Self-Para-Consistency: Improving Reasoning Tasks at Low Cost for Large Language Models

Wenqing Chen¹, Weicheng Wang¹, Zhixuan Chu²,³*, Kui Ren²,³, Zibin Zheng¹, Zhichao Lu⁴

¹School of Software Engineering, Sun Yat-sen University
²The State Key Laboratory of Blockchain and Data Security, Zhejiang University
³School of Cyber Science and Technology, Zhejiang University
⁴Department of Computer Science, City University of Hong Kong

{chenwq95, zhzibin}@mail.sysu.edu.cn, wangwch7@mail2.sysu.edu.cn, {zhixuanchu,kuiren}@zju.edu.cn, luzhichaocn@gmail.com

Abstract

Recently, the self-consistency decoding strategy has shown the ability to improve performance for complex reasoning tasks with large language models (LLMs). However, the costs may be high because the sampling process of the strategy generates some low-probability text, resulting in low-quality reasoning paths. As a consequence, it requires a relatively large sampling number to obtain good aggregation performance. In this paper, we propose an alternative strategy, self-para-consistency. It first generates multiple paraphrases for each test question, then generates reasoning paths for the original and all the paraphrased questions based on greedy decoding, and finally selects the most consistent answer. Since all the candidate paths have relatively high probabilities, the sampling number could be much smaller than the self-consistency strategy. Extensive experiments on complex reasoning datasets demonstrate the effectiveness of our method in reducing the sampling number.

1 Introduction

Recently, large language models (LLMs) like GPT-3 were considered to have an emergent ability of Chain of Thought (CoT) prompting (Wei et al., 2022) to perform multi-step reasoning (Wei et al., 2023), though the term "emergent" is still debated (Schaeffer et al., 2023). While promising, the CoTs generated through greedy decoding may fall into local optima. To alleviate this problem, Wei et al. (2022) proposed to sample a diverse set of CoTs and then aggregate them through majority voting, as shown in Figure 1(a).

However, the self-consistency strategy may encounter a key challenge in reality, which is the high cost due to the sampling process producing low-probability reasoning paths and therefore needing a relatively large number to achieve considerable performance. For example, Wei et al. (2022) and Chen et al. (2022) set the sampling number to 40, making the sampling process expensive, especially for complex reasoning tasks with long reasoning paths.

To alleviate this problem, we need to achieve a better trade-off between the quality and the diversity of generated reasoning paths. In this paper, we propose an alternative strategy, referred to as self-para-consistency. Its key idea is to retain the advantage of greedy decoding, which is likely to have a higher average probability than the sampling process. Then the diversity comes from generating different paraphrases of the given question. The intuition of paraphrasing is that LLMs are shown to be sensitive to spurious features in the prompts (Sclar et al., 2023), and therefore may

* Corresponding Author

Figure 1: The comparison of self-consistency and self-para-consistency: (a) The self-consistency strategy samples diverse reasoning paths but includes some low-probability paths; (b) The self-para-consistency transfers the given questions into multiple paraphrases and then generates the corresponding reasoning paths for each paraphrase and the original question with greedy decoding.
generate different text in response to paraphrases in different expressions.

As shown in Figure 1(b), the proposed self-para-consistency strategy consists of three steps. We first prompt an LLM to generate paraphrases and then generate the corresponding reasoning path for each paraphrase and the original question through greedy decoding. The last step is the same as the self-consistency strategy, which is to find the most consistent answer via majority voting. We conduct extensive experiments on 6 reasoning datasets, of which the results demonstrate effectiveness in reducing the sampling number.

The contributions of this work are as follows:

• We propose the self-para-consistency strategy to improve the reasoning performance of LLMs, which is a lower-cost alternative to self-consistency.

• The self-para-consistency strategy can serve as a kind of robustness or uncertainty measurement method.

• The extensive experiments demonstrate the effectiveness of reducing the generation costs by reducing the sampling number.

2 Related Work

Recently, CoT Prompting has demonstrated impressive performance (Wei et al., 2022) but has two challenges. The first is the inconsistency between the reasoning path and the result. The second is the local optimality of the generated reasoning path through greedy decoding (Wang et al., 2022). To address the inconsistency problem, a line of work has employed program languages instead of natural language to depict the reasoning path, referred as program-of-thought (PoT) prompting (Chen et al., 2022; Gao et al., 2023; Lyu et al., 2023). To address the problem of local optimality, Wang et al. proposed the self-consistency strategy based on CoT prompting (Wang et al., 2022).

The Quality-Diversity Trade-Off. Due to the diversity of natural language, there is a quality-diversity trade-off in text generation systems (Montahaei et al., 2019; Zhang et al., 2021). By sampling a latent variable z, text generation models can further improve diversity while still using greedy decoding or beam search. Our method is inspired by this work, as we suppose that different prompts can involve extra diversity while still keeping considerable quality through greedy decoding.

3 Methodology

3.1 The Framework and Notation

As shown in Figure 1, the proposed self-para-consistency strategy consists of three steps: 1) paraphrasing the given question into multiple paraphrases, 2) generating corresponding reasoning paths via greedy decoding, and 3) aggregating the answers based on majority voting. We formalize the three steps in the following paragraphs.

Given a testing question \( x \), we first prompt an LLM (parameterized by \( \theta \)) to generate \( k - 1 \) paraphrased questions \( G_{\text{para}} = x^{i} | i = 1 \rightarrow k - 1 \) where \( k > 1 \). The prompt for paraphrasing is denoted by \( \mathcal{I}_{\text{para}} \), then the paraphrasing process can be formalized as:

\[
P_{\theta}(G_{\text{para}} \mid x, \mathcal{I}_{\text{para}}) = \prod_{i=1}^{k-1} P_{\theta}(x_{i}^{i} \mid x, \mathcal{I}_{\text{para}}, G_{\text{para}}^{<i})
\]

(1)

where \( G_{\text{para}}^{<i} \) denotes the subgroup containing the already generated \( i - 1 \) paraphrases. It means that all the \( k - 1 \) paraphrases are generated sequentially. In this way, the LLM tends to generate different paraphrases.

In the second step, we collect the original question and the \( k - 1 \) generated paraphrases, and then prompt the LLM to generate \( k \) reasoning paths \( R_{\text{path}} = r_{i}, i = 1 \rightarrow k \) in parallel. This step can be formalized as:

\[
P_{\theta}(R_{\text{path}} \mid x, G_{\text{para}}, \mathcal{I}_{\text{inst}}) = \\
P_{\theta}(r_{1} \mid x, \mathcal{I}_{\text{inst}}) \cdot \prod_{i=1}^{k-1} P_{\theta}(r_{i+1} \mid x_{i}^{i}, \mathcal{I}_{\text{inst}})
\]

(2)

where \( \mathcal{I}_{\text{inst}} \) denotes the instruction prompt to generate reasoning paths. We denote \( r_{1} \) as the reasoning path for the original question \( x \). Equation 2 means that the generation of reasoning paths can be parallelized. Greedy decoding is then performed following Equation 1 and 2. The diversity is no longer from the sampling process of the LLM’s decoder but comes from the paraphrasing process instead.
In the final step, the process is the same as with self-consistency (Wang et al., 2022). We also assume each reasoning path \( r_i \) is coupled with the answer \( a_i \), where \( r_i \rightarrow a_i \). This can be easily achieved because CoT prompting can generate the answer in the final tokens of \( r_i \), and PoT prompting can get the answer by executing the codes, which takes a majority vote over the answers \( a_{ii=1} \) by: \[ \arg \max_a \sum_{i=1}^{k} 1 (a_i = a). \]

### 3.2 Prompting Details

We then introduce the two prompts \( I_{\text{para}} \) and \( I_{\text{inst}} \) in detail. We show a case in numerical reasoning in Figure 2 where \( k = 3 \) and the LLM needs to generate 2 paraphrases following the instruction \( I_{\text{para}} \) in the first step. \{question\} in \( I_{\text{para}} \) denotes the original question. \{[text1]\} and \{[text2]\} denote the placeholders for the output paraphrases.

In the second step, when prompting the LLM with \( I_{\text{inst}} \), there is a slight difference between the paraphrased question \( x^i \) and the original question \( x \). As shown in Figure 2, for generating reasoning paths for paraphrased questions, we put both \( x_i \) and \( x \) into \( I_{\text{inst}} \) because only including \( x_i \) will lose some key information in practice (with GPT-3.5) due to the imperfect paraphrasing process. For the original question, we simply remove the line starting with "Rephrased Question: ", as shown in the left gray box of Figure 2.

It worth noting that \( I_{\text{inst}} \) can also be combined with in-context learning, where \( I_{\text{inst}} \) will consist of \( T \) few-shot demonstrations denoted by \( D'_{\text{demo}} = \{ x_i, x_i', r_i \}_{i=1}^{T} \) without the instruction text for the paraphrased question. For the original question, \( I_{\text{inst}} \) will consist of \( D_{\text{demo}} = \{ x_i, r_i \}_{i=1}^{T} \).

### 4 Experiments

#### 4.1 Datasets

We conducted experiments on 5 reasoning datasets, which we categorized into three main classes: (1) The in-distribution numerical reason-
In-Distribution | Out-of-Distribution
--- | --- | --- | ---
GSM8K | SVAMP | ASDIV | GSMHARD

CoT | 70.3 | 80.1 | 84.7 | 31.1
Zero-Shot-PAL | 76.8 | 82.5 | 85.8 | 56.8
+ Self-Consistency\(T=0.4,k=5\) | 80.9 | 84.9 | 87.0 | 58.8
+ Self-Consistency\(T=0.7,k=5\) | 82.0 | 86.3 | 87.2 | 59.8
+ Self-Consistency\(T=1.0,k=5\) | 78.5 | 86.9 | 86.3 | 58.7
+ Self-Consistency\(T=0.4,k=10\) | 81.5 | 86.2 | 86.1 | 58.8
+ Self-Consistency\(T=0.7,k=10\) | 83.4 | 86.8 | 88.4 | 60.6
+ Self-Consistency\(T=1.0,k=10\) | 83.7 | 85.9 | 87.9 | 60.5

+ Self-Para-Consistency\(k=3\) | **83.8** | **88.0** | **87.8** | **66.3**

Table 1: Results of different methods on datasets for numerical reasoning. The **bold** represents the best scores.

Table 2: Results of different methods on the dataset for date understanding. The **bold** represents the best scores.

4.3 Main Results

The results of our method and baseline methods are shown in Table 1 and 2. Overall, our method, self-para-consistency, achieves the best performance on 4 out of 5 datasets.

For numerical reasoning, Self-Para-Consistency\(k=3\) improves the accuracy of the baseline Zero-Shot-PAL by 7.0, 5.0, and 9.5 points on GSM8K, SVAMP and GSMHARD, respectively. For date understanding, Self-Para-Consistency\(k=3\) improves the accuracy of the baseline Zero-Shot-PAL by 1.8 points.

Compared with Self-Consistency\(T=0.7,k=5\), Self-Para-Consistency\(k=3\) outperforms it on all 5 datasets. Compared with Self-Consistency\(T=0.7,k=10\), Self-Para-Consistency\(k=3\) outperforms it on 4 out of 5 datasets with 7 less reasoning paths required.

Compared with different datasets, our method improves most on GSM-HARD, which is typically...
Figure 3: A case study where the paraphrasing process misses some necessary information. Self-para-consistency alleviates the negative influence of incomplete paraphrases by concatenating the original and the paraphrased questions. As a result, it still outperforms the baseline self-consistency with fewer decoding paths.

4.4 Analysis of the Paraphrasing Process

A concern of this work is that the paraphrased question may not always be faithful to the original question, which may cause error accumulation for downstream tasks. To alleviate the problem, we concatenate the original and the paraphrased questions in the prompt as shown in Figure 2, which gives the LLM an opportunity to answer correctly even if the paraphrased question is wrong or incomplete.

We show a case in Figure 3 where the paraphrased questions miss some necessary information. The baseline Self-Consistency $\gamma_{T=1.0,k=5}$ yields low-quality reasoning paths and then gets a wrong answer. Our method Self-Para-Consistency $k=3$ can tolerate imperfect paraphrasing and finally answer correctly.

4.5 Limitations and Future Work

The paraphrasing process may have more applications that need to be explored. In the future, we would like to explore the following scenarios: 1) Integrating our method with self-consistency. For instance, generating $k_1$ paraphrases where each results in $k_2$ sampled decoding paths, resulting in a total of $k_1 \times k_2$ paths. This approach could potentially achieve a more favorable balance between quality and diversity in the decoding paths. 2) Combining our method with a paraphrase verifier to eliminate low-quality paraphrases, ensuring only the most accurate versions are used. 3) Employing self-para-consistency as an indicator of uncertainty, which could serve as a measure of the robustness of LLMs in reasoning tasks.

5 Conclusion

In this paper, we propose the self-para-consistency strategy, which serves as an alternative to the self-consistency strategy with lower costs. The proposed method first prompts the LLM to generate multiple paraphrases sequentially, and then generate reasoning paths in parallel. The diversity comes from paraphrasing instead of non-deterministic decoding strategies. Extensive experiments show the effectiveness of the proposed method.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62306344, and Natural Science Foundation of Guangdong Province of China under Grant 2024A1515010253.
References


