Part A) Baseline Prompting Results

I tested GPT-4o on a random sample of 300 questions from the GSM8K evaluation dataset. The model was applied to the same 300 questions using zero-shot, few-shot, and chain of thought prompting strategies.

Zero shot prompt context:

```
zero_shot_context = "You will receive a math problem. Solve the problem and ensure your answer ends with #### followed by the numerical answer."
```

Few shot prompt context:

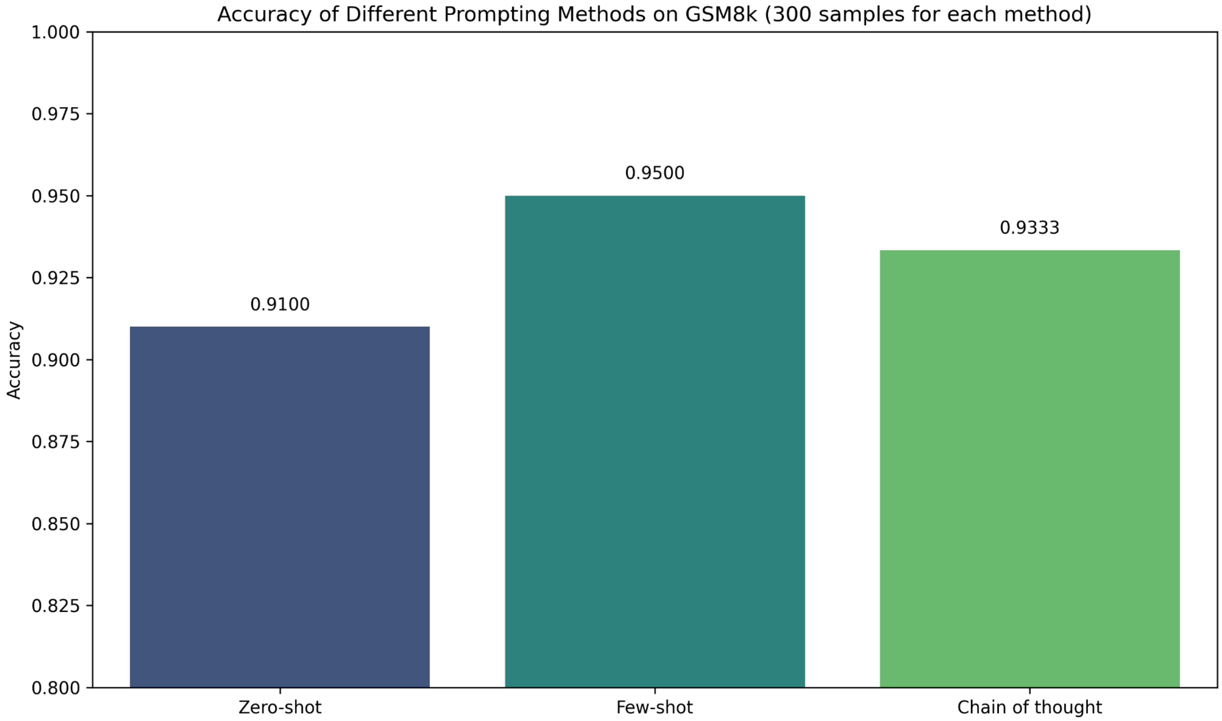
```
few_shot_context = f"""
You will receive a math problem. Solve the problem similarly to the following examples, ensuring that your answer ends with #### followed by the numerical answer.
Question #1. {df_train.iloc[0,]['question']},
Answer #1. {df_train.iloc[0,]['answer']},
Question #2. {df_train.iloc[1,]['question']},
Answer #2. {df_train.iloc[1,]['answer']},
Question #3. {df_train.iloc[2,]['question']},
Answer #3. {df_train.iloc[2,]['answer']},
Remember, every answer must include #### followed by the numerical answer.
"""
```

Where the few shot examples were drawn from the shuffled df_train dataset, so as not to contaminate the model during testing.

Chain of thought prompt context:

```
cot_context = """You will receive a math problem. Use Chain of Thought reasoning to solve the problem, working through it step by step with clearly explained reasoning.
At the end, ensure your answer concludes with #### followed by the numerical answer."""
```

Results

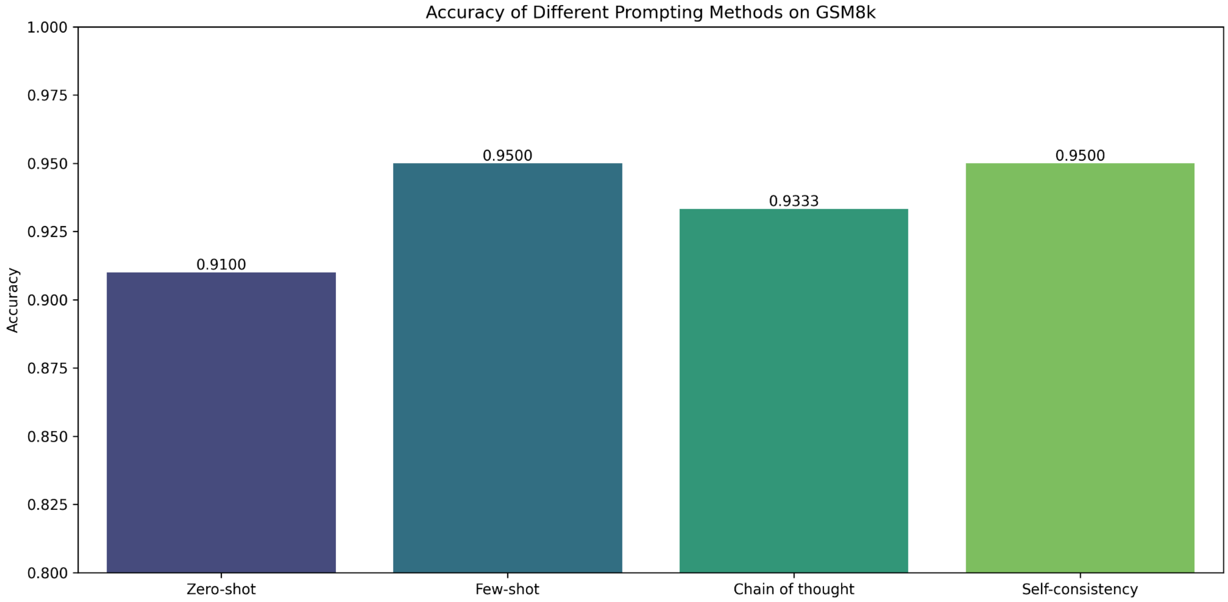


I expected performance to increase substantially when going from zero-shot to few-shot prompting, and then to increase at least marginally when going to the chain of thought approach. As expected, few-shot significantly outperformed zero-shot on the 300 question samples, but I was surprised by the result that few-shot also marginally outperformed the chain of thought approach.

A priori, this shows that chain of thought itself is a powerful approach—the fact that we can get higher performance on these questions simply by adding a "work through it step by step" instruction is a cool result. It's also possible slight variations in the chain of thought context (i.e., slightly tweaking the chain of thought description in the instructions) could marginally increase performance, too.

On the other hand, we see how powerful few-shot prompting is here, given that it excels over zero-shot and also outperforms CoT. A few explanations here could include that the 3 random examples from df_train given in the few-shot context were particularly well-suited to the test set. More generally, it could be the case that the questions are straightforward enough for an already powerful model like GPT-4o that additional reasoning can result in an "overthinking" effect, such that the chain of thought is less effective than it might otherwise be on more complicated problems.

B) Improved CoT Performance with Self-Consistency



I did a number of searches for “improving chain of thought” and “improving reasoning models,” with a ton of results from the past 2 years or so. A number of approaches incorporate fine-tuning and/or reinforcement learning (c.f. DeepSeek’s V3 and R1 models) to improve reasoning model performance. For the scope of this assignment, I wanted to find a simpler strategy based on prompting strategy alone that could try to increase the baseline chain of thought performance and see if I could beat few-shot prompting.

Although I’m not sure if it’s exactly “recent literature” in AI terms, I decided to focus on “Self-Consistency Improves Chain of Thought Reasoning in Language Models” by Wang et al. from ICLR 2023.[1] The idea here is to use temperature sampling to produce diverse reasoning paths (in up to 40 separate chain of thought prompts for the same problem), and then use simple majority voting to find the consensus answer. For the scope of this assignment, I used the paper’s most basic prompting framework, applying my same chain of thought prompt to the same 300 problems as before, only this time holding temperature at 0.7 (which yielded the best results in the paper) across 5 rounds for each question (for a total of 1,500 API calls). I then produced an answer via majority voting across the 5 rounds, and compared accuracy as before with the other methods. This “self-consistency” approach indeed improved performance from the baseline chain of thought result, which is great. Unfortunately, it was still tied with the few-shot approach, which was a bit disappointing. It should be noted that Wang et al. noticed improvements over CoT baseline after just 3 rounds, however they achieved their best result after 40 rounds (c.f., pg. 7). Therefore, we might expect that adding additional self-consistency rounds, and/or further varying temperature could have increased performance even further, thereby beating out few-shot. But for my purposes in this assignment I was satisfied to see the improvement over baseline CoT from part a) using the self-consistency approach.

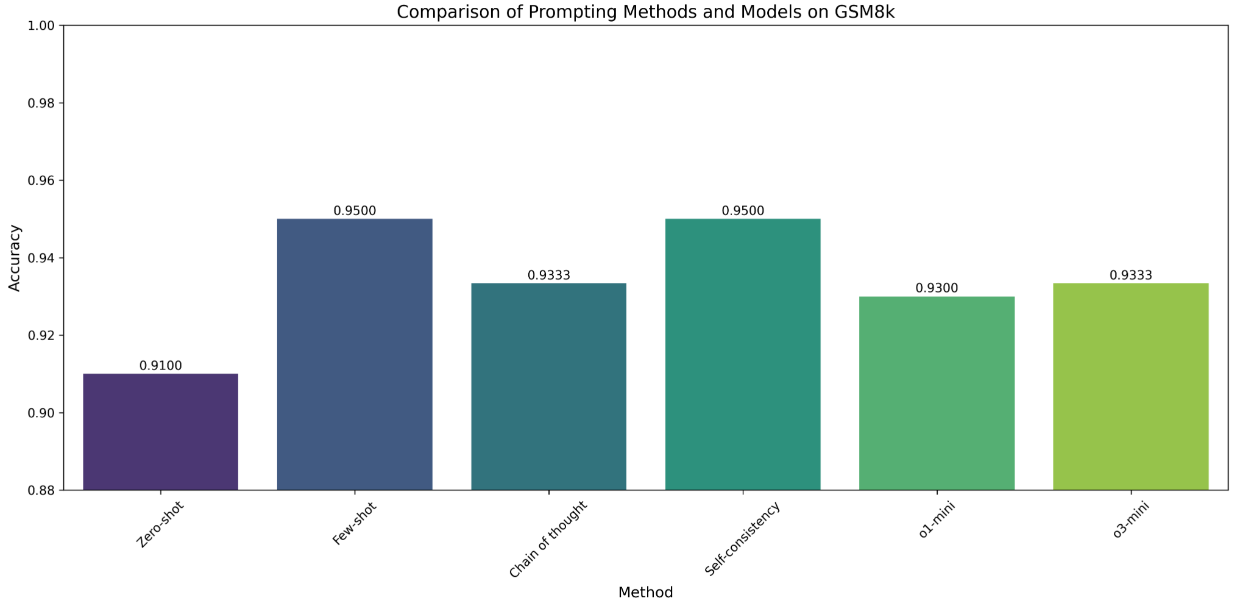
[1] https://arxiv.org/abs/2203.11171

Comparing results with o1-mini and o3-mini

Next, I sought to compare the baseline results and the self-consistency CoT improvement result with the performance of o1-mini and o3-mini, holding constant the following prompting scheme:

```
prompt + " Return #### followed by the final numerical answer without any additional punctuation or units"
```

The results were surprising.

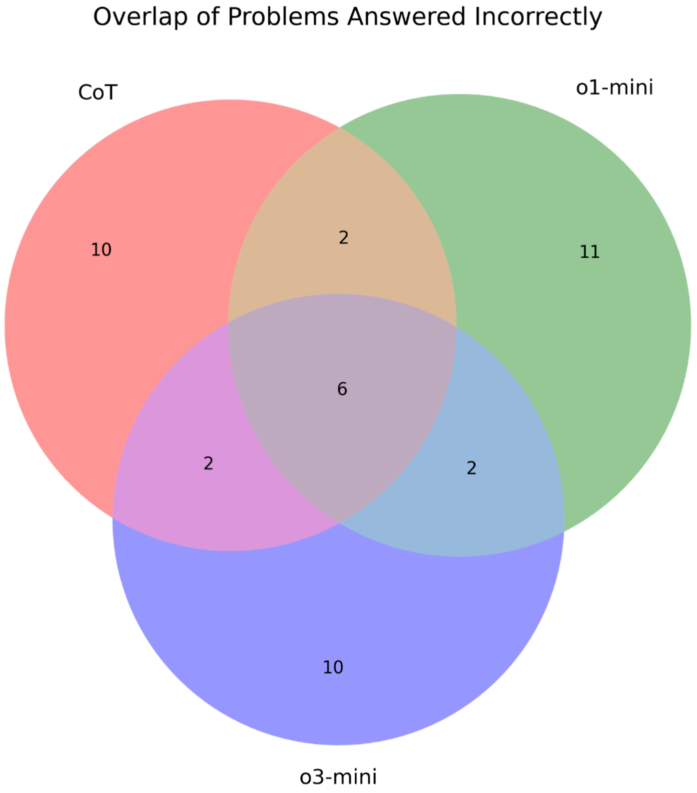


o1-mini and o3-mini both underperformed my baseline few-shot and self-consistency prompting with 4o. Additionally, o3-mini was only marginally more performative than o1-mini. This is surprising given that these models are apparently so capable. But there are some important caveats here.

First, I noticed that o1-mini and o3-mini both had slightly worse instructions-following than 4o. In particular, the reasoning models liked to return answers with units or additional punctuation, which required some massaging of the prompt, and an updated extract solution function to get as close as possible to full extraction accuracy, but still may have admitted to a few errors. In addition, super annoyingly, o1-mini repeatedly returned error 400, "inappropriate prompt" detected, often for simple questions such as, ”A plague infects ten people. Every day, each infected person infects six others. How many people are infected after three days? Return #### followed by the final numerical answer without any additional punctuation or units."

I found similar reports from other o1-mini API users (cf. https://community.openai.com/t/invalid-prompt-in-using-o1-preview-and-o1-mini/1061767). For example, this resulted in 7 NaN answers for o1-mini, which, if answered correctly, would have boosted accuracy roughly to the level of few-shot and self-consistency.

Interestingly, I also noticed that there appeared to be a pattern in which problems o1-mini and o3-mini got wrong—specifically, their wrong answers appeared to correlate with the baseline CoT wrong answers. For example, almost half of the incorrect answers in baseline CoT were also answered incorrectly by the reasoning models.



This result may lend credence to my earlier thinking that apparently, these models can "overthink" certain simple questions and fail to produce the correct answer. For example, it may be the case that they "overthink" any slight ambiguity in the prompt, leading them down a false reasoning path. And This may also help explain why producing multiple reasoning paths in the self-consistency approach yields the highest chain of thought performance. And it may also help explain why the few shot prompting, which ostensibly shows the models that a relatively straightforward reasoning path can yield the correct answer, is so relatively effective here. All this said, it's important to remember that all 3 of the models in question are highly performant on these questions, achieving over 90% accuracy in all cases.

C) Ablation Studies

I conducted 3 ablations, similar to the ones in Wei et al. (2023) to try to isolate the effects of chain of thought prompting on 4o. I used the same 300 GSM8K problems as were used to evaluate the earlier approaches and reasoning models, but this time with new contexts for the 3 ablations:

```
equation_only_context = """Solve this math problem by first writing the equation(s) needed to solve it, then providing the final numerical answer.
```

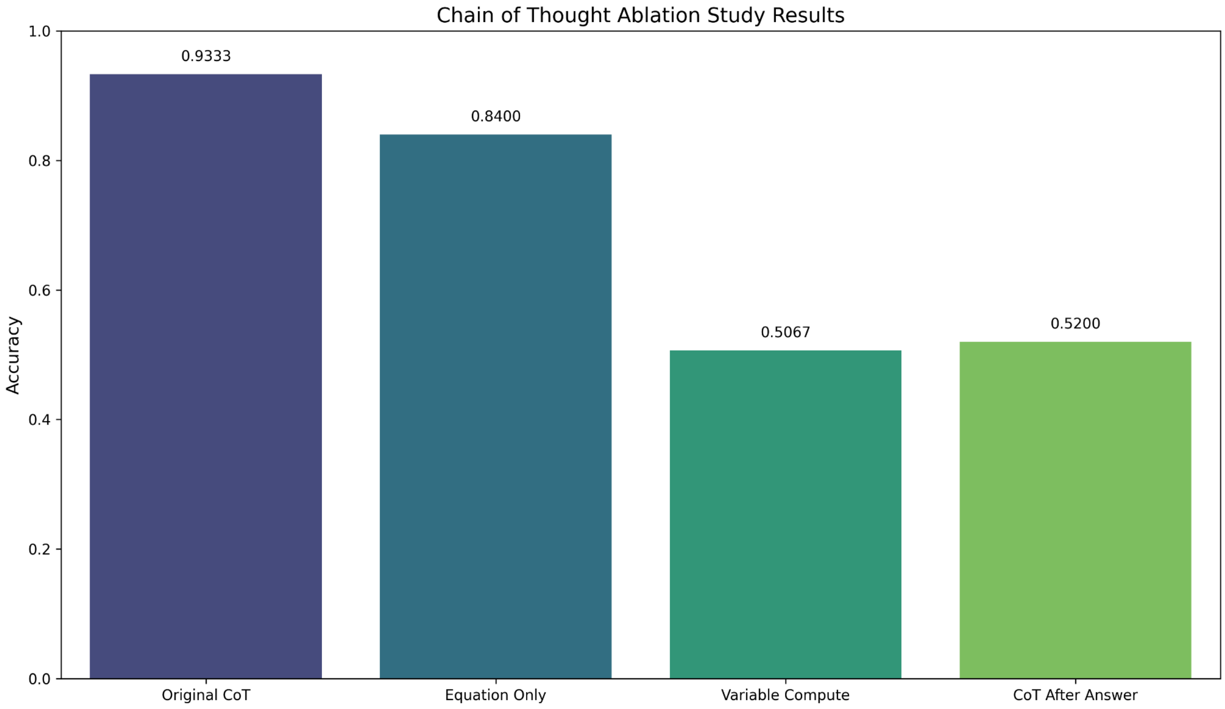
```
Write your answer in the format:
Equation: [your equation here]
#### [final numerical answer]"""

variable_compute_context = """Solve this math problem.
First, output a sequence of dots (.) where the number of dots is approximately equal to the complexity of the problem.
Then provide the final numerical answer.
Write your answer in the format:
[dots]
#### [final numerical answer]"""

cot_after_context = """Solve this math problem.
First provide only the final numerical answer, then explain your reasoning.
Write your answer in the format:
#### [final numerical answer]
Explanation: [your step-by-step reasoning]"""
```

We have 1) equation only, where the model is instructed to provide an equation before the final numerical answer, 2) variable compute, where the model is asked to insert dots reflecting the approximate complexity of the problem before the final answer, and 3) chain of thought after context, where the model is prompted to provide its reasoning after providing the numerical solution. In all cases, I gave the exact same formatting instructions to maintain as much consistency as possible.

The results show that equation only performs reasonably well at 84%, but still well below the performance of baseline CoT. Still, it suggests that we gain about 9% by prompting the model to conduct step by step reasoning, suggesting that there is still something important in the natural language reasoning steps. Variable compute and CoT after answer showed dramatic decreases in performance, barely achieving above 50% accuracy in both cases. This suggests that time spent on computation alone and "reasoning after" both dramatically decrease performance of 4o as compared to even the zero shot task. One might wonder if these ablations cause interference in the model's "internal reasoning" process—even though 4o isn't ostensibly a reasoning model, it seems plausible that it has been context-distilled, reinforced, and/or fine-tuned to internalize a form of base reasoning (although likely short of full CoT as in the full-fledged reasoning models). Interfering with that internalized reasoning capability leads to severe degradation in performance.


Chain of Thought Ablation Study Results
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
0.9333
0.8400
0.5067
0.5200
Original CoT
Equation Only
Variable Compute
CoT After Answer