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

Wenqing Chen[1], **Weicheng Wang**[1], **Zhixuan Chu**[2,3*], **Kui Ren**[2,3], **Zibin Zheng**[1], **Zhichao Lu**[4]
[1]School of Software Engineering, Sun Yat-sen University
[2]The State Key Laboratory of Blockchain and Data Security, Zhejiang University
[3]School of Cyber Science and Technology, Zhejiang University
[4]Department 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.

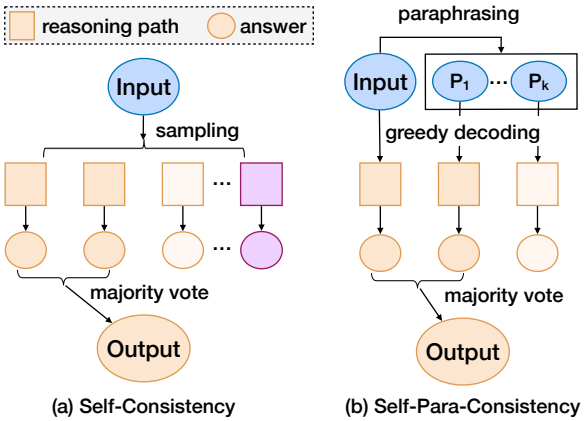

Figure 1: The comparison of self-consistency and self-para-consistency: (a) The self-consistency strategy samples diverse reasoning paths but includes some low-probability paths; (b) The self-para-consistency transfers the given questions into multiple paraphrases and then generates the corresponding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

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

Findings of the Association for Computational Linguistics: ACL 2024, pages 14162–14167

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). Although tuning the temperature or other parameters of LLMs can switch on the quality-diversity trade-off curve, previous studies suggest another type of diversity improvement (Hu et al., 2017; Wang et al., 2017; Ruan et al., 2020; Shao 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 θ) to generate $k-1$ paraphrased questions $G_{\text{para}} = x'ii = 1^{k-1}$ where $k > 1$. The prompt for paraphrasing is denoted by $\mathcal{I}$para, then the paraphrasing process can be formalized as:

$$\mathcal{P}_\theta(G_{\text{para}} \mid x, \mathcal{I}_{\text{para}}) = \prod_{i=1}^{k-1} \mathcal{P}_\theta(x'_i \mid x, \mathcal{I}_{\text{para}}, G^{<i}_{\text{para}}) \tag{1}$$

where $G^{<i}_{\text{para}}$ 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 $R_{\text{path}} = r_i i = 1^k$ in parallel. This step can be formalized as:

$$\begin{aligned} &\mathcal{P}_\theta(R_{\text{path}} \mid x, G_{\text{para}}, \mathcal{I}_{\text{inst}}) = \\ &\mathcal{P}_\theta(r_1 \mid x, \mathcal{I}_{\text{inst}}) \cdot \prod_{i=1}^{k-1} \mathcal{P}_\theta(r_{i+1} \mid x'_i, \mathcal{I}_{\text{inst}}) \end{aligned} \tag{2}$$

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.

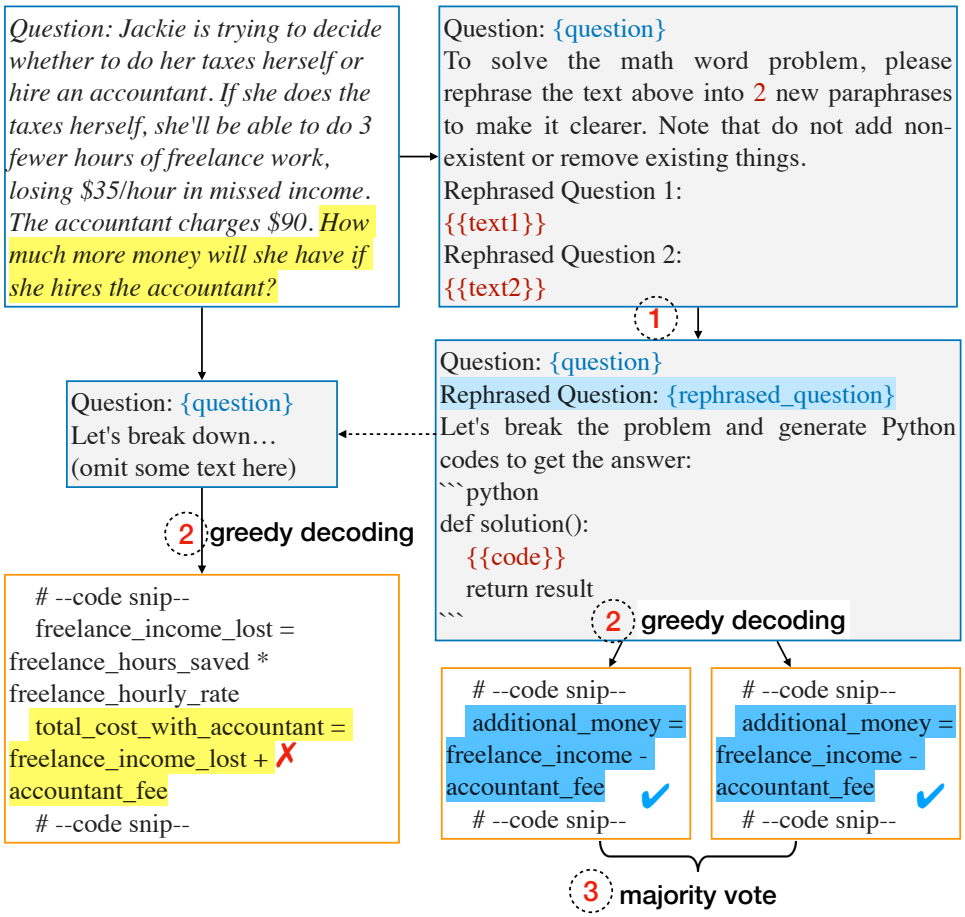

Figure 2: Illustration of the self-para-consistency with $k = 3$, which first prompts the LLM to generate 2 paraphrased questions based on PoT prompting for numerical reasoning. Then the LLM generates PoTs for the original question and paraphrased questions through greedy decoding. Finally, the answer is aggregated via majority voting.

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 $a_i{}_{i=1}^{k}$ by:$\arg\max_a \sum_{i=1}^{k} 1\,(a_i = a)$.

3.2 Prompting Details

We then introduce the two prompts $\mathcal{I}_{\text{para}}$ and $\mathcal{I}_{\text{inst}}$ 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 $\mathcal{I}_{\text{para}}$ in the first step. {question} in $\mathcal{I}_{\text{para}}$ denotes the original question. {{text1}} and {{text2}} denote the placeholders for the output paraphrases.

In the second step, when prompting the LLM with $\mathcal{I}_{\text{inst}}$, 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 $\mathcal{I}_{\text{inst}}$ 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 $\mathcal{I}_{\text{inst}}$ can also be combined with in-context learning, where $\mathcal{I}_{\text{inst}}$ 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, $\mathcal{I}_{\text{inst}}$ 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.

ing datasets, comprising GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), and ASDIV (Miao et al., 2020). (2) An out-of-distribution numerical dataset, GSM8K-hard, modified from GSM8K where the numbers in the questions were replaced with significantly larger values (Gao et al., 2023). (3) A symbolic reasoning dataset, date understanding sourced from BIG-bench (Suzgun et al., 2022), requiring inference of dates based on contextual information.

We realize that there are more reasoning datasets in previous work (Wang et al., 2022; Chen et al., 2022; Gao et al., 2023) like MAWPS (Koncel-Kedziorski et al., 2016), but those datasets are relatively easy for GPT-3.5 with PoT prompting, making them less distinguishable for different methods.

4.2 Baselines

Since PoT prompting has outperformed CoT prompting in complex reasoning tasks and PoT can obtain more consistent output formats than CoTs, the basic baseline in this paper is PoT prompting, also known as PAL (Gao et al., 2023). Based on PoT, the self-consistency strategy is performed with different temperatures T and sampling number k. The sampling number of our method, self-para-consistency, is set to $k = 3$ as we do not want to increase the costs and $k = 3$ is the smallest number to perform majority voting.

The GPT-3.5-turbo-0613 version is used for both baselines and our method. The temperature of our method is set to 0.0 in both the paraphrasing and reasoning stages. For date understanding, we use the same few-shot PoT examples as (Gao et al., 2023).

	DATE
Few-Shot-PAL$_{\text{GPT-3.5}}$	77.2
+ Self-Consistency$_{T=0.4,k=5}$	76.2
+ Self-Consistency$_{T=0.7,k=5}$	78.0
+ Self-Consistency$_{T=1.0,k=5}$	76.7
+ Self-Consistency$_{T=0.4,k=10}$	77.0
+ Self-Consistency$_{T=0.7,k=10}$	77.8
+ Self-Consistency$_{T=1.0,k=10}$	77.8
+ Self-Para-Consistency$_{k=3}$	**79.0**

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

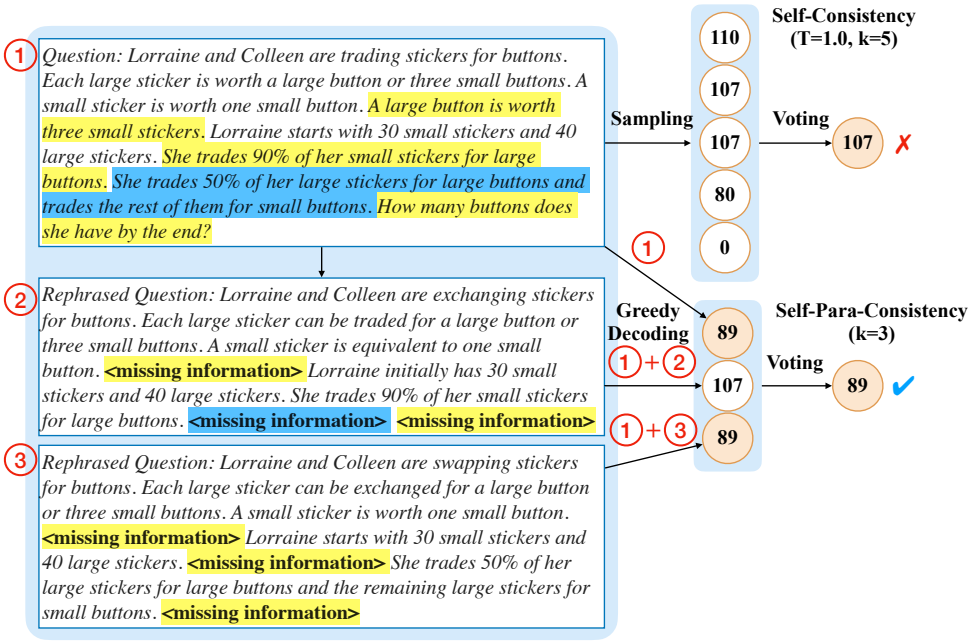

Figure 3: A case study where the paraphrasing process misses some necessary information. Self-para-consistency alleviates the negative influence of incomplete paraphrases by concatenating the original and the paraphrased questions. As a result, it still outperforms the baseline self-consistency with fewer decoding paths.

an OOD dataset where the training data of LLMs do not contain such large numbers. It shows that paraphrasing may introduce extra improvements for OOD data.

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$_{T=1.0,k=5}$ 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

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

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

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

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379*.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984.

Ehsan Montahaei, Danial Alihosseini, and Mahdieh Soleymani Baghshah. 2019. Jointly measuring diversity and quality in text generation models. *NAACL HLT 2019*, page 90.

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.

Yu-Ping Ruan, Zhen-Hua Ling, and Xiaodan Zhu. 2020. Condition-transforming variational autoencoder for generating diverse short text conversations. *ACM Trans. Asian Low Resour. Lang. Inf. Process.*, 19(6):79:1–79:13.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. 2023. Are emergent abilities of large language models a mirage? *arXiv preprint arXiv:2304.15004*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.

Huajie Shao, Jun Wang, Haohong Lin, Xuezhou Zhang, Aston Zhang, Heng Ji, and Tarek F. Abdelzaher. 2021. Controllable and diverse text generation in e-commerce. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 2392–2401. ACM / IW3C2.

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

Liwei Wang, Alexander G. Schwing, and Svetlana Lazebnik. 2017. Diverse and accurate image description using a variational auto-encoder with an additive gaussian encoding space. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5756–5766.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2023. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2021. Trading off diversity and quality in natural language generation. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 25–33.