FRI Natural language processing course 2024
Project report

University *of Ljubljana*
Faculty *of Computer and Information Science*

LLM Prompt Strategies for Commonsense-Reasoning Tasks

Žan Počkar, Amer Mujagić, and Ivan Nikolov

Abstract

Large Language Models (LLMs) have made significant advancements through extensive training, yet enhancing their commonsense reasoning remains challenging. This report evaluates four prompting strategies for their effectiveness in improving LLMs' commonsense reasoning using a subset of the aNLI, Physical IQa, and WinoGrande datasets. Our findings show that Few-Shot prompting significantly outperforms other methods, achieving over 20% higher accuracy on the aNLI dataset. Conversely, Chain of Thought with Hints underperforms compared to zero-shot prompting. These results highlight the need for more sophisticated evaluation metrics and the potential of structured prompting techniques to enhance LLM performance on commonsense tasks. Future work will refine these techniques and expand evaluation to additional models and strategies.

Keywords
Prompt Strategies, Prompt Evaluation

Advisors: Aleš Žagar

Introduction

Large Language Models (LLMs) have demonstrated remarkable progress in various tasks when provided with extensive training material. However, enhancing their commonsense reasoning abilities remains a significant challenge. Commonsense reasoning, which entails making judgments based on everyday knowledge and experiences, is an essential aspect of human intelligence, and incorporating this capability into LLMs is a complex endeavor.

In this paper, we compare four prompting strategies using the same problem set and model to evaluate their performance. Our flexible evaluation framework is designed to accommodate multiple models and strategies in the future and serves as a simple blueprint for adding similar strategies. We employed a subset of aNLI, Physical IQa, and WinoGrande datasets, inspired by the Rainbow dataset, using the first 300 questions as a proof of concept and a good general idea of their comparative performance. The primary objective of this paper is to demonstrate the effectiveness of our evaluation framework and provide a roadmap for comparing different prompt strategies on various LLMs.

Problem

Commonsense reasoning

Commonsense reasoning in Large Language Models (LLMs) is a complex problem due to the implicit and context-dependent nature of commonsense knowledge. For instance, consider the sentence, "The ice cream was too hot to eat." Humans intuitively understand this sentence is likely incorrect because ice cream is typically cold. However, an LLM might not flag this as an anomaly unless it has been explicitly trained on similar examples or has learned to associate certain objects with their typical properties.

Prompt generation

The idea of generating prompts from prompts seems straightforward, and for the most part, depending on the technique, it can be. However, comparing techniques based on these prompts remains a challenge. It is difficult to quantify the impact of prompt modifications on complexity or simplicity when adapting them to fit specific techniques. This issue may result in a biased comparison between techniques. We tried to mitigate this problem by selecting diverse datasets, but it was challenging to quantify the impact in a meaningful way. As a result, our findings should be interpreted while considering that changing the prompts might have simplified them, potentially biasing the comparison between techniques. This is particularly relevant for Chain of Thought with hints, where automatically generated hints were often extremely directly connected to the answer.

Prompting Techniques

Prompts techniques are the proposed input structures that guide LLMs to produce outputs with the desired form and content quality. That is, to simplify, we can say that we use different prompting techniques to ”program” LLM inputs by following specific patterns in order to obtain an output. [1]

Few-shot and zero-shot Prompting

Few-shot prompting [2] is a technique that involves providing a prompt along with a limited number of demonstrations to help guide the Large Language Model (LLM) in understanding and performing a specific task. In contrast to zero-shot prompting, where no demonstrations are used, few-shot prompting allows for a more nuanced understanding by providing examples that showcase the desired output.

As illustrated in Figure 1, few-shot prompting lies between zero-shot and one-shot prompting in terms of the number of demonstrations provided. By including examples, the LLM can better grasp the context and the expected results. However, the challenge lies in balancing the complexity of the prompt and the number of demonstrations provided. Too few examples might not provide sufficient guidance, while too many may overfit the LLM to the specific examples and hinder generalization. Overall, few-shot prompting offers a flexible approach to enhancing LLM performance by combining the simplicity of natural language instructions with the additional context provided by demonstrations.

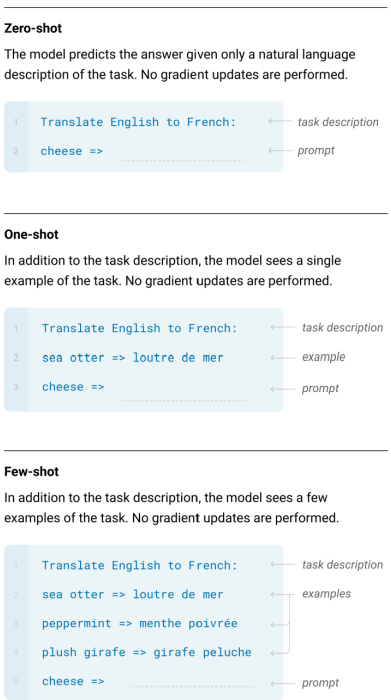

Figure 1. Zero-shot, one-shot, and few-shot prompts

Chain of Thought

Chain of Thought (CoT) [3] prompting involves adding a series of intermediate reasoning steps to the intended task. It has been shown to improve performance in more complex prompts. These intermediate reasoning steps can be thought of as giving an answer to a similar problem but with detailed steps on how to get to that answer as shown in Figure 1. This is similar to few-shot prompting, which also provides examples of answers to similar questions, however few-shot prompts include less detail that is they usually provide only questions and answers, no intermediate reasoning steps in between. [3] An example of Chain of Thought prompting can be seen in Figure 2

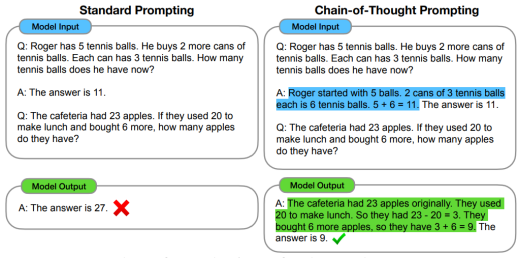

Figure 2. Example of a Chain of Thought prompt versus a regular prompt

Chain of Thought with hints

Chain of Thought with hints other than the standard thought process contains also a hint pointing to the correct answer. The hint should not be too obvious to answer the question directly (because then the prompt uses any usefulness) but not too vague so it confuses the model.

Datasets

Because commonsense reasoning is a complex topic and encompasses a wide collection of different tasks, it is difficult to generate a dataset that will evaluate the model's generalization abilities.

The Winograd schema challenge [4] was the standard for testing the commonsense reasoning, however from 2019 it is considered defeated since numerous transformer-based models achieved over 90% accuracy. [5] For that reason, we will use more recent and challenging datasets.

On the one side, we have different SOTA methods [3] evaluated on task specific datasets like GSM8K [6]. This dataset focuses on linguistically diverse grade school math problems created by human problem writers.

On the other side, there are benchmarks that combine multiple commonsense datasets. For this project we will use Rainbow which is a suite of commonsense benchmarks that contain multiple choice question-answering datasets. For our experiments, we selected the following three datasets from Rainbow

- αNLI [7] - tests abductive reasoning in narratives. Models need to find the best explanation among the presented options connecting a beginning and ending;

- Physical IQa [8] — question answering benchmark for physical commonsense.
- WinoGrande [9] — large-scale collection of Winograd schema-inspired problems that test reasoning about social and physical interactions.

Methodology

Our evaluation pipeline consists of three main components. First, we modify the datasets to obtain hints and examples for few-shot prompts. Second, we generate results using these modified prompts. Finally, we evaluate the results and compare them.

Prompt Modification

We utilized the DSPy framework's ChainOfThought and Predict modules to modify the prompts to fit a given strategy. Since the framework already contained the ChainOfThought and ChainOfThoughtWithHints modules, we decided to use them directly for prompt generation. However, our datasets lacked hints and examples for our Few-shot prompts. To generate the examples and hints, we employed the Mistral-7B model and the Predict module. We used custom signatures fine-tuned to each dataset's structure to maximize the effectiveness of the models. These signatures only explained the structure of the prompt and the expected output, without any additional information about the prompt. We made 1 signature per output type (that is examples or hint) and per module, so a total of 6 custom signatures for this part of the assignment.

Answer Generation

We once again utilized the DSPy ChainOfThought, Predict, and ChainOfThoughtWithHints modules to generate answers for the prompts using different techniques. For Few-shot prompts, we used the Predict module, incorporating the examples into the questions themselves. We created custom signatures for each dataset, as their structures varied slightly. We used one signature per module for three of the strategies and a separate signature for Few-shot prompts due to their additional examples in the questions.

Data preprocessing

Before we could compare the generated answers, we firstly need to process the data into a singular form. We agreed to use JSON format for our data, but even with the set format some prompting techniques field the format incorrectly. To prepare the data we mostly had to transform string into integers, add indexing and in the few cases we actually need to find the given answers in the raw field of the format.

Results

Evaluation

We are currently performing evaluation of the models answers with two different methods. In the first method where the model has to select between two choices and then give his answer we use a simple comparison algorithm. If the model gives multiple answers, we count up the choices and use his most often given answer as the answer we use for comparison. The comparison algorithm is then a simple checker which has the list of correct answers. The checker compares the question id and content from the correct list with those give by the model. And then prints the results.

The other comparison method is a much more involved process for when the model gives a descriptive answer. In this method, we are using an uncased BERT model to generate the embedding and calculate the cosine similarity between the correct answer and the given one. As an additional layer of grading we are comparing cosine similarity with Word2Vec model to better determine the meaning of the sentence these two scores are then combine and if the answer pass the threshold of correctness we are considering the answer as correct. This model also supports comparison of different techniques, and not just direct comparison between the result and the technique. This way we can determine which technique is closer to the right answer even if this particular answer is incorrect. Sadly, even with all the steps we took to produce reproducible results, we just couldn't get the model to give consistent results. So in the end in the name of reproducibility of results we only used the first method in our report.

Results

The results were not all as expected, while we did generally see an improvement when using prompting techniques the improvement varied very much from small to big and in a few cases even worsening performance.

The best result we got from the Winograd and the Anli datasets, where we can see improvements with using the chain of thought and the few shot prompting techniques. But we see the worst performance with the chain of thought with hints techniques performance.

While the results for Physical dataset do follow the general trend, the overall performance of the model was very bad. We still don't understand, even if we used a lot of our time to troubleshoot it

The last image clearly shows that the techniques have effect non withstanding which dataset we used, but that they are more effective for certain datasets.

Discussion

The framework serves as a valuable blueprint, even though the results might not fully represent the techniques themselves but rather showcase the DSPy framework's implementation. Despite following the official documentation, some results were weaker than anticipated. Notably, Chain of Thought, especially with hints, was expected to outperform Few-Shot examples, but it did not always yield the desired outcome, even when the generated hints were evident.

The framework's modules, while being easy to use, provided answers in a format that was difficult to parse, often

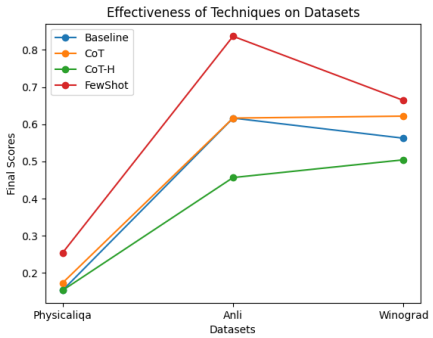

Figure 3. Technique Effectivnes across Datasets

mixing random questions and answers in the same section. This structure made it challenging to differentiate and extract relevant information automatically.

Additional challenges included the time needed to generate results and limited resources. These factors might have impacted the quality of the results. Running the experiments multiple times would help account for the inherent uncertainty of LLMs and further refine the analysis.

In conclusion, the framework demonstrates significant potential as a blueprint. Future work should focus on investigating and optimizing the effectiveness of these prompt strategies while addressing the identified issues. Using the in-build prompt optimization can also be investigated. This approach will enable a more comprehensive assessment of the framework's performance and its ability to contribute to the ongoing advancement of LLM prompting techniques.

References

[1] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt, 2023.

[2] 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. Language models are few-shot learners, 2020.

[3] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022.

[4] Hector Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012.

[5] Hector J. Levesque. On our best behaviour. *Artificial Intelligence*, 212:27–35, 2014.

[6] Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. Making language models better reasoners with step-aware verifier. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333, Toronto, Canada, July 2023. Association for Computational Linguistics.

[7] Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Yih, and Yejin Choi. Abductive commonsense reasoning. *ArXiv*, abs/1908.05739, 2019.

[8] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. 2020.

[9] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. 2020.