Rationale Belief Aggregation for Self-Verified Reasoning

Stanford CS224N Custom Project

Vaishnavi Shrivastava
Department of Computer Science
Stanford University
vaish1@stanford.edu

Abstract

Large language models such as GPT-3 [1] have a great deal of information encoded within their parameters, however, our ability to access this information is bottlenecked by how we communicate or interface with these models, namely through prompting. Chain-of-thought prompting [2] demonstrates the value of producing step-by-step reasoning chains (rationales) before answering a question, and self-consistency [3] shows that sampling multiple rationales and aggregating their outputs can allow for more robust reasoning. In this work we postulate that rationales consist of multiple beliefs, or informational phrases that the model uses as context for its reasoning (which may or may not be factual), and some inference over these beliefs. Empirically, we observe that different rationales expose different beliefs and hypothesize that performing a principled aggregation over the beliefs surfaced by different rationales would allow us to reduce internal contradictions within a language model and produce more consistent rationales to reason over. We propose two such aggregation strategies, Belief Aggregation and Belief Majority Voting, and evaluate their performance over three challenging QA datasets [4]. Our method results in modest gains in accuracy over self-consistency and greedy decoding for chain-of-thought prompting, while providing strong gains in coverage (% of questions a model is able to answer without abstaining), thereby resulting in confident reasoning over a larger set of questions.

1 Key Information to include

- External collaborators (if you have any): Michi Yasunaga, Percy Liang, Tatsu Hashimoto
- External mentor (if you have any): Percy Liang, Tatsu Hashimoto
- CS 224N mentor: Irena Gao
- Sharing project: CS 324: Foundation Models

2 Introduction

Large pre-trained language models [1] have increasingly encapsulated more and more world-knowledge in their weights. However, as chain-of-thought prompting [2] has shown, prompting models with the step-by-step reasoning required to solve a question can elicit higher-order reasoning capabilities from large language models. Self-Consistency based decoding [3] has also shown that generating multiple reasoning paths (which we will refer to as rationales) and computing a final output by majority voting on the sampled outputs can significantly improve performance over greedily sampling a single rationale, which can suffer from local optimization and repetitive generations [5]. In most cases, chain-of-thought rationales consist of a set of beliefs followed by some inference based on these beliefs. We define beliefs as informational statements such as *'Barack Obama has 2 children'*, which the model uses to reason about related questions. These beliefs could be true or false (i.e. the model could equivalently generate *'Barack Obama has 3 children'*, which is false), but are implicitly treated by the LM to be true. When sampling multiple rationales, we empirically observe that different rationales can expose different beliefs that the language model may have about the

given question. For example, rationale 1 may surface beliefs A and B, while rationale 2 may surface beliefs C and D. The problem we seek to solve is, how can aggregate the beliefs surfaced in different rationales, use them to verify each other and gain a more consistent view of what the language model believes, and use this set of self-verified beliefs that we are more confident in to reason more robustly about complex problems?

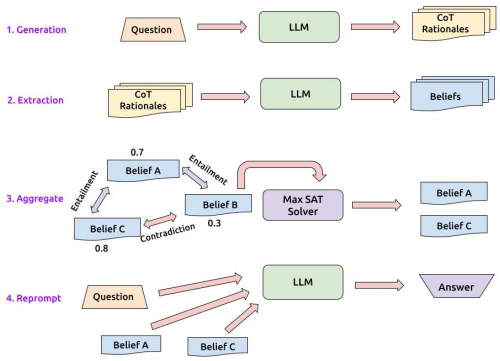
This research question of aggregating rationales for more confident and consistent reasoning is important because no single sampled rationale can provide a full sense of the information a language model contains about a particular question. By relying on just one rationale, the language model answers questions using an incomplete set of information. Furthermore, the information exposed in a given rationale may be an unfaithful reflection of the 'true beliefs' that the LM actually has. Given the diversity and volume of data used to train an LM, and the problem of hallucination during generation [6], LMs most likely encode and generate many facts that contradict themselves. Therefore employing an element of self-verification is also crucial for consistent reasoning by surfacing a large set of beliefs relevant to a question, checking for contradictions between these beliefs, and finding the most confident set of non-contradictory beliefs that can be provided as context to the model. This set would ideally represent a relatively complete set of information relevant to a given question the model is most confident about, and therefore serve as more robust context to base the language model's reasoning on than a single rationale.

3 Related Work

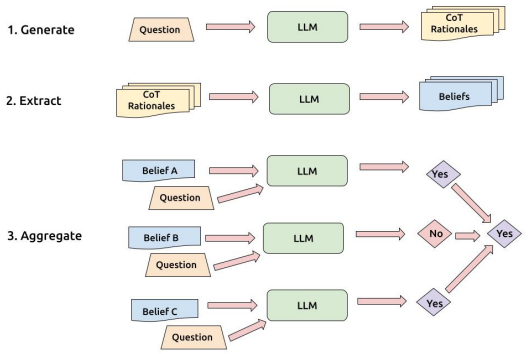
There has been some prior work done on sampling multiple rationales from a model and using them to answer an overall question. For example, the Self-Consistency paper samples a series of rationales from a language model and then aggregates across these rationales. The primary difference in this approach is that with Self-Consistency, the aggregation is done over the final answers instead of over on the beliefs or factual information exposed in each rationale. Aggregating over final answers can be insufficient in cases where the sampled outputs are evenly distributed over the space of final answers. Self-Consistency also doesn't improve rationale quality or improve performance in cases where information from multiple rationales is required to faithfully and factually reason about and answer a question. The Ask Me Anything technique [7] also prompts a model multiple times for a given question, but instead of generating chain-of-thought style rationales the model is prompted using a series of templates to generate questions related to the original question and the final answers are aggregated. Our technique would not require template-based questions and would perform aggregation at a more granular level based on the beliefs within a rationale.

There are also works related to making the outputs of a model more consistent. The ConCord framework [8] samples multiple output answers for a given question and then aims to make a model's output for different questions consistent with each other, also by checking for entailments between different sampled answers across questions and using a MAX-SAT solver to ensure that the selected outputs are consistent. Our work notably differs from this since we sample rationales (instead of directly sampling answers) and instead of checking for consistency across responses to different questions, we check for consistency regarding model beliefs surfaced pertaining to a given question. Maieutic prompting [9] is another technique that pursues logically consistent reasoning by abductively prompting a model to rationalize both possible answers of True or False questions and using a MAX-SAT solver to unify the resulting output explanations. Our method differs from this technique since instead of abductively prompting a model, we sample and aggregate across multiple rationales, and our technique can be applied to datasets where the answer is more nuanced than just True or False.

Related methods also exist which provide model generated content as context for the model to answer a given question. Liu et. al. [10] prompt the model to specifically elicit more knowledge about a given commonsense question and then reprompt the model with the produced knowledge. This work differs from ours in that we sample chain-of-thought style rationales from the model, which include a series of related chained facts ultimately leading to an answer and aggregate across these chains, instead of reprompting for individual pieces of information from the model. Yu et. al. [11] generate a diverse set of documents related to a given question, instead of retrieving related documents, and answer questions in the context of the retrieved documents. This differs from our technique since there is no self-verification step of extracting information across documents and finding an internally consistent set of information.



(a) Belief Aggregation: Generate, Extract, Aggregate, Reprompt



(b) Belief Majority Voting: Generate, Extract, Aggregate

Figure 1: Diagrams of the Belief Aggregation and Belief Majority Voting Techniques

4 Methods

Task. The main tasks to evaluate this method on will be complex multi-step reasoning tasks, such as commonsense reasoning or reasoning with world knowledge, and arithmetic reasoning. Though the specifics may vary, the task in each of these cases is given a question to reason about it explicitly and then produce the final answer predicated on this reasoning. In our method, we will sample multiple such rationales and aggregate the information found across them to answer the final question. An example input question (from the StrategyQA dataset) may be "Do placozoa get learning disabilities?", with an example chain-of-thought output being "Placozoa are simple animals. They do not have a nervous system. They cannot learn. Thus, placozoa do not get learning disabilities. So the answer is no." We will sample and aggregate multiple such outputs.

Methods. Our first aggregation method **Belief Aggregation** has the following steps:

1. **Generate**: We begin by sampling several rationales from a language model, similar to Self-consistency. In the current version of this technique, we use chain-of-thought prompting with temperature sampling to generate rationales. In further iterations, we may also incorporate other prompting techniques like self-ask or decomposition-based prompting [12] to extract more diverse information from the model.
2. **Extract**: After the rationales are sampled, we extract the model beliefs or factual information from the rationales, experimenting with different few-shot prompts to extract this information. Currently we use two different types of extraction, annotated as 'Keypoints' and 'Distinct Facts' in the results. In 'Keypoints' extraction we prompt the model with some in-context examples and the instruction 'What are the keypoints that can be extracted from the following statements?'. With the 'Distinct Facts' extraction we change the instruction to 'Remove any repeated information from the following statements.' and correspondingly change the in-context examples. The statements in each case include the sampled rationales. In addition, we post-process the beliefs by manually filtering 'uninformative' beliefs, since models (text-davinci-002 in particular) have a tendency to often express uncertainty even if they contain a particular answer. We further deduplicate beliefs with a large amount of shared content by computing an n-gram based overlap metric and filtering out beliefs with overlap above a given threshold.
3. **Aggregate**: For each extracted belief, we compute a confidence score, based on the language modeling probability of generating that belief given the question - P(belief | question), which is further normalized by dividing by P(belief) to prevent up-weighting generic beliefs [**?**]. Additionally, in order to self-verify and shed contradictory beliefs, we check the pairwise entailments between every pair of beliefs and determine which ones are contradictory to each other. After these beliefs have been extracted, and their entailment relationships and

confidence scores have been computed, we use a MAX-SAT solver to select the highest confidence, non-contradictory subset of beliefs.

4. **Reprompt**: Then, we reprompt the model with the question and the selected subset of beliefs, to generate the final answer.

Our second method **Belief Majority Voting** shares the same first two steps as the Belief Aggregation method - generating rationales and then extracting beliefs from those rationales, but instead performs majority voting to aggregate the beliefs.

1. **Generate**: Same as in Belief Aggregation
2. **Extract**: Same as in Belief Aggregation
3. **Majority Voting Aggregation**: For each extracted belief, we reprompt the LM by providing the extracted belief as additional context before prompting the model to provide the answer conditioned on the question and the given belief. Then the outputted answer for each belief are aggregated using a majority vote to produce the final answer.

We evaluate our method against chain-of-thought with greedy decoding and self-consistency with 10 sampled rationales with temperature sampling. We will re-implement these baselines and regenerate the corresponding accuracy numbers. We report results on the Instruct-GPT3 (openai/text-davinci-002) model [13], which is used for each step of prompting in both aggregation methods.

5 Experiments and Results

Dataset We will use several datasets including StrategyQA [4], OpenbookQA [14], and CommonsenseQA2 [15]. Each of these datasets have several thousand examples in their train sets, and their dev set sizes can range from 200 examples to several thousand examples. No additional pre-processing is done on the data. Given the token quota constraints for this project, we evaluated our method on 200 randomly sampled questions from the datasets, instead of evaluating on their entire dev sets.

Evaluation The chain-of-thought and self-consistency papers primarily report percentage accuracy of the final answer across the test set for each dataset as their metric of choice. So our primary metric of choice is also accuracy on the final outputted answer. We also report coverage on the StrategyQA dataset, where we empirically saw more cases of the model abstaining an answer (by a tied vote in self-consistency or by outputting *I don't know* or *Maybe*). Coverage is defined as the % of questions for which the model answers the question instead of abstaining.

5.1 Experimental Details

All experiments are done using the Instruct-GPT3 model, prompted using the CRFM API. All of the above steps have been implemented froms scratch by creating a prompting pipeline specifically for this project (using the CRFM API to prompt the models). The MAX-SAT solver was downloaded and used as a python package (https://pysathq.github.io/docs/html/api/examples/rc2.html). The prompts for the generation step were borrowed from the chain-of-thought paper for StrategyQA and from the Maieutic prompting paper for CSQA2, and written using randomly sampled examples for OpenbookQA. All other prompts were written using randomly sampled examples from the training subsets of the datasets. All results are on 200 randomly sampled questions from the training and dev subsets of each dataset (given token allocation constraints with CRFM).

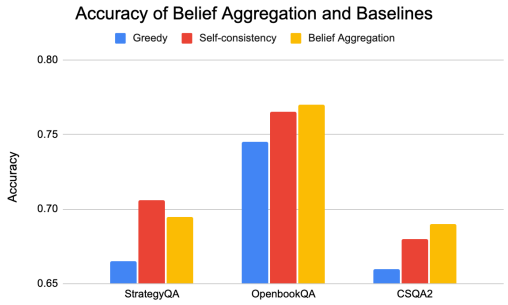
5.2 Results

Table 1 displays the results on a few variations of our Belief Aggregation method on the StrategyQA dataset. Figure 2a shows the results of Belief Aggregation compared to the baseline approach on the StrategyQA, OpenbookQA, and CSQA2 datasets. Overall Belief Aggregation slightly exceeds performance as compared to both the chain-of-thought and self-consistency baselines on OpenbookQA and CSQA2, while it slightly underperforms self-consistency on StrategyQA but still supercedes the greedy decoding baseline. The technique however does result in a 3-4% boost in coverage compared to self-consistency and greedy decoding.

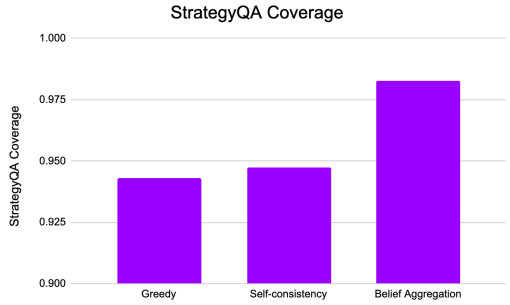
5.2.1 Analysis of Different Variants of Belief Aggregation

Technique	Accuracy
Greedy Chain-of-Thought	0.665
Self-Consistency Chain-of-Thought	**0.706**
Keypoints + Remove and Ignore Irrelevant Content	0.680
Distinct Facts + Remove and Ignore Irrelevant Content	0.680
Distinct Facts + Remove and Ignore Irrelevant Content + Ngram Deduplication	0.685
Distinct Facts + Remove and Ignore Irrelevant Content + Ngram Deduplication + Normalized Confidence Score	0.680
Distinct Facts + Remove and Ignore Irrelevant Content + Ngram Deduplication + Normalized Confidence Score + Top-5 Confidence Thresholding	**0.695**

Table 1: Accuracies of variants of Belief Aggregation on 200 examples from the StrategyQA dataset for the text-davinci-002 model, with 10 sampled rationales.



(a) Accuracy of Belief Aggregation against Baselines on StrategyQA, OpenbookQA, and CSQA2



(b) Coverage of Belief Aggregation against Baselines on StrategyQA

Figure 2: Accuracy and Coverage of Belief Aggregation compared to Greedy Decoding and Self-consistency

Technique	Accuracy
Greedy Chain-of-Thought	0.665
Self-Consistency Chain-of-Thought	**0.706**
Majority Vote Beliefs + Ignore Irrelevant + Ngram Deduplication	0.690
Majority Vote Beliefs + Ignore Irrelevant + Ngram Deduplication + Normalized Confidence Score + Top-5 Confidence Thresholding	**0.695**

Table 2: Accuracies of variants of Belief Majority Voting on 200 examples from the StrategyQA dataset for the text-davinci-002 model, with 10 sampled rationales.

Extracting Beliefs
We consider two separate prompts to extract beliefs from a given rationale. The first 'Keypoints' is an instruction + few-shot example designed to extract the most relevant, unique beliefs from a given rationale. The second 'Distinct Facts' has a similar format but an instruction and few-shot example designed to retain all unique information, unburdening the belief extraction stage from needing to identify the most relevant content. Empirically, we see better performance from the 'Distinct Facts' prompt since asking the model to identify the most important beliefs is often more likely to result in a loss of information causing us to lose important beliefs to reason with.

Removing Irrelevant Content
Prompting an LM while including irrelevant content in provided context often results in the LM being distracted by the irrelevant information. We saw the same problem occur in our technique, given that LMs can at times output irrelevant beliefs in their chains-of-thought. For example, given the following question from the StrategyQA dataset, *"Was Hillary Clinton's deputy chief of staff in 2009 baptised?"*, one belief outputted in a chain-of-thought rationale could be *"Huma Abedin, Hillary Clinton's deputy chief of staff in 2009, was born in 1976."*, even though the birth year of the deputy chief of staff is irrelevant to answering the question. We hypothesize that this may occur because LMs like humans may 'brainstorm' and surfacing such irrelevant information may occasionally indirectly trigger the formation of other more useful connections between concepts. To remove such irrelevant content from distracting the LM, we use a zero-shot instruction prompt to post-process the beliefs by directing the LM to consider both the question and the beliefs and retain only the beliefs useful in answering the question. Additionally, we modify the instruction when reprompting the model to ignore any content in the context. While this doesn't result in a performance improvement for the 'Keypoints' technique, it does improve performance for 'Distinct Facts', perhaps given that more beliefs are retained so there's a greater chance of more beliefs being irrelevant.

Confidence Scores
In our technique, we compute confidence scores for each belief to provide as weights to the MAX-SAT solver, in addition to the entailment-contradiction relationships between beliefs. The original confidence score of belief $b = P(b|$*"Q: question A: "*$)$, i.e. the probability of seeing the belief as an answer to the given question. We experiment with normalizing the confidence score computation to divide this score by $P(b|$*"A: "*$)$, to reduce the weights of beliefs that may be seen frequently, regardless of any given question. This normalization coupled with confidence score thresholding to select only the beliefs that the LM is most confident about results in a 1% performance gain.

5.2.2 Analyzing the Success of Belief Aggregation
Further analysis reveals where our method is better than the self-consistency approach of majority voting over final answers. There are several cases where the model's outputs are evenly distributed across the space of all possible answers. For example, in StrategyQA where the answer space consists of $True, False$, if half of the rationales result in an output of $True$ while half result in an output of $False$, this suggest an inherent lack of confidence in the model's outputted answer, similar to a model outputting *"I don't know"* or *"Maybe"* as the final answer. In such cases, where the outputs across all sampled rationales are fairly evenly distributed, our method of belief aggregation is able to better extract the relevant information often present in a minority of the rationales and compose or leverage this information to answer the overall question. Currently the best metric we have to assess this improvement is coverage, or the % of questions to which the model outputs an answer instead of abstaining (by saying *"I don't know"* or *"Maybe"* in the case of greedy decoding or belief aggregation or having a tie in majority voting with self-consistency). Our technique results in a 3-4% increase in coverage compared to both self-consistency and greedy decoding, allowing us respond to more questions.

5.2.3 Analyzing the Failure Cases of Belief Aggregation
While belief aggregation is a well-motivated technique, there are different pieces of the overall process that that could be more defined and more carefully explored in our future work to further improve our results. The first area of improvement is around belief extraction. Currently the beliefs produced by prompt-based belief extraction can run into the issues of being either too atomic or not atomic enough in their formulation. For example, given the question *'Is the Asian black bear multicolored?'*, a few extracted beliefs could be *'The Asian black bear is black with a white crescent on its chest.'* and *'Black and white are two colors.'* In this case, the second belief is not one that is

relevant to the question in a standalone way; it requires the first belief to provide additional context. Similarly given the question *'Do the directors of The Matrix advocate for transgender rights?'*, an extracted belief could be *'The Wachowskis, who are the directors of The Matrix, are both transgender women.'*, which may not be atomic enough, since it actually consists of two separate beliefs *'The Wachowskis are directors of The Matrix.'* and *'The Wachowskis are both transgender women.'*. One way of improving this could be more nuanced in-context examples that give more concrete examples of these two cases. We leave other considerations to future work.

Another potential concern with our current method is the lack of calibration in LLMs, which affects our confidence score computation. Specifically beliefs that LLMs may consider to be true may actually be factually incorrect, therefore when we threshold on the top-N beliefs we are most confident in, we may be filtering out beliefs that have a lower weight but are actually correct. Another related issue is LLMs tending to upweight beliefs that are more generic and therefore more likely to be true, over beliefs that are more nuanced and relevant to the question under consideration, but also more likely to possibly be false.

Furthermore, our method currently relies on entailment relationships between beliefs to find a set of factually consistent beliefs. However, it might be more reasonable to be able to cluster beliefs that all imply one particular answer and separately cluster beliefs that imply a different answer, by considering the beliefs as evidence we are accruing towards one answer or the other. For example, given the question *'The Wachowskis are directors of The Matrix.'*, the beliefs *'The directors of The Matrix have not made any public statements about transgender rights.'* and *'The Wachowskis, who are the directors of The Matrix, are both transgender women.'* are not directly contradictory (may have a neutral relationship with each other), but may be more likely to imply one particular answer or another, and therefore may be more likely to constitute evidence towards one particular answer or another.

5.2.4 Analyzing Belief Majority Voting

We motivate **Belief Majority Voting** by empirically observing that self-consistency often fails in cases when several sampled rationales are quite similar to one another, and only a minority of the rationales contain the information necessary to answer the question correctly. In such a case, majority voting ignores the critical minority opinion. Table 2 shows the results of two variants of belief majority voting compared to the greedy decoding and self-consistency baselines on the StrategyQA dataset. Overall, belief majority slightly underperforms self-consistency. Concretely analyzing outputs, we observe that this occurs for a few different reasons. One shortcoming of this method is that it relies on ngram-based deduplication of beliefs, however this often fails to filter all of the beliefs that are very similar to each other, since beliefs can be semantically similar without being syntactically similar. Another challenge this method faces is the issue of language model calibration, where again incorrect beliefs may be assigned lower confidence scores compared to factually correct ones. This poses a problem if we try to filter out less relevant beliefs by thresholding on the belief confidence scores, since we may end up retaining incorrect beliefs and losing correct ones. The final concern this method faces is that here beliefs are used in a standalone way as context to answer a question. However, often beliefs must be coupled with other beliefs in order to be understood in context of the question. In future work, we will investigate ways of modeling the dependencies between beliefs using models such as probabilistic graphical models.

6 Discussion and Conclusion

In this work we introduce two novel prompting techniques for more robust question answering by aggregating chain-of-thought rationales - **Belief Aggregation** and **Belief Majority Voting**. We postulate that chain-of-thought based rationales consist of a set of beliefs in addition to a basic inference over those beliefs. Since no single rationale can surface all possible beliefs required to answer a given question, we follow self-consistency in generating multiple rationales to surface a larger set of possible beliefs relevant to the question. We consider a principled means of aggregating the surfaced beliefs by computing entailment-contradiction relationships between each pair of beliefs, computing confidence scores for each belief, and leveraging a MAX-SAT solver to find the highest confidence set of consistent beliefs surfaced by the LLM and then reprompting the LLM with this consistent set of beliefs. We also implement and investigate an alternate strategy around reprompting the model with each belief as context and majority voting the outputs to arrive at our final answer. We evaluate our method on the Instruct-GPT3 model across three challenging question answering datasets, StrategyQA, OpenbookQA, and CSQA2, and in general see our approach result in modest gains over the greedy decoding and self-consistency baselines for chain-of-thought reasoning. Additionally,

we see substantive 3% gains on coverage for StrategyQA with Belief Aggregation, indicating that our approach allows us to confidently answer more questions in the dataset. We further analyze our outputs and reason about areas of improvement to our methods, which we leave as future work in the important area of rationale aggregation.

References

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, and et. al. Jared Kaplan. Language models are few-shot learners. In *Conference and Workshop on Neural Information Processing Systems (NeurIPS)*, 2020.

[2] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Conference and Workshop on Neural Information Processing Systems (NeurIPS)*, 2022.

[3] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[4] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. In *Transactions of the Association for Computational Linguistics (TACL)*, 2021.

[5] Ari Holtzman, Jan Buys, Li Du , Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020.

[6] ZIWEI JI, NAYEON LEE, RITA FRIESKE, and TIEZHENG YU et. al. Survey of hallucination in natural language generation. In *https://arxiv.org/pdf/2202.03629.pdf*, 2022.

[7] Simran Arora, Avanika Narayan , Mayee F. Chen , Laurel Orr , Neel Guha , and et. al. . Ask me anything: A simple strategy for prompting language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[8] Eric Mitchell, Joseph J. Noh, Siyan Li, and William S. Armstrong et. al. Enhancing self-consistency and performance of pretrained language models with nli. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

[9] Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. Maieutic prompting: Logically consistent reasoning with recursive explanations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

[10] Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Ronan Le Bras Peter West, Yejin Choi, and Hannaneh Hajishirzi. Generated knowledge prompting for commonsense reasoning. In *Association for Computational Linguistics (ACL)*, 2022.

[11] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators. In *International Conference on Learning Representations (ICLR)*, 2023.

[12] Ofir Press, Muru Zhang , Sewon Min , Ludwig Schmidt , Noah A. Smith , and Mike Lewis . Measuring and narrowing the compositionality gap in language models. 2022.

[13] Long Ouyang, Jeffrey Wu, and Xu Jiang et. al. Training language models to follow instructions with human feedback. In *Neurips 2022*.

[14] Tushar Khot Ashish Sabharwal Todor Mihaylov, Peter Clark. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

[15] Ronan Le Bras Chandra Bhagavatula Yoav Goldberg Yejin Choi Jonathan Berant Alon Talmor, Ori Yoran. Commonsenseqa 2.0: Exposing the limits of ai through gamification. In *NeurIPS*, 2021.