Counterfactual Reasoning Capacity of Large Language Models

By

Sushovit Nanda

Dept. of Electrical Engineering and Computer Science (EECS)

Indian Institute of Science Education and Research Bhopal

Under the Guidance of

Dr. Shreya Ghosh

Assistant Professor

Department of Computer Science and Engineering

School of Electrical and Computer Sciences (SECS)

Indian Institute of Technology Bhubaneswar

Abstract

Suppose we have a reasoning question, such as a verbal or logical reasoning question. As humans, we can find the correct answer to that question with a supporting explanation or context which helps us in finding the solution. In this internship project, we are interested in understanding how large language models (LLMs) are learning to answer such questions and what are their reasoning capacity. For this, we prompt few LLMs with questions from a dataset with certain prompting techniques such as zero-shot and few-shot prompting, and check how the LLMs perform in giving the answers. We repeat this first without any form of training the model and then with a trained model to see how it affects the performance. We also find out the difference in performance among different types of reasoning questions primarily in comparison to analogical reasoning type questions. The results show that LLMs exhibit very low accuracy at this task of question-answering, depending on the specific model and testing dataset. Smaller models such as those with a lesser number of parameters have been used for this internship project due to no availability of additional resources.

Contents

1. Introduction

Reasoning is a fundamental cognitive process that allows individuals to draw conclusions and make decisions based on information. It has been classified into various types by renowned psychologists, including deductive, inductive, analogical, and counterfactual reasoning. Among these, counterfactual reasoning is often considered the most challenging due to its hypothetical nature, which involves considering alternative scenarios and their possible outcomes. This type of reasoning requires a deeper level of abstraction and the ability to think beyond the present reality, making it difficult to comprehend and accurately predict.

In recent years, advancements in natural language processing (NLP) have led to significant improvements in the performance of large language models (LLMs) on reasoning tasks [1]. Notably, the ChatGPT-4 model from OpenAI has demonstrated the highest accuracy in handling complex reasoning questions, including counterfactual scenarios. This model's ability to generate coherent and contextually appropriate responses has set a new benchmark in the field.

For this project, I utilized the base models and various fine-tuned versions of the Llama2-13b and Llama3-8b models developed by Meta, accessed through the huggingface repository [2]. These models were selected because these are traditionally text-generation models that can also be used for question-answering based on the prompt given to them. These models also have a lower resource requirement they have for being smaller models. The software used for this project is the free version of Google Colab notebook with the T4 GPU it provides along with System RAM of 11.4 GB, GPU RAM of 15.0 GB, and Disk Space of 50.9 GB.

To assess the accuracy of these models on the testing dataset, we employ both ROUGE scores and human evaluation, with a greater emphasis on the latter. while ROUGE scores [3] offer valuable insights into the lexical and semantic alignment of responses, they may not fully capture the nuances of reasoning and the depth of understanding demonstrated by the models.

2. Methods for Large Language Model Experiments

With a large counterfactual dataset as input, this project aimed to test and fine-tune Large Language Models to evaluate counterfactual explanations automatically. The models selected for this were Llama2-13b-hf, Llama2-13b-chat-hf, Llama2-13b-chat, Llama3-8b, Llama3-8b-Instruct; and the models were accessed and fine-tuned using the transformers library [4] by huggingface.

2.1 Large Language Models

Language models are a type of machine learning model that can process and generate text. Large Language Models (LLMs) are a new paradigm in this field, as they are many times larger than traditional models and can generate fluent and coherent text in many languages.

Llama 2 is an open-access Large Language Model (LLM) family developed by Meta in 2023 as a successor to its previous Llama 1 model. It consists of models ranging from 7 billion parameters to 70 billion parameters. Parameters refer to the tunable weights present in a neural network and are often used as a rough estimate of a model's performance and potential. Llama 2 is one of the most widely used LLMs, since it is free for both research and commercial and strikes a good balance between performance and size [5]. In April of 2024, Meta released the successor to Llama 2, Llama 3 [6]. Similarly to Llama 2, Llama 3 contained models with 8 billion parameters and 70 billion parameters. A significantly better Llama3.1 has been released (on 24th July, 2024) ranging up to 405b parameters but as of the writing of this document, those models haven't been worked upon. Meta claimed that despite the Llama 3 models' similar sizes to Llama 2, the new models are trained on 7 times more data than Llama 2

In this project, both the older Llama 2 13 billion parameter model (13B) and the newly released Llama 3 8 billion parameter model (8B) were used and compared. For both Llama 2 and Llama 3 models, a version that is fine-tuned for instructions was used instead of the base model, as specified by the suffix "Chat", "hf" or "Instruct".

For the experiments in this project, all of the abovementioned models were sourced from huggingface's repositories through the Transformers library [4].

2.2 Efficient fine-tuning of LLMs

As models based on neural networks, LLMs consist of billions of tunable parameters, also known as weights. During the training process, these parameters are adjusted to achieve optimal predictive power on the training data. Fine-tuning is the process of taking a pre-trained model and further optimizing these parameters for specific tasks. However, fine-tuning LLMs is computationally intensive due to the need to update billions of parameters, requiring substantial memory and processing power. For instance, a model with 13 billion parameters may need over 26 GB of memory for storage and additional memory will be required for inference. Thus for Google Colab with limited amount of space provided for free, the model size needs to be reduced without affecting the efficiency by a lot.

Low-Rank Adaptation (LoRA) [7] is a technique used to reduce the memory requirements for fine-tuning LLMs by "freezing" the original weights and adding a set of lower-rank weights that are specifically fine-tuned. A more advanced method, QLoRA [8], enhances this approach by incorporating quantization, which reduces the precision of parameters to save memory. Despite these techniques, fine-tuning an LLM still requires considerable GPU resources.

3. Experiments

Using the methods described above in previous sections, four experiments were conducted. As a prerequisite, the dataset for counterfactual reasoning was searched for in Google. Subsequently, the goal of the first experiment was to compare the effects of different LLM system prompts and its parameters on the accuracy of the counterfactual response. The second experiment was used to compare the answers and accuracy of the given questions without any form of training the model and with zero shot prompting, among all 5 different models. This also gives us an overview of which model performs better over others and in which tasks. The third experiment is aimed at finetuning the LLM with trial and testing of various parameters as supported by the Colab notebook. Finally, in the fourth experiment, we evaluate the best model using few-shot prompting. The codes and datasets used for the experiments in this project can be found in GitHub.

Dataset

In order to fine-tune Large Language Models, the data needs to be presented in a format specific to the LLM in question. Generally, an instance of training data contains three things: a system prompt, an instruction, and an output. A system prompt gives the LLM general information on how to approach any tasks it is presented with. For example, a system prompt may contain information such as “you are an AI language model” or “avoid giving false or misleading information”. An instruction usually contains the task that the model is meant to perform.

For this project text generation has been used to have the LLM answer the question, similar to how a human would. Alternatively, the problem could be considered as a classification task, and the LLM could be used to classify text into different classes, with different classes corresponding to high, medium or low values in this case. Both approaches have their merits and problems, but due to the nature of the available dataset, and the experience of the intern, text generation was considered the superior option and used for this project.

The dataset used for the project is the IFQA dataset [9]. IfQA is an open-domain question-answering dataset where each question is based on a counterfactual presupposition via an ``if'' clause. For example, if Los Angeles was on the east coast of the U.S., what would be the time difference between Los Angeles and Paris? Such questions require models to go beyond retrieving direct factual knowledge from the Web: they must identify the right information to retrieve and reason about an imagined situation that may even go against the facts built into their parameters. The IfQA dataset contains 3,800 questions annotated by crowdworkers. We hope the unique challenges posed by IfQA will push open-domain. QA research on both retrieval and reasoning fronts, while also helping endow counterfactual reasoning abilities to today's language understanding models. This dataset has been uploaded to the GitHub repository too.

3.1 Experiment I: System Prompt comparison without fine-tuning

The first experiment consisted of comparing 2 formulated basic system prompts on their accuracy. To comprehensively compare the system prompts and their performance, Llama2-13b-chat model has been used. The model and system prompts were tested on 2 counterfactual questions from the dataset and their responses were compared. We executed the code multiple times, observing variations in the outputs generated during each run due to certain parameters. Following cases below demonstrate how differing system prompts generate differing responses by the LLM:

- Ques1: 'If sea levels had risen significantly over the past decade, which country would have been the first to be submerged?'

```
prompt = Ques1
prompt_template=f'''SYSTEM: You are a helpful, respectful and honest assistant.
Always answer in very brief with simple 2 lines of explanation.

USER: {prompt}

ASSISTANT:
'''
response=lcpp_llm(prompt=prompt_template, max_tokens=256, temperature=0.5,
top_p=0.95,
                  repeat_penalty=1.2, top_k=150,
                  echo=False)
print(response["choices"][0]["text"])
```

If sea levels had risen significantly over the past decade, Bangladesh would likely have been one of the first countries to be submerged due to its low elevation and exposure to tropical cyclones.

OR

1) Kiribati
2) Tuvalu
3) Maldives
4) Bangladesh
5) None of the above

The model provides different answers each time the code is executed thus demonstrating that there is no uniformity in the answers. This is so because the "temperature" parameter has a high value of 0.5 to it which has increased the randomness of the generated texts. It influences the model's likelihood of selecting less probable words when generating a response. A lower temperature results in more deterministic and conservative outputs, making the model more likely to choose words with higher probabilities. Conversely, a higher temperature allows for more diversity and creativity, increasing the chance of selecting less common words.

- ```
  response=lcpp_llm(prompt=prompt_template, max_tokens=256, temperature=0,
  top_p=0.95,
                    repeat_penalty=1.2, top_k=150,
                    echo=False)
  print(response["choices"][0]["text"])
  ```
  If sea levels had risen significantly over the past decade, the country that would have been the first to be submerged is likely to be one of the low-lying coastal areas such as the Maldives or Tuvalu. These countries are already experiencing frequent flooding and erosion due to rising sea levels caused by climate change.

Maldives is the correct answer as per the dataset. Thus with a lower temperature, we get a more deterministic answer that doesn't change even after repeated execution of the code.

- Ques2 = 'If Leonardo da Vinci had not passed away and was still alive, how old would he have been in the year 2020?'
  ```

```
prompt = Ques2
prompt_template=f'''SYSTEM: You are a helpful, respectful and honest assistant. Always answer in very brief with simple 2 lines of explanation.

USER: {prompt}

ASSISTANT:
'''
response=lcpp_llm(prompt=prompt_template, max_tokens=256, temperature=0, top_p=0.95,
          repeat_penalty=1.2, top_k=150,
          echo=False)
print(response["choices"][0]["text"])
```

Leonardo da Vinci passed away on May 2, 1519 at the age of 67. If he were still alive today (2020), he would be approximately 543 years old.

- ```
  prompt = Ques2
  prompt_template=f'''SYSTEM: You are a helpful, respectful and honest assistant. Always make your answers to be factually coherent.

  USER: {prompt}

  ASSISTANT:
  '''
  response=lcpp_llm(prompt=prompt_template, max_tokens=256, temperature=0, top_p=0.95,
            repeat_penalty=1.2, top_k=150,
            echo=False)
  print(response["choices"][0]["text"])
  ```

  Leonardo da Vinci was born on April 15th, 1452. Therefore, if he were still alive today (in 2020), he would be approximately 568 years old. However, it's important to note that this is purely hypothetical as Leonardo da Vinci passed away on May 2nd, 1519 at the age of 67.

The correct answer is 568 years old. We can see that despite the temperature being 0 in both cases, it is the system prompt here that made the difference. In case 3 the model went by the death anniversary of Da Vinci and gave an incorrect answer but when the prompt was changed, it switched its approach and went by the birth anniversary. Also the length and semantic meaning of the response changed as well due to changing the system prompt.

Thus this experiment is able to prove satisfactorily how system prompts and their parameters affect the responses generated by any LLM. Thus prompt engineering and parameter finetuning plays an important role in improving the efficiency of any LLM.
  ```

3.2 Experiment III: Comparing the 5 LLMs without fine-tuning

The second experiment was designed to test and accuracy of the 5 Meta-Llama models and compare their performance with each other on certain metrics and find out the best performing model of them for further experimentation. The specific metrics used to assess performance includes various rouge scores such as Rouge1, Rouge2, and RougeL scores [3] along with human evaluation done by me on the counterfactual reasoning dataset.

ROUGE score metric is particularly useful for evaluating the fluency and relevance of the generated text by comparing the overlap of n-grams, word sequences, and word pairs; but it is not a robust metric as there are multiple ways to generate and formulate the sentences of an answer. In such a scenario human evaluation becomes crucial. Human evaluators can assess the quality of the answers in a more comprehensive manner, considering factors such as the correctness of the reasoning process, contextual appropriateness, and the coherence of the response.

Based on the evaluation matrix for the 5 models shown in the tables below we get an idea of each model's performance. In Table 1. we have the model performance without any context in the prompt whereas in Table 2. We have context given to the LLM within the prompt.

Model	Rouge-1 F1 Score	Rouge-2 F1 Score	Rouge-L F1 Score	Human Evaluation
Llama2-13b-Chat	0.24	0.07	0.15	0.46
Llama2-13b-hf	0.13	0.02	0.09	0.49
Llama2-13b-Chat-hf	0.28	0.08	0.17	0.64
Llama3-8b	0.20	0.04	0.12	0.52
Llama3-8b-Instruct	0.27	0.07	0.16	0.62

Table 1. Evaluation Matrix for LLM without context in prompt.

Model	Rouge-1 F1 Score	Rouge-2 F1 Score	Rouge-L F1 Score	Human Evaluation
Llama2-13b-Chat	0.24	0.11	0.19	0.57
Llama2-13b-hf	0.20	0.03	0.14	0.64
Llama2-13b-Chat-hf	0.28	0.08	0.17	0.72
Llama3-8b	0.13	0.03	0.09	0.62
Llama3-8b-Instruct	0.29	0.09	0.18	0.8

Table 2. Evaluation Matrix for LLM with context in prompt.

We observe a dip in Rouge scores for the Llama3-8b model as the base model without an form of finetuning generates more sentences with the same system prompt as mentioned above. This creates a significant increase in unigrams, resulting in more possible combinations that don’t match the answer. For scenarios like these, human evaluation is the more trusted performance metric. These human evaluation scores pretty much match the evaluation shown by Meta in their model card.

This experiment also proves that context related to the given question being provided to the LLM, results in a significant increase in the performance. Because of these contexts and additional information, the LLM can draw some inferences regarding the counterfactual question.

With this experiment we can conclude that the Llama3-8b-Instruct model is the best model out of all the 5 models in this project. This conclusion is also verified by how this model performs on the analogical QA GSM8K dataset [10]. On the GSM8K dataset the Llama3-8b model shows an accuracy of 79.6%.and upon experimentation myself the Instruct model shows an accuracy close to 86% which further proves that's the 8b-Instruct model is the best out of all its predecessors in terms of small LLM.

3.3 Experiment III: Effects of fine-tuning LLMs on evaluation accuracy

The third experiment was designed to test the efficacy and expedience of fine-tuning LLMs to evaluate counterfactual explanations. The optimal hyperparameters for every model were discerned through extensive testing and can be viewed in Table 3. All models were fine-tuned using a completion-only data collator from Hugginface's trl library [11]. This means that the models were only fine-tuned to predict the answers to the questions, not the text of the questions themselves. This type of data collator was chosen to focus on improving the predictive performance of the models. With a typical language modeling data collator, the model would have learned to predict the question text as well, but this was unnecessary for the task at hand.

Models	General			LORA	
	Batch Size	**Learning Rate**	**Epoch**	**r**	**alpha**
Llama2-13b-Chat	4	2e-4	4	64	16
Llama2-13b-hf	4	2e-4	4	64	16
Llama2-13b-Chat-hf	4	2e-4	4	64	16
Llama3-8b	4	2e-4	4	64	16
Llama3-8b-Instruct	4	2e-4	4	64	16

Table 3. Hyperparameter values used for fine-tuning models

All the other values for LORA [7] peft configuration and training arguments for the SFTT Trainer have been upload in the GitHub repository for this project.

The results for the Llama3-8b-Instruct model are as follows in Table 4.

Steps	Training Loss
25	1.942400
50	1.504600
75	1.384800
100	1.332100

Table 4. SFTT Train output

Global Steps of 100, metrics = { 'train_runtime': 487.2255, 'train_samples_per_second': 0.821, 'train_steps_per_second': 0.205, 'total_flos': 3037522237194240.0, 'train_loss': 1.5409628677368163}. Upon texting the trained model with the same dataset we experience an accuracy of 90% through the metric of human evaluation. The ROUGE score on the other hand diminishes close to zero as now the answers given by the model are very precise or within few words or no answer is given at all.

The third experiment convincingly showed that relatively small LLMs can be trained to be as capable as larger pre-trained models, even with a very limited dataset, such as the one used in this project. The most accurate model in this project being the Llama3-8b-Instruct model which reached 90% accuracy after finetuning could perform on par with the larger models from the same family without any finetuning. This is a promising result, since smaller LLMs are far superior in terms of both speed and resource requirements, enabling them to be used in wider contexts.

3.4 Experiment IV: Comparison between “Zero-Shot” and “Few-Shot”

The results of this experiment showed no consistent differences between the answers given by the LLMs. The lowest average accuracy of 41% was seen for the prompt with examples, with the other two prompts achieving similar average accuracy scores of around 50-60%. This result is somewhat unexpected since it appears to suggest that “zero-shot” learning achieves better results than “few-shot” learning for this task, which contradicts the common view that "few-shot" learning is superior

This discrepancy between the results of the experiment and generally accepted truths likely stems from the following possible cause. The cause is that the examples chosen for the prompt were simply not representative of the overall data and therefore did not provide useful new information to the model. Counterfactual questions range from questions that already have an answer or suggestion available in the free internet as well as those that don’t have an answer and a deeper inference of the context and available true data is needed for the model to arrive at the right decision. Certain hypothetical mathematical questions also exist within the dataset in which case it is virtually impossible to include few examples in the prompt for “few-shot” learning which encompasses all possible scenarios of counterfactuals. Thus including “few-shot” examples in the prompt may very well include bias in the LLM further deteriorating the performance.

Overall, larger LLMs seem to be significantly more capable of evaluating the quality of counterfactual explanations by default. Llama2-13b model gives a comparable performance that of Llama3-8b model which suggests that despite Llama3 being more advanced than Llama2 due to being trained on more data, the size of the model plays a significant role in the model’s capacity to understand and evaluate counterfactual explanations. Thus in this case, the additional 5b parameters that the Llama2-13b model has over the Llama3-8b model can bridge the gap between the two. This also suggests that in addition to scale, advancements in model architecture and training data can have a significant impact as well, making small and fast models increasingly more viable in the near future.

4. Limitations

Due to limited resources and time availability in the working of this project, several compromises were to be made.

Firstly, the dataset used for the entirety of this project was sourced from public domain. The IFQA dataset [9] was hand-crafted collectively by the crowdworkers and since there is a human component to it, there are several errors and incomplete sentences present in the context part of each dataset. These incomplete sentences might have interfered with the tokenization by the models.

Secondly, the small LLMs were used for the project due to the low availability of software. The base Llama2-13b model and Llama2-13b-Chat version doesn't have a .config file in their data card in the huggingface repo. There are no available quantized model for the base Llama2-13b model in the huggingface repo thus it wasn't included in this project. The quantized model of Llama2-13b-Chat-ggml was used from the huggingface using the llama-cpp-python library [12] and a publicly available quantized model was used for the experiments. A separate code is posted in the GitHub repo, related to the public available quantized Llama2-13b-Chat-ggml model.

Thirdly, the human evaluation has been done by only 1 person in this project. A scale of 0-5 was used where 5 represents the exact perfect answer and 1 represents a complete incorrect answer. 0 is given when the model in unable to answer any particular question and gives filler responses such as – "please note that this is speculative as there is no concrete evidence to support this hypothetical scenario." , "I strive to provide accurate and reliable information, but if you find anything incorrect, please let me know so I can correct it." , etc. The points in between are given to a model answer based on how close or off they are to the true answer. Also the entire context of the answer is taken into account when scoring points.

Despite these limitations, the experiments performed in this project showed potential and suggested that fine-tuning Large Language Models may be a viable strategy for automating the evaluation of counterfactual explanations.

5. Conclusion

The results in this project showed that large LLMs are capable of evaluating counterfactual explanations. The smaller models can be fine-tuned to achieve desirable performance. Fine-tuning large models results in accuracies of 80% to 90%, which is sufficient for many real-world applications. These results lay the groundwork for further experiments and research, both in further improving automated evaluation of counterfactual explanations and in improving existing explanation generation algorithms with LLM integration.

References

[1] F. Z. V. C. R. S. M. F. D. S. H. &. E.-A. M. Cheng, “Interactive Analysis of LLMs using Meaningful Counterfactuals.,” *arXiv preprint arXiv:2405.00708.,* 2024.

[2] S. Mandvikar, “Factors to consider when selecting a large language model: A comparative analysis,” *International Journal of Intelligent Automation and Computing,* vol. 6, no. 3, pp. 37-40, 2023.

[3] M. Grusky, “Rogue scores. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics,” vol. 1, pp. 1914-1934, july 2023.

[4] [Online]. Available: https://huggingface.co/transformers/v3.0.2/main_classes/pipelines.html.

[5] [Online]. Available: https://ai.meta.com/blog/meta-llama-3/.

[6] [Online]. Available: https://ai.meta.com/blog/meta-llama-3/.

[7] X. A. L. &. R. M. Wang, “LoRA ensembles for large language model fine-tuning.,” *arXiv preprint arXiv:2310.00035.,* 2023.

[8] T. P. A. H. A. &. Z. L. Dettmers, “ Qlora: Efficient finetuning of quantized llms. Advances in Neural Information Processing Systems,” p. 36, 2024.

[9] W. a. J. M. a. C. P. a. S. A. Yu, “IfQA: A Dataset for Open-domain Question Answering under Counterfactual Presuppositions,” 2023.

[10] K. a. K. V. a. B. M. a. C. M. a. J. H. a. K. L. a. P. M. a. T. J. a. H. J. a. N. R. a. H. C. a. S. J. Cobbe, “Training Verifiers to Solve Math Word Problems,” *arXiv preprint arXiv:2110.14168,* 2021.

[11] Y. B. L. T. E. B. T. T. N. L. a. S. H. L. v. Werra, “TRL: Transformer Reinforcement Learning,” 2020.

[12] [Online]. Available: https://pypi.org/project/llama-cpp-python/0.1.9/.