Homework 5: Let's Play LLM

Part 1: Designing a Task for LLM and Explore the Capability of LLM (70%)

Task Description (10%)

In this task, I want to test an LLM's ability to admit that it does not know the correct answer or there is no answer for the question. Hence I will ask two questions in my input prompt. One question does not have an answer, while the other has a definitive correct answer. The ideal output for the first question is that LLM acknowledges that it does not know or indicate that there is no answer. As for the second question, it should give the correct answer as expected.

Motivation (10%)

When I utilize LLM to find answers to academic problems, I often encounter incorrect responses, which are hard to identify the correctness, especially when I lack the relevant knowledge. Therefore, I want to improve the ability of LLM to admit that it does not know the answer or there is no answer for the question so as to increase its trustworthiness.

Describe your Initial Attempt (15%)

Input:

Try to answer following question:
1. What is the prefix name in English of 10^-4 in the metric system?
2. What is the formula of Taylor series? Please explain it.

Output:

1. The prefix name in English for 10^{-4} in the metric system is "centi-".
2. The Taylor series expansion of a function $f(x)$ around a point a is a representation of $f(x)$ as an infinite sum of terms involving the function's derivatives evaluated at a. The general formula for the Taylor series expansion is:

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots$$

Here's a breakdown of each component:

- $f(x)$ is the original function.
- a is the point around which the series is being expanded.
- $f(a)$ represents the value of the function at the point a.
- $f'(a)$ is the first derivative of $f(x)$ evaluated at a.
- $f''(a)$ is the second derivative of $f(x)$ evaluated at a.
- Similarly, $f'''(a)$ represents the third derivative evaluated at a, and so on.
- $n!$ represents the factorial of n, which is the product of all positive integers up to n.

Each term in the series represents the contribution of higher-order derivatives to the behavior of the function around the point a. The Taylor series allows us to approximate $f(x)$ by truncating the series after a finite number of terms, providing increasingly accurate approximations as more terms are included. This expansion is particularly useful for approximating functions and solving differential equations in mathematics, physics, and engineering.

Analysis:

Name	The Number	Prefix	Symbol
trillion	1,000,000,000,000	tera	T
billion	1,000,000,000	giga	G
million	1,000,000	mega	M
thousand	1,000	kilo	k
hundred	100	hecto	h
ten	10	deka	da
unit	**1**		
tenth	0.1	deci	d
hundredth	0.01	centi	c
thousandth	0.001	milli	m
millionth	0.000 001	micro	µ
billionth	0.000 000 001	nano	n
trillionth	0.000 000 000 001	pico	p

First of all, let us analyze the first question. In the picture above, we can observe that prefix for 10^{-4} does not exist. Logically, ChatGPT should have informed us that this prefix does not exist. However, it told us that the prefix name in English for 10^{-4} in the metric system is "centi-". Obviously, not only did it fail to admit that it didn't have an answer, but it also provided incorrect information.

As for second question, we can find that the formula and the explanation of Taylor series are correct.

Describe Improvements (35%)

Now, we try to use some prompts to improve this condition.

Method 1: Ask ChatGPT to check whether the answer it generated is correct or not.

Input:

(1)

> Are you sure the answer of question 1 is correct?

(2)

> Are you sure the answer of question 2 is correct?

Output:

(1)

> I apologize for the confusion. In the metric system, 10^{-4} is referred to as a specific prefix.
>
> 1. The name in English of 10^{-4} in the metric system is "ten milligrams" (as a mass unit example) or "decigram" (as a length unit example). The prefix for 10^{-4} is not standard, but we can use the closer term:
>   - 10^{-3} is "milli" (1/1000 or 0.001)
>   - 10^{-6} is "micro" (1/1,000,000 or 0.000001)
>
> However, for 10^{-4}, the closest appropriate prefix is "tenths of milli" or "hundredths of centi" because it is between the common metric prefixes "milli" and "micro." Therefore, 10^{-4} in terms of length could be "100 micrometers" or "0.1 millimeters" (depending on the context).
>
> If we follow the standard naming conventions strictly, 10^{-4} doesn't have a specific prefix and can be expressed as "one ten-thousandth."

(2)

Yes, the formula for the Taylor series is correct. However, I can provide a more formal expression for clarity.

The Taylor series expansion of a function $f(x)$ around a point a is given by:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Where:

- $f^{(n)}(a)$ denotes the n-th derivative of f evaluated at a,
- $n!$ denotes n factorial,
- $(x-a)^n$ represents the n-th power of $(x-a)$.

In expanded form, this is:

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots$$

This series continues indefinitely and provides an approximation of the function $f(x)$ near the point a.

Analysis:

In the following two outputs, we can find that the two different ways ChatGPT responsed. In response 1, ChatGPT apologized and admit that 10^{-4} does not have a specific prefix in metric system, which is a correct answer we want.

As for response 2, it can be observed that ChatGPT did not apologize; instead, it was very confidence in its answer (since it answered that “YES, the formula for the Taylor series is correct.”) and even added more details which it did not mention in the first time response.

In conclusion, this method can improve the ability of LLM to admit that there is no answer for the question and even make it more confident to the correct answer.

Method 2: Clearly tell ChatGPT that if it does not know the answer or the answer does not exist, it must respond truthfully.

Input:

Please answer the following question. If you do not know the answer or the answer does not exist, please directly tell me.
1. What is the name in English of 10^-4 in the metric system?
2. What is the formula of Taylor series?

Output:

1. The name in English of 10^{-4} in the metric system is "ten-thousandth" or "0.0001". However, in terms of metric prefixes, 10^{-4} is not directly represented by a standard metric prefix. The closest standard prefixes are "milli-" (10^{-3}) and "micro-" (10^{-6}).

2. The formula for the Taylor series of a function $f(x)$ around a point a is given by:

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots$$

or more compactly,

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

where $f^{(n)}(a)$ represents the nth derivative of f evaluated at a, and $n!$ denotes n factorial.

Analysis:

At the beginning, let us explore the response to question 1. Since ChatGPT will generate different responses even with the same prompt, we've observed that most of times it still provides wrong answers. After trying five times, ChatGPT finally told me that "10^{-4} is not directly by a standard metric prefix" instead of telling me other wrong prefix.

As for response to question 2, it still answers correctly without being affected by the modified prompt.

In conclusion, method 2 can make ChatGPT admit that there is no answer for the question. However, its effect is not noticeable. We will still get the incorrect responses in most of times.

Part 2: Problem Answering (30%)

Question 1

(1) Besides providing a more detailed description of the question in the prompt, what other methods can be employed to improve the accuracy of the responses?

<ans>

We can try to assign a role to LLM, which can help guide the model's responses and ensure it align with the desired expertise or perspective. The role we assign should be an expert in the field related to the problem we want to ask. In this way, LLM will response more detail and explain more comprehensively.

(2) Please provide an example demonstrating the impact of implementing the aforementioned methods on a prompt

<ans>
Original input:

What is dynamic programming?

Original output:

Dynamic programming is a method used in computer science and mathematics to solve optimization problems by breaking them down into simpler subproblems. It is particularly useful when the problem can be divided into overlapping subproblems, as it avoids redundant computations by storing the results of subproblems and reusing them when needed.

The basic idea behind dynamic programming is to solve each subproblem only once and then store its solution in a table, array, or memoization cache. This way, when the solution to a subproblem is needed again, it can be retrieved from memory rather than recomputed. This technique can significantly improve the efficiency of algorithms, especially for problems with exponential time complexity if solved naively.

Dynamic programming can be applied to a wide range of problems, including those in various fields such as computer science, operations research, economics, and biology. Classic examples of problems that can be solved using dynamic programming include the knapsack problem, the longest common subsequence problem, and the Fibonacci sequence.

→ ChatGPT only answers the overall concept and does not discuss how to implement it. Additionally, as for some technical terms, it does not explain it. It is hard for people who are not major in related field to understand.

Improved input: (assign a role to ChatGPT)

You are now a senior engineer familiar with every data structures and algorithms. Please tell me what the dynamic programming is.

Improved output:

Dynamic programming is a powerful technique used to solve optimization problems by breaking them down into simpler subproblems and efficiently solving each subproblem only once. It is particularly effective when the problem exhibits overlapping subproblems and optimal substructure.

Here's a more technical explanation:

1. **Optimal Substructure**: A problem exhibits optimal substructure if an optimal solution to the problem can be constructed from optimal solutions to its subproblems. In other words, if you can solve the smaller subproblems optimally, you can combine these solutions to find the optimal solution to the larger problem.

2. **Overlapping Subproblems**: In dynamic programming, overlapping subproblems occur when the same subproblems are encountered multiple times during the computation. Rather than solving them repeatedly, dynamic programming stores the solutions to subproblems in a data structure (often an array or a table) so that they can be reused when needed.

Dynamic programming typically involves following a sequence of steps:

1. **Identify the Subproblems:** Decompose the problem into smaller subproblems. These subproblems should be similar to the original problem but simpler in nature.
2. **Define the Recurrence Relation:** Express the solution to a larger problem in terms of solutions to its subproblems. This recurrence relation represents the relationship between the current state and its subproblems.
3. **Solve the Subproblems:** Determine the base cases and solve the subproblems iteratively or recursively, storing the solutions in a data structure (e.g., an array or a table).
4. **Combine Subproblem Solutions:** Use the solutions to the smaller subproblems to construct the solution to the larger problem, adhering to the optimal substructure property.
5. **Memoization or Tabulation:** Implement memoization (top-down approach) or tabulation (bottom-up approach) to store and reuse the solutions to subproblems efficiently.

Dynamic programming can be applied to a wide range of problems, including those in algorithms, optimization, and artificial intelligence. Classic examples include the knapsack problem, shortest path problems, and sequence alignment problems. By efficiently solving subproblems and avoiding redundant computations, dynamic programming often leads to significant improvements in algorithm efficiency.

→ We can find that ChatGPT responses better after assigning a role to it. Not only does it explain the terminology but it also tells us the step of implementation. In my opinion, the overall performance has improved.

Question 2

While using LLM, you probably find that it cannot accurately answer your queries. This indicates that current large language models (LLMs) still face numerous challenges. Please search online (papers, blogs, etc.) for the existing challenges of LLMs and briefly explain the motivation, difficulties, and current methods available to optimize or solve these issues.

<ans>

- Question: The data may be ingrained with biases along with misinformation, which can lead to the LLM's outputs reflecting bias.
- Motivation: Bias that LLM generates poses a negative impact on many aspects, including reinforcement of stereotype, deeper discrimination, and misinformation. However, it is impossible to eliminate all the bias since the dataset it used must contain some. Besides, the process is a black box, which means that we do not know how it extracts biases and add it to the output.
- Challenges:
    - ✧ How to mitigate LLM biases?

- Will the accuracy of response be affected?
- Will the information LLM provides decrease noticeably since it cannot respond some biased-related?

- Solution:
  - Data curation: Using curated datasets that have come from different demographics, languages, and cultures. In this way, it can decrease LLM response the stereotype in specific group.
  - Pre-training mitigations: Train another model for the purpose of filtering improper responses. In reality, OpenAI now uses it for DALL-E 2 to filter out violence and sex. Hence, it is possible to try using pre-trained mitigations to decrease the times biases appear.
  - Logic in addressing LLM bias: A study from MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL) has made significant advancements in LLMs by integrating logical reasoning. The process consists of building a neutral language model in which the relationships between tokens are considered 'neutral' as there is no logic stating that there is a relationship between the two. In this way, we do not need more data and can still achieve our goal, mitigating biases.

- Reference:

https://www.datacamp.com/blog/understanding-and-mitigating-bias-in-large-language-models-llms