ↓0.2	18.3 ↓1.5	27.8 ↓0.5	59.6 ↑7	79.3 ↓0.5	93.1 ↑0.1	38.8 ↑1.3	40.6 ↑0.4	67.6 ↑1.6	6.7
Math-Prompt	79.6 ↑3.0	93.9 ↓0.0	19.5 ↓0.3	30.6 ↑2.3	59.8 ↑7.2	81.2 ↑1.4	93.6 ↑0.6	37.2 ↓0.3	41.5 ↑1.3	68.7 ↑0.5	4.5
Self-Contrast	**84.4 ↑7.8**	**95.4 ↑1.5**	20.5 ↑0.7	**42.3 ↑9.2**	**64.2 ↑11.6**	**89.0 ↑9.2**	**94.0 ↑1**	44.5 ↑7	**54.6 ↑14.4**	**75.3 ↑9.3**	7.8

Table 4: The performance on mathematical reasoning. Self-Consistency (SC-Vote, -Select, -Reflect) samples eight responses and then performs voting, selecting, or reflection. For the Multi-Agent, we configure three agents to engage in a three-round debate. ↑ and ↓ means accuracy changes over the CoT prompt. `L-` denotes Llama2-chat.

	GPT3.5	L-7B	L-13B	L-70B
CoT Prompt	69.1	53.7	62.5	63.2
ExpertPrompt	69.6 ↑0.5	53.8 ↑0.1	62.9 ↑0.4	63.4 ↑0.2
Self-Reflection	69.3 ↑0.2	48.8 ↓4.9	61.5 ↓1.0	62.2 ↓1.0
Self-Consistency				
– SC-Vote	–	–	–	–
– SC-Select	68.6 ↓0.5	52.1 ↓1.6	62.8 ↑0.3	63.0 ↓0.2
– SC-Reflect	69.0 ↓0.1	54.0 ↑0.3	62.2 ↓0.3	63.2 ↑0
Multi-Agent	69.9 ↑0.8	51.9 ↓1.8	**63.1 ↑0.6**	65.8 ↑2.6
Hint-Prompt	69.6 ↑0.5	**54.2 ↑0.5**	62.5 ↑0	64.6 ↑1.4
Self-Contrast	**70.7 ↑1.6**	52.1 ↓1.6	62.8 ↑0.3	**66.7 ↑3.5**

Table 5: The performance on Creative Translation.

LLMs	Strategy	Invalid ✗⇒✗ Cases ↓	Toxic ✓⇒✗ Cases ↓
GPT3.5	Self-Reflection	269	52
	SC-Reflect	245	73
	Self-Contrast	**186** ↓30.8%	**11** ↓78.9%
L-70B	Self-Reflection	528	140
	SC-Reflect	468	127
	Self-Contrast	**401** ↓24.8%	**71** ↓49.2%

Table 6: Self-Contrast is evaluated on two cases.

and SC-Reflect, lag significantly behind ours.

Fewer manual efforts and more reasonable call overheads. Compared to the multi-agent debate, Self-Contrast gains more significant improvements with less call overhead (>10% reduction). From a unified perspective, it can be viewed as a multi-agent contrastive mechanism. Instead of a free-form debate among multiple agents, our strategy fosters a more explicit and purposeful debate by contrasting the differences between agents and summarizing the reasons for their disagreements. Moreover, Self-Contrast is flexible, dynamically designing multiple perspectives tailored to user requests, without the need for manually preconfiguring agent roles and quantities.

5 The Effect of the Different Components

The above results show that Self-Contrast inspires reflection more accurately and stably than direct evaluation. It encompasses a self-curated prompt process, which fosters diverse solving perspectives to mitigate self-evaluation biases. Besides, it involves a checklist generation process to facilitate re-examination. We analyze their effect as follows:

Self-curated Prompt Vs. Sampling Multiple Responses. Instead of self-curated prompt process, we directly sample multiple responses from LLMs for subsequent contrast and reflection. Figure A2 shows that the final accuracy improves as the number of sampled responses increases, yet it is still lower than Self-Contrast with self-curated prompts process, where full strategy achieves 84.4% compared to the maximum of 81.8% when sampling 5 responses. We find that the top-n responses are sometimes strikingly similar, diminishing the effectiveness of the contrastive strategy.

Reflection Without Checklist. We eliminate the checklist generation process, i.e., directly instruct the LLM to reflect on the differences among perspectives. In Table A1, it brings a significant impact on mathematical reasoning (-3.5%), but a slight impact on translation (-0.1%), since translation tasks tend to focus more on local features. Even without a checklist, the LLM also can reflect based on the comparisons of lexical, syntactic.

6 Analysis

6.1 Reducing Invalid and Toxic Reflections

As mentioned in Table 2, due to overly confident or highly random in the self-evaluate process, vanilla

Strategy	Acc.(%)
Self-Evaluate - *An **Incorrect** Solution*	70.1
Self-Contrast	
- *A **Correct** and an **Incorrect** Solutions*	83.6
- *Two **Incorrect** Solutions with **Similar** Error*	70.9
- *Two **Incorrect** Solutions with **Different** Error*	75.5

Table 7: We conduct comparisons across four cases on a subset of GSM8K. LLM self-evaluates or self-contrasts different initial responses and reflects on their results.

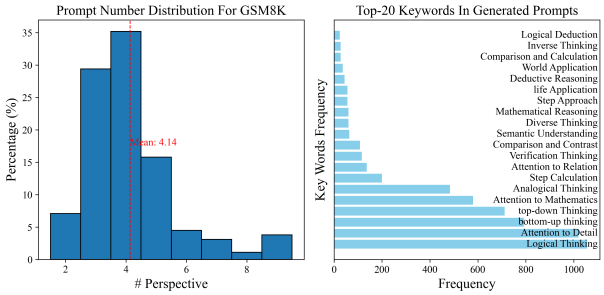

Figure 3: Left: The distribution of the prompt number generated when Self-curated. Right: We visualize the top-20 keywords and frequencies in the prompt name.

self-reflection contains a large amount of invalid (✗→ ✗: 20.3%) or toxic reflections (✓→ ✗: 4%). Therefore we investigate how Self-Contrast improves these two scenarios on GSM8K. As shown in Table 6, we observe that with Self-Contrast, the occurrences of invalid and toxic cases significantly reduced. In particular toxic cases decreased by 78.9% and invalid cases by 30.8% using GPT3.5. In contrast, the SC-Reflect does not significantly mitigate either of these scenarios.

The results indicate that through exploration, comparison, and summarization, the uncertainty in the reflection process is greatly reduced, thereby enhancing the error-correction capability of the LLM.

6.2 Contrasting Incorrect Solutions is also Instructive

Self-Contrast inspires reflection by contrasting the differences. An intuitive explanation is that the errors in different responses are dissimilar or randomized, so they can be used to compare with each other and eliminate uncertainties or biases. To verify this, we sample 200 questions from GSM8K, each manually annotated with a correct solution, two incorrect solutions with similar errors (e.g., Error1), and an incorrect solution with a different error (Error2). We design four experiments: 1. Self-evaluate **one incorrect** solution followed by reflection. 2. Self-Contrast a **correct** and an **incorrect** solution. 3. Self-Contrast **two similar incorrect** solutions. 4. Self-Contrast **two dissimilar incorrect** solutions. Table 7 shows that contrasting a correct and an erroneous solution, or contrasting two incorrect solutions with different errors both yield significant enhancements of 13.5% and 5.4%. In contrast, comparing two solutions with similar errors does not result in perceptible changes.

This result aptly explains that the improvement of Self-Contrast stems from contrasting the differences between dissimilar solutions. Therefore, even if candidate solutions are both incorrect, as long as their errors are different, Self-Contrast has the potential to eliminate errors. In other words, Self-Contrast can mitigate the random errors arising from the inherent uncertainty of the LLM.

6.3 Diverse Solving Perspectives Maximize Contrast Effect

Prior analysis indicates that only contrasting dissimilar solutions can foster reflection. Reviewing our strategy, we employ a self-curated prompt process to create multiple solving perspectives (§ 3.1), thereby providing diverse solutions for subsequent comparison. Here, we analyze the distribution of perspectives generated by this process in Figure 3. For most requests, LLM generates four prompts. We also analyze the frequency of keywords within these perspective's names. For mathematical reasoning, the LLM indeed adaptively designs numerous unique solving perspectives, then generating a variety of results. These dissimilar results can maximize the efficacy of our contrastive strategy.

7 Conclusion

We conduct a comprehensive investigation into the inherent reflection capabilities of LLMs. Our findings reveal a notable challenge: in the absence of external feedback, LLMs struggle to correct errors in previous responses on their own. After analyzing their self-evaluate process, we discover that LLMs are unable to accurately evaluate prior solutions and often provide overconfident or inconsistent feedback, which impedes reflection. To mitigate this, we introduce Self-Contrast, a contrastive strategy that inspires reflection by contrasting the differences between multiple perspectives, providing an informative checklist for reflection. Our experiments show that Self-Contrast performs well across a variety of scenarios and with different LLMs.

Limitations

For some smaller-scale LLMs, their instruction-following capability is weaker, hindering their potential to conduct precise comparisons and reflection. In such scenarios, the effectiveness of Self-Contrast might be slightly inferior to ensemble strategies. For instance, the performance of Self-Contrast with Llama2-7B is marginally lower than self-consistency. A viable approach is to utilize an external tool to compare differences between multiple perspectives, rather than LLM itself. For instance, we explore utilizing sequences comparison library *difflib*[2] to contrast two generated equations (e.g., differ.compare(a+b÷c, a-b÷c)) or some rule-based strategy to compare two responses. It can provide us with more accurate and flexible comparisons at different granularity (e.g., character level). We leave this as future work.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, et al. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *NeurIPS*.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shan Zhang, Jie Fu, and Zhiyuan Liu. 2023. Chateval: Towards better llm-based evaluators through multi-agent debate. *ArXiv*, abs/2308.07201.

Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F Karlsson, Jie Fu, and Yemin Shi. 2023a. Autoagents: A framework for automatic agent generation. *arXiv preprint arXiv:2309.17288*.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Cheng Qian, Chi-Min Chan, Yujia Qin, Ya-Ting Lu, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023b. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *ArXiv*, abs/2308.10848.

[2] https://docs.python.org/3/library/difflib.html

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

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2023c. Universal self-consistency for large language model generation.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023d. Teaching large language models to self-debug. *ArXiv*, abs/2304.05128.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and others. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168.

Roi Cohen, May Hamri, Mor Geva, and Amir Globerson. 2023. Lm vs lm: Detecting factual errors via cross examination. In *Conference on Empirical Methods in Natural Language Processing*.

A. Deshpande, Vishvak S. Murahari, Tanmay Rajpurohit, A. Kalyan, and Karthik Narasimhan. 2023. Toxicity in chatgpt: Analyzing persona-assigned language models. *ArXiv*, abs/2304.05335.

Yihong Dong, Xue Jiang, Zhi Jin, and Ge Li. 2023. Self-collaboration code generation via chatgpt. *ArXiv*, abs/2304.07590.

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Simon Frieder, Luca Pinchetti, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Christian Petersen, Alexis Chevalier, and J J Berner. 2023. Mathematical capabilities of chatgpt. *ArXiv*, abs/2301.13867.

Yao Fu, Hao-Chun Peng, Tushar Khot, and Mirella Lapata. 2023. Improving language model negotiation with self-play and in-context learning from ai feedback. *ArXiv*, abs/2305.10142.

Yao Fu, Hao-Chun Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. *ArXiv*, abs/2210.00720.

Deep Ganguli, Amanda Askell, Nicholas Schiefer, Thomas Liao, Kamil.e Lukovsiut.e, Anna Chen, Anna Goldie, Azalia Mirhoseini, Catherine Olsson, Danny Hernandez, Dawn Drain, Dustin Li, Eli Tran-Johnson, Ethan Perez, John Kernion, Jamie Kerr, Jared Mueller, Joshua D. Landau, Kamal Ndousse, Karina Nguyen, Liane Lovitt, Michael Sellitto, Nelson Elhage, Noem'i Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Sandipan Kundu, Saurav Kadavath, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, T. J. Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Benjamin Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom B. Brown, Christopher Olah, Jack Clark, Sam Bowman, and Jared Kaplan. 2023. The capacity for moral self-correction in large language models. *ArXiv*, abs/2302.07459.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. Pal: Program-aided Language Models. *ArXiv*, abs/2211.10435.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *ArXiv*, abs/2309.17452.

Jie He, Tao Wang, Deyi Xiong, and Qun Liu. 2020. The box is in the pen: Evaluating commonsense reasoning in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3662–3672, Online. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *ArXiv*, abs/2103.03874.

Sirui Hong, Xiawu Zheng, Jonathan P. Chen, Yuheng Cheng, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zi Hen Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, and Chenglin Wu. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *ArXiv*, abs/2308.00352.

Baizhou Huang, Shuai Lu, Weizhu Chen, Xiaojun Wan, and Nan Duan. 2023a. Enhancing large language models in coding through multi-perspective self-consistency. *ArXiv*, abs/2309.17272.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *ArXiv*, abs/2210.11610.

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

Shima Imani, Liang Du, and H. Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *ArXiv*, abs/2303.05398.

Siddhartha Jain, Xiaofei Ma, Anoop Deoras, and Bing Xiang. 2023. Self-consistency for open-ended generations. *ArXiv*, abs/2307.06857.

Zhanming Jie, Jierui Li, and Wei Lu. 2022. Learning to reason deductively: Math word problem solving as complex relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Saurav Kadavath, Tom Conerly, Amanda Askell, T. J. Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zachary Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, John Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom B. Brown, Jack Clark, Nicholas Joseph, Benjamin Mann, Sam McCandlish, Christopher Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *ArXiv*, abs/2207.05221.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. 2023. Language models can solve computer tasks. *ArXiv*, abs/2303.17491.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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*, abs/2305.19118.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *ArXiv*, abs/2305.20050.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2023. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization. *ArXiv*, abs/2310.02170.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jian-Guang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for

large language models via reinforced evol-instruct. *ArXiv*, abs/2308.09583.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Sean Welleck, Bodhisattwa Prasad Majumder, Shashank Gupta, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *ArXiv*, abs/2303.17651.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin T. Vechev. 2023. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. *ArXiv*, abs/2305.15852.

Deepak Nathani, David Wang, Liangming Pan, and William Yang Wang. 2023. Maf: Multi-aspect feedback for improving reasoning in large language models. *ArXiv*, abs/2310.12426.

OpenAI. 2022. Chatgpt.

OpenAI. 2023. Gpt-4 technical report.

Liangming Pan, Michael Stephen Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies. *ArXiv*, abs/2308.03188.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *ArXiv*, abs/2304.03442.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. 2023. Refiner: Reasoning feedback on intermediate representations. *ArXiv*, abs/2304.01904.

Silviu Pitis, Michael Ruogu Zhang, Andrew Wang, and Jimmy Ba. 2023. Boosted prompt ensembles for large language models. *ArXiv*, abs/2304.05970.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, M. Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. *ArXiv*, abs/2302.04761.

Timo Schick, Jane Dwivedi-Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. 2022. Peer: A collaborative language model. *ArXiv*, abs/2208.11663.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. *ArXiv*, abs/2302.00618.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. In *Advances in Neural Information Processing Systems*.

Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. *ArXiv*, abs/2303.11366.

Kumar Shridhar, Koustuv Sinha, Andrew Cohen, Tianlu Wang, Ping Yu, Ram Pasunuru, Mrinmaya Sachan, Jason Weston, and Asli Celikyilmaz. 2023. The art of llm refinement: Ask, refine, and trust. *arXiv preprint arXiv:2311.07961*.

Kaya Stechly, Matthew Marquez, and Subbarao Kambhampati. 2023. Gpt-4 doesn't know it's wrong: An analysis of iterative prompting for reasoning problems.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and Efficient Foundation Language Models. *ArXiv*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Karthik Valmeekam, Matthew Marquez, and Subbarao Kambhampati. 2023. Can large language models really improve by self-critiquing their own plans? *ArXiv*, abs/2310.08118.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim.

2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *ArXiv*, abs/2305.04091.

Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghuai Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. 2023b. Making large language models better reasoners with alignment. *ArXiv*, abs/2309.02144.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y.Wu, and Zhifang Sui. 2023c. Math-shepherd: Verify and reinforce llms step-by-step without human annotations.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023d. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *ICLR 2023 poster*, abs/2203.11171.

Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. 2023e. Unleashing cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. 2022. Generating sequences by learning to self-correct. *ArXiv*, abs/2211.00053.

Yixuan Weng, Minjun Zhu, Shizhu He, Kang Liu, and Jun Zhao. 2022. Large language models are reasoners with self-verification. *ArXiv*, abs/2212.09561.

Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. 2023a. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023b. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *ArXiv*, abs/2308.08155.

Zhiheng Xi, Senjie Jin, Yuhao Zhou, Rui Zheng, Songyang Gao, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. Self-polish: Enhance reasoning in large language models via problem refinement. *ArXiv*, abs/2305.14497.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, MingSung Kan, Junxian He, and Qizhe Xie. 2023. Decomposition enhances reasoning via self-evaluation guided decoding. *ArXiv*, abs/2305.00633.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *IJCAI*, pages 5299–5305.

Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. 2023. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate. In *Conference on Empirical Methods in Natural Language Processing*.

Benfeng Xu, An Yang, Junyang Lin, Quang Wang, Chang Zhou, Yongdong Zhang, and Zhendong Mao. 2023a. Expertprompting: Instructing large language models to be distinguished experts. *ArXiv*, abs/2305.14688.

Xiaohan Xu, Chongyang Tao, Tao Shen, Can Xu, Hongbo Xu, Guodong Long, and Jian-Guang Lou. 2023b. Re-reading improves reasoning in language models. *ArXiv*, abs/2309.06275.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *ArXiv*, abs/2305.10601.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *ArXiv*, abs/2210.03629.

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*, abs/2304.13007.

Long Long Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zheng Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *ArXiv*, abs/2309.12284.

Zheng Yuan, Hongyi Yuan, Cheng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023a. Scaling relationship on learning mathematical reasoning with large language models. *ArXiv*, abs/2308.01825.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, and Songfang Huang. 2023b. How well do large language models perform in arithmetic tasks? *ArXiv*, abs/2304.02015.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *ArXiv*, abs/2309.05653.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. Glm-130b: An Open Bilingual Pre-trained Model. *ICLR 2023 poster*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel

Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022a. Opt: Open Pre-trained Transformer Language Models. *ArXiv*, abs/2205.01068.

Wenqi Zhang, Yongliang Shen, Weiming Lu, and Yueting Zhuang. 2023a. Data-copilot: Bridging billions of data and humans with autonomous workflow. *arXiv preprint arXiv:2306.07209*.

Wenqi Zhang, Yongliang Shen, Qingpeng Nong, Zeqi Tan, Yanna Ma, and Weiming Lu. 2023b. An expression tree decoding strategy for mathematical equation generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 439–456, Singapore. Association for Computational Linguistics.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alexander J. Smola. 2022b. Automatic chain of thought prompting in large language models. *ArXiv*, abs/2210.03493.

Xu Zhao, Yuxi Xie, Kenji Kawaguchi, Junxian He, and Qizhe Xie. 2023. Automatic model selection with large language models for reasoning. *ArXiv*, abs/2305.14333.

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

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed Huai hsin Chi, Quoc V Le, and Denny Zhou. 2023b. Take a step back: Evoking reasoning via abstraction in large language models. *ArXiv*, abs/2310.06117.

Shen Zheng, Jie Huang, and Kevin Chen-Chuan Chang. 2023c. Why does chatgpt fall short in answering questions faithfully? *ArXiv*, abs/2304.10513.

Denny Zhou, Nathanael Scharli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Huai hsin Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *ArXiv*, abs/2205.10625.

Xinyu Zhu, Junjie Wang, Lin Zhang, Yuxiang Zhang, Yongfeng Huang, Ruyi Gan, Jiaxing Zhang, and Yujiu Yang. 2023. Solving math word problems via cooperative reasoning induced language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada. Association for Computational Linguistics.

Appendix

A Complementary Experiments

A.1 LLM is More Likely to Trust Previous Response

We investigate whether LLMs are prone to uncritically trusting previous responses during reflection, rather than meticulously examining and rectifying errors. Typically, self-reflection often contains three stages, i.e., initial response, self-evaluate, and revision stage. We employ different LLMs to provide a poorer quality response as the initial response for the subsequent two stages. We observe whether this affects the results of the reflection, e.g., we replace gpt3.5→gpt3.5→gpt3.5 with Llama-2-70b→gpt3.5→gpt3.5. If LLMs tend to place undue trust in prior responses, the efficacy of the final reflective process will be adversely impacted.

However, as shown in Figure A1, the reflection results are severely impacted by the quality of the initial response. E.g., compared with using gpt3.5 for three phases, Llama2-70b→gpt3.5→gpt3.5 exhibits a marked decrease (-8.4% for GSM8K). Furthermore, we also observe the weaker the LLM replaced, the poorer the performance after reflection. It suggests that LLMs tend to trust the initial solution rather than detect and revise the errors during the self-evaluate phase.

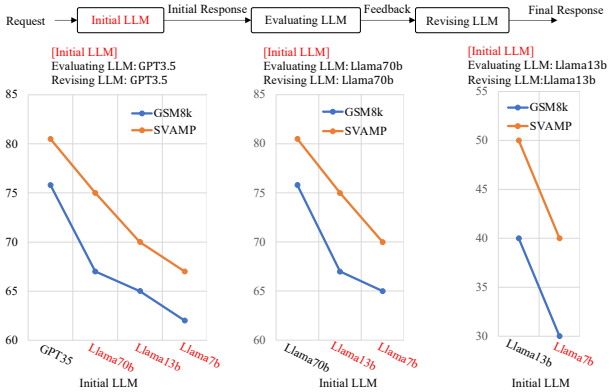

Figure A1: The Reflection Accuracy with Different LLM for Initial Response. *Left*: different LLMs provide initial responses when GPT3.5 is utilized for Evaluation and Revision. *Center*: different LLMs provide initial responses when Llama2-70B is utilized for Evaluation and Revision. *Right*: different LLMs provide initial responses when Llama2-13B is utilized for Evaluation and Revision. The results indicate that LLMs are easily influenced during reflection. LLM is predisposed to trust previous responses over diligently examining and correcting errors.

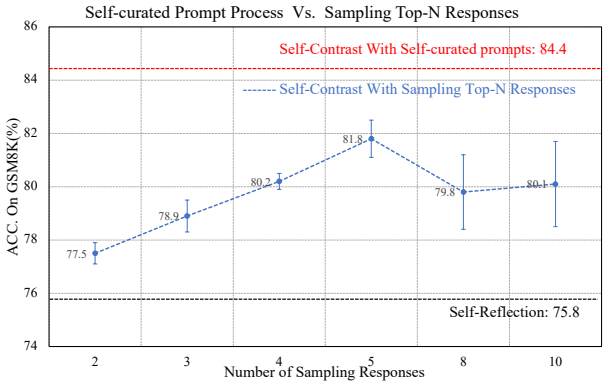

Figure A2: We replace the self-curated prompt process with a simple strategy: directly sampling top-n responses for contrast. We observe that as N increases, the performance also improves, yet it still remains lower than self-contrast with the self-curated prompts. All results are conducted on GSM8K using GPT-3.5.

Strategy	GSM8K	CommonMT
Self-reflection	75.8	69.3
Self-Contrast	84.4	70.7
- *w/o Checklist Generation*	80.9 ↓3.5	70.6 ↓0.1
Selecting Strategies		
- *Random Selecting*	76.4 ↓8	69.5 ↓1.2
- *Clustering + Random Selecting*	81.2 ↓3.2	69.7 ↓1.0
- *Clustering + LLM Selecting*	82.6 ↓1.8	69.9 ↓0.8
- *Clustering + Negative Perspective*	83.9 ↓0.5	70.8 ↑0.1

Table A1: We eliminate the checklist generation process, instructing the LLM to directly reflect on the differences from contrasting multiple perspectives. Besides, we also analyze the impact of different selecting strategies.

A.2 Self-Evaluate Vs. Self-Contrast

Self-Contrast inspires reflection by contrasting the differences, rather than evaluating directly. The underlying assumption is contrast is more accurate and stable than direct evaluation for LLM. To validate this, we conduct an experiment using 200 samples from GSM8K, each containing a correct and an incorrect solution. We design two tasks: Taks 1: Contrasting two solutions. Task 2: Evaluating the incorrect solution. We manually check the results of two tasks, i.e., whether LLM can perform contrast or evaluate correctly. As shown in Figure A3, we observe contrasting is more accurate than direct evaluating (171 correct Vs. 140 incorrect).

Further, we divide all samples into four cases: 1. both tasks are correct. 2. Contrasting: correct, Evaluating: wrong. 3. Contrasting: wrong, Evaluating: correct. 4. Both are wrong. In Figure A3, the results show that when LLM can correctly evaluate a solution, it is often able to contrast correctly, with

few exceptions (only 8 samples for Evaluating Correct Only). Notably, in 39 cases, the LLM fails in direct evaluation but succeeds in contrast. These results indicate that contrasting two solutions is more accurate and stable than direct evaluation, leading to more reliable results.

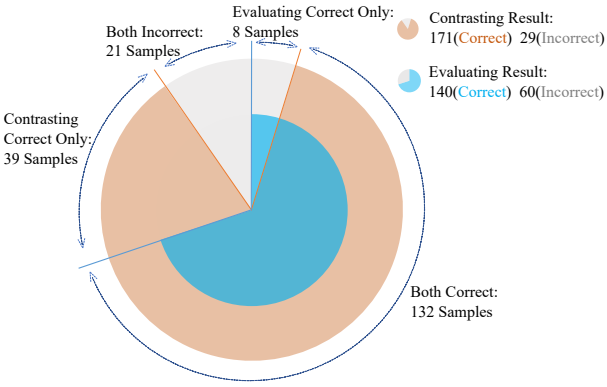

Figure A3: We compare the results of the *Evaluating* and *Contrasting* using two pie charts. It shows *Contrasting* is more accurate and stable than direct *Evaluating*.

A.3 Ablation Study For Selection Strategy

As introduced in Section 3.2, we cluster multiple responses generated by the self-curated process and then select the cluster center from each category for contrast. We design four different selection strategies. 1) *Random Selecting*: We randomly choose K responses from all candidates. 2) *Clustering + Random Selecting*: We first cluster all responses into k categories, then randomly select one from each category. 3) *Clustering + LLM Selecting*: Similarly, we first cluster all responses into k categories, then instruct the LLM to choose a potentially correct response from each category. 4) *Clustering + Negative Perspective*: We first instruct the LLM to consider what are common errors for the user request. Then LLM should intentionally generate an imperfect solution based on these common errors. Finally, we instruct the LLM to select one response from each category that is least similar to the intentionally generated imperfect solution. As shown in Table A1, we observe that compared to Self-Contrast, the performance of several selection strategies experiences a certain degree of decline.

B Experiments Details

B.1 Benchmarks

Mathematical Reasoning: We leverage multiple datasets with different complexity and languages, including GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021) as benchmarks to evaluate performance. Notably, GSM8K presents higher levels of difficulty, encompassing complex mathematical operations, while SVAMP is slightly simpler and consists of combinations of addition, subtraction, multiplication, and division.

Creative Translation In addition to mathematical reasoning tasks, we introduce a generation task: creative translation. We utilize the CommonMT (He et al., 2020), which includes a vast body of Chinese-to-English pair examples. Unlike conventional translations, most samples contain non-standard expressions such as idioms and metaphors, necessitating an understanding of local cultural and linguistic habits for accurate translation. Following the Multi-agent debate (Liang et al., 2023), we adopt the samples with "hard" categories from CommonMT as testing benchmarks.

B.2 Other Details

In the Self-Curated Prompt phase, we limit LLMs to design at least two different prompts and a maximum of nine prompts for each request. In selecting stage (Section 3.2), we set k to 3, which means that all perspective results are divided into three categories, and then we select a result from each category. We instruct LLM sequentially output comparisons among three results, subsequently synthesizing these differences into a comprehensive checklist in a single pass, eliminating the need for multiple prompts. Besides, due to the diversity of translation tasks, we also introduce a negative perspective for translation. Specifically, we instruct LLMs to consider what common errors might be made for the user request, then actively adopt a careless persona to generate an incorrect response with some common mistakes. The result of this negative perspective serves as a negative demonstration for subsequent selection and reflection.

C Baseline Prompts

Standard Prompt We use a simple prompt for CoT Prompt and self-consistency baselines. For each experiment, we run 10 times and averaged their results.

```
Math Reasoning: You are a math teacher.
Let us solve the math question step by
step. The question is {input}.
```

```
Creative Translation: You are an expert
translator, please translate Chinese
into English accurately. The Chinese
sentence is {input}.
```

Reflection Prompts We designed 10 prompts for the self-reflection baseline. Each experiment follows *Initial response-Evaluation-Revision* pattern. The prompt for the *Initial response* remains consistent with previous experiments (Standard Prompt).

```
1:
Evaluation: Please carefully examine the
previous responses for correctness, and
provide detailed feedback.

Revision: Please refine the previous
response based on the feedback.
2:
Evaluation: Please review your previous
responses for any errors, and provide
detailed feedback.

Revision: Please refine the previous
response based on the feedback. If there
 are no questions, you can repeat the
previous solution
3:
Evaluation: Do you think the previous
response is correct or not, and if not
please point out where is wrong.

Revision: Please refine the previous
response.
4:
Evaluation: Please carefully evaluate the
quality of the previous response and
point out if you feel something is not
appropriate

Revision: Please carefully consider the
comments in the feedback and re-generate
 the answer.
5:
Evaluation: Please double-check the
previous response for any errors. If
there are any errors, please point them
out.

Revision: Please read the feedback
carefully, and improve your answer.
6:
Evaluation: There may have been some
mistakes with your previous response, so
 please double-check and find out the
mistake. If you think there are no
errors at all, please just reply, "
Exactly correct".

Revision: Please refine your response. If
 you think it's acceptable, then just
repeat your last response.
7:
Evaluation: Please check that your
previous response matches the question.
Please point out if it does not fit

Revision: Please refine your response
based on the feedback. If the feedback
points out something that is not perfect
 please fix it!
8:
Evaluation: Please consider whether your
response addresses the problem. If not
or if there is an error please point it
out

Revision: Please reflect based on the
feedback and improve your response.
9:
Evaluation: Please assess in detail
whether your previous response solves
the problem and provide feedback.

Revision: Please refine your response
based on the feedback.
10:
Evaluation: Please check your previous
response for correctness and whether it
can be further enhanced.

Revision: Please further refine your
response based on the feedback. If you
don't feel it is necessary then restate
the previous response
```

D Our Prompt

D.1 Prompt for Self-Curated Process

Different requests may require some unique solving perspectives. We design a self-curated prompts process, enabling LLMs to design their prompts based on specific user requests. The prompt for the self-curated process is as follows:

```
Translation Task:
You are a translation specialist who
specializes in translating from diverse
perspectives. Given a Chinese source
sentence, you need to carefully analyze
the source sentence and dynamically
generate several useful prompt
instructions. These prompt instructions
should be diverse and also relevant to
the source sentence. These prompt
instructions are used to guide the
language model to think in different
ways, attention to different emphases,
and reason from different perspectives
for a more accurate translation.

For instance, you can design different
translation styles, different
expressions of emotion, different
emphases, and different tones for input
sentences in prompt instruction. Besides
 you can create different knowledge
backgrounds, identities, personalities,
different concerns, etc for more
relevant translation.

Here are some guidance rules for Prompt
Generation:
1. Tone Requirement: Please generate
prompt instructions in the third person.
2. Content Requirement: Each prompt
instruction should be different, and
include at least three parts:
translation styles, attention emphasis,
and tones and emotion design. Please do
not state them separately.
```

3. Number Requirement: Dynamically generate the most valuable 2 to 9 prompt instructions based on the input Chinese source sentences.
4. Format Requirement: Each prompt instruction should start with ###.
5. Others: Prompt should focus on translation. So don't ask any other irrelevant questions in the prompt.

Here is an example:
The input Chinese sentence is: 他想拉同村的干部一起下水去贩毒. Please generate the most suitable prompts.
Output:
Literal Perspective: ###You are a meticulous translator, proficient in direct translation, and highly focused on specifics. Your translation approach prioritizes precise replication of the original text's expression.

Liberal Perspective: ###You are an inventive translator, characterized by a dynamic and liberal translation approach, often reimagining the original text's meaning in your own linguistic style.

The input Chinese sentence is {input}. Please generate the most suitable prompts:

Reasoning Task:
You are a math specialist who specializes in math solving from diverse perspectives. Given a math question, you need to carefully analyze the question and dynamically generate several useful prompt instructions. These prompt instructions should be diverse and also useful for math-solving. These prompt instructions are used to guide the language model to think in different ways, attention to different emphases, and reason from different perspectives for more accurate math solving.

For instance, you can adopt multi-faceted thinking (logical thinking, lateral thinking, analogical thinking, etc.), different reasoning perspectives (e.g., top-down, bottom-up, step-by-step), and different emphases of concern, (entity words, numbers, units, percentages, math knowledge, etc) for input question in prompt instruction.

Here are some guidance rules for Prompt Generation:
1. Tone Requirement: Please generate prompt instructions in the third person.
2. Content Requirement: Each prompt instruction should adopt a different way of thinking, or focus on a different perspective, or different emphases to solve the question.
3. Number Requirement: Dynamically generate the most valuable 2 to 9 prompt instructions based on the input math question.
4. Format Requirement: Each prompt instruction should start with ### and end with @@@.
5. Others: Prompt instructions should focus on math solving. So don't ask any other irrelevant questions in the prompt.

Here is an example: The math question is: Mark works at his job for 8 hours a day for 5 days a week. He used to make $10 an hour but they raised his pay by $2 per hour. How much does he make a week?

Output:
bottom-up perspective: ###As a mathematician, you have to solve the given problem from a bottom-up perspective. Please focus initially on the foundational elements of the problem. Start with the simplest parts and their interrelations. Progressively build upon these foundational components, joining them together until a complete solution emerges

The input math question is {input}. Please generate the most suitable prompts:

D.2 Prompt for Contrasting Process

Translation Task:
You are an expert translator. Given some candidate English translations for a Chinese source sentence, you should carefully compare the difference between each two translations in terms of semantics, syntax, words (e.g., nouns and verbs), and any other aspects.

When you compare, you need to consider the following questions:
1: Are there differences between the two translations?
2: Where are the differences?
3: What causes these differences?

After contrasting, you should generate a checklist based on these differences between candidate translations. You should carefully consider each discrepancy and the reasons behind it, summarizing them into a few checking instructions in the checklist. This checklist can guide others to re-examine the input sentence and these candidate translations to eliminate these discrepancies.

Input Format:
The Chinese sentence is {Chinese sentence}.
All Results: {Result1},{Result2}, {Result3},....

```
Output Format:
For Result1 and Result2: {Difference1}.
For Result1 and Result3: {Difference2}
For Result2 and Result3: {Difference3}
Checklist: {Directive1, Directive2, ...}
....
```

Reasoning Task:

```
You are a math specialist who
specializes in math solving. Given some
candidate solutions for a math question,
 you should carefully compare the
difference for each two solutions in
their solving steps.

When you compare, you need to consider
the following questions:
1: Are the two solutions have different
final answers and mathematical
expressions?
2: Where are the differences in their
solution steps and mathematical
expressions?
3: Why are the answers of the two
solutions different?

After contrasting, you should generate a
 checklist based on these differences
between candidate solutions. You should
carefully consider each discrepancy and
the reasons behind it, summarizing them
into a few checking instructions in the
checklist. This checklist can guide
others to re-examine the input question
and these candidate solutions to
eliminate these discrepancies.


Input Format:
The math question is {Question}.
All solutions: {Solution1}, {Solution2}, {
Solution3}, ....

Output Format:
For Solution1 and Solution2: {Difference1}
For Solution1 and Solution3: {Difference2}
For Solution2 and Solution3: {Difference3}
Checklist: {Directive1, Directive2, ...}
```

D.3 Prompt For Reflection Stage

We record all candidate responses, their differences, and the checklist in a JSON format. The whole prompt for math reasoning is as follows:

Reflection Instruction:

```
Given a math question, multiple
inconsistent solutions, their
differences in their solving processes,
and a checklist. You should revise the
inconsistent solving step for each
solution, eliminate the differences, and
 output a new solving process for each
solution.

Guidance Rules for Reflection:
1. Please check carefully according to
the requirements on the checklist. It
helps you to resolve conflicts between
different solutions.
2. When you finish revising inconsistent
 solutions, please ensure all revised
solutions should have the same answer.
If not, please revise again until all
inconsistencies are removed, and all
candidates are consistent.
3. Please output all revised solutions
in JSON format as input, without any
other text.


The math question is {question}.
The candidate solutions and their
discrepancy are as follows:
{
  "Candidate": {
        "result_1": {
              "answer": "{answer1}",
              "solution": "{solution1}"},
        "result_2": {
              "answer": "{answer2}",
              "solution": "{solution2}"},
        "result_3": {
              "answer": "{answer3}",
              "solution": "{solution3}"},
        ....
  },
  "Discrepancy": {
        "difference_1_2": {
              "source": "result_1",
              "target": "result_2",
              "relation": "{difference}"
              },
        "difference_1_3": {
              "source": "result_1",
              "target": "result_3",
              "relation": "{difference}"
              },
        "difference_2_3": {
              "source": "result_2",
              "target": "result_3",
              "relation": "{difference}"
              },
        ....
  }
}
Checklist: {Directive1, Directive2,....}
Please revise each inconsistent solution
.
```

E Related Works

E.1 Self-correction Ability of LLM

Recently, one exciting discovery is that LLMs appear to possess advanced cognitive intelligence: self-correction, where LLMs can refine their previous responses based on feedback (Shinn et al., 2023; Madaan et al., 2023; Paul et al., 2023). This capacity endows LLMs to harness external feedback, or even self-evaluated feedback to refine the prior responses (Welleck et al., 2022; Kadavath et al., 2022; Chen et al., 2023d; Kim et al., 2023;

Xi et al., 2023; Ganguli et al., 2023; Pan et al., 2023; Nathani et al., 2023). This capacity, particularly when it is solely reliant on inherent reflection, has generated significant interest in the academic community. It appears that a simple iterative prompt strategy could facilitate self-correction in an LLM-based system. However, recent studies (Huang et al., 2023b; Stechly et al., 2023; Liang et al., 2023; Valmeekam et al., 2023) have cast doubt on LLM's inherent reflection capability. Their research indicates that without external feedback, LLMs have difficulties in amending prior responses.

E.2 Prompting for Better Problem-Solving

Drawing on cognitive science, human reasoning involves two different reasoning patterns: breadth reasoning, i.e., exploring various reasoning perspectives, and depth reasoning, which involves continually refining ideas and minimizing errors. Based on this concept, we can view previous prompting strategies as either breadth or depth reasoning. Self-consistency and some contemporaneous works (Wang et al., 2023d; Huang et al., 2022; Yoran et al., 2023; Jain et al., 2023; Chen et al., 2023c) mimic breadth reasoning by sampling diverse reasoning processes and voting the final answer, while self-reflection, abstraction reasoning strategies (Shinn et al., 2023; Madaan et al., 2023; Paul et al., 2023; Zheng et al., 2023a; Wang et al., 2023a; Yoran et al., 2023; Zheng et al., 2023b; Xu et al., 2023b; Shridhar et al., 2023) represent depth reasoning, refining reasoning through iterative prompting strategy. Except for these, Self-Verification (Weng et al., 2022) designs a reverse generation from the answer to given conditions, which is widely used in machine translation (Edunov et al., 2018). Cohen et al. (2023); Mündler et al. (2023) propose a method for detecting self-contradictions or factual errors in responses to enhance quality. However, our Self-Contrast combines both breadth and depth reasoning. It creates multiple perspectives to enhance the breadth of reasoning and also reflects on the differences for better depth reasoning, offering more reliable problem-solving.

E.3 Agent-based Methods

Recent studies (Li et al., 2023; Deshpande et al., 2023; Xu et al., 2023a; Du et al., 2023; Xiong et al., 2023) have found that when an LLM is assigned a specific role personas, it can generate higher-quality responses. This suggests that LLMs are powerful enough, and the appropriate prompt can elicit this capability. Moreover, recent works (Wang et al., 2023e; Fu et al., 2023; Liang et al., 2023; Schick et al., 2022; Dong et al., 2023; Park et al., 2023; Liu et al., 2023) have utilized a multi-role dialogue to collaborate or debate with each other for a more comprehensive response. Furthermore, some studies (Chen et al., 2023b; Chan et al., 2023; Huang et al., 2023a; Chen et al., 2023a; Hong et al., 2023; Wu et al., 2023b) have integrated this concept with complex tasks such as code generation by decomposing a complex task into several sub-tasks and employing multiple agents with different identities for each sub-task. However, most agent-based approaches necessitate careful manual design of each agent's role and pattern of interaction. Our approach, in contrast, does not require pre-defined agents' roles and numbers by humans, as it is entirely designed by the LLMs based on the user request, offering greater flexibility.

E.4 Learning Mathematical Reasoning

In recent years, mathematical reasoning has become a significant benchmark (Cobbe et al., 2021; Hendrycks et al., 2021) to evaluate the capabilities of artificial intelligence models. Within the paradigm of supervised learning, a vast amount of research (Xie and Sun, 2019; Patel et al., 2021; Jie et al., 2022; Zhang et al., 2023b) has been dedicated to translating human language into mathematical equations. In the era of LLMs, the advent of Chain-of-Thought and other prompting strategies have notably augmented the reasoning capabilities (Zhu et al., 2023; Yuan et al., 2023b; Frieder et al., 2023; Zhou et al., 2022).

Prompting Method PAL and Program-of-Thoughts (Gao et al., 2022; Chen et al., 2022) separate the computation and reasoning process using code as the intermediate process. Mathprompter, Auto-Model (Imani et al., 2023; Zhao et al., 2023) encourage LLMs to generate diverse reasoning paths in different forms simultaneously, including text (CoT), code (PAL), and symbols (Equation) for a higher confidence answer. Automatic-CoT, Complexity-CoT, Synthetic Prompt and Boosted Prompt (Zhang et al., 2022b; Fu et al., 2022; Shao et al., 2023; Pitis et al., 2023) enhance reasoning performance by optimizing the selection of demonstrations within the prompt. Tree-of-thought and Self-Evaluation (Yao et al., 2023; Xie et al., 2023) extend the CoT into a search tree, obtaining more accurate answers through self-evaluation.

Finetuning-based Method Another domain of study involves methods based on finetuning. These approaches involve finetuning open-source models, such as LLaMA, by incorporating insights from sophisticated closed-source LLMs. The fine-tuning approaches(Yuan et al., 2023a; Luo et al., 2023; Yue et al., 2023; Wang et al., 2023b; Yu et al., 2023; Gou et al., 2023) also have the potential to improve the mathematical reasoning capabilities of LLMs. The essence of fine-tuning is centered around the development of quality datasets comprising question-response pairs. Additionally, process-supervised training methods Lightman et al. (2023); Wang et al. (2023c) can also enhance the reasoning abilities of the LLMs.