The Benefits of a Concise Chain of Thought on Problem Solving in Large Language Models

Matthew Renze
Johns Hopkins University
mrenze1@jhu.edu

Abstract

In this paper, we introduce Concise Chain-of-Thought (CCoT) prompting. We compared standard CoT and CCoT prompts to see how conciseness impacts response length and correct-answer accuracy. We evaluated this using GPT-3.5 and GPT-4 with a multiple-choice question-and-answer (MCQA) benchmark.

CCoT reduced average response length by 48.70% for both GPT-3.5 and GPT-4 while having a negligible impact on problem-solving performance. However, on math problems, GPT-3.5 with CCoT incurs a performance penalty of 27.69%. Overall, CCoT leads to an average per-token cost reduction of 22.67%.

These results have practical implications for AI systems engineers using LLMs to solve real-world problems with CoT prompt-engineering techniques. In addition, these results provide more general insight for AI Researchers studying the emergent behavior of step-by-step reasoning in LLMs.

All code, data, and supplemental materials are available on GitHub[1].

Keywords

Large Language Model (LLM), Generative Pre-trained Transformer (GPT), Chain-of-Thought (CoT)

1. Introduction

1.1 Background

In recent years, Large Language Models (LLMs) have transformed the field of artificial intelligence by offering unprecedented new capabilities for AI systems. As a result, LLMs have become a standard component in many AI systems that automate solutions to real-world problems.

However, to create effective LLM solutions, prompt engineering is often necessary. As a result, AI systems engineers have developed various techniques to improve the performance of LLMs for specific use cases and problem domains. These techniques include Chain of Thought (CoT) prompting [1], [2].

[1] https://github.com/matthewrenze/jhu-concise-cot

1.2 Chain-of-Thought (CoT) Prompting

CoT is a prompt engineering technique that instructs the LLM to reason through a problem in a step-by-step manner. Reasoning step-by-step increases the likelihood of the LLM producing a correct solution to the problem. As a result, CoT improves LLM performance on many problem-solving tasks [3]–[5].

There are multiple versions of CoT prompting with various pros and cons. Zero-shot CoT instructs the LLM to "think step-by-step" through the problem in the system prompt [4], [5]. Few-shot CoT provides a series of examples as problem-solution pairs with the CoT explicitly stated in each example solution [3].

CoT prompting has been shown to improve LLM performance by up to 80% for certain problem tasks and problem domains [4]. However, this performance increase comes at the expense of increased LLM response length. As a result, the cost of using the LLM with CoT grows in proportion to response length.

1.3 Concise Prompting

Concise prompting is a prompt-engineering technique used to reduce LLM response verbosity. This technique helps reduce the per-token cost of using the LLM. In addition, it can reduce the LLM's energy consumption, minimize response wait time, and improve communication efficiency with the end user.

There are two main implementations of concise prompting. Zero-shot prompting instructs the LLM to "be concise" in its response [6], [7]. Few-shot prompting requires the prompt engineer to create a series of problem-solution example pairs with concisely written text in each example solution.

While concise prompting is beneficial for reducing resource costs, it may negatively impact the performance of the LLM on some problem-solving tasks [7]. This is because the LLM requires additional verbosity to fully elaborate the steps in its thought process to produce a correct solution.

As a result, prompt engineers often provide LLMs with instructions and examples designed to ensure higher verbosity so that all steps in the LLM's thought process are explicitly stated. However, the adoption of verbose CoT conventions appears to be based on anecdotal rather than empirical evidence.

1.4. Hypotheses

It is still an open question how conciseness impacts response length and problem-solving capabilities for an LLM with CoT. To answer this question, we combined concise prompting and CoT to create Concise Chain-of-Thought (CCoT) prompting. We used CCoT to answer this question in two parts:

To test the impact of CCoT on an LLM's response length, we developed the following hypotheses:

- **Response-Length Null Hypothesis (RL-H_0):** CCoT has no effect on the number of response tokens produced by the LLM compared to standard CoT prompting.
- **Response-Length Alternative Hypothesis (RL-H_1):** CCoT decreases the number of response tokens produced by the LLM compared to standard CoT prompting.

To test the impact of CCoT on an LLM's problem-solving performance, we developed the following hypotheses:

- **Performance Null Hypothesis (P-H_0):** CCoT has no effect on the correct-answer accuracy of the LLM compared to standard CoT prompting.
- **Performance Alternative Hypothesis (P-H_1):** CCoT decreases the correct-answer performance of the LLM compared to standard CoT prompting.

To determine the validity of these hypotheses, we performed a hypothesis test using a significance level (α) of 0.05.

1.5 Significance

These results have practical implications for AI systems engineers using LLMs with CoT in their solutions.

If CCoT reduces response length (i.e., if RL-H_1 is true), then AI systems engineers can use CCoT to reduce LLM costs. Third-party LLM APIs are typically priced per token [8], [9]. So, reducing response length will reduce total costs. Reducing response length also reduces energy consumption and response wait times.

In addition, if CCoT does not decrease performance (i.e., if P-H_0 is true), then there is no performance penalty for implementing CCoT. As a result, AI systems engineers should prefer CCoT over standard CoT.

These results also have theoretical implications for AI researchers studying CoT reasoning in LLMs.

If we can reduce the length of a CoT without impacting performance, then only some aspects of a CoT are relevant to the LLM's problem-solving performance. This discovery raises new questions about which specific tokens or aspects of an LLM's CoT are necessary vs. which are superfluous.

1.6 Prior Literature

Few-shot CoT prompting was introduced by Wei et al. in Jan 2022 [3]. Zero-shot CoT was introduced four months later, in May 2022, by Kojima et al. [4]. The zero-shot CoT method was then further refined by Zhou et al. using the Automatic Prompt Engineer (APE) method in Oct 2022 [5].

In Sept 2022, Madaan and Yazdanbakhsh attempted to separate symbols, patterns, and text to determine their individual effects on CoT and develop their own concise CoT prompt technique [10]. However, as of Jan 2023, their paper was withdrawn from publication due to technical issues [11].

Concise prompting, in general, has not been studied to the same extent as CoT prompting. It has been used in scientific research papers for practical reasons but has not been studied directly. Most guidance on concise prompting comes from best practices in the prompt-engineering community.

After an extensive literature search, we found no research papers specifically studying concise prompting. In addition, we could not find any other publications exploring concise CoT prompting.

2. Methods

2.1 Data

Our test dataset consists of Multiple-Choice Question-and-Answer (MCQA) problems from standard LLM benchmarks.

We reviewed existing literature to identify a set of candidate benchmarks that spanned multiple problem domains and difficulty levels. Then, we pre-processed the data into a standard data format and randomly selected 100 questions from each of the ten benchmarks to create an exam with 1,000 MCQA problems.

Source Problem Sets				
Problem Set	**Benchmark**	**Domain**	**Questions**	**Source**
ARC Challenge Test	ARC	Science	1,173	[12]
AQUA-RAT	AGI Eval	Math	254	[13]
Hellaswag Val	Hellaswag	Common Sense Reasoning	10,042	[14]
LogiQA (English)	AGI Eval	Logic	651	[13], [15]
LSAT-AR	AGI Eval	Law (Analytic Reasoning)	230	[13], [16]
LSAT-LR	AGI Eval	Law (Logical Reasoning)	510	[13], [16]
LSAT-RC	AGI Eval	Law (Reading Comprehension)	260	[13], [16]
MedMCQA Valid	MedMCQA	Medicine	6,150	[17]
SAT-English	AGI Eval	English	206	[13]
SAT-Math	AGI Eval	Math	220	[13]

Table 1 - Source of problem sets used to create the MCQA test set

```
{
  "id": 3,
  "source": "agi-eval/aqua-rat",
  "source_id": 35,
  "topic": "Math",
  "context": "",
  "question": "A rectangular solid, 3 x 4 x 15, is inscribed in a sphere,
              so that all eight of its vertices are on the sphere.
              What is the diameter of the sphere?",
  "choices": {
    "A": " 13.3542",
    "B": " 15.8113",
    "C": " 18.3451",
    "D": " 19.5667",
    "E": " 20.8888"},
  "answer": "B",
  "solution": "In an inscribed rectangle in a sphere, we will have a line joining
              the opposite vertices as the diameter. According to the Pythagoras theorem,
              sides 3, 4 give diagonal as 5 ==> with 5 and 15, we get 5sqrt(10).
              5sqrt(10) or 15.8113 is the diameter of the sphere.\nanswer = B"
}
```

Figure 1 - Sample of a multiple-choice question in standardized data format – with whitespace added for readability.

2.2 Models

We used two popular LLMs to test our hypotheses – namely GPT-3.5 and GPT-4.

GPT-3.5 is a Generative Pre-trained Transformer (GPT) created by OpenAI and publicly released as "ChatGPT" in Nov 2022 [18], [19]. GPT-4 is a more powerful GPT with more advanced capabilities [20], [21]. However, GPT-4 costs roughly 30x more per output token than GPT-3.5 [8].

GPT-3.5 and GPT-4 can be accessed via an Application Programming Interface (API) hosted by OpenAI or Microsoft. We used Microsoft Azure Open AI Service for our experiments – though our results should hold for either platform – since they both use the same underlying foundational models [22]–[24].

2.3 Prompts

We used three prompt-engineering techniques to test our hypotheses. These prompts consisted of an answer-only prompt, a standard (i.e. verbose) CoT prompt, and a concise CoT prompt.

The answer-only prompt instructed the LLM to respond with only the answer to the question. The prompt included a single (i.e., one-shot) example to demonstrate how to complete the task. This prompt provided a baseline for evaluating the minimum response length and task performance (see Figure 2).

```
[System Prompt]
You are an intelligent assistant.
Your task is to answer the following multiple-choice questions.
You MUST answer the question using the following format 'Action: Answer("[choice]")'
The parameter [choice] is the letter or number of the answer you want to select (e.g. "A",
"B", "C", or "D").
For example, 'Answer("C")' will select choice "C" as the best answer.
You MUST select one of the available choices; the answer CANNOT be "None of the Above".

[Example Problem]
Question: What is the capital of the state where Johns Hopkins University is located?
Choices:
  A: Baltimore
  B: Annapolis
  C: Des Moines
  D: Las Vegas

[Example Solution]
Action: Answer("B")
```

Figure 2 - Sample of the answer-only system prompt and one-shot example

The standard CoT prompt instructed the LLM to think step-by-step through its thought process before answering. The one-shot example included a verbose CoT for each reasoning step in the solution. This prompt provided a ceiling for maximum response length and task performance (see Figure 3).

```
[System Prompt]
You are an intelligent assistant.
Your task is to answer the following multiple-choice questions.
Think step-by-step through the problem to ensure you have the correct answer.
Then, answer the question using the following format 'Action: Answer("[choice]")'
The parameter [choice] is the letter or number of the answer you want to select (e.g. "A",
"B", "C", or "D").
For example, 'Answer("C")' will select choice "C" as the best answer.
You MUST select one of the available choices; the answer CANNOT be "None of the Above".

[Example Problem]
Question: What is the capital of the state where Johns Hopkins University is located?
Choices:
  A: Baltimore
  B: Annapolis
  C: Des Moines
  D: Las Vegas

[Example Solution]
Thought:
  Johns Hopkins University is located in Baltimore.
  Baltimore is a city located in the State of Maryland.
  The capital of Maryland is Annapolis.
  Therefore, the capital of the state where Johns Hopkins University is located is
Annapolis.
  The answer is B: Annapolis.
Action: Answer("B")
```

Figure 3 - Sample of standard CoT system prompt and one-shot example

The CCoT prompt included the standard CoT prompt but also had a final instruction to “Be concise.” The one-shot example also included a more concise CoT in its solution. This prompt allowed us to compare CCoT to both the answer-only and standard CoT prompts (see Figure 4).

```
[System Prompt]
You are an intelligent assistant.
Your task is to answer the following multiple-choice questions.
Think step-by-step through the problem to ensure you have the correct answer.
Then, answer the question using the following format 'Action: Answer("[choice]")'
The parameter [choice] is the letter or number of the answer you want to select (e.g. "A",
"B", "C", or "D").
For example, 'Answer("C")' will select choice "C" as the best answer.
You MUST select one of the available choices; the answer CANNOT be "None of the Above".
Be concise.

[Example Problem]
Question: What is the capital of the state where Johns Hopkins University is located?
Choices:
  A: Baltimore
  B: Annapolis
  C: Des Moines
  D: Las Vegas

[Example Solution]
Thought:
  Johns Hopkins University is located in Baltimore, Maryland.
  The capital of Maryland is Annapolis.
Action: Answer("B")
```

Figure 4 - Sample of CCoT system prompt and one-shot example

2.4 Metrics

To test our hypotheses, we measured the LLM’s response length and correct-answer accuracy.

Response length was measured as the number of output tokens produced by the LLM in its response. A token is the smallest unit of information an LLM can process. Both GPT-3.5 and GPT-4 work with sub-word tokens. On average, there are approximately 1.3 tokens per word [25].

Performance was measured as the number of correctly answered questions divided by the total number of questions asked. This performance metric allowed us to directly compare the LLM’s ability to correctly solve a problem presented in the MCQA format.

2.5 Analysis

To test the response-length hypothesis (RL-H_0), we performed a Mann-Whitney U (MWU) test on the means of the CoT and CCoT response lengths. To test our performance hypothesis (P-H_0), we performed an MWU test on the means of correct-answer accuracy for standard CoT and CCoT [26], [27].

We used an MWU test instead of a t-test due to the non-normal distribution of the data. A Shapiro-Wilk normality test indicated that the response-length data and accuracy data were not normally distributed. This condition violates the normality assumption of the t-test, so an MNU test was used instead.

3. Results

3.1 Response-Length

Our analysis revealed that CCoT reduced average response length by 48.70% compared to CoT.

For GPT-3.5, we saw a 47.62% decrease in average response length when comparing CCoT to CoT. The MWU test yielded (U=873,046.00, $p < 0.001$), indicating a significant difference between the two means.

For GPT-4, we saw a 49.77% decrease in average response length between CCoT and CoT. The MWU test yielded (U=807,523.50, $p < 0.001$), also indicating a significant difference between the means.

Given our significance level of 0.05, the low p-values indicate that the differences in response lengths between CCoT and CoT are statistically significant for both GPT-3.5 and GPT-4. Thus, we have sufficient evidence to reject the null hypothesis (RL-H_0) in favor of the alternative hypothesis (RL-H_1).

A visual analysis of the data supports these findings.

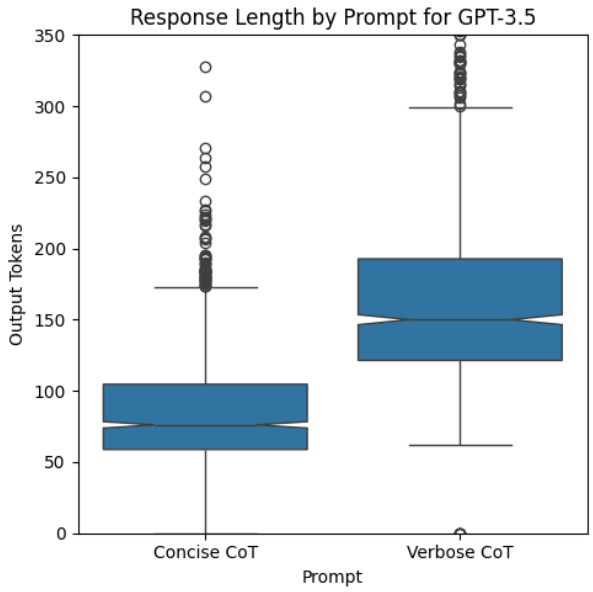

Figure 5 - GPT-3.5 with CCoT reduces response length by an average of 47.62% compared to verbose CoT.

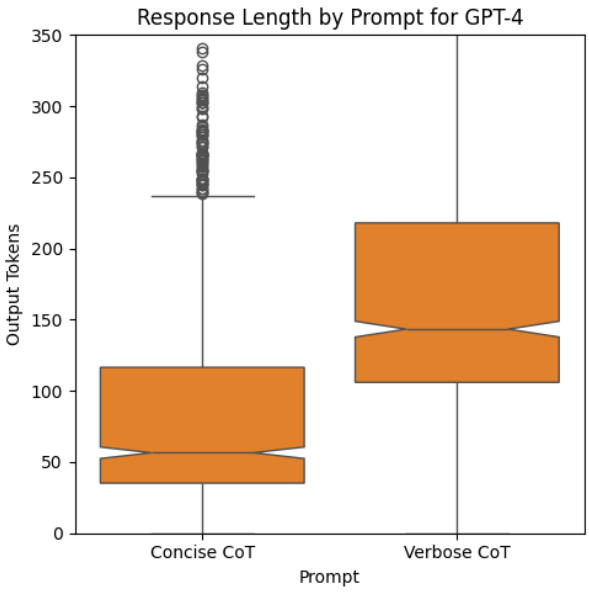

Figure 6 - GPT-4 with CCoT reduces response length by an average of 49.77% compared to verbose CoT.

A visual analysis of response length by exam shows this decrease in response length holds across all problem domains. This pattern can be observed for both GPT-3.5 and GPT-4 (see Figures 7 and 8).

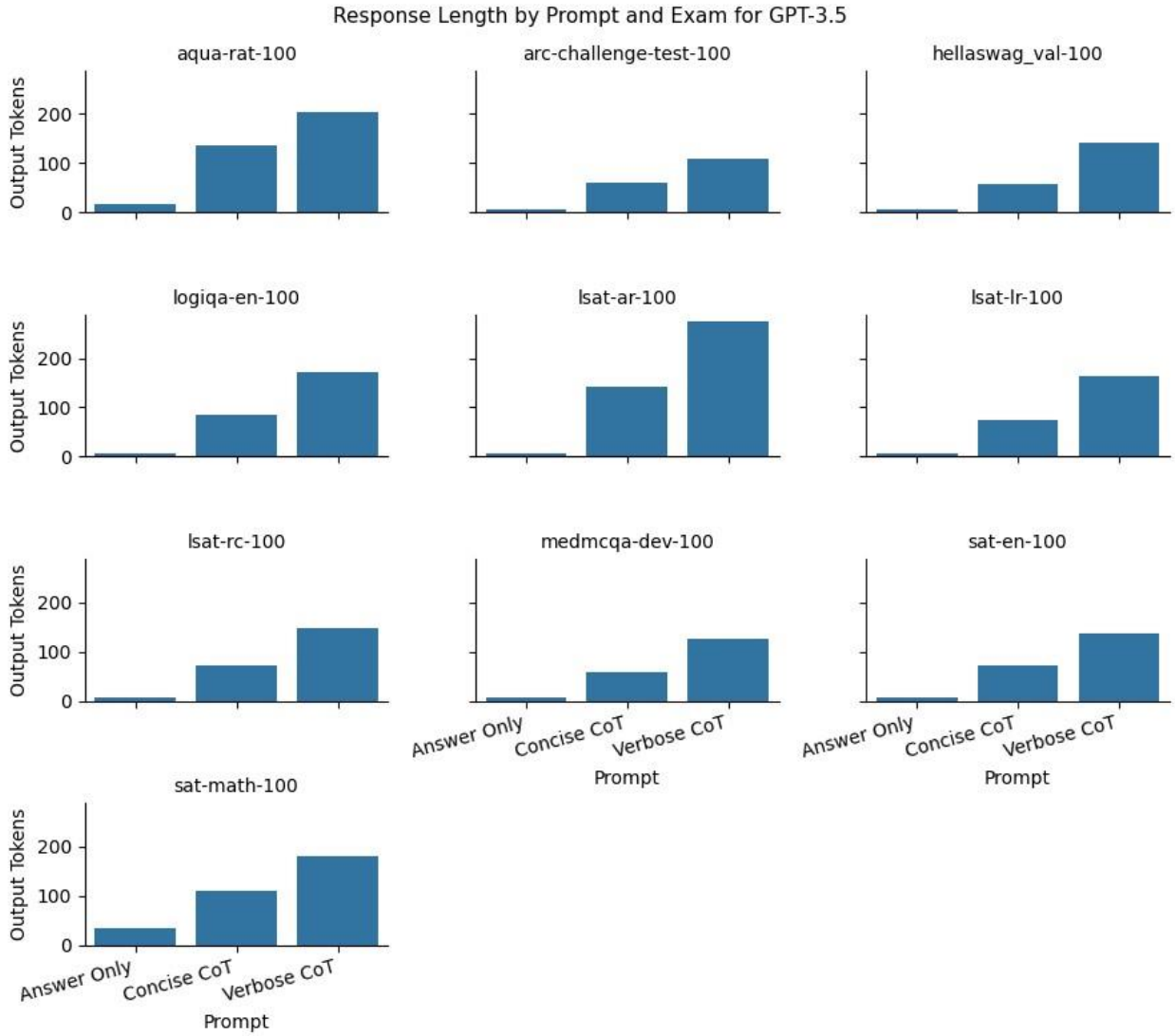

Figure 7 - GPT-3.5 with CCoT decreases response length significantly across all problem domains compared to CoT.

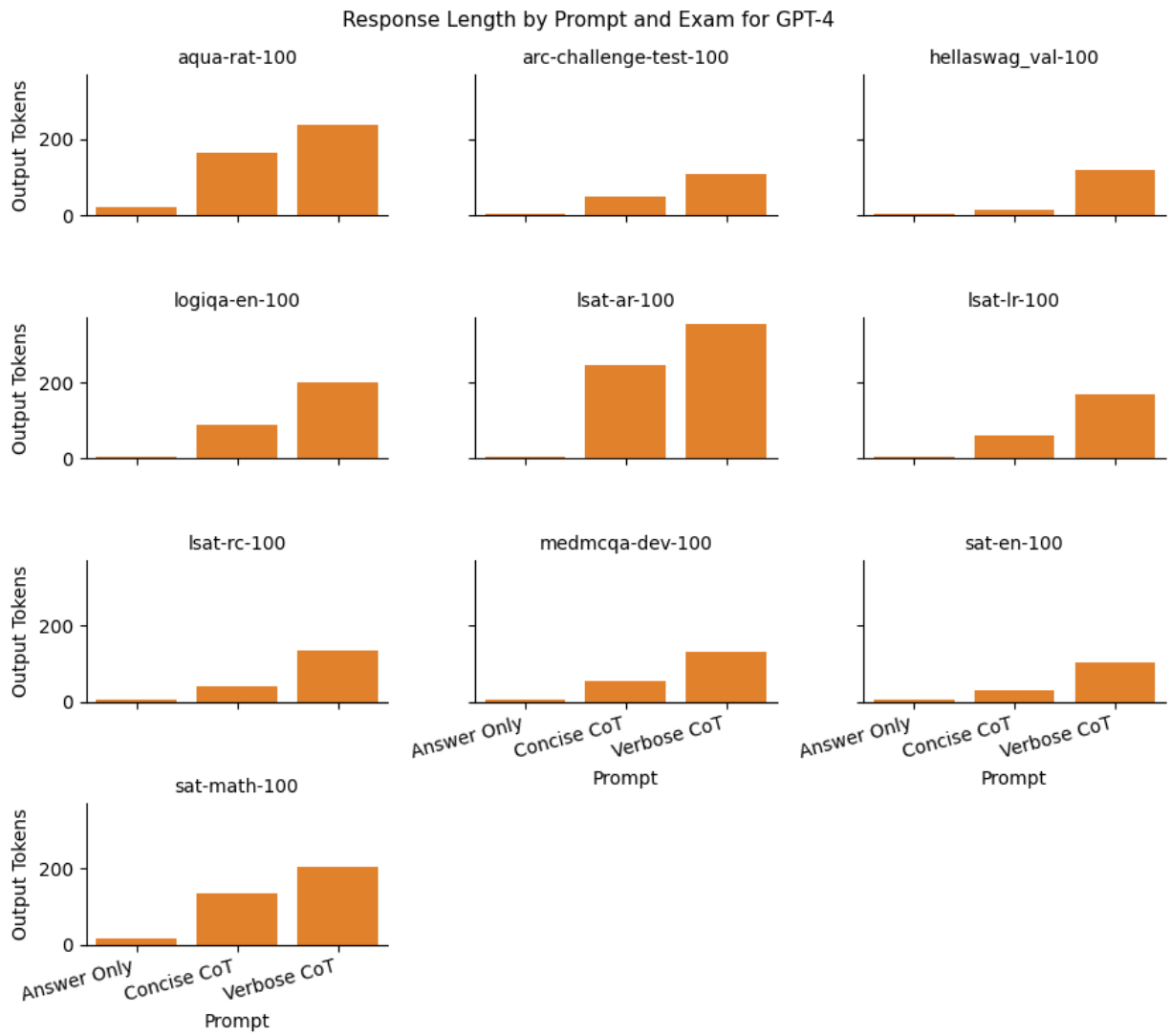

Figure 8 - GPT-4 with CCoT decreases response length significantly across all problem domains compared to CoT.

3.2 Performance

Our analysis revealed that CCoT did not reduce problem-solving performance – in any statistically meaningful way – compared to standard CoT.

For GPT-3.5, we saw a 2.95% decrease in average accuracy when comparing CCoT to CoT. The MWU test yielded (U=507,648.50, p = 0.55), indicating the differences in the means are not statistically significant.

For GPT-4, we saw an *increase* in response length of 0.25% when comparing CCoT to CoT prompting. The MWU test yielded (U=485,396.00, p = 0.21), also indicating a non-significant difference in the means.

Given our significance level of 0.05, the high p-values indicate that the differences in accuracy between CCoT and CoT are not statistically significant for either GPT-3.5 or GPT-4. Thus, we do not have sufficient evidence to reject the null hypothesis for performance ($P\text{-}H_0$).

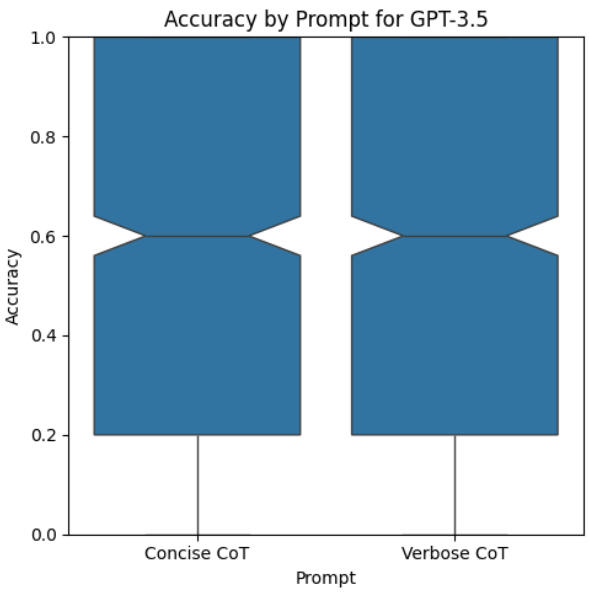

Figure 9 - GPT-3.5 with CCoT has similar accuracy compared to verbose CoT.

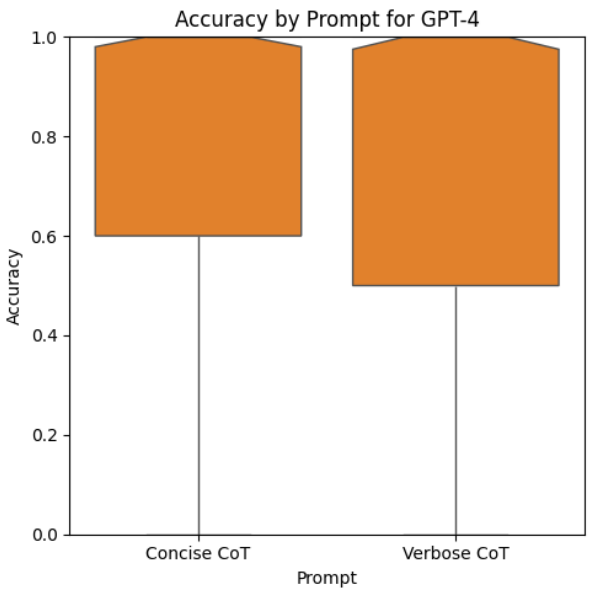

Figure 10 - GPT-4 with CCoT has similar accuracy compared to verbose CoT. (Note: the median value for both prompts is 1.0, making the box plot more difficult to interpret visually.)

A more in-depth analysis by exam revealed that GPT-3.5 with CCoT produced a statistically significant decrease in performance *only* for math problems (i.e., the aqua-rat-100 and sat-math-100 exams) compared to CoT. Other problem domains saw no decrease in accuracy (see Figure 11).

For these two math exams, GPT-3.5 with CCoT resulted in an average reduction in accuracy of 27.69%. An MWU test on the means of CoT and CCoT yielded ($U=26{,}546.00$, $p < 0.001$). Thus, for the special case of math problems, we must reject the null hypothesis ($P\text{-}H_0$) in favor of the alternative hypothesis ($P\text{-}H_1$).

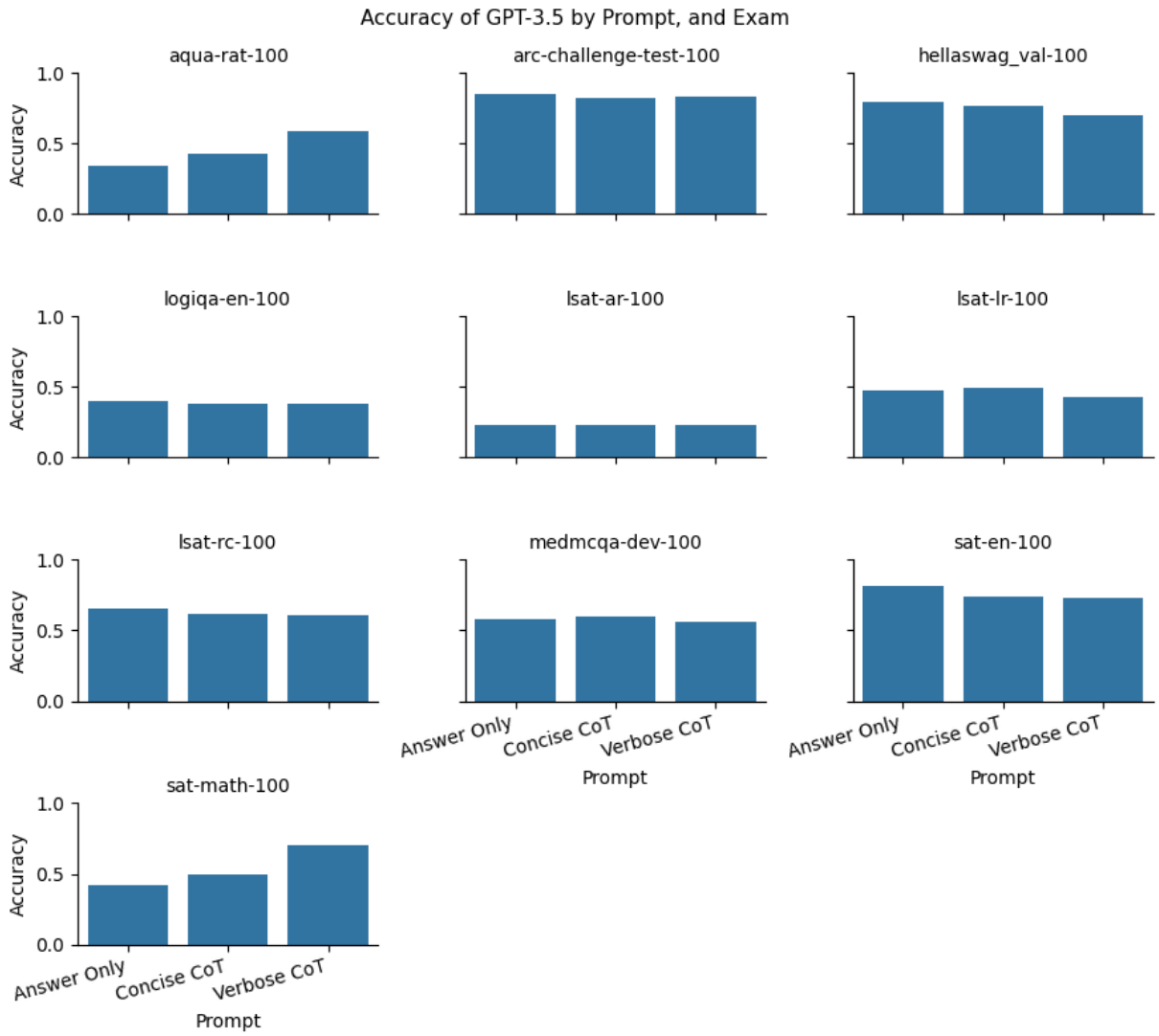

Figure 11 – Verbose CoT prompting with GPT-3.5 significantly improved performance on math problems (i.e., aqua-rat-100 and sat-math-100) compared to CCoT but had minimal impact on other problem domains

On the other hand, GPT-4 with CCoT did not result in a statistically significant decrease in performance on math problems compared to CoT. Both CCoT and CoT had roughly equivalent performance on all problem domains. CCoT also significantly outperformed the answer-only prompt on math problems.

An MWU test on the means yielded (U=21,471.50, p=0.17). Thus, in the special case of the mathematics problem domain, the null hypothesis (P-H_0) still holds for GPT-4.

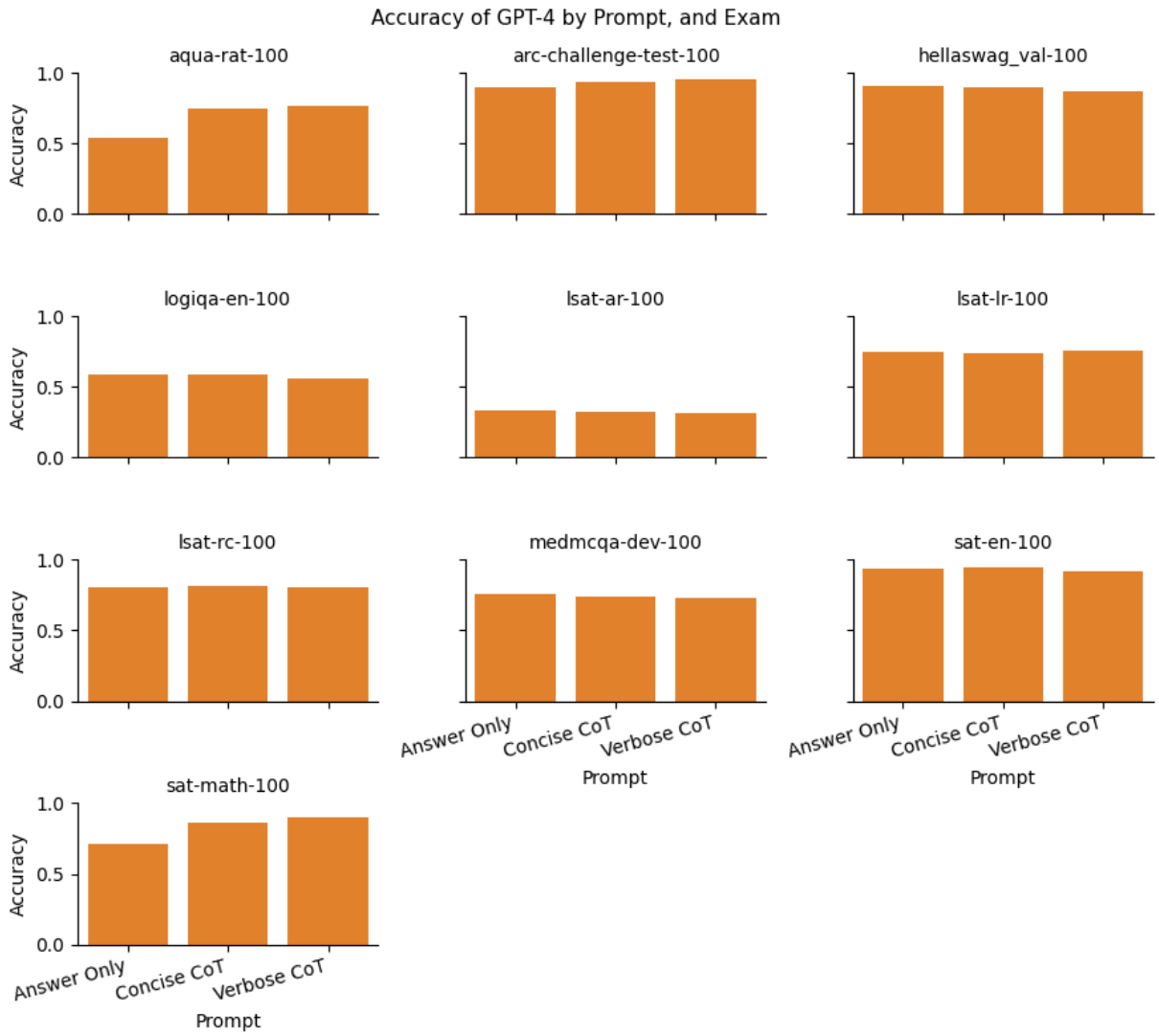

Figure 12 – Verbose CoT and CCoT prompting with GPT-4 significantly improved accuracy for math problems (i.e., aqua-rat-100 and sat-math-100) but had minimal impact on other problem domains.

3.3 Cost Analysis

To understand the practical implications of these results, we computed the cost of solving all 1,000 problems using the current per-token pricing model for GPT-3.5 and GPT-4[2].

Currently, GPT-3.5 is priced at $0.001 per 1,000 input tokens and $0.002 per 1K output tokens[3]. GPT-4 is priced at $0.03 per 1K input tokens and $0.06 per 1K output tokens [8]. As a result, CCoT produced a total cost savings of 21.85% for GPT-3.5 and 23.49% for GPT-4. These cost savings should scale linearly.

[2] For our study, we solved each MCMQ problem 10 times. So our total cost was 10x the values we are reporting.
[3] All prices are in US Dollars (USD)

Cost Analysis				
	GPT-3.5 with CoT	**GPT-3.5 with CCoT**	**GPT-4 with CoT**	**GPT-4 with CCoT**
Input Cost	$0.55	$0.51	$16.37	$15.29
Output Cost	$0.33	$0.17	$10.53	$5.29
Total Cost	$0.88	$0.69	$26.90	$20.58
Cost Savings		21.85%		23.49%

Table 2 - CCoT significantly reduces total costs per 1,000 problems for both GPT-3.5 and GPT-4.

4. Discussion

4.1 Limitations

There were several limitations in this research study:

First, our study involved only two LLMs – both of which were versions of GPT. As a result, these findings may not be replicable using other proprietary and open-source LLMs like Llama 2, PaLM, and Claude.

Second, our study only tested a single CoT and CCoT prompt. As a result, other variations of CoT and CCoT prompts may produce different results.

Third, our study was limited to ten problem domains. As a result, these findings may not hold for other problem domains or sub-domains within each problem domain.

Finally, in the case of GPT-4, the median accuracy was 1.0. As a result, the data were compressed at the ceiling of accuracy. This compression may have caused issues with our statistical analysis of performance.

4.2 Implications

These results have practical implications for AI systems engineers building AI systems with LLMs.

Since most proprietary LLM APIs charge on a per-token pricing model, reducing the number of output tokens in an LLM's response has direct cost savings [8], [9]. In addition, getting more concise responses without sacrificing accuracy has direct savings in energy consumption and response wait times.

These results also have theoretical implications for AI researchers studying how LLMs perform step-by-step reasoning using CoT.

CCoT can reduce the number of output tokens by roughly half while maintaining the same level of accuracy. As a result, only a subset of the CoT tokens contribute to the performance of the LLM. This opens new questions about which aspects of a CoT lead to a correct solution and which are superfluous.

4.3 Future Research

To improve upon this research, we suggest the following follow-up experiments:

First, we recommend performing additional experiments with other proprietary and open-source LLMs. It would be beneficial to know if these results hold for all LLMs or only GPT-3.5 and GPT-4.

Second, we recommend testing more CCoT prompt variations. Instructing the LLM to be even more concise in the system prompt may further decrease response length. In addition, even more concisely written few-shot examples may produce even more succinct responses without sacrificing accuracy.

Third, we recommend testing more task types and problem domains beyond those tested in our MCQA test set. These results may not hold for non-MCQA task types. In addition, they may not hold for other problem domains. Other math sub-domains may also exhibit different behaviors than what we observed.

Finally, we recommend performing an in-depth error analysis to improve our understanding of the nature of CCoT errors. The LLMs may be making specific types of errors in their CoT that impact their performance. Understanding these errors may provide additional insight into how to mitigate them.

5. Conclusion

In this research study, we introduced the CCoT prompt-engineering technique. We demonstrated that it reduced response token length for GPTs by 48.70% while performing as well as standard CoT.

GPT-4 incurred no performance penalty in any problem domain. However, for math problems, GPT-3.5 incurred a 27.69% reduction in accuracy.

In practice, CCoT can reduce the per-token cost of solving multi-step problems by 22.67%. These cost savings should also extend to reduced energy consumption and LLM response wait times.

These results have practical implications for AI systems engineers using CoT to build problem-solving AI systems. They also have theoretical implications for AI researchers studying LLM reasoning processes.

6. References

[1] J. White *et al.*, "A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT," *ArXiv*, Feb. 2023, Accessed: Nov. 26, 2023. [Online]. Available: https://arxiv.org/abs/2302.11382

[2] G. Mialon *et al.*, "Augmented Language Models: a Survey," *ArXiv*, Feb. 2023, Accessed: Apr. 28, 2023. [Online]. Available: https://arxiv.org/abs/2302.07842v1

[3] J. Wei *et al.*, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," *ArXiv*, Jan. 2022, Accessed: Dec. 07, 2023. [Online]. Available: https://arxiv.org/abs/2201.11903

[4] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large Language Models are Zero-Shot Reasoners," in *Advances in Neural Information Processing Systems*, May 2022, pp. 22199–22213. Accessed: Dec. 07, 2023. [Online]. Available: https://arxiv.org/abs/2205.11916

[5] Y. Zhou *et al.*, "Large Language Models Are Human-Level Prompt Engineers," *The Eleventh International Conference on Learning Representations*, Nov. 2023, Accessed: Dec. 06, 2023. [Online]. Available: https://arxiv.org/abs/2211.01910

[6] W. Kadous, "Numbers Every LLM Developer Should Know," Anyscale. Accessed: Dec. 06, 2023. [Online]. Available: https://www.anyscale.com/blog/num-every-llm-developer-should-know

[7] N. Crispino, K. Montgomery, F. Zeng, D. Song, and C. Wang, "Agent Instructs Large Language Models to be General Zero-Shot Reasoners," *ArXiv*, Oct. 2023, Accessed: Dec. 07, 2023. [Online]. Available: https://arxiv.org/abs/2310.03710

[8] OpenAI, "Pricing," Open AI. Accessed: Dec. 06, 2023. [Online]. Available: https://openai.com/pricing

[9] Anyscale, “Pricing,” Anyscale. Accessed: Dec. 06, 2023. [Online]. Available: https://docs.endpoints.anyscale.com/pricing/

[10] A. Madaan and A. Yazdanbakhsh, “Text and Patterns: For Effective Chain of Thought, It Takes Two to Tango,” *ArXiv*, Sep. 2022, Accessed: Dec. 07, 2023. [Online]. Available: https://arxiv.org/abs/2209.07686

[11] A. Madaan and A. Yazdanbakhsh, “Text and Patterns: For Effective Chain of Thought It Takes Two to Tango,” OpenReview.net - ICLR 2023 Conference Withdrawn Submission. Accessed: Dec. 06, 2023. [Online]. Available: https://openreview.net/forum?id=z9fXRC5XdT

[12] P. Clark *et al.*, “Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge,” *ArXiv*, Mar. 2018, Accessed: Nov. 22, 2023. [Online]. Available: https://arxiv.org/abs/1803.05457

[13] W. Zhong *et al.*, “AGIEval: A Human-Centric Benchmark for Evaluating Foundation Models,” *ArXiv*, Apr. 2023, Accessed: Dec. 07, 2023. [Online]. Available: https://arxiv.org/abs/2304.06364

[14] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, “HellaSwag: Can a Machine Really Finish Your Sentence?,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. Accessed: Nov. 22, 2023. [Online]. Available: https://arxiv.org/abs/1905.07830

[15] J. Liu, L. Cui, H. Liu, D. Huang, Y. Wang, and Y. Zhang, “LogiQA: A Challenge Dataset for Machine Reading Comprehension with Logical Reasoning,” in *International Joint Conference on Artificial Intelligence*, 2020. Accessed: Nov. 22, 2023. [Online]. Available: https://arxiv.org/abs/2007.08124

[16] S. Wang *et al.*, “From LSAT: The Progress and Challenges of Complex Reasoning,” *IEEE/ACM Trans Audio Speech Lang Process*, vol. 30, pp. 2201–2216, Aug. 2021, Accessed: Nov. 22, 2023. [Online]. Available: https://doi.org/10.1109/TASLP.2022.3164218

[17] A. Pal, L. K. Umapathi, and M. Sankarasubbu, “MedMCQA: A Large-scale Multi-Subject Multi-Choice Dataset for Medical domain Question Answering,” in *Proceedings of the Conference on Health, Inference, and Learning*, PMLR, 2022, pp. 248–260. Accessed: Nov. 22, 2023. [Online]. Available: https://proceedings.mlr.press/v174/pal22a.html

[18] Open AI, “Introducing ChatGPT,” Open AI. Accessed: Apr. 28, 2023. [Online]. Available: https://openai.com/blog/chatgpt

[19] T. Brown *et al.*, “Language Models are Few-Shot Learners,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[20] Open AI, “GPT-4 Technical Report,” *ArXiv*, Mar. 2023, Accessed: Apr. 28, 2023. [Online]. Available: https://arxiv.org/abs/2303.08774

[21] Open AI, “GPT-4,” Open AI. Accessed: Apr. 28, 2023. [Online]. Available: https://openai.com/research/gpt-4

[22] OpenAI, “Models,” OpenAI. Accessed: Dec. 06, 2023. [Online]. Available: https://openai.com/product#models

[23] Open AI, “https://platform.openai.com/docs/api-reference/chat/create,” Open AI - API Reference. Accessed: Nov. 25, 2023. [Online]. Available: https://platform.openai.com/docs/api-reference/chat/create

[24] Microsoft, “Azure OpenAI Service,” Microsoft. Accessed: Dec. 06, 2023. [Online]. Available: https://azure.microsoft.com/en-us/products/ai-services/openai-service/

[25] OpenAI, “Tokens,” OpenAI. Accessed: Dec. 06, 2023. [Online]. Available: https://platform.openai.com/docs/introduction/tokens

[26] H. B. Mann and D. R. Whitney, “On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other,” *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, Mar. 1947, doi: 10.1214/aoms/1177730491.

[27] SciPy Community, “SciPy Manual - scipy.stats.mannwhitneyu,” SciPy. Accessed: Dec. 06, 2023. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.mannwhitneyu.html