ANALYSIS OF "SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS"

Ali Khalaji
ali.khalaji@stud.tu-darmstadt.de
July 25, 2023

Abstract

Language models are increasingly being used to solve complex reasoning tasks. However, they often struggle with these tasks because they can generate several different answers that are consistent with the prompt. This can make it difficult for the language model to select the correct answer. In this paper, I provide an analysis of a new decoding strategy called self-consistency. This strategy tackles a problem by sampling various reasoning paths and selecting the most consistent answer. The paper evaluates self-consistency decoding on a number of arithmetic and common sense reasoning benchmarks and shows that it significantly improves the performance of language models on these tasks. In addition, the limitations of the self-consistency method are discussed, and future research directions are suggested.

1 Introduction

Large language models (LLMs) have been increasingly used in recent years to solve complex reasoning tasks. They have a number of advantages, including accuracy, speed, and flexibility. However, LLMs can sometimes have difficulty solving maths problems or answering factual questions and can make mistakes such as generating incorrect or misleading answers. In addition, they can sometimes generate different answers to the same question, making it difficult to choose the right one.

The paper "Self-consistency improves chain of thought reasoning in language models" (Wang et al., 2023) discuss how chain of thought (CoT) prompting addresses this problem by improving the performance of LLMs on reasoning tasks. In their study, Wei et al. (2022) put forward an innovative approach to enhance the effectiveness of LLMs in tackling reasoning tasks. This approach, known as chain of thought prompting, involves utilizing a prompt to direct the LLM through a series of steps aimed at resolving a reasoning problem.

Consistency in LLMs To use the benefits of chain of thought and make consistent answers, it is important to understand what consistency means in the context of language models. Consistency is not just about making the same decision every time. It is about making decisions that are meaningful and relevant to the context of the question. (Elazar et al., 2021)

Consistency is an attractive feature of language understanding models. It refers to the ability of a model to make consistent decisions in semantically equivalent contexts. For example, if a model is asked to answer the question "What is the capital of France?" (Elazar et al., 2021), it should give the same answer whether the question is asked in English, French, or any other language. It allows models to generalize across language variability. This means that a consistent model can understand and interpret language in a way independent of the specific language being used.

In the next section, we will examine several works that investigate the consistency of language models by generating multi-step reasoning paths and different ways to choose the majority vote across the generated answers.

2 Related Work

Enhancing language model outputs through re-ranking Cobbe et al. (2021) tackle the issue of multi-step mathematical reasoning tasks by training a verifier to re-rank generated solutions to math word problems generated by the model. The verifier judges the correctness of model completions and selects the highest-ranked solution from a set of candidates, annotated by humans. The approach is presented on the GSM8K, a dataset of diverse grade school math word problems, and demonstrates sig-

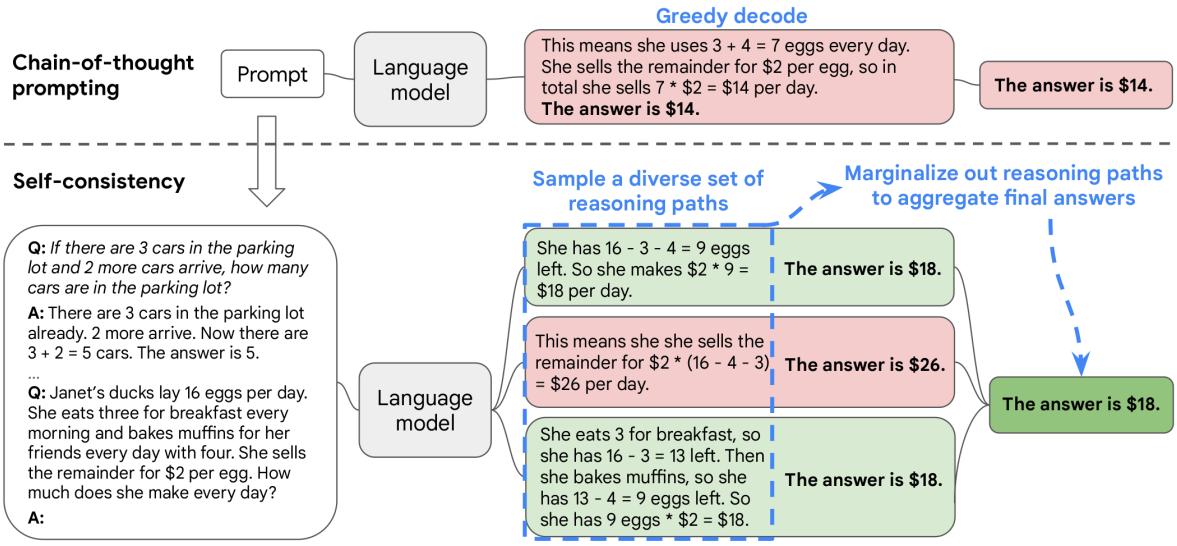

Figure 1: The steps of Self-Consistency method compared to greedy decode

nificant performance improvements over standard language model fine-tuning.

Improving consistency in pre-trained language models Elazar et al. (2021) address the challenge of assessing the consistency of pre-trained language models (PLMs) with respect to factual knowledge. To tackle this issue, they introduce the PARAREL dataset, specifically designed to evaluate PLM consistency. The authors propose a method for enhancing model consistency by incorporating additional consistency loss during pre-training. By focusing on factual knowledge consistency, their approach aims to improve the reliability and accuracy of language models in handling real-world information. Welleck et al. (2020) address inconsistency issues in decoding algorithms for recurrent language models. It introduces two remedies: consistent variants of top-k and nucleus sampling, and a self-terminating recurrent language model. These methods prevent inconsistency and improve the quality of generated sequences, the reliability of the model and its performance on various tasks, Nye et al. (2020) neural and symbolic approaches have different strengths. Neural approaches are good at learning statistical patterns, while symbolic approaches are good at representing and reasoning about knowledge. By combining the strengths of these two approaches, we can create models that are more robust. This will enable the system to integrate domain knowledge more effectively and enhance its overall performance.

Rationale generation of reasoning paths Multimodal CoT incorporates both text and image modalities. The approach (Zhang et al., 2023), use a two-stage framework that separates rationale generation and answer inference. This allows the model to access more information about the problem, such as the input text and image, and generate more accurate rationales. The rationales can then be used to improve the performance of answer inference.

3 Self-Consistency Method

Before discussing the self-consistency method, we need to understand how greedy decoding works. Greedy decoding (Wei et al., 2022) is a simple text generation method that chooses the word with the highest probability of being the next word in the sequence. While this method is both easy to use and efficient, it may result in repetitive or nonsensical text since it solely focuses on the most probable word at each step, rather than considering the bigger picture for a coherent sequence.

This is where the self-consistency method comes in. The self-consistency method presented involves three steps (Figure 1):

1. Use CoT prompting to elicit responses from a language model.

2. Extending CoT prompting by sampling from the language model decoder and generating a diverse set of reasoning paths rather than relying on greedy decoding.

	GSM8K	MultiArith	AQuA	SVAMP	CSQA	ARC-c
Greedy decode	56.5	94.7	35.8	79.0	79.0	85.2
Weighted avg (unnormalized)	56.3 ± 0.0	90.5 ± 0.0	35.8 ± 0.0	73.0 ± 0.0	74.8 ± 0.0	82.3 ± 0.0
Weighted avg (normalized)	22.1 ± 0.0	59.7 ± 0.0	15.7 ± 0.0	40.5 ± 0.0	52.1 ± 0.0	51.7 ± 0.0
Weighted sum (unnormalized)	59.9 ± 0.0	92.2 ± 0.0	38.2 ± 0.0	76.2 ± 0.0	76.2 ± 0.0	83.5 ± 0.0
Weighted sum (normalized)	74.1 ± 0.0	99.3 ± 0.0	48.0 ± 0.0	86.8 ± 0.0	80.7 ± 0.0	88.7 ± 0.0
Unweighted sum (majority vote)	74.4 ± 0.1	99.3 ± 0.0	48.3 ± 0.5	86.6 ± 0.1	80.7 ± 0.1	88.7 ± 0.1

Figure 2: Accuracy comparison of different answer aggregation strategies on PaLM-540B.

3. Aggregate the reasoning paths by marginalizing them and selecting the most consistent answer from the final set of answers.

We have a fixed answer Set A with m candidate outputs and each generated answers $a_i \in A$.

Taking the majority vote between the generated reasoning paths by a marginalization over different reasoning paths r_i:

$$argmax_a \sum_{i=1}^{m} \mathbb{1}(a_i = a)$$

Figure 2 compares the accuracy of greedy decoding and self-consistency using majority voting. As can be seen, majority voting is more effective at aggregating different reasoning paths than greedy decoding and achieves a better result.

> This is probably because majority voting considers a wider range of possibilities and is therefore less likely to be misled by local maxima.

They have also used the calculation of the unnormalized probability of each pair of r_i,a_i, given prompt and question, and normalized it by the output length:

$$P(r_i, a_i) =$$

$$exp^{\frac{1}{K}} \sum_{k=1}^{K} logP(t_k|prompt, question, t_1, ..., t_{k-1})$$

with the assumption that each token t_k depends on the previous tokens (t_1,...,t_{k-1}). Unnormalized probability refers to the probability value before it undergoes normalization, which is then divided by the output length.

Weighted sum and unweighted sum are two methods for combining multiple scores or probabilities. (Cai1 et al., 2023) Weighted sum is more accurate than unweighted sum, because it allows the scores or probabilities to be combined in a more nuanced way. However, a weighted sum can also be more computationally expensive. An unweighted sum is often used when accuracy is not as important, or when the scores or probabilities are already preweighted. However, the results in Figure 2 show that the accuracy of majority vote is as good as the normalized weighted sum.

> I find this surprising. I suspect that the majority voting algorithm may have been implemented in a way that was particularly effective for the tasks used in the study. For example, the algorithm may have been able to identify and discard outliers in the reasoning paths, which would have improved the accuracy of majority voting.

4 Experiments

To evaluate the self-consistency method, they used several benchmarks, including Arithmetic Reasoning, Commonsense Reasoning, and Symbolic Reasoning, and across four language models, UL2, LaMDA, GPT-3, and PaLM. The results (Figure 8 and Figure 9) show that self-consistency outperforms CoT-prompting on every benchmark, and is the new state-of-the-art on almost all tasks.

> I think it is a good point to evaluate the method on different language models, as different LMs have different strengths and weaknesses, which can help us to better understand the method and its limitations.

For example, GPT-3 is known for its ability to generate creative text, (Bi et al., 2023) while PaLM is

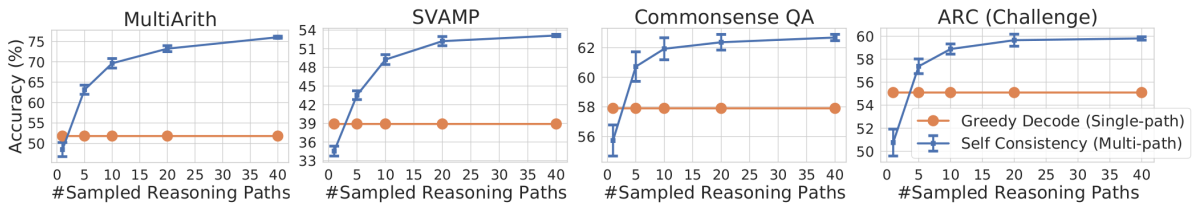

Figure 3: More diverse reasoning paths improve accuracy in self-consistency over CoT-prompting with greedy decoding

	ANLI R1 / R2 / R3	e-SNLI	RTE	BoolQ	HotpotQA (EM/F1)
Standard-prompting (no-rationale)	69.1 / 55.8 / 55.8	85.8	84.8	71.3	27.1 / 36.8
CoT-prompting (Wei et al., 2022)	68.8 / 58.9 / 60.6	81.0	79.1	74.2	28.9 / 39.8
Self-consistency	**78.5 / 64.5 / 63.4**	**88.4**	**86.3**	**78.4**	**33.8 / 44.6**

Figure 4: Standard CoT prompting vs. self-consistency on common NLP tasks

known for its ability to answer questions accurately. This means that by evaluating the self-consistency method on different LMs, we can get a better understanding of their strengths and weaknesses.

> In addition, the different LMs are of different sizes. This means that we can also evaluate the scalability of the self-consistency method. For example, we can see if the method works as well on a large language model like PaLM as it does on a smaller language model like GPT-3.

Figure 3 shows the effect of the number of reasoning paths generated. As can be seen, the more reasoning paths a model generates, the more accurate and robust its results will be. This may be because the model can gain a more diverse understanding of the problem and consider a variety of possible solutions. This is particularly beneficial for complex tasks involving arithmetic and common sense reasoning.

Comparison with CoT on common NLP tasks Ye and Durrett (2022) discovered that CoT prompting can sometimes harm performance compared to standard prompting (Brown et al., 2020) in few-shot learning for certain NLP tasks. To address this gap, a study using self-consistency was conducted on common NLP tasks using PaLM-540B (Chowdhery et al., 2022). The results in Figure 4 reveal that while adding CoT can hurt performance in some tasks, self-consistency boosts performance, surpassing standard prompting. This makes self-consistency a reliable approach for enhancing rationales in few-shot in-context learning for various NLP tasks, particularly when CoT prompting is detrimental.

Self-Consistency vs. Beam Search The authors have compared in Figure 5 the self-consistency method with the decoding algorithm (Li and Jurafsky, 2016) that increases diversity, based on beam search, where the decoder generates a sequence of words by iteratively selecting the most likely word given the previous words, while keeping track of the top-k (beam size) most likely sequences. However, Self-Consistency significantly outperforms beam search on the UL2-20B model, as it yields a higher diversity of inference paths. The ability of self-consistency to outperform beam search on diverse reasoning path tasks makes it a promising method for improving neural generation models.

Effect of Self-Consistency on imperfect prompts The authors have investigated how self-consistency can impact one's performance, even when dealing with nonsensical prompts. Human-generated prompts can contain small errors, which can reduce the accuracy of greedy decoding. Self-consistency, on the other hand, can help the model to be more robust to imperfect prompts by filling in the gaps and improving the results. Self-consistency can also provide an estimate of the uncertainty of the solutions generated by the model, as the consistency of decoded responses is highly correlated with accuracy. This allows the model to "know when it doesn't know" and to acknowledge its lack

	Beam size / Self-consistency paths	1	5	10	20	40
AQuA	Beam search decoding (top beam)	23.6	19.3	16.1	15.0	10.2
	Self-consistency using beam search	23.6	19.8 ± 0.3	21.2 ± 0.7	24.6 ± 0.4	24.2 ± 0.5
	Self-consistency using sampling	19.7 ± 2.5	**24.9** ± **2.6**	**25.3** ± **1.8**	**26.7** ± **1.0**	**26.9** ± **0.5**
MultiArith	Beam search decoding (top beam)	10.7	12.0	11.3	11.0	10.5
	Self-consistency using beam search	10.7	11.8 ± 0.0	11.4 ± 0.1	12.3 ± 0.1	10.8 ± 0.1
	Self-consistency using sampling	9.5 ± 1.2	11.3 ± 1.2	**12.3** ± **0.8**	**13.7** ± **0.9**	**14.7** ± **0.3**

Figure 5: Self-Consistency vs. Beam Search on the UL2-20B model with the same number of beams and reasoning paths.

of confidence in certain situations. (Figure 6)

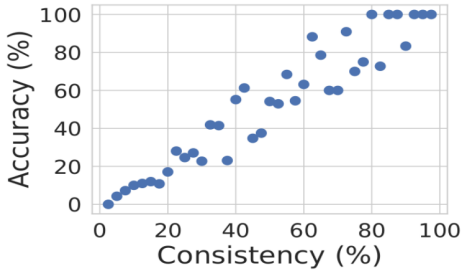

Figure 6: The relationship between model accuracy and consistency.

Self-Consistency and non-natural reasoning paths Self-consistency increases model accuracy by generating intermediate equations, although gains may be sometimes limited by the lack of diversity in equation decoding. For example, transforming a natural language sentence ("There are already 3 cars in the car park. 2 more arrive. Now there are 3 + 2 = 5 cars.") to an intermediate equation ("3 + 2 = 5") shows some limitations. In this case, there is little room for the creation of alternative reasoning paths. The accuracy improvement, as shown in Figure 7, prompt with equations, isn't as significant when generating intermediate equations compared to generating natural language reasoning paths. This is because equations are much shorter than natural language, so there is less opportunity for the decoding process to generate different solutions.

LaMDA-137B	Prompt with equations	5.0
	+ Self-consistency (40 paths)	**6.5**
PaLM-540B	Zero-shot CoT (Kojima et al., 2022)	43.0
	+ Self-consistency (40 paths)	**69.2**

Figure 7: Self-Consistency in zero-shot scenarios.

5 Discussion and Limitations

CoT Reasoning, a good starting point

In my opinion, taking advantage of the newly introduced chain of thought prompting would be a good starting point for developing a more robust method for reasoning and question answering. CoT can be used to develop more robust methods for these tasks by providing a clear and structured framework to guide the reasoning and question-answering processes. This framework can help ensure that the methods are able to perform well under a variety of different conditions, such as when the data is noisy or incomplete.

Self-Consistency and encoder-only language models Encoder-only language models are typically less accurate than decoder-only language models, but they are also faster. (Fu et al., 2023) This means that they may be more suitable for some applications, such as real-time dialogue.

Self-Consistency is relatively new and has not been evaluated on a wide range of language models. It would be interesting to see how it performs on an encoder-only language model.

A potential challenge of using the self-consistency method on an encoder-only language model may be that the method requires the language model to be able to generate several different reasoning paths. Encoder-only language models are typically unable to do this because they can only generate a single sequence of text. However, the self-consistency method may be adapted to work with encoder-only language models.

Despite the potential benefits of self-consistency in language models, there are certain limitations that need to be considered.

Firstly, one notable limitation is the computational cost of the method, particularly for larger language models. Implementing self-consistency decoding may require more time and resources, which could impact real-time applications or systems with limited computational capabilities.

Moreover, the method may not be well-suited for tasks that involve complex reasoning, (Singhal et al., 2023) such as scientific or medical tasks. These tasks often require deeper understanding and domain-specific knowledge, which may not be adequately captured by the self-consistency decoding method. The method relies on the given prompt for generating answers, and tasks with insufficient context may result in less accurate responses.

Furthermore, the self-consistency decoding method may have limitations in tasks that require more diverse or creative answers, as discussed in the last section by generating intermediate equations. Language models may struggle to generate highly creative responses, leading to less diverse outputs for such tasks.

> The paper mentioned that the proposed method required more computing resources, but didn't provide a thorough analysis of its exact cost. This lack of detailed analysis could be worrying about real-world applications.

In addition, the paper didn't explore the cases where the method might not work well, which could have helped us better understand its limitations.

6 Conclusion and Future Work

In this paper, the authors introduced a novel decoding strategy called "Self-Consistency" to address a challenge faced by language models, where they sometimes produce multiple consistent answers to complex reasoning tasks, leading to ambiguity and uncertainty in their responses. To evaluate the effectiveness of self-consistency decoding, they conducted extensive testing on several arithmetic and common sense reasoning benchmarks, using multiple language models. The results showed significant improvements in the performance of the models when self-consistency decoding was applied. By incorporating reasoning paths, the decision-making process became more transparent and understandable, providing clearer insights into how they arrived at their answers. Moreover, the implementation of self-consistency improved the output calibration and uncertainty estimation of the model's predictions, making them more trustworthy, which is particularly valuable in critical applications where accurate and reliable results are essential.

Looking ahead, the paper suggests potential future work to explore the full potential of self-consistency decoding. One exciting prospect is to utilize self-consistency decoding to generate better-supervised data for fine-tuning language models. This can lead to further accuracy improvements in a single inference run after fine-tuning, enhancing the overall effectiveness of the model. The paper also highlights the importance of addressing inconsistencies and inaccuracies that may arise in the reasoning paths generated by the models. Further research in this area can help improve the trustworthiness and reliability of the models, making them more suitable for real-world applications.

Use of AI writing Assistances

To improve the overall quality of my paper analysis, I used AI writing tools for language checking. These tools fixed any mistakes in grammar and spelling and helped me find appropriate words that precisely express my viewpoints and ensure that the paper was well-organized. They also helped me to follow the rules of academic writing and made my work look more professional. With the help of AI tools, my analysis became clearer and more suitable for academic readers.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Bin Bi, Chenliang Li, Chen Wu, Ming Yan, Wei

Wang, Songfang Huang, Huang Fei, and Luo Si. 2023. Palm: Pre-training an autoencodingautoregressive language model for context-conditioned generation. *rXiv:2004.07159v2 [cs.CL] 20 Sep 2020*.

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. *arXiv:2005.14165*.

Zefan Cai1, Baobao Chang1, and Wenjuan Han. 2023. Human-in-the-loop through chain-of-thought. *arXiv:2306.07932v2 [cs.CL] 23 Jun 2023*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *rXiv:2204.02311v5 [cs.CL] 5 Oct 2022*.

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. *rXiv:2110.14168v2 [cs.LG] 18 Nov 2021*.

Yanai Elazar, Nora Kassner, Shauli Ravfogel1, Abhilasha Ravichander, Eduard Hovy, Hinrich Schutze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *TACL journal 2021*.

Zihao Fu, Wai Lam, Qian Yu, Anthony Man-Cho So, Shengding Hu, Zhiyuan Liu, and Nigel Collier. 2023. Decoder-only or encoder-decoder? interpreting language model as a regularized encoder-decoder. *arXiv:2304.04052*.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Yihuai Lan, Lei Wang, Qiyuan Zhang, Yunshi Lan, Bing Tian Dai, Yan Wang, Dongxiang Zhang, and Ee-Peng Lim. 2021. Mwptoolkit: An open-source framework for deep learning-based math word problem solvers. *CoRR*, abs/2109.00799.

Jiwei Li and Dan Jurafsky. 2016. Mutual information and diverse decoding improve neural machine translation. *arXiv:1601.00372v2 [cs.CL] 22 Mar 2016*.

Maxwell Nye, Michael Henry Tessler, Joshua B. Tenenbaum, and Brenden M. Lake. 2020. Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning. *NeurIPS 2021*.

Xinyu Pi, Qian Liu, Bei Chen, Morteza Ziyadi, Zeqi Lin, Qiang Fu, Yan Gao, Jian-Guang Lou, and Weizhu Chen. 2022. Reasoning like program executors.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Abubakr Babiker, Nathanael Schärli, Aakanksha Chowdhery, Philip Mansfield, Dina Demner-Fushman, Blaise Agüera y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, and Vivek Natarajan. 2023. Large language models encode clinical knowledge. *10.1038/s41586-023-06291-2*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self consistency improves chain of thought reasoning in language models. *ICLR 2023*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *Conference on Neural Information Processing Systems*.

Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. 2020. Consistency of a recurrent language model with respect to incomplete decoding. *10.18653/v1/2020.emnlp-main.448*.

Yichong Xu, Chenguang Zhu, Shuohang Wang, Siqi Sun, Hao Cheng, Xiaodong Liu, Jianfeng Gao,

Pengcheng He, Michael Zeng, and Xuedong Huang. 2021. Human parity on commonsenseqa: Augmenting self-attention with external attention. *CoRR*, abs/2112.03254.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. *NeurIPS 2022 Conference Program Chairs*.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. 2023. Multimodal chain-of-thought reasoning in language models. *arXiv:2302.00923v4 [cs.CL] 17 Feb 2023*.

Appendix

	Method	AddSub	MultiArith	ASDiv	AQuA	SVAMP	GSM8K
	Previous SoTA	**94.9**[a]	60.5[a]	75.3[b]	37.9[c]	57.4[d]	35[e] / 55[g]
UL2-20B	CoT-prompting	18.2	10.7	16.9	23.6	12.6	4.1
	Self-consistency	24.8 (+6.6)	15.0 (+4.3)	21.5 (+4.6)	26.9 (+3.3)	19.4 (+6.8)	7.3 (+3.2)
LaMDA-137B	CoT-prompting	52.9	51.8	49.0	17.7	38.9	17.1
	Self-consistency	63.5 (+10.6)	75.7 (+23.9)	58.2 (+9.2)	26.8 (+9.1)	53.3 (+14.4)	27.7 (+10.6)
PaLM-540B	CoT-prompting	91.9	94.7	74.0	35.8	79.0	56.5
	Self-consistency	93.7 (+1.8)	99.3 (+4.6)	81.9 (+7.9)	48.3 (+12.5)	86.6 (+7.6)	74.4 (+17.9)
GPT-3 Code-davinci-001	CoT-prompting	57.2	59.5	52.7	18.9	39.8	14.6
	Self-consistency	67.8 (+10.6)	82.7 (+23.2)	61.9 (+9.2)	25.6 (+6.7)	54.5 (+14.7)	23.4 (+8.8)
GPT-3 Code-davinci-002	CoT-prompting	89.4	96.2	80.1	39.8	75.8	60.1
	Self-consistency	91.6 (+2.2)	**100.0** (+3.8)	**87.8** (+7.6)	**52.0** (+12.2)	**86.8** (+11.0)	**78.0** (+17.9)

Figure 8: Arithmetic reasoning accuracy by self-consistency compared to chain-of-thought prompting (Wei et al., 2022). The previous SoTA baselines are obtained from: a: Relevance and LCA operation classifier (Roy and Roth, 2015), b: Lan et al. (2021), c: Amini et al. (2019), d: Pi et al. (2022), e: GPT-3 175B finetuned with 7.5k examples (Cobbe et al., 2021), g: GPT-3 175B finetuned plus an additional 175B verifier (Cobbe et al., 2021). The best performance for each task is shown in bold.

	Method	CSQA	StrategyQA	ARC-e	ARC-c	Letter (4)	Coinflip (4)
	Previous SoTA	**91.2**[a]	73.9[b]	86.4[c]	75.0[c]	N/A	N/A
UL2-20B	CoT-prompting	51.4	53.3	61.6	42.9	0.0	50.4
	Self-consistency	55.7 (+4.3)	54.9 (+1.6)	69.8 (+8.2)	49.5 (+6.8)	0.0 (+0.0)	50.5 (+0.1)
LaMDA-137B	CoT-prompting	57.9	65.4	75.3	55.1	8.2	72.4
	Self-consistency	63.1 (+5.2)	67.8 (+2.4)	79.3 (+4.0)	59.8 (+4.7)	8.2 (+0.0)	73.5 (+1.1)
PaLM-540B	CoT-prompting	79.0	75.3	95.3	85.2	65.8	88.2
	Self-consistency	80.7 (+1.7)	**81.6** (+6.3)	**96.4** (+1.1)	**88.7** (+3.5)	70.8 (+5.0)	91.2 (+3.0)
GPT-3 Code-davinci-001	CoT-prompting	46.6	56.7	63.1	43.1	7.8	71.4
	Self-consistency	54.9 (+8.3)	61.7 (+5.0)	72.1 (+9.0)	53.7 (+10.6)	10.0 (+2.2)	75.9 (+4.5)
GPT-3 Code-davinci-002	CoT-prompting	79.0	73.4	94.0	83.6	70.4	99.0
	Self-consistency	81.5 (+2.5)	79.8 (+6.4)	96.0 (+2.0)	87.5 (+3.9)	**73.4** (+3.0)	**99.5** (+0.5)

Figure 9: Commonsense and symbolic reasoning accuracy by self-consistency compared to chain-of-thought prompting (Wei et al., 2022). The previous SoTA baselines are obtained from: a: DeBERTaV3-large + KEAR (Xu et al., 2021), b: Chowdhery et al. (2022), c: UnifiedQA-FT (Khashabi et al., 2020). The best performance for each task is shown in bold.

	GSM8K	StrategyQA
Question	*Henry made two stops during his 60-mile bike trip. He first stopped after 20 miles. His second stop was 15 miles before the end of the trip. How many miles did he travel between his first and second stops?*	*Yes or no: Is Albany, Georgia the most populous US Albany?*
Greedy Decode	He traveled 60 miles in total. He stopped after 20 miles, so he traveled 60 - 20 = 40 miles between the first and second stops. The answer is 40.	Albany, Georgia is the most populous US Albany. Thus, the answer is yes.
Sampled Path 1	Henry travelled a total of 60 miles. His first stop was 20 miles in, and his second stop was 15 miles before the end. So between his first and second stops he travelled 60 - 20 -15 = 25 miles. The answer is 25.	The most populous US Albany is Albany, New York. Thus, Albany, Georgia is not the most populous US Albany. So the answer is no.
Sampled Path 2	He made two stops during a 60-mile trip. The first was 20 miles into the trip. The second was 15 miles before the end of the trip. This means the second stop was 60 - 15 = 45 miles into the trip. Since he made the stops in order, the second stop must have been 45 - 20 = 25 miles after the first stop. The answer is 25.	Albany, Georgia has a population of about 88,000. Albany, New York has a population of about 95,000. Thus, Albany, Georgia is not the most populous US Albany. So the answer is no.

Figure 10: Examples where self-consistency helps repair the errors over greedy decode, on PaLM-540B. Two sampled reasoning paths that are consistent with the ground truth are shown.

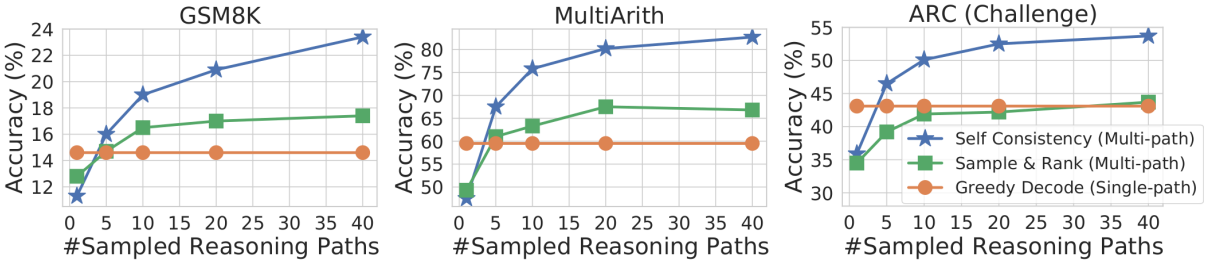

Figure 11: Self-consistency significantly outperforms sample-and-rank with the same # of samples.

	GSM8K	MultiArith	SVAMP	ARC-e	ARC-c
CoT (Wei et al., 2022)	17.1	51.8	38.9	75.3	55.1
Ensemble (3 sets of prompts)	18.6 ± 0.5	57.1 ± 0.7	42.1 ± 0.6	76.6 ± 0.1	57.0 ± 0.2
Ensemble (40 prompt permutations)	19.2 ± 0.1	60.9 ± 0.2	42.7 ± 0.1	76.9 ± 0.1	57.0 ± 0.1
Self-Consistency (40 sampled paths)	**27.7** ± **0.2**	**75.7** ± **0.3**	**53.3** ± **0.2**	**79.3** ± **0.3**	**59.8** ± **0.2**

Figure 12: Self-consistency outperforms prompt-order and multi-prompt ensembles on LaMDA-137B.