Outcome-Process Combined Supervision of Multi-Hop Knowledge-Augmented Generation

Parshan Teimouri
Department of Computer Science
parshanteimouri@cs.ucla.edu

Chengdi Cao
Department of Computer Science
chengdicao@g.ucla.edu

Tang Mohan
Department of Computer Science
tangmohanp@outlook.com

Abstract

In this paper, we present a novel framework for enhancing the reasoning capabilities of Large Language Models (LLMs) through outcome-process combined supervision. Our approach synergizes Chain-of-Thought reasoning and external knowledge integration via the ReAct framework, augmented with iterative fine-tuning inspired by the STaR method. By incorporating both outcome correctness and the quality of intermediate reasoning steps, we aim to achieve more accurate and robust multi-hop knowledge-augmented generation. Evaluated on the HotpotQA dataset, our method demonstrates significant improvements in reasoning accuracy. However, our findings indicate that supervision based solely on outcome correctness yields the best results. Our contributions provide insights into effective model training and the advancement of LLM capabilities in natural language understanding and generation.

1 Introduction

Large Language Models (LLMs) [18, 2, 3] have revolutionized natural language processing by demonstrating impressive capabilities across various tasks. However, they are not without limitations, and current research is investigating how to enable LLMs to exhibit more human-level intelligence. Two notable streams of approaches are Chain-of-Thought reasoning [20] and utilization of external knowledge [24].

Chain-of-Thought reasoning aims to enhance LLM performance by allowing them to articulate intermediate steps in their reasoning process, rather than jumping directly to conclusions. This method has shown promise in improving accuracy and transparency by providing a structured way for the model to tackle complex problems [20]. However, early mistakes in the reasoning chain can propagate, leading to incorrect final answers. Additionally, LLMs may require additional training to effectively perform this step-by-step reasoning. The STaR method enhances Chain-of-Thought reasoning through iterative fine-tuning with outcome supervision, focusing on refining training data based on the correctness of the final answer [25].

On the other hand, leveraging external knowledge involves integrating information from sources beyond the model's pre-trained data. This can significantly enhance the model's ability to handle queries that require up-to-date or highly specific information, reducing instances of hallucination where the model generates plausible but incorrect responses [24]. Techniques such as ReAct (Reasoning and Acting) combine internal model knowledge with external data retrieval to create a more robust and accurate problem-solving approach.

Preprint. Under review.

In this paper, we propose a novel framework that synergizes these approaches by combining the STaR (Self-Taught Reasoner) method and the ReAct framework [24]. We also extend the STaR method by incorporating process supervision, and evaluating the quality of intermediate reasoning steps in addition to the final outcomes. Our experiments aimed to determine whether this additional layer of supervision would enhance performance. However, our findings indicate that while outcome supervision significantly improves reasoning accuracy, adding process supervision does not yield additional gains. This suggests that focusing on the correctness of the final outcomes may be sufficient for optimizing the reasoning capabilities of LLMs.

The rest of the paper is organized as follows: we first review related works in Section 2, then propose our framework in Section 3 that combines outcome and process supervision by LLM self-evaluation to enhance the reasoning ability of LLMs. After presenting our experimental results in 4, we draw conclusions and discuss future prospects in Section 5 and 6.

2 Related Work

2.1 Large Language Models

Large Language Models (LLMs) [18, 2, 3] represent a significant advancement in the field of natural language processing (NLP). Pre-trained on large-scale corpora, these models have shown great potential in various NLP tasks, including question answering, text summarization, machine translation, and logic reasoning. Most LLMs are based on the Transformer architecture [19], which features encoder and decoder modules empowered by a self-attention mechanism. By leveraging billions of parameters, LLMs have demonstrated an unprecedented ability to understand and generate human-like text, performing a wide range of tasks with impressive accuracy. One of the key factors behind their success is their emergent ability to utilize a "zero-shot" or "few-shot" learning approach [2], where a task description or a few examples are provided to guide their reasoning process without any gradient updates.

However, LLMs suffer from hallucinations, i.e. generating seemingly logical but incorrect text, and are not transparent concerning the background process of generation. Hence, despite their impressive characteristics, LLMs have also raised important questions about their ability to reason and answer complex questions. In face of this, researchers have proposed to let LLM perform multi-hop reasoning [20, 23] or augment LLM with external knowledge [24].

2.2 Prompt Engineering

Prompt engineering is a novel field that focuses on creating and refining prompts to maximize the effectiveness of large language models (LLMs). This technique aims to improve the capacity of LLMs, in diverse complex tasks including question answering [6, 12], sentiment classification [10], and common sense reasoning [16]. By clearly describing the instructions for questions or providing specific examples, users can guide the model to perform tasks more precisely [2]. Effective prompt engineering leverages the model's vast knowledge base and contextual understanding, optimizing its ability to perform zero-shot or few-shot learning. This approach enhances the model's performance by reducing ambiguity and providing clear context, which is especially important given the model's sensitivity to input phrasing. As a result, prompt engineering enables more precise and reliable outputs from LLMs across diverse applications.

In the context of LLM reasoning, researchers have proposed prompting methods to instruct LLM to perform multi-hop reasoning, such as the Chain-of-Thought prompting [20] and Tree-of-Thoughts prompting [23]. Researchers also found that LLM can be prompted to evaluate its own reasoning [23, 11], which provides the possibility to improve LLM using its own supervision.

2.3 Finetuning LLMs

Finetuning Large Language Models (LLMs) is the process of adapting a pre-trained model to improve its performance on a specific downstream task instead of training a new model from scratch. This adaptation is usually done via finetuning, which updates all the parameters of the pre-trained model. While finetuning enables LLMs to achieve high accuracy, it has a major disadvantage: finetuning is computationally infeasible, due to the massive size of the model. LLMs consist of billions of parameters, and the size of state-of-the-art models is growing rapidly. However, there are a few methods that tackle this problem.

2.3.1 Injecting Trainable Parameters into Model Architecture

Low-Rank Adaptation (LoRA) [8] suggests freezing the layers of the pre-trained language models and injecting trainable matrices into the transformer layers. In this way, the number of trainable parameters becomes almost 10000 times smaller, leading to less time and computational resources needed for training the language model. In mathematical terms, for the task of autoregressive language modeling, the original objective function is:

$$\sum_{(x,y)\in Z}\sum_{t=1}^{|y|}\log\left(P_{\Phi}(y_t|x,y_{<t})\right) \tag{1}$$

Where Φ is the parameters of the model, y_t is the token language model has to predict, and $x, y_{<t}$ are inputs and previous outputted tokens respectively. After injecting the matrices into transformer layers, the objective function would be:

$$\sum_{(x,y)\in Z}\sum_{t=1}^{|y|}\log\left(p_{\Phi_0+\Delta\Phi(\Theta)}(y_t|x,y_{<t})\right) \tag{2}$$

Here, $\Delta\Phi = \Delta\Phi(\Theta)$ is the task-specific parameters encoded by a much smaller-sized set of parameters Θ with $|\Theta| \ll |\Phi_0|$.

2.3.2 Editing a Subset of Parameters

Factual knowledge might be mislearned during the training time or obsolete with the passing of time. Authors of ROME [5] suggest a way to efficiently update incorrect facts inside a language model. In this approach, first, a hyper-network is trained to take the atomic facts and predict the components of the network that need to be updated. They empirically showed that not only is this method successful in updating the incorrect facts, but it also maintains consistent output for other tasks, i.e. not affecting other model predictions.

They overcome this task by training the hyper-network using the following objective:

$$\begin{aligned} \min_{\phi} \quad & \sum_{\hat{x}\in\mathcal{P}^x}\mathcal{L}(\theta';\hat{x},a) \\ \text{s.t.} \quad & \mathcal{C}(\theta,\theta',f;\mathcal{O}^x) \leq m\ , \end{aligned} \tag{3}$$

where $\mathcal{L}(\theta';\hat{x},a)$ is the loss when using parameters θ', input $\hat{x}$ and targeting fact a. $\mathcal{P}^x$ is the set of semantically-equivalent fact to a, and $\mathcal{C}$ is the constraint we aim to satisfy. Here, the constraint is that the model outputs $f(\cdot;\theta')$ must not change for $x' \in \mathcal{O}^x$ when the model parameters change from θ to θ'.

2.4 Reasoning Methods and Benchmarks

Since the dawn of Large Language Models [18, 2] there has been a growing interest in discovering the extent of LLM reasoning potential and ways to adopt LLM natural language understanding in complex reasoning tasks [15]. To better study the reasoning ability of LLMs, it has been divided into many categories, namely arithmetic, commonsense, mathematical, etc.; Furthermore, numerous benchmarks and datasets have been curated to help quantify the ability in each category.

- **Logical reasoning** which itself can further be divided into three categories: inductive, deductive, and abductive. Inductive reasoning pertains to extracting a general pattern from seen instances, and deductive reasoning, the counterpart of inductive reasoning, is the process of deriving conclusions based on a general rule. Abductive reasoning is finding hypotheses to justify and explain an observation. [21, 1] are examples of benchmarks for logical reasoning evaluation.

- **Mathematical reasoning** pertains to the ability to understand, perform, and manipulate mathematical equations. GSM8K [4] and MATH [7] are among the most used benchmarks for mathematical reasoning evaluation, and are collected from high school math exams and math competitions respectively.
- **Commonsense reasoning** is the ability to understand fundamental real-world knowledge. For example, the model must know that "rain makes roads slippery" [16] or when asked "Where do you put your grapes just before checking out?," the model should output "grocery cart" [15]. CommonsenseQA [17] and OpenBookQA [12] are amongst the most cited benchmarks for commonsense reasoning.
- **Multi-hop reasoning** tasks require the model to take multiple reasoning steps in a sequence to arrive at the correct answer. As an example, the model has to answer "Was the director of "Interstellar" born in Paris?" by first identifying the director of the movie and then output their birthplace [15]. HotpotQA [22], which are method is evaluated on and StrategyQA [6] are the most prominent benchmarks in this reasoning category.

A multitude of prior works explore methods to improve reasoning ability, among which Chain-of-Thought prompting [20] is one of the most seminal. This method demonstrates that allowing the model to take intermediate reasoning steps, or in other words, producing a chain of thoughts, instead of only generating the final answer can drastically increase the accuracy of LLMs in complex reasoning tasks. However, this method comes with several drawbacks. First, early mistakes in the chain of thought might cause all the next steps to be incorrect, i.e., the error propagates through the output of LLM. Second, LLMs might not be trained for performing such small-step reasonings, and need an additional fine-tuning step. In the rest of this section, we will discuss several important approaches that lay our method's foundation.

2.4.1 Synergizing Reasoning and Acting

Following recent works on improving the reasoning abilities of LLMs, such as Chain-of-Thought [20], the Idea of ReAct [24] is to interleave task-oriented actions and verbal reasoning, to help reduce hallucinations when the model faces unforeseen circumstances or information uncertainties. This method can be viewed as a combination of internal and external knowledge, i.e., information stored in the LLM during the pertaining phase and knowledge retrieved from external sources, such as Wikipedia. More precisely, ReAct produces a trace of thought, action, and observation triplets, where the model first generates a thought, then searches the external knowledge base for relevant information (action step), and then interprets the results using its understanding of the natural language. Figure 1 demonstrates an overview of the prompting technique adopted in ReAct [24].

2.4.2 Self-Taught Reasoner

The STaR (Self-Taught Reasoner) method, proposed by Zelikman et al., improves language model reasoning through an iterative process of rationale generation and fine-tuning [25]. Initially, a model is prompted with a few examples that include questions and their corresponding rationales. The model then generates rationales and answers for a larger dataset. Crucially, only rationales that lead to correct answers are retained, providing outcome-based supervision.

For questions answered incorrectly, the correct answers are provided, and the model generates rationales in a reverse-engineering process called rationalization. These rationales are also added to the training dataset. The model is then fine-tuned on this combined set of generated and rationalized rationales. This iterative process, repeated with the improved model generating new rationales each time, progressively enhances its reasoning capabilities.

Our work builds on the STaR approach, applying similar iterative fine-tuning and filtering techniques to ReAct, thereby improving our model's performance in reasoning tasks.

3 Method

An overview of our framework is shown in Figure 2. The LLM will perform complex tasks by ReAct-style multi-hop reasoning augmented by knowledge from external resources (shown as the environment "Env" in the figure) [24]. Specifically, in every step, the LLM can generate step-wise

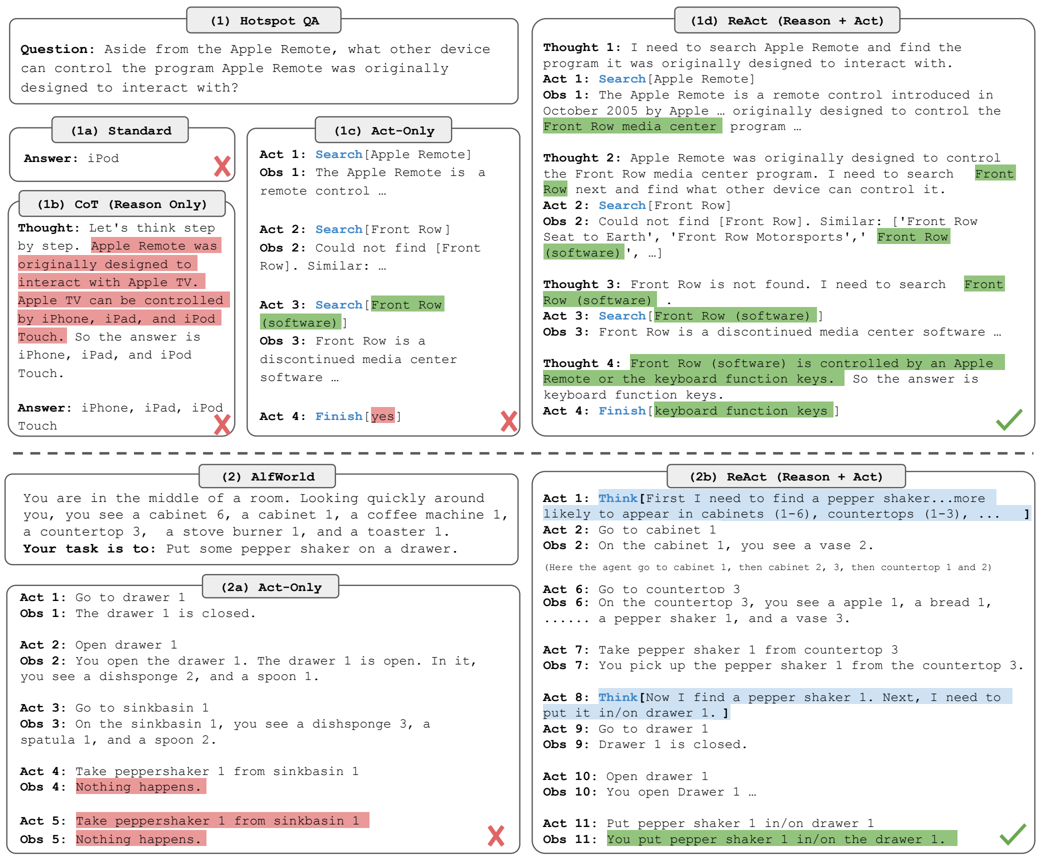

Figure 1: An overview of the ReAct method and how it improves the performance of the LLM in comparison with standard and Chain-of-Thought prompting [20].

thoughts and actions to query the environment, the environment will return retrieved information as observations to the LLM. In accordance with the STaR-style finetuning [25], We will record the trajectories of such reasoning processes on the training set, which will serve as the candidates for fine-tune data. In order to achieve better performance, careful filtering of the trajectories is required. While previous works simply filter the trajectories according to correctness of outcome [25, 24], we propose to combine the supervision from the outcome and the process (intermediate steps) to collect diverse and high-quality fine-tune data.

3.1 Text Generation Endpoints

Latest LLMs usually adopt two different paradigms for text generation: completions and chat completions. The latter would specify the roles ("system", "user" and "assistant") of the text to allow the LLM to better understand the instructions from the system and the user. In terms of implementation, the role specification is usually transformed to a pre-defined sequence of tokens. For example, in Llama3 [18], the user text will be followed by the header `<|start_header_id|>user<|end_header_id|>`, where `<|start_header_id|>` and `<|end_header_id|>` are two special tokens.

Based on the completions template in ReAct [24], we developed another template for chat completions endpoint. An example of question answering on HotPotQA [22] using chat completions endpoint is shown below.

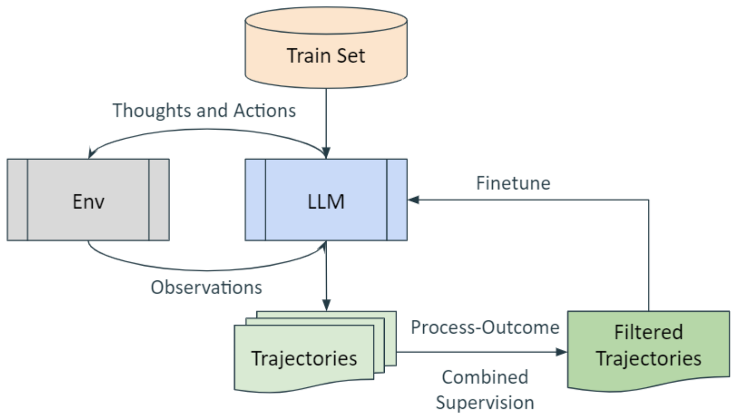

Figure 2: Framework Overview

Table 1: Performance of off-the-shelf LLMs on the first 100 questions in the development set of HotPotQA. Accuracy is measured by exact match.

Model	Endpoint	Accuracy (%)
`gpt-3.5-turbo-instruct`	completions	26
`Meta-Llama-3-8B-Instruct`	text completions	27
`gpt-3.5-turbo-0125`	text completions	32
`gpt-4o-2024-05-13`	text completions	40

```
System: Solve a question answering task with interleaving Thought,
 ↪  Action, Observation steps. Thought can reason about the
 ↪  current situation, and Action can be three types:\n(1)
 ↪  Search[entity], which ...\n(2) Lookup[keyword], which ...\n(3)
 ↪  Finish[answer], which returns the answer and finishes the
 ↪  task.\nHere are some examples: ...
User: Question: Musician and satirist Allie Goertz wrote a song
 ↪  about the "The Simpsons" character Milhouse, who Matt Groening
 ↪  named after who?\nThought 1:
Assistant: I only need to search Milhouse and find who it is named
 ↪  after.\nAction 1: Search[Milhouse]\nObservation 1:
User: Milhouse Mussolini Van Houten is ...\nThought 2:
Assistant: The paragraph does not tell who Milhouse is named
 ↪  after, maybe I can look up "named after".\nAction 2:
 ↪  Lookup[named after]\nObservation 2:
User: (Result 1 / 1) Milhouse was designed by Matt Groening for
 ↪  ...\nThought 3:
Assistant: Milhouse was named after U.S. president Richard Nixon,
 ↪  so the answer is Richard Nixon.\nAction 3: Finish[Richard
 ↪  Nixon]
```

The modification brings us the following two benefits:

1. This allows us to utilize latest LLMs with most powerful capabilities, including open-sourced Llama-3 [18] and proprietary GPT-3.5-turbo [13] and GPT-4o [14]. The performance of these off-the-shelf LLMs are shown in Table 1.

2. This allows us to only fine-tune on the reasoning and action generation by specifying related text as assistant text. In this way, the LLMs learn to better reason instead of memorizing the knowledge from external resources.

3.2 Process Supervision by LLM Self-Evaluation

Current works only utilize outcome supervision [25, 24] and disregard all trajectories that lead to incorrect answers. Given the low accuracy on hard datasets (less than 40% for HotPotQA), a large portion of trajectories do not contribute to the fine-tuning data. In our observation, some trajectories do not lead to exactly matched answers due to format error and minor mismatches. We argue that including some helpful steps from these trajectories in the fine-tuning data can achieve better utilization of the trajectories and better reasoning performance. We call such selection "process supervision", which complements the outcome supervision in previous works.

In order to enrich the fine-tuning data by process supervision, we propose to incorporate LLM itself for evaluation of reasoning process. This utilized the versatile abilities of LLMs and is proved effective by previous works [23, 11]. Inspired by the "score" and "vote" scheme in Tree-of-Thoughts Prompting [23], we developed two strategies for LLM self-evaluation. The first strategy is based on classification and lets LLM evaluate intermediate steps individually. The second strategy is based on comparison and lets LLM process multiple trajectories at the same time and give preference.

3.2.1 Classification of Intermediate Steps

To make use of helpful intermediate thought steps in the trajectories that do not lead to correct answers, we prompt the LLM to classify intermediate steps as "good" or "bad" and cap the trajectories that lead to incorrect answers at the last "good" action. To exploit the few-shot in-context learning ability of LLMs, we manually composed 6 examples and provide them in the system prompt. The prompt and examples can be seen in Appendix A.

3.2.2 Comparison of Trajectories

In our observation, LLMs tend to compliment its own generated text, which aligns with previous works [9]. In order to make better comparison, we proposed a second method for process supervision. In the second method, we choose exactly half of the reasoning trajectories by using LLM to pick the better trajectory for each pair, as shown in Figure 3. The prompt and example output can be found in Appendix B.

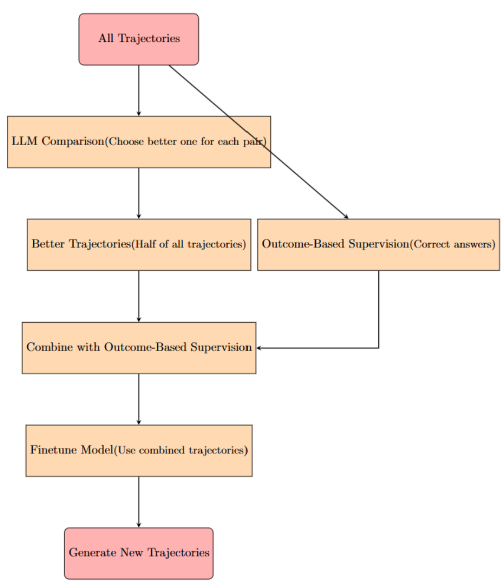

Figure 3: Comparison-based process supervision

4 Experiments

We evaluated the effectiveness of our proposed framework on HotpotQA [22] where the LLM is allowed to access Wikipedia as an external resource to answer open questions. For off-the-shelf models, We access the Llama3 model [18] via HuggingFace and the GPT models via OpenAI APIs [13, 14]. Our baseline method is ReAct with `gpt-3.5-turbo-0125` as the backbone model. To prove the effectiveness of self-evaluation, we use the same model for throughout the experiments. The model is fine-tuned via API provided by OpenAI using the default hyperparameters.

4.1 Trajectory Collection and Filtering

Due to limited time and cost budget, we collect the reasoning trajectories on 300 questions from the training set of HotpotQA, among which 104 lead to exactly matched answers. The baseline accuracy is 32%. The trajectories that lead to exactly matched answers are added to fine-tune data as the result of outcome supervision. Then, we expand the fine-tuning data by process supervision using our proposed two LLM self-evaluation strategies.

In the classification-based method, we filtered out 117 partially good trajectories, effectively expanding the fine-tuning data to around $2\times$. In the comparison-based method, we expanded the fine-tuning data by 150 trajectories, which is roughly an expansion to $2.5\times$ with some duplication.

4.2 fine-tune Results

The performance of fine-tuned GPT is shown in Table 2. We observe that fine-tuning with output supervision effectively improves performance. On the other hand, we don't observe an advantage of using process supervision vs. output-only supervision.

We also observe that using LLM classification to pick out partially good steps does not yield as good performance as other fine-tuning methods. We provide two explanations for this phenomenon: 1) LLM tends to adhere to its own generated text, thus leaving a lot of low-quality reasoning steps in the filtered trajectories; 2) the filtered partial trajectories are incomplete in format, which can misguide the model to generate wrong question answering format.

Table 2: Accuracy of Different Methods on the first 100 questions in the development set of HotPotQA. Accuracy is measured by exact match.

Method	**Accuracy (%)**
Baseline	32
Finetuning with Output Supervision	35
Finetuning with Output Supervision + LLM Ranking	35
Finetuning with Output Supervision + LLM Picked Partially Good Steps	33

5 Conclusion

In this work, we first discussed the limitations LLMs have in complex reasoning tasks. We probed into different subcategories of reasoning and available benchmarks for quantifying this quality. Next, we discussed some of the most seminal works on improving the quality of reasoning In LLMs. Furthermore, we proposed to improve the reasoning abilities of fine-tuned LLM by using LLM-self-evaluation-based process supervision to expand fine-tune data. Concrete evaluations demonstrate the efficiency of our approach in reducing the hallucinations and generating higher-quality reasoning traces.

6 Future Work

Due to computational limitations, we were not able to perform the fine-tuning experiments on the full HotPotQA dataset and on the Llama-3 model [18]. Instead, we used GPT-3.5 API and performed our fine-tuning experiments on a subset of HotPotQA. Expanding the experiments and investigating other approaches for trajectory evaluation could be promising future directions for our work.

Contributions

The contribution of each team member is shown in Table 3.

Table 3: Contributions to the Project

Task	Members Contributed
Literature Review	All
Brainstorming	All
Fine Tuning GPT 3.5	Chengdi
Prompt Engineering	Mohan
Experiments on Llama3	Parshan
Preparing the presentations	All
Writing the report	All

References

[1] Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. Abductive commonsense reasoning. *arXiv preprint arXiv:1908.05739*, 2019.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[5] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

[6] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

[7] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[8] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[9] Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive verification of chain-of-thought reasoning. *Advances in Neural Information Processing Systems*, 36, 2024.

[10] Md Saef Ullah Miah, Md Mohsin Kabir, Talha Bin Sarwar, Mejdl Safran, Sultan Alfarhood, and MF Mridha. A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and llm. *Scientific Reports*, 14(1):9603, 2024.

[11] Ning Miao, Yee Whye Teh, and Tom Rainforth. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *arXiv preprint arXiv:2308.00436*, 2023.

[12] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

[13] OpenAI. Gpt-3.5-turbo-0125, 2023. `https://www.openai.com/`.

[14] OpenAI. Gpt-4o, 2024. `https://openai.com/index/hello-gpt-4o/`.

[15] Ji-Lun Peng, Sijia Cheng, Egil Diau, Yung-Yu Shih, Po-Heng Chen, Yen-Ting Lin, and Yun-Nung Chen. A survey of useful llm evaluation. *arXiv preprint arXiv:2406.00936*, 2024.

[16] Maarten Sap, Vered Shwartz, Antoine Bosselut, Yejin Choi, and Dan Roth. Commonsense reasoning for natural language processing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 27–33, 2020.

[17] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.

[18] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[20] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[21] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

[22] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. 2018.

[23] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[25] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. 2022.

A Prompts Used for Classification-Based LLM Self-Evaluation

```
Given a question solving process, evaluate if the last action is
 ↪  promising after giving supporting rationale.
End the response with "good" or "bad".
Here are some examples:

Question: Megalyn Echikunwoke played a comic book character in
 ↪  what movie?
Thought 1: I need to search for Megalyn Echikunwoke and find out
 ↪  in which movie she played a comic book character.
Action 1: Search[Megalyn Echikunwoke]
Observation 1: Ebubennem Megalyn Ann Echikunwoke // \u24d8
 ↪  (/\u02c8m\u025b\u0261\u0259l\u026an
 ↪  \u025b\u02cct\u0283\u026ak\u028an\u02c8wo\u028ake\u026a/
 ↪  MEG-\u0259-lin etch-IK-uun-WOH-kay;[1] born May 28,
 ↪  1983[2][3]), also known as Megalyn E.K.,[4] is an American
 ↪  actress,[5] known primarily for her roles in television and
 ↪  film.. Born in Spokane, Washington, she was discovered while
 ↪  performing in a theatrical production for an arts academy, and
 ↪  began working professionally at age 15, appearing in an
 ↪  episode of The Steve Harvey Show. Between 2001 and 2002, she
 ↪  starred as Nicole Palmer, daughter of David Palmer, on the Fox
 ↪  series 24, followed by guest roles on ER and Buffy the Vampire
 ↪  Slayer. Between 2004 and 2005, she portrayed Angie Barnett,
 ↪  the half-sister of Steven Hyde, on the sitcom That '70s Show.
 ↪  In 2006, she starred as a lead in the series The 4400 which
 ↪  ran until 2007, followed by recurring roles on 90210 and House
 ↪  of Lies..
Thought 2: The passage provides information about Megalyn
 ↪  Echikunwoke's career in television and film, but it does not
 ↪  mention any specific movie where she played a comic book
 ↪  character. I should look up her filmography to find the
 ↪  answer.
Action 2: Search[Megalyn Echikunwoke filmography]
Rationale: The last action searched for a keyword that is unlikely
 ↪  to exist as an individual page. Instead, I should lookup
 ↪  "filmography" in the last searched page.
Evaluation: bad

Question: Which actress, who at age twelve gained widespread
 ↪  recognition from her role as Claudia in "Interview with the
 ↪  Vampire", stars in The Virgin Suicides?
Thought 1: I need to search for the actress who starred in
 ↪  "Interview with the Vampire" as Claudia and also stars in The
 ↪  Virgin Suicides. Let's start by searching for the actress who
 ↪  played Claudia in "Interview with the Vampire".
Action 1: Search[Claudia Interview with the Vampire]
Rationale: The last action searched for a keyword that combines
 ↪  the role and the title and is unlikely to exist as an
 ↪  individual page. Instead, I should search for the title
 ↪  "Interview with the Vampire" alone.
Evaluation: bad

Question: Galgameth was a film that was a loose remake of the 1985
 ↪  film directed by whom?
Thought 1: I need to search for the film Galgameth and find out
 ↪  who directed the 1985 film it was a loose remake of. Let's
 ↪  start by searching for Galgameth.
```

```
Action 1: Search[Galgameth]
Observation 1: Galgameth (also released under the titles The
 ↪  Legend of Galgameth and The Adventures of Galgameth) is a 1996
 ↪  American fantasy children's film directed by Sean McNamara.
 ↪  The film stars Devin Neil Oatway, Johna Stewart and Stephen
 ↪  Macht.[2][3] The film's script is loosely based on Shin
 ↪  Sang-ok's 1985 film Pulgasari, which he had directed while
 ↪  being held in North Korea and which was itself a remake of a
 ↪  lost 1962 film.[4][5] The film was produced through Shin's
 ↪  production company Sheen Communications.[6]. In the medieval
 ↪  kingdom of Donnegold, a young prince named Davin (Devin
 ↪  Oatway) lives with his father, the noble King Henryk (Sean
 ↪  McNamara). This comes to an end when the King's black knight,
 ↪  El El (Stephen Macht), poisons him. As he lays dying, Henryk
 ↪  gives his son a small black statue of a creature.
Thought 2: The film Galgameth is a 1996 American fantasy
 ↪  children's film directed by Sean McNamara. It was a loose
 ↪  remake of Shin Sang-ok's 1985 film Pulgasari. Now, I need to
 ↪  find out who directed the 1985 film Pulgasari. Let's search
 ↪  for Pulgasari.
Action 2: Search[Pulgasari]
Rationale: The last action successfully guided the answering
 ↪  process to a reasonable next step.
Evaluation: good

Question: How long is the footpath devised by the author of the
 ↪  book which devotes an entire chapter to the Maiden Moor?
Thought 1: I need to search for the book that devotes an entire
 ↪  chapter to Maiden Moor and find out the length of the footpath
 ↪  devised by the author.
Action 1: Search[book devotes entire chapter to Maiden Moor]
Observation 1: Could not find book devotes entire chapter to
 ↪  Maiden Moor. Similar: ['List of songs about Oklahoma',
 ↪  'Richard Nixon (section Return to public life)', 'Gian Lorenzo
 ↪  Bernini (section Visit to France and service to King Louis
 ↪  XIV)', 'Maximilian I, Holy Roman Emperor (category Pretenders
 ↪  to the Hungarian throne)', 'Filippo Brunelleschi'].
Thought 2: I should try to search for Maiden Moor directly and see
 ↪  if I can find any information about the footpath devised by
 ↪  the author.
Action 2: Search[Maiden Moor]
Rationale: The last action modified the search keyword to a more
 ↪  precise one and has higher possibility to exist as an
 ↪  individual page.
Evaluation: good

Question: What FIA drag racing class is similar to the Australian
 ↪  Top Doorslammer clas?
Thought 1: I need to search for the FIA drag racing class similar
 ↪  to the Australian Top Doorslammer class. Let me start by
 ↪  searching for the Australian Top Doorslammer class.
Action 1: Search[Australian Top Doorslammer class]
Observation 1: Could not find Australian Top Doorslammer class.
 ↪  Similar: ['Top Doorslammer', 'Victor Bray (category Use
 ↪  Australian English from April 2018)', 'Australian National
 ↪  Drag Racing Association', 'Drag racing (section Classes)',
 ↪  'ANDRA Pro Series (category Motorsport in Australia)'].
```

```
Thought 2: To find the FIA drag racing class similar to the
 ↪  Australian Top Doorslammer class, I can search for the Top
 ↪  Doorslammer class first. Let me search for the Top Doorslammer
 ↪  class.
Action 2: Search[Top Doorslammer class]
Rationale: The last action modified the search keyword, but the
 ↪  new keyword does not exist in the suggested similar keywords.
 ↪  Therefore, it has a low possibility in finding a page. I
 ↪  should search for "Top Doorslammer" in the suggestions
 ↪  instead.
Evalluation: bad

Question: Are Nuphar and Geum both types of aquatic plants?
Thought 1: I need to search for information about Nuphar and Geum
 ↪  to determine if they are both types of aquatic plants.
Action 1: Search[Nuphar]
Observation 1: Nuphar is a genus of aquatic plants in the family
 ↪  Nymphaeaceae, with a temperate to subarctic Northern
 ↪  Hemisphere distribution. Common names include water-lily
 ↪  (Eurasian species; shared with many other genera in the same
 ↪  family), pond-lily, alligator-bonnet or bonnet lily, and
 ↪  spatterdock (North American species).[2]. The genus is closely
 ↪  related to Nymphaea. Nuphar differs in that its petals are
 ↪  much smaller than its 4-6 bright yellow-coloured sepals,
 ↪  whereas in Nymphaea, the petals are much larger than the
 ↪  sepals. The genera also differ in the maturation of their
 ↪  fruit; while maturing, Nuphar fruit remain above water level
 ↪  on their scapes, whereas fruit of Nymphaea sink below water
 ↪  level immediately after their flowers close, and there they
 ↪  mature.
Thought 2: Nuphar is indeed an aquatic plant, belonging to the
 ↪  genus of water-lilies. Now, I need to search for information
 ↪  about Geum to determine if it is also an aquatic plant.
Action 2: Search[Geum]
Rationale: The last actions decompose the answering process into
 ↪  searching for the two species mentioned in the question. It is
 ↪  a promising strategy to do so and compare the results.
Evaluatoin: good
```

B Prompts Used for Comparison-Based LLM Self-Evaluation

```
For the following thought-action trajectories, which one is a
 ↪  better attempt to answer the question provided? A good attempt
 ↪  should at least find something useful and promising. You may
 ↪  do some analysis, but don't get too long.
```

Example output:

```
In comparing the two thought-action trajectories provided,
 ↪  trajectory B is the better attempt to answer the question
 ↪  provided. While trajectory A successfully identified the
 ↪  countries where Wenling and Hechuan District are located,
 ↪  trajectory B demonstrated a more thorough and persistent
 ↪  effort in finding the correct answer to the question. Despite
 ↪  facing challenges in directly finding the co-recipient of
 ↪  Subrahmanyan Chandrasekhar's Nobel Prize, trajectory B
 ↪  showcased adaptability by considering the historical context
 ↪  and ultimately arriving at the correct answer, William A.
 ↪  Fowler. This approach of exploring different avenues and
 ↪  utilizing contextual knowledge led to a more accurate and
 ↪  comprehensive response.
```

LLM is then prompted again to summarize the response:

```
Does this analysis prefer A or B? Give the answer directly, with
 ↪  only a character (A or B).
```