

Midterm Report: The Practicality of Prompt Engineering

Isaac Liu

University of California, Berkeley
ijyliu@berkeley.edu

Abstract

This paper examines the practicality of prompt engineering in improving the performance of Large Language Models (LLMs). Through empirical analysis, we evaluate the trade-offs between costs and benefits of prompting using novel metrics. Different prompting methods are assessed using standardized tasks and both state-of-the-art and older models.

Introduction

Prompt engineering, the practice of developing specialized prompts and queries to improve the accuracy of Large Language Models after training, is a prominent topic of interest in the NLP community, and among the general public. The practice is believed to allow for improvements in LLM performance a variety of domains without investment in underlying training (Martineau, 2021). It is not, however, without its critics. Some commentators believe that the practice will become irrelevant as models grow larger and more powerful, becoming more capable of directly interpreting a user's intent (Ethan Mollick [emollick], 2023). Others question the need for specialized professionals or training to attain minimal improvements which are often not repeatable across contexts (Shackell, 2023; Acar, 2023).

Despite such controversy, it is difficult to find empirical analyses of the tradeoff between costs and accuracy benefits associated with advanced prompting. Papers introducing new prompting techniques often only include performance benchmarks concerning the techniques' efficacy, typically within a limited domain. Some authors briefly mention problems associated with human-tailored prompts, such as the increased complexity induced by prompt-chaining and limitations on creativity and randomness (Wu et al., 2022), and others suggest the automation of prompting to avoid these costs (Diao et al., 2023). It is known that token

costs, degradation of quality with increased prompt context lengths, and the uncertain nature of accuracy gains are all important practical considerations (Gao, 2023). However, the extent of these issues for various techniques, so as to enable standardized comparison between them and with a control baseline (no special prompting), has not been (to my knowledge) quantified.

The quality, length, and complexity of LLM responses have been analyzed within several individual task and technique domains. Some research with GPT-3 series models on math and non-math reasoning tasks suggests the addition of length and complexity through the introduction of extra reasoning steps for both input prompts and output responses improves performance when using chain-of-thought prompting techniques (Fu et al., 2023). Effects are on the magnitude of several points of accuracy per added step, with generally low costs as long as prompt examples are selected carefully. Improvements from complex chain-of-thought prompting are not fully accepted in the literature, however - other work has noted a tendency for the method to lead to worse performance on simple questions (Shum et al., 2023).

Complexity has also been studied with the context of prompting for summarization and story generation tasks. The Chain of Density prompting technique seeks to optimize the named entity density of generated summaries through choice from a series of repeated, increasingly dense iterations (Adams et al., 2023). Human preferences tend to align with 0.1 - 0.15 named entities per token, a point near the middle of the usual sequence of generations, demonstrating the existence of a tradeoff between informativeness and clarity. At the same time, other work has shown is difficult to control language model output complexity, meaning that the choice of specific techniques is important. Research demonstrates current models are not yet able to achieve compliance with desired readabil-

ity instructions for the tasks of story generation, simplification, and summarization, though a small amount of improvement is achievable through careful prompt word choice and the use of few-shot examples (Pu and Demberg, 2023; Imperial and Madabushi, 2023).

This paper uses several metrics to evaluate the costs of prompt engineering methods systematically, and analyzes the tradeoffs inherent in their application to standardized data. Such an assessment is valuable on several dimensions. Beyond quantifiably testing the practicality of prompt engineering as a whole, it can be used to compare the performance of different approaches, useful in a world where so many competing techniques are available. I also provide a newly constructed dataset summarizing the wide variety of existing techniques and data on their popularity as measured by Semantic Scholar citations, which may be useful for future surveys of the field. Next, I offer a new look at these prompting methods in a period long after ideas were introduced. The current environment is one in which far greater capabilities are native to underlying foundation models. Finally, I introduce and adapt some useful measures of costs and complexity, such as the ratio of interaction length with prompting to the length of an accepted human-generated answer, to the challenge of LLM evaluation.

Prompting Methods Assessed

The following methods were selected based on their popularity (see Appendix Section A) and ease of implementation.

- Zero-Shot Control Baseline/Direct Prompting: This method consists of just providing the question/task directly.
- Zero-Shot Chain of Thought Prompting (Original): Initial advances in chain of thought prompting to improve reasoning were achieved by simply including the following before the question/task: "Let's think step by step." (Kojima et al., 2023)
- Zero-Shot Chain of Thought Prompting (Automatic Prompt Engineer): Automated testing has indicated that an optimal zero-shot Chain of Thought prompt is "Let's work this out in a step by step way to be sure we have the right answer." (Zhou et al., 2022)
- Self-Refine Prompting: The model produces an initial response, then is prompted for feedback which is used for refinement. (Madaan et al., 2023)
- Least-to-most Prompting: The model is given few-shot examples that demonstrate how to first break down the task into smaller and simpler subproblems, then solve them sequentially. (Zhou et al., 2023)
- Few-Shot Prompting: The prompter provides a few examples of successfully/answered questions or tasks before the main question/task.
- Few-Shot Chain-of-Thought Prompting: The model is provided worked examples of answers in which the reasoning steps are written out. (Wei et al.) Note, however, that such steps are not initially planned out as is the case in least-to-most prompting.

Data

To evaluate performance, I attempt to use tasks that are general-purpose, close to real-world applications, and standardized in the literature. To this end, I selected the GSM8K dataset, a collection of elementary-level math word problems (Cobbe et al., 2021), and a creative writing task involving the generation of a coherent two-paragraph passage with two random, predetermined ending sentences (Yao et al., 2023). The original creative writing task in Yao et al., 2023 uses four sentences, but this is too difficult for older models and for the production of good chain-of-thought/few-shot demonstrations - I simply take the first two sentences for each original question. These tasks carry several key benefits. They cover both the mathematical and linguistic domains - two types of tasks that form the foundation of the standardized testing of humans. GSM8K is studied in the majority of the papers introducing techniques I mentioned and is among the most common datasets used in the far larger list of papers I initially surveyed. Text and story generation is a widespread foundational task in NLP. Importantly, these sets of tasks are known to be free of data contamination. The GSM8K test set has been intentionally withheld from the training of OpenAI's models (OpenAI, 2023). The creative writing task was released only in 2023, and the code and data

provided with the associated paper includes only questions and not LLM responses.¹

I perform the analysis on one cutting-edge model and one older model, closer to the time that these techniques were introduced. This provides a picture of the changing costs and benefits of advanced prompting, a trend that may even be extrapolated into the future if current LLM scaling laws continue to hold. As the most widely used models and the ones behind much original work in the field, I select two models from the OpenAI series: GPT-4 and text-davinci-003 on OpenAI playground. These models are available both via the OpenAI API and in web interfaces - where possible I use web interfaces to limit resources required.

To the extent possible, I report accuracy scores on the domain dataset as they are in the original paper introducing each technique. It is possible to use any prompts, LLM responses, and correct responses provided along with original papers to calculate other simple metrics such as response lengths and complexity. However, some metrics (time taken, human assessments of complexity) require my own evaluations.

Metrics

Quality

Improved response quality can be a benefit of prompt engineering. I report:

For GSM8K problems:

- Correct/Incorrect accuracy at the point a technique has been fully implemented (the end of the chain of thought, etc., modelling the real world)

For Creative Writing:

- Human assessment of coherence (on a scale of 1 to 5, 1 being incoherent and 5 being very coherent)

I grade LLM responses manually and with assistance from GPT-4

¹It is also relatively simple to find random sentence generators online (such as <https://www.thewordfinder.com/random-sentence-generator/>), which would work for this task - at the cost of comparability with the results of (Yao et al., 2023). In any case, the 2-sentence task represents a modification of the original, and the test overall makes use of subjective human evaluations of the coherence of generated stories, limiting comparisons anyway.

Length

The length of responses and interactions can affect the practicality of prompting. It could indicate that a model is carefully and correctly solving through the steps of a problem (though it may actually be a confounder of other factors in such cases <https://browse.arxiv.org/pdf/2210.00720.pdf>). It can also impose time and financial costs to users, or become an indicator of degraded performance as models sometimes tend to go off on tangents or repeat themselves (to the extent some platforms have imposed length limitations) (Mann).

Prompt length is helpful up to 20 tokens, detrimental past 100 tokens. p.5 of <https://arxiv.org/pdf/2104.08691.pdf>

I report:

- Length of the entire interaction in tokens
- Financial cost of the entire interaction in tokens
- Length of the entire interaction in tokens relative to the length of the task/question + a human/solved out/generally accepted as correct answer (or relative to direct prompting). How much is prompt engineering stretching the interaction out? This ratio can be informative.
- The change in accuracy (in percentage points, 0 to 100) divided by the change in tokens (difference in token counts), between the prompt engineering technique and direct prompting. Is any stretching of output adding value/improving accuracy?

$$\frac{Accuracy_{PE} - Accuracy_B}{Tokens_{PE} - Tokens_B}$$

- Length of the entire interaction in time (seconds). This can include time writing a response, waiting for a response, or reviewing a response. More granular data on these each of the component steps may be hard to collect, but it might be possible to look at human assessments of time spent on these activities. Attempts will be made to have queries to models made at a consistent time during off-peak hours to minimize confounding due to server load, connectivity issues, etc.

Complexity

Similarly to length, complexity could be an indicator of high accuracy. In the case of summarization tasks, increased output complexity achieved by requests for summaries at an "expert" level can improve precision and recall in the face of concerns about named entity hallucination and detection <https://aclanthology.org/2023.acl-srw.1.pdf>. However, complexity has substantial costs in potentially making review of LLM output more difficult, and on simple questions it may even lead to degraded accuracy <https://browse.arxiv.org/pdf/2302.12822.pdf>.

Complicated prompts may hurt robustness to irrelevant context: <https://proceedings.mlr.press/v202/shi23a/shi23a.pdf>

Chaining prompts creates complexity but also enables fluency and transparency - qualitative statements <https://arxiv.org/pdf/2110.01691.pdf>. In a study of prompt chaining, participants qualitatively reported...

I report:

- Vocabulary - share of words on the Academic Vocabulary List (AVL) for natural language and non-code components of responses. (Gardner and Davies, 2014) Share of novel n-grams in the response (words not in the prompt), presence of contrasting words 'while', 'but', 'though', 'although', 'other', 'others', 'however'. <https://aclanthology.org/2023.findings-acl.591.pdf>
- Number of named entities <https://browse.arxiv.org/pdf/2309.04269.pdf>
- Number of reasoning steps - line-breaks, periods, "step i" strings, and semicolons serve as separators. <https://browse.arxiv.org/pdf/2210.00720.pdf>
- Sentence length and Flesch reading ease (implemented via the textstat Python package) for natural language and non-code components of responses. (Flesch, 2016; Aggarwal)
- Cyclomatic complexity for code responses (implemented via the radon Python package). (Lacchia)
- Ratio or difference of these scores in prompts vs. responses, responses vs. accepted/outside correct answer

- Human assessment of need for specialized knowledge/difficulty of implementation of the technique. This could be task specific (done for some novel real world example questions/prompting scenarios), or it could be done overall based on a pre-existing description of the technique. Perhaps a balance of both is best. The actual metric will be a numeric score and a qualitative description.
- Human assessment of output complexity (ease of evaluating results). This could be task specific (done for some novel real world example questions/prompting scenarios), or it could be done overall based on pre-existing examples of the technique. Perhaps a balance of both is best. The actual metric will be a numeric score and a qualitative description.
- Human assessment of amount of irrelevant text generated

Analyses

I provide summary statistics of the metrics for each prompting method by model by question/task type. In cases where human/textual assessment and comments have been provided, it might be interesting to use NLP methods to evaluate responses (ex: for sentiment).

The results can provide insight into several questions:

Finish filling this out with points from the introduction.

Do larger/more modern models benefit more from prompt engineering, or are the techniques becoming obsolete? Earlier evidence demonstrated that gains from few-shot learning increased with scale - has this trend continued to hold? (Brown et al., 2020) Are the most recent prompt engineering techniques more powerful and useful?

Does increased complexity limit creativity and randomness (investigating the creative writing task in particular might be helpful)? Does it lead to worse performance on simple questions?

Limitations

It was difficult to select prompt engineering methods to try for this paper, and there is potential for my choice of methods to be somewhat biased. I mostly picked methods based my perception of their popularity and ease of implementation. If anything, this may lead to an underestimation of costs

if easy-to-implement and high quality methods are likely to be popular.

Just as my evaluation comes at a time with significantly more capable LLMs relative to those available when much work began on prompting, I expect the underlying calculus concerning prompting to continue to change in the future. However, I again only expect relative costs of complex engineering to increase as models get better.

Another potential problem is the extent that prompting techniques have been absorbed into default LLM behavior, likely through reinforcement learning. GPT-4 in particular does seem to automatically implement chain-of-thought methods when presented with a sufficiently complex problem. In this environment, this paper become less of an evaluation of prompting techniques themselves, but more of an evaluation of their intentional and manual implementation.

Finally, though I have taken steps to limit it, data contamination remains a real concern. The questions/tasks I use are unlikely to have been used in pretraining, but they may have been introduced to LLMs through reinforcement learning and other evaluations. On the other hand, this seems unlikely to bias the results for any one particular prompting method relative to the others or versus the control/direct prompting - comparisons internal to this paper are still likely to be useful.

Acknowledgements

The template for this document was adapted by Jordan Boyd-Graber, Naoaki Okazaki, and Anna Rogers.

References

[Semantic Scholar | AI-Powered Research Tool.](#)

Oguz A. Acar. 2023. [AI Prompt Engineering Isn't the Future](#). *Harvard Business Review*. Section: Technology and analytics.

Griffin Adams, Alexander Fabbri, Faisal Ladhak, Eric Lehman, and Noémie Elhadad. 2023. [From Sparse to Dense: GPT-4 Summarization with Chain of Density Prompting](#). ArXiv:2309.04269 [cs].

Shivam Bansal Aggarwal, Chaitanya. [textstat: Calculate statistical features from text](#).

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 [cs].

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training Verifiers to Solve Math Word Problems](#). ArXiv:2110.14168 [cs].

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. [Active Prompting with Chain-of-Thought for Large Language Models](#). ArXiv:2302.12246 [cs].

Ethan Mollick [@emollick]. 2023. [I have a strong suspicion that “prompt engineering” is not going to be a big deal in the long-term & prompt engineer is not the job of the future AI gets easier. You can already see in Midjourney how basic prompts went from complex in v3 to easy in v4. Same with ChatGPT to Bing.](#) <https://t.co/BTtSN4oVF4>.

Rudolf Flesch. 2016. [How to Write Plain English](#).

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. [Complexity-Based Prompting for Multi-Step Reasoning](#). ArXiv:2210.00720 [cs].

Andrew Gao. 2023. [Prompt Engineering for Large Language Models](#).

Dee Gardner and Mark Davies. 2014. [A New Academic Vocabulary List](#). *Applied Linguistics*, 35(3):305–327.

Joseph Marvin Imperial and Harish Tayyar Madabushi. 2023. [Flesch or Fumble? Evaluating Readability Standard Alignment of Instruction-Tuned Language Models](#). ArXiv:2309.05454 [cs].

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. [Large Language Models are Zero-Shot Reasoners](#). ArXiv:2205.11916 [cs].

Michele Lacchia. [radon: Code Metrics in Python](#).

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-Refine: Iterative Refinement with Self-Feedback](#). ArXiv:2303.17651 [cs].

Jyoti Mann. [Microsoft limits Bing chat exchanges and conversation lengths after 'creepy' interactions with some users](#).

Kim Martineau. 2021. [What is prompt tuning?](#)

OpenAI. 2023. [GPT-4 Technical Report](#). ArXiv:2303.08774 [cs].

Dongqi Pu and Vera Demberg. 2023. [ChatGPT vs Human-authored Text: Insights into Controllable Text Summarization and Sentence Style Transfer](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 1–18, Toronto, Canada. Association for Computational Linguistics.

Cameron Shackell. 2023. [Prompt engineering: is being an AI ‘whisperer’ the job of the future or a short-lived fad?](#)

KaShun Shum, Shizhe Diao, and Tong Zhang. 2023. [Automatic Prompt Augmentation and Selection with Chain-of-Thought from Labeled Data](#). ArXiv:2302.12822 [cs].

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](#).

Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. [AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts](#). In *CHI Conference on Human Factors in Computing Systems*, pages 1–22, New Orleans LA USA. ACM.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. [Tree of Thoughts: Deliberate Problem Solving with Large Language Models](#). ArXiv:2305.10601 [cs].

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. [Least-to-Most Prompting Enables Complex Reasoning in Large Language Models](#). ArXiv:2205.10625 [cs].

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. [Large Language Models are Human-Level Prompt Engineers](#).

A The Popularity of Some Prompting Methods

Table 1 displays the popularity (in terms of Semantic Scholar citations ([noa](#))) of some of the most popular generalizable prompt engineering methods based on the lists at <https://www.promptingguide.ai/papers#approaches> and https://en.wikipedia.org/wiki/Prompt_engineering#Text-to-text as of October 22, 2023. Please contact the author for a full list of the 162 papers considered.

Another good resource for prompt engineering methods and evaluations is the paperswithcode website: <https://paperswithcode.com/task/prompt-engineering>. I did not make use of this page, however, as it seemed to be missing many prominent approaches, contains text-to-vision methods, and focuses on GitHub implementations (which are often low integer numbers difficult to compare) and currently trending social media items.

Table 1: Popularity of Selected Prompt Engineering Methods

Paper Title	Prompt Engineering Method	Citations Per Day Since Release
Language Models are Few-Shot Learners	Few-Shot Learning	13.23
Chain of Thought Prompting Elicits Reasoning in Large Language Models	Chain-of-Thought Prompting	3.33
Large Language Models are Zero-Shot Reasoners	Zero-Shot Chain-of-Thought	1.71
Tree of Thoughts: Deliberate Problem Solving with Large Language Models	Tree-of-Thought	1.43
Self-Refine: Iterative Refinement with Self-Feedback	Self-Refine	0.97
ReAct: Synergizing Reasoning and Acting in Language Models	ReAct	0.87
Least-to-most prompting enables complex reasoning in large language models	Least-to-Most Prompting	0.71
PAL: Program-aided Language Models	Program Aided Language Models	0.58
Large Language Models Are Human-Level Prompt Engineers	Automatic Prompt Engineer	0.55
How Can We Know What Language Models Know?	Prompt Mining, Prompt Paraphrasing	0.54
Automatic Chain of Thought Prompting in Large Language Models	Automatic Chain of Thought Prompting	0.53
Show Your Work: Scratchpads for Intermediate Computation with Language Models	Scratchpads	0.46
Multimodal Chain-of-Thought Reasoning in Language Models	Multimodal CoT	0.35
Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm	Metaprompt	0.32
CAMEL: Communicative Agents for "Mind" Exploration of Large Scale Language Model Society	Role-Playing	0.31
Chain-of-Verification Reduces Hallucination in Large Language Models	Chain-of-Verification	0.31
Complexity-Based Prompting for Multi-Step Reasoning	Complexity-Based Prompting	0.3
Decomposed Prompting: A Modular Approach for Solving Complex Tasks	Decomposed Prompting	0.27
Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models	Plan-and-Solve Prompting	0.27
Large Language Models Can Be Easily Distracted by Irrelevant Context	Instruction to Ignore Irrelevant Information	0.26
Connecting Large Language Models with Evolutionary Algorithms Yields Powerful Prompt Optimizers	EvoPrompt	0.21
AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts	Chaining	0.2
Prompting GPT-3 To Be Reliable	Prompting for Reliability	0.19
Demonstrate-Search-Predict: Composing retrieval and language models for knowledge-intensive NLP	Demonstrate-Search-Predict	0.19
ART: Automatic multi-step reasoning and tool-use for large language models	Automatic Reasoning and Tool-Use	0.18
Promptagator: Few-Shot Dense Retrieval From 8 Examples	Few-Shot Dense Retrieval	0.17
Maieutic Prompting: Logically Consistent Reasoning with Recursive Explanations	Maieutic Prompting	0.17
Reframing Instructional Prompts to GPTk's Language	Reframing	0.16
Generated Knowledge Prompting for Commonsense Reasoning	Generated Knowledge Prompting	0.15
Teaching Algorithmic Reasoning via In-context Learning	Algorithmic Prompting	0.15
Hard Prompts Made Easy: Gradient-Based Discrete Optimization for Prompt Tuning and Discovery	Gradient-Based Prompt Optimization	0.15

B Prompts Used

Below I have listed question and prompt examples for each method.

B.1 GSM8K

The sample problem is the first one in the GSM8K test dataset. All prompt examples for non-zero-shot methods are drawn from the training dataset.

<Question> "Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?"

Zero-Shot Control Baseline/Direct Prompting:
<Question>

Zero-Shot Chain-of-Thought: <Question> Let's think step by step.

APE Improved Zero-Shot Chain-of-Thought:
<Question> Let's work this out in a step by step way to be sure we have the right answer.

Tree-of-Thought: <https://github.com/princeton-nlp/tree-of-thought-llm/tree/master> <https://github.com/kyegomez/tree-of-thoughts>

One reasoning step:

Problem: <Question> Current plan of reasoning: <Current Plan> Task: Generate k different possible one-sentence thoughts to serve as step X in solving the problem. Only work on step X. Put each thought on a new line. Do not number them. Response:

Problem: <Question> Current plan of reasoning: <Current Plan> Potential next thoughts: <Potential Next Thoughts> Task: State the thought among "Potential next thoughts" that is most likely to contribute to solving the problem. If one of the thoughts fully solves the problem correctly, instead state the solution to the problem and output the word STOP on a new line. State only a thought or the solution and the word STOP. Response:

NOTE: This is very hard for older models to follow! davinci-002 can't really do any form of ToT tried

Baked in calculations:

Problem: <Question> Current plan of reasoning: <Current Plan> Task: Generate k different possible one-step calculations to serve as step X in solving the problem. Only work on step X. Put each calculation on a new line. Do not number them. Response:

Still hard to follow...

Zero-Shot Tree-of-Thought: Imagine three different experts are answering this question. All experts will write down 1 step of their thinking, then share it with the group. Then all experts will go on to the next step, etc. If any expert realises they're wrong at any point then they leave. The question is... <Question>

Self-Refine: <https://selfrefine.info/> See python code

Least-to-most Prompting (1-shot):
<https://arxiv.org/pdf/2205.10625.pdf> Q: Elsa has 5 apples. Anna has 2 more apples than Elsa. How many apples do they have together? A: Let's break down this problem: 1. How many apples does Anna have? 2. How many apples do Elsa and Anna have together? 1. Anna has 2 more apples than Elsa. So Anna has $2 + 5 = 7$ apples. 2. Elsa and Anna have $5 + 7 = 12$ apples together.

Q: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for \$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market? A: Let's break down this problem:

Append to this: "The answer is:" in a second pass.

Manual Few-Shot:

Randomly draw examples from the training set, as per <https://arxiv.org/pdf/2005.14165.pdf>

Manual Chain-of-Thought:

Page 35 of <https://arxiv.org/pdf/2201.11903.pdf>

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today? A: There are 15 trees originally. Then there were 21 trees after some more were planted. So there must have been $21 - 15 = 6$. The answer is 6. Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot? A: There are originally 3 cars. 2 more cars arrive. $3 + 2 = 5$. The answer is 5. Q: Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total? A: Originally, Leah had 32 chocolates. Her sister had 42. So in total they had $32 + 42 = 74$. After eating 35, they had $74 - 35 = 39$. The answer is 39. Q: Jason had 20 lollipops. He gave Denny some lollipops. Now Jason has 12 lollipops. How many lollipops did Jason give to Denny? A:

Jason started with 20 lollipops. Then he had 12 after giving some to Denny. So he gave Denny $20 - 12 = 8$. The answer is 8. Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now? A: Shawn started with 5 toys. If he got 2 toys each from his mom and dad, then that is 4 more toys. $5 + 4 = 9$. The answer is 9. Q: There were nine computers in the server room. Five more computers were installed each day, from monday to thursday. How many computers are now in the server room? A: There were originally 9 computers. For each of 4 days, 5 more computers were added. So $5 * 4 = 20$ computers were added. $9 + 20$ is 29. The answer is 29. Q: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday? A: Michael started with 58 golf balls. After losing 23 on tuesday, he had $58 - 23 = 35$. After losing 2 more, he had $35 - 2 = 33$ golf balls. The answer is 33. Q: Olivia has \$23. She bought five bagels for \$3 each. How much money does she have left? A: Olivia had 23 dollars. 5 bagels for 3 dollars each will be $5 * 3 = 15$ dollars. So she has $23 - 15$ dollars left. $23 - 15$ is 8. The answer is 8. Q: <Question> A:

B.2 Last Letter

Sources: kojima for data but also original chain of thought paper

There is no training set, only a small provided set of examples.

<Question>: Take the last letters of each words in Whitney Erika Tj Benito and concatenate them. (sic)

Zero-Shot Control Baseline/Direct Prompting: <Question>

Zero-Shot Chain-of-Thought: <Question> Let's think step by step.

APE Improved Zero-Shot Chain-of-Thought: <Question> Let's work this out in a step by step way to be sure we have the right answer.

Tree-of-Thought: <https://github.com/princeton-nlp/tree-of-thought-llm/tree/master>
<https://github.com/kyegomez/tree-of-thoughts>

One reasoning step:

Problem: <Question> Current plan of reasoning: <Current Plan> Task: Generate k different possible one-sentence thoughts to serve as step X in solving the problem. Only work on step X. Put each thought on a new line. Do not number them.

Response:

Problem: <Question> Current plan of reasoning: <Current Plan> Potential next thoughts: <Potential Next Thoughts> Task: State the thought among "Potential next thoughts" that is most likely to contribute to solving the problem. If one of the thoughts fully solves the problem correctly, instead state the solution to the problem and output the word STOP on a new line. State only a thought or the solution and the word STOP. Response:

NOTE: This is very hard for older models to follow! davinci-002 can't really do any form of ToT tried. davinci-003 struggles.

Zero-Shot Tree-of-Thought: Imagine three different experts are answering this question. All experts will write down 1 step of their thinking, then share it with the group. Then all experts will go on to the next step, etc. If any expert realises they're wrong at any point then they leave. The question is...

Self-Refine: <https://selfrefine.info/> See python code

Least-to-most Prompting (4-shot): <https://arxiv.org/pdf/2205.10625.pdf>

Q: "think, machine" A: The last letter of "think" is "k". The last letter of "machine" is "e". Concatenating "k", "e" leads to "ke". So, "think, machine" outputs "ke". Q: "think, machine, learning" A: "think, machine" outputs "ke". The last letter of "learning" is "g". Concatenating "ke", "g" leads to "keg". So, "think, machine, learning" outputs "keg". Q: "transformer, language" A: The last letter of "transformer" is "r". The last letter of "language" is "e". Concatenating: "r", "e" leads to "re". So, "transformer, language" outputs "re". Q: "transformer, language, vision" A: "transformer, language" outputs "re". The last letter of "vision" is "n". Concatenating: "re", "n" leads to "ren". So, "transformer, language, vision" outputs "ren". Q: <Question> A:

Manual Few-Shot:

There is no training set, only a small provided set of examples.

Q: Take the last letters of the words in "Elon Musk" and concatenate them. A: The answer is nk. Q: Take the last letters of the words in "Larry Page" and concatenate them. A: The answer is ye. Q: Take the last letters of the words in "Sergey Brin" and concatenate them. A: The answer is yn. Q: Take the last letters of the words in "Bill Gates" and concatenate them. A: The answer is ls. Q:

<Question> A:

Manual Chain-of-Thought:

Page 36 of <https://arxiv.org/pdf/2201.11903.pdf>

Q: Take the last letters of the words in "Elon Musk" and concatenate them. A: The last letter of "Elon" is "n". The last letter of "Musk" is "k". Concatenating them is "nk". The answer is nk. Q: Take the last letters of the words in "Larry Page" and concatenate them. A: The last letter of "Larry" is "y". The last letter of "Page" is "e". Concatenating them is "ye". The answer is ye. Q: Take the last letters of the words in "Sergey Brin" and concatenate them. A: The last letter of "Sergey" is "y". The last letter of "Brin" is "n". Concatenating them is "yn". The answer is yn. Q: Take the last letters of the words in "Bill Gates" and concatenate them. A: The last letter of "Bill" is "l". The last letter of "Gates" is "s". Concatenating them is "ls". The answer is ls. Q: <Question> A:

B.3 Creative Writing

Source: <https://arxiv.org/pdf/2305.10601.pdf>

No training data was provided, so I used random sentences courtesy of <https://www.thewordfinder.com/random-sentence-generator/>

<Question>: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. It isn't difficult to do a handstand if you just stand on your hands. 2. It caught him off guard that space smelled of seared steak.

Zero-Shot Control Baseline/Direct Prompting:
<Question>

Zero-Shot Chain-of-Thought: <Question> Let's think step by step.

APE Improved Zero-Shot Chain-of-Thought:
<Question> Let's work this out in a step by step way to be sure we have a correct and coherent answer.

Tree-of-Thought: <https://github.com/princeton-nlp/tree-of-thought-llm/tree/master>
<https://github.com/kyegomez/tree-of-thoughts>

NEEDS HELP

<Question> Generate 3 plans for potential passages. Do not write the passage.

Select the most coherent outline that follows the rules of the task. Do not write the passage.

Write three drafts of the passage based on this outline.

Select the most coherent draft that follows the rules of the task and write it out.

Self-Refine: <https://selfrefine.info/> See python code

Least-to-most Prompting (1-shot):
<https://arxiv.org/pdf/2205.10625.pdf>

Q: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. She couldn't get a job because she didn't have a permanent address. 2. He didn't have any hidden talents. A: Let's break down this problem into steps: First, we will write a few ideas for the passage. Second, draft the passage. Ideas: 1. Make the passage about travelling circus performers looking for other work as their circus shuts down. 2. Make the passage about the struggles of a homeless person who is trying to get a job. 3. Make the passage about perceptions and preconceptions of people's skills and social status as factors in hiring. Passage: Laura sat on the park bench, watching the people walk by. She was homeless, and had been for a few months now. She couldn't get a job because she didn't have a permanent address. She had tried to talk to career counselors about her situation, but the conversations often seemed fruitless. She didn't feel she had any marketable skills. Her situation was similar to that of her friend, Rodrigo, who openly shared a similar attitude with counselors in his meetings. He didn't have any hidden talents. Q: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. He had always wanted to be a Youtuber but never thought it would actually happen. 2. My sweater got caught on the door hinge. A: Let's break down this problem into steps: First, we will write a few ideas for the passage. Second, draft the passage. Ideas: 1. Make the passage about a sister visiting her brother; the brother has recently become a successful Youtuber - she excitedly gets her sweater caught leaving a meeting with him. 2. Make the passage about a men's fashion reviewer who is working on a video review of a sweater. 3. Make the passage about a Youtuber preparing for a video shoot - as they hurry through things, their sweater gets caught but this becomes an amusing part of their vlog. Passage: My brother, John, had been making home videos for years, but they never got much attention. He was always disappointed when he saw other people's videos getting thousands of views. Then one day, he got a call from a company that wanted to

sponsor him. They offered him a lot of money to make videos for them. He was so excited that he couldn't sleep that night. He had always wanted to be a Youtuber but never thought it would actually happen. As it turned out, John would need his own production staff to help with script writing and video editing. As I lived in the area and had prior experience in these fields, I was a natural choice for a part-time role on his channel. The company's sponsorship was very generous, and I would get a large portion of the profits. I was glad to finally be able to earn a substantial income in a more exciting and engaging role than my current position as a barista. I was smiling for most of our first business meeting, and strutted with pride out of our new studio. My sweater got caught on the door hinge. Q: <Question> A:

Manual Few-Shot:

Q: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. She couldn't get a job because she didn't have a permanent address. 2. He didn't have any hidden talents. A: Laura sat on the park bench, watching the people walk by. She was homeless, and had been for a few months now. She couldn't get a job because she didn't have a permanent address. She had tried to talk to career counselors about her situation, but the conversations often seemed fruitless. She didn't feel she had any marketable skills. Her situation was similar to that of her friend, Rodrigo, who openly shared a similar attitude with counselors in his meetings. He didn't have any hidden talents. Q: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. He had always wanted to be a Youtuber but never thought it would actually happen. 2. My sweater got caught on the door hinge. A: My brother, John, had been making home videos for years, but they never got much attention. He was always disappointed when he saw other people's videos getting thousands of views. Then one day, he got a call from a company that wanted to sponsor him. They offered him a lot of money to make videos for them. He was so excited that he couldn't sleep that night. He had always wanted to be a Youtuber but never thought it would actually happen. As it turned out, John would need his own production staff to help with script writing and video editing. As I lived in the area and had prior experience in these fields, I was a natural choice for a part-time role on his channel.

The company's sponsorship was very generous, and I would get a large portion of the profits. I was glad to finally be able to earn a substantial income in a more exciting and engaging role than my current position as a barista. I was smiling for most of our first business meeting, and strutted with pride out of our new studio. My sweater got caught on the door hinge. Q: <Question> A:

Manual Chain-of-Thought:

Page 36 of <https://arxiv.org/pdf/2201.11903.pdf>

Q: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. She couldn't get a job because she didn't have a permanent address. 2. He didn't have any hidden talents. A: Ideas: 1. Make the passage about travelling circus performers looking for other work as their circus shuts down. 2. Make the passage about the struggles of a homeless person who is trying to get a job. 3. Make the passage about perceptions and preconceptions of people's skills and social status as factors in hiring. Passage: Laura sat on the park bench, watching the people walk by. She was homeless, and had been for a few months now. She couldn't get a job because she didn't have a permanent address. She had tried to talk to career counselors about her situation, but the conversations often seemed fruitless. She didn't feel she had any marketable skills. Her situation was similar to that of her friend, Rodrigo, who openly shared a similar attitude with counselors in his meetings. He didn't have any hidden talents. Q: Write a coherent passage of 2 short paragraphs. The end sentence of each paragraph, respectively, must be: 1. He had always wanted to be a Youtuber but never thought it would actually happen. 2. My sweater got caught on the door hinge. A: Ideas: 1. Make the passage about a sister visiting her brother; the brother has recently become a successful Youtuber - she excitedly gets her sweater caught leaving a meeting with him. 2. Make the passage about a men's fashion reviewer who is working on a video review of a sweater. 3. Make the passage about a Youtuber preparing for a video shoot - as they hurry through things, their sweater gets caught but this becomes an amusing part of their vlog. Passage: My brother, John, had been making home videos for years, but they never got much attention. He was always disappointed when he saw other people's videos getting thousands of views. Then one day, he got a call from a company that wanted to sponsor him. They offered him a lot of money to

make videos for them. He was so excited that he couldn't sleep that night. He had always wanted to be a Youtuber but never thought it would actually happen. As it turned out, John would need his own production staff to help with script writing and video editing. As I lived in the area and had prior experience in these fields, I was a natural choice for a part-time role on his channel. The company's sponsorship was very generous, and I would get a large portion of the profits. I was glad to finally be able to earn a substantial income in a more exciting and engaging role than my current position as a barista. I was smiling for most of our first business meeting, and strutted with pride out of our new studio. My sweater got caught on the door hinge.

Q: <Question>