Navigating the Fermi Multiverse: Assessing LLMs for Complex Multi-hop Queries

Mostafa Rahgouy[1], Hamed Babaei Giglou[2], Dongji Feng[3], Taher Rahgooy[4], Gerry Dozier[1] and Cheryl D. Seals[1]

[1]*Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA*
[2]*TIB Leibniz Information Centre for Science and Technology, Hannover, Germany*
[3]*Department of Mathematics, Computer Science, and Statistics, Gustavus Adolphus College, MN, USA*
[4]*Meta, Menlo Park, CA, USA*

Abstract

Recently, large language models (LLMs) have gained significant attention in the field of Natural Language Processing (NLP) and have shown promise across various tasks, even when given only a few examples to learn from. However, their ability to understand and reason with natural language remains uncertain. While there have been attempts to evaluate these models using reasoning tests, these evaluations have mostly focused on models' final answers, often overlooking the step-by-step reasoning processes behind their performance. Additionally, these analyses have typically concentrated on just one or a few aspects of reasoning, especially for tasks that do not require much complex thinking to find the answer. This limits our understanding of LLMs' potential and limitations when it comes to more complex and realistic questions. To address this issue, we conduct a comprehensive analysis of LLMs using the existing Fermi reasoning challenge, a task that combines different aspects of reasoning into a single question-answering format, requiring deeper levels of reasoning. In this paper, we examine various advanced LLMs in this reasoning challenge and explore how their performance is affected by their size (i.e., the number of parameters). We also investigate how these models behave with different levels of supervision, ranging from having all the information to no evidence at all. Furthermore, we compare the two primary methods of teaching these LLMs, fine-tuning, and few-shot learning, using the *Chain-of-Thought* approach. We provide a detailed case study highlighting the most common limitations of these models. While our results imply that these models may have a long journey ahead to reach human-level reasoning, our work can be considered a robust baseline for the community to strive toward achieving this ambitious goal. Our code is available on GitHub https://github.com/MostafaRahgouy/LLMs_for_FPs for the community.

Keywords

NLP, Natural Language Reasoning, LLMs, QA, Fermi Problems, Few-shot Learning, Fine-tuning

1. Introduction

Throughout history, humans have grappled with the concept of reasoning, seeking to define and understand it. This pursuit dates back to the early Greek philosophers, who posed profound questions such as "What can be known?" and "What does that mean someone knows something?" in their quest to illuminate the nature of reasoning [2]. A more recent definition involves step-by-step or systematic thinking that guides humans toward correct answers [3]. With the advent

NL4AI 2023: Seventh Workshop on Natural Language for Artificial Intelligence, November 6-7th, 2023, Rome, Italy [1]

MZR0108@auburn.edu (M. Rahgouy); hamed.babaei@tib.eu (H. Babaei Giglou); djfeng@gustavus.edu (D. Feng); trahgooy@meta.com (T. Rahgooy); doziegv@auburn.edu (G. Dozier); sealscd@auburn.edu (C. D. Seals)

CEUR Workshop Proceedings (CEUR-WS.org)

of Artificial Intelligence (AI), the development of systems facilitating reasoning has emerged as a paramount objective for researchers in this domain ([4, 5]). This aspiration has drawn nearer to realization with the introduction of LLMs. These models undergo a two-phase training process, commonly referred to as pre-training and fine-tuning. In the pre-training phase, they are exposed to vast volumes of data, equipping them with a foundational understanding of language. Subsequently, in the fine-tuning phase, these models refine their capabilities by learning to excel at specific downstream tasks. Prominent exemplars of such LLMs include BERT [6], BART [7], and T5 [8], which have consistently outperformed their predecessors across a spectrum of NLP tasks, including reasoning challenges. Researchers have pushed the boundaries further by introducing super-large language models capable of addressing various questions with minimal or even zero examples, denoted as few-shot and zero-shot learning, respectively [9]. Models such as GPT-4 and LLaMA [10], have demonstrated remarkable prowess in these areas. Nevertheless, in numerous NLP tasks, LLMs often approach or even surpass human-level performance. However, their ability to reason falls significantly short of human capabilities, warranting further in-depth investigation to enhance these models. For example, [11] has pointed out a significant issue where LLMs like GPT-3 and BLOOM struggle with simple common-sense planning tasks, which humans find easy. Moreover, [12] conducted experiments and found that existing LLMs are still unable to pass the Theory-of-Mind tests, where Theory-of-Mind tests aim to assess LLMs abilities to understand and infer the intentions, emotions, and mental states of others. The root cause of the uncertainty regarding the ability of LLMs in reasoning can be traced back to the initial works that published tasks and benchmark datasets that are overly simplistic for these large models. Such simplicity inadvertently provides opportunities for models to employ suboptimal techniques, thus potentially skewing their performance [13, 14]. To address this issue, we have recently witnessed significant efforts within the community aimed at devising more intricate tasks that LLMs to not only comprehend but also engage in reasoning when responding to questions [15, 16]. These endeavors have resulted in the emergence of more intricate reasoning tasks, typically taking the form of QA formats, which require multi-hop reasoning capabilities. Nevertheless, while these studies have been invaluable and have contributed intriguing insights into LLMs, they have inadvertently overlooked certain crucial factors. Firstly, many of these tasks involve a limited number of hops, often restricted to two or three steps. Secondly, these tasks are typically structured as true/false or multiple-choice questions, which may not capture the nuanced behavior of LLMs in responding. Additionally, these investigations failed to consider the correlation between the level of supervision and LLMs performance. In essence, it is essential to explore how LLMs perform when provided with various levels of information, ranging from complete information as seen in mathematical word problems to partial or even no information, to gain a comprehensive understanding of their behavior and decision-making processes. To this end, we selected the existing Fermi Reasoning Challenge introduced by [17]. This selection addressed the aforementioned issues by presenting multi-level tasks that demand a more profound level of reasoning. Furthermore, Fermi Problems (FP) inherently require an approximation in their responses, as precise answers are often impossible or impractical to attain. Our contribution can be summarized as:

1. We present a comprehensive assessment of LLMs applied to Fermi Problems across

different levels of supervision. Our study focuses on leading LLMs, including T5, Flan-T5, and models from the GPT family, marking the first of its kind in applying these models to the FP reasoning task. Therefore, this paper can serve as a reference point for approaching this challenging task.

2. We investigate various approaches for training and inferring from LLMs. Specifically, we assess the impact of fine-tuning, both with and without prompting. We also explore the utility of *Chain-of-Thought* prompting [18] and evaluate the effectiveness of few-shot learning while varying the number of provided examples. Furthermore, we delve into the application of zero-shot learning for implicit reasoning in FP.
3. We offer a more in-depth analysis of the behavior exhibited by LLMs, particularly focusing on the most common errors they tend to make. Additionally, we investigate the relationship between the size of LLMs and their performance for FP. Furthermore, we also unearth intriguing latent insights that offer valuable guidance for potential enhancements in these models.

2. Fermi Problem

The Fermi challenge [17] inspired by Enrico Fermi, the Nobel winner in physics known for his remarkable skill in making accurate estimates of complex numerical problems, is often referred to as "Fermi problems". These problems typically involve making assumptions and approximations to arrive at a rough estimate, rather than a precise calculation. Owing to the intrinsic complexity of the reasoning questions involved, FPs have been appropriated for use in science Olympiads and interviews.

2.1. Tasks

FP encompasses three distinct tasks: PERFECT-CONTEXT, DISTRACTOR-CONTEXT, and FULL, which are designated as Task 1, Task 2, and Task 3, respectively. These tasks can be seen as different levels of supervision or evidence provided to a model. Figure 1 illustrates an example of the FP for these three tasks.

2.1.1. Task 1: Perfect-Context

At this level, alongside the given question, all essential knowledge (defined as a set of facts) required to answer the question is integrated into the input. This task bears resemblance to math word problems, which often feature a concise narrative outlining a scenario and presenting a question related to an unknown quantity [19]. In both of these scenarios, information is explicitly provided, eliminating the need for retrieving knowledge, and instead emphasizing how such information is interconnected and can be used as guidelines to arrive at the final answer.

2.1.2. Task 2: Distractor-Context

In realistic scenarios, the input context often comprises non-relevant information, and models must effectively discern which pieces are pertinent to the given question. In line with this

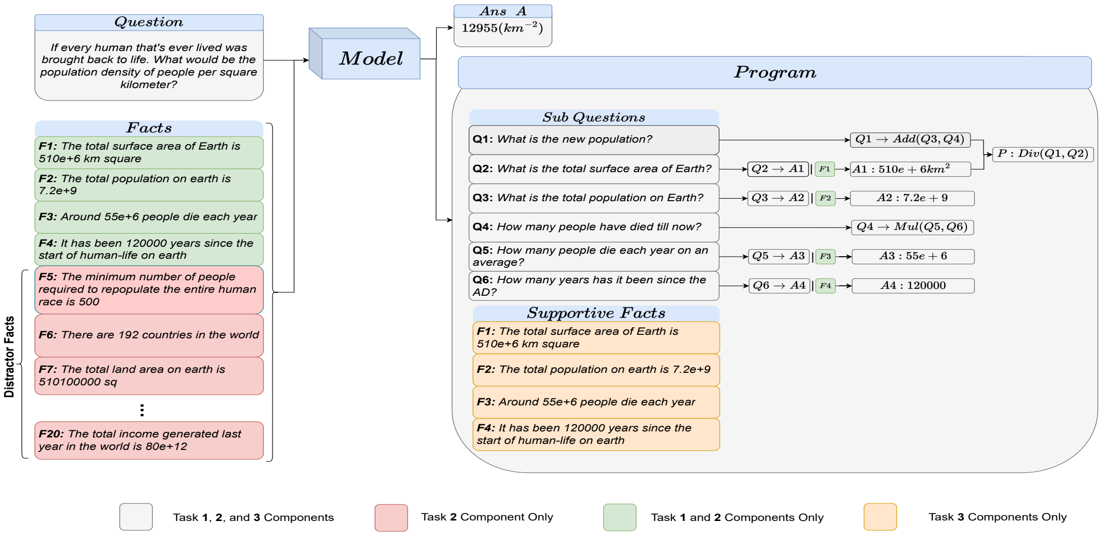
Question
If every human that's ever lived was brought back to life. What would be the population density of people per square kilometer?
Model
Ans A
12955(km^-2)
Program
Facts
F1: The total surface area of Earth is 510e+6 km square
F2: The total population on earth is 7.2e+9
F3: Around 55e+6 people die each year
F4: It has been 120000 years since the start of human-life on earth
F5: The minimum number of people required to repopulate the entire human race is 500
F6: There are 192 countries in the world
F7: The total land area on earth is 510100000 sq
⋮
F20: The total income generated last year in the world is 80e+12
Distractor Facts
Sub Questions
Q1: What is the new population?
Q2: What is the total surface area of Earth?
Q3: What is the total population on Earth?
Q4: How many people have died till now?
Q5: How many people die each year on an average?
Q6: How many years has it been since the AD?
Q1 → Add(Q3, Q4)
Q2 → A1 | F1 → A1 : 510e + 6km^2
Q3 → A2 | F2 → A2 : 7.2e + 9
Q4 → Mul(Q5, Q6)
Q5 → A3 | F3 → A3 : 55e + 6
Q6 → A4 | F4 → A4 : 120000
P : Div(Q1, Q2)
Supportive Facts
F1: The total surface area of Earth is 510e+6 km square
F2: The total population on earth is 7.2e+9
F3: Around 55e+6 people die each year
F4: It has been 120000 years since the start of human-life on earth
Task 1, 2, and 3 Components
Task 2 Component Only
Task 1 and 2 Components Only
Task 3 Components Only

Figure 1: FP's Tasks: In Task 1, the model receives both the question and the relevant facts (represented as green facts). In Task 2, in addition to the question and relevant facts, the model also receives distractor facts (represented as red facts). Finally, in Task 3, the model only receives the question without any additional information. Regardless of the task, the model can produce two alternative outputs: Ans A, which is a direct answer and implies implicit reasoning, and Program, which represents an executable explicit reasoning process. In the program, fact matching (assigning sub-questions to facts) is excluded from Task 3 because this task is meant to generate its supportive facts independently (orange component).

real-world complexity, the FP includes this task that involves the deliberate inclusion of some distractor facts alongside the question and relevant information.

2.1.3. Task 3: Full

The ultimate goal is to enable models to answer complex questions without any provided information. In essence, the model should learn to retrieve supporting facts and demonstrate the reasoning behind the answer based on the retrieved supporting facts. Achieving this level of abstraction is akin to human-style problem-solving.

2.1.4. Outputs

Regardless of the chosen task, two acceptable outputs can be provided, which may serve as alternatives to each other: *Ans A* and *Program P*. Ans A signifies the direct answer to a given question. Furthermore, each FP question is complemented by an explanation in the form of an executable program. This program delineates the facts, values, and mathematical computations essential for deriving the answer. Program P has the potential to demonstrate how models engage in reasoning across various facets, including question decomposition, fact matching, and the relationship between questions in terms of mathematical operations. Given the paper's primary focus on LLM behavior and Program P's role in facilitating the interpretability of

Name	Train	Validation	Test
REALFP	185	125*	558
SYNTHFP	8000	1000	1000

Table 1
Statistics for both RealFP and SynthFP datasets. However, [17] mentioned that the validation set (*) of RealFP contains 185 samples, but we found only 125 in their released dataset.

models, we will exclusively present experiments based on this output, excluding Ans A. This decision allows us to delve deeper into the model's reasoning processes and provides valuable insights into its decision-making capabilities.

2.2. Datasets

[17] presented two distinct datasets for analysis: RealFP, comprising of real-world FP collected from various internet pages, quizzes, and Fermi problem Olympiads; and SynthFP, a larger synthetic dataset created manually from 12 different templates. Table 1 shows statistics of the datasets. As the test set of RealFP better represents the Fermi challenges in the real world, throughout this paper, we will use this set to report the performance of the models. Additionally, the RealFP validation set is employed for early stopping mechanisms for the models in our experiments, and we chose not to use the validation and test sets of Synthetic to maintain consistency and fairness across all models.

2.3. Metrics

Answer Evaluation: To evaluate the performance of a predicted direct answer A' over the true answer A, the fp_score with the following definition will be employed:

$$fp\_score = max\left\{0, 1 - \frac{1}{3}\left|log_{10}\frac{A'}{A}\right|\right\} \tag{1}$$

This metric considers the imprecision and uncertainty of answers by assigning a full score to a prediction that produces an answer within the same order of magnitude as the reference gold answer. Conversely, for each order of magnitude that the prediction diverges from the reference answer, the score is reduced by 1/3 points.

Program Evaluation: Explanations (programs) are evaluated along three criteria:

- Validity (*valid?*): This assesses whether the program is syntactically valid by evaluating it to determine if it results in a numerical output. A Python program executor is employed for this purpose. A score of 1 is assigned if the execution is successful, otherwise, it receives a score of 0.
- Answer Accuracy Evaluation ($PAns$): After the program successfully generates a numeric answer (passed the *valid?* evaluation with score 1), this metric assesses the accuracy of the generated numeric answer A' with ground truth A based on the previously defined fp_score.

- Fact Identification (*Facts*): Determines whether the program includes all and only the specified gold facts F, using an F1 measure for assessment.

3. Experimental Design

Our experiments are structured around different LLMs training approaches, including fine-tuning, few-shot, and zero-shot learning, with and without Chain-of-Thought (CoT) prompting.

3.1. Fine-tuning Setting

Fine-tuning without Chain-of-Thought: The introduction of Transformers, along with the attention mechanism [20], marked a new era in transfer learning. This innovation enabled the training of large models in an unsupervised manner on extensive datasets, allowing them to acquire semantic knowledge of the language, thus forming a robust foundation for adapting to downstream NLP tasks. This approach has significantly improved performance across a wide range of NLP tasks with limited training samples, a practice commonly referred to as fine-tuning. Notable examples of these models include BERT, BART, and T5. Among the aforementioned models, T5 stands out as particularly adept at handling reasoning tasks due to its inherent architecture of sequence-to-sequence (seq2seq), which aligns well with the demands of such tasks. Consequently, we have chosen this model as the foundation for our fine-tuning setting. To achieve this, we employ a straightforward approach: we concatenate the available supportive facts with the question (for tasks 1 and 2) to construct the input, which the model processes, ultimately yielding Program P as the output. Furthermore, we assess the performance of T5 using various versions, including *T5-small* and *T5-base*. It is worth noting that we also explored a larger variant of T5, namely *T5-large*, which boasts 770 million parameters. However, our findings indicated a degradation in results, which could potentially be attributed to the limited number of samples available within the datasets.

Fine-tuning with Chain-of-Thought: In the realm of instruction-based fine-tuning, [21] conducted an investigation into the impact of various factors, including scaling the number of tasks and model size, and the incorporation of CoT data during the fine-tuning process. Specifically, in their paper, they outlined their objective as follows:

> *"The goal of Flan finetuning is to produce an improved checkpoint across a range of evaluations, which includes multi-step reasoning ability in addition to traditional NLP tasks".*

To achieve this goal, they integrated nine CoT datasets into the fine-tuning phase and demonstrated the positive impact of this approach on unseen reasoning tasks. Additionally, they made Flan-T5 checkpoints publicly available, maintaining consistency with prior versions of publicly released T5 checkpoints. As a result of these considerations, we selected the Flan-T5 model as our experimental model, which incorporates the CoT capability. However, prompt engineering can be beneficial in tailoring prompts to suit specific tasks. Nonetheless, we opted to maintain consistency by using the original CoT prompt ("Answer the following question step by step"), as utilized in the original paper, to ensure fairness across different tasks and model sizes in our experiments.

3.2. Few-Shot Setting

LLMs have popularized the notion of few-shot learning, where these models can acquire new tasks with just a small number of examples [22, 23]. Few-shot learning offers significant benefits as it reduces the necessity for extensive data collection, which can be costly. Investigating few-shot reasoning for FPs can determine whether such models can address intricate questions in an interpretable manner. To this aim, we explore the capabilities of the GPT-based family with varying numbers of provided examples, as typically encountered in few-shot learning scenarios.

3.3. Zero-Shot Setting

Embracing the use of super-large LMs has opened the door to unlocking zero-shot reasoning capabilities. Prominent models in this category include GPT-4, LLama, Flan-PaLM, Flan-T5, and Bloom. Furthermore, the fusion of these models with the CoT prompting has yielded significant improvements in various tasks [24]. However, CoT imposes a requirement for these models to generate answers through step-by-step reasoning processes. Nonetheless, these answers may not align with FP's *Program P* output. This discrepancy arises because the specified program P does not conform to the standard output generated by such models, and they encounter challenges in producing responses without any prior exposure to relevant samples. Consequently, we explored and evaluated zero-shot reasoning by focusing solely on direct answers (*Ans A*). While this approach may limit interpretability to some extent, assessing their performance under these conditions can offer valuable insights.

3.4. Experiments Setup

In our fine-tuning process, we used a constant random seed value throughout all experiments. We maintained a batch size of 8 for the entire duration. Additionally, we set the learning rate to 1e-3 and utilized the Adam optimizer [25]. To monitor model performance and ensure reproducibility, we employed the validation set from the real dataset. The best model checkpoints were saved based on this validation set loss. For consistency and fairness in reporting results, we conducted fine-tuning for 50 epochs on real data and 5 epochs each on synthetic and combined (both) data. Moreover, our GPU of choice was the NVIDIA A100 SXM4 40 GB. In few-shot and zero-shot settings, we adjusted the temperature parameter to <= 0.1 to enhance model determinism and minimize variation.

4. Results and Performance Analysis

4.1. Quantitative Results

Program-Based results: Table 2 provides an overview of our findings obtained through fine-tuning and few-shot learning experiments. As observed in the table, increasing the model

Model	**Task 1**: Perfect-Context			**Task 2**: Distractor-Context			**Task 3**: Full	
	Program P			Program P			Program P	
	$PAns$	Valid?	Facts	$PAns$	Valid?	Facts	$PAns$	Valid?
Fine-tuning								
T5-small								
real	0.36	0.67	0.97	0.22	0.87	0.66	0.18	0.95
synth	0.17	0.39	0.85	0.08	0.43	0.58	0.19	0.84
both	0.35	0.63	0.95	0.18	0.78	0.79	0.13	0.75
T5-base								
real	0.37	0.75	0.94	**0.28**	0.88	0.59	0.16	0.93
synth	0.16	0.26	0.91	0.12	0.49	0.89	0.15	0.83
both	0.44	0.77	0.89	0.16	0.56	0.87	0.16	0.83
FLAN-T5-small								
real	0.34	0.62	0.97	0.21	0.89	0.56	0.15	0.94
synth	0.15	0.33	0.91	0.09	0.57	0.5	0.14	0.82
both	0.39	0.64	0.94	0.18	0.71	0.88	0.18	0.78
FLAN-T5-base								
real	**0.49**	0.86	0.94	0.24	0.85	0.62	0.16	0.93
synth	0.15	0.35	0.88	0.05	0.27	0.85	0.14	0.83
both	0.43	0.75	0.95	0.23	0.89	0.93	0.14	0.87
Few-Shot Learning								
GPT-3.5-Turbo								
1-shot	0.23	0.44	0.90	0.13	0.26	0.86	0.10	0.34
3-shot	0.49	0.68	0.96	0.28	0.51	0.92	0.21	0.52
5-shot	**0.52**	0.70	0.95	**0.32**	0.56	0.92	**0.23**	0.58

Table 2
Program-based results: T5, FLAN-T5 fine-tuned without and with CoT respectively on three distinct datasets: *real*, synthetic (*synth*), and combined (*both*) datasets. For few-shot learning, GPT was utilized exclusively on the real dataset. Evaluation criteria for *Program P* include whether it executes (Valid?), and if so, whether the execution produces a correct answer ($PAns$), as well as whether it utilizes the required (gold) facts included in the input for Tasks 1 and 2. All the results reported above were obtained from the real dataset's test set.

size from *small* to *base* resulted in performance improvements for both the T5 and FLAN-T5 models. Notably, *FLAN-T5* achieved the highest performance in *task-1* with a precision score of **0.49**, while in *task-2*, *T5-base* obtained the highest score (**0.28**) among all fine-tuning settings. However, it is important to highlight that *task 3* yielded comparatively lower results across all fine-tuning settings. Of particular interest is the performance in the 5-shot learning scenario, where GPT-3.5-turbo surpassed all fine-tuned models and achieved a new state-of-the-art benchmark on all three tasks. Another noteworthy observation relates to the "Valid?" score in Task 3 when compared to Task 1 and Task 2 in the Fine-tuning setting. In most instances, the model tends to generate more valid programs in Task 3. This can be attributed to the model's greater freedom in generating its own approach, which results in more preferred ways of generating solutions, often leading to shorter answers and reducing the likelihood of producing invalid responses.

Model	**Task 1**: Perfect-Context	**Task 2**: Distractor-Context	**Task 3**: Full
	Ans A	Ans A	Ans A
GPT-4	0.66	**0.55**	0.22
GPT-3.5-Turbo	**0.72**	0.46	**0.29**
FLAN-T5-XL	0.34	0.12	0.25

Table 3
Zero-Shot results: Due to the infeasibility of generating *Program P*, this setting was evaluated using direct answers (*Ans* A) with the fp_score function.

Direct-Answer-based results As previously mentioned, attempting to generate the executable *Program P* without providing any examples to LLMs is not practically achievable in a zero-shot context. Therefore, in this section, we evaluate the results based on *Ans* A. Table 3 showcases significant findings in this scenario, with scores of **0.72** and **0.55** for tasks 1 and 2, respectively. These scores suggest that LLMs can offer reasonably accurate answers through implicit reasoning when provided with useful information(either with only pertinent information or in conjunction with distractors.). However, they struggle to clearly explain how they arrived at their answers (see the superior results in table 3 compared to table 2). Regardless of the output format, it becomes evident that the presence of distractor information can pose challenges for models and adversely affect their performance. Importantly, the results indicate that the provision of relevant information can enable models to generate estimations, whereas the absence of such knowledge reduces LLMs to a trivial baseline level. In essence, [17] reported a constant model that predicts a random value (a logarithmic sweep between 10^{10} and 10^{-10}), which can achieve a result of **0.22** of *Ans* A.

4.2. Qualitative Results

Figure 2 illustrates various examples generated by fine-tuned and few-shot models for tasks 1 and 3. In the leftmost example in task 1, we observe that the few-shot model correctly produced a program, while the fine-tuned model made an error concerning the selection of the appropriate mathematical operation, mistakenly choosing multiplication instead of division to derive the answer. Conversely, on the right side of the figure, the opposite situation occurred, where the few-shot model erroneously included the original question in the decomposition question and incorporated it into the calculation (Program: div(Q1, Q2, Q3)). More interesting, in task 3 where models have the freedom to forge their unique path of reasoning, leveraging their supportive facts and calculations, the left example showcases the performance of the few-shot model. In this task, the few-shot model interprets the question and deduces a suitable estimated number based on its own knowledge. Notably, in this particular example, the few-shot learner employs “liters” as a unit of measurement to provide the answer. Conversely, the fine-tuned model, while correctly identifying the appropriate mathematical operation (division) for the two facts, falls slightly short in its generation of numbers and decomposition questions. Finally, in the rightmost example, where the question can be considered more challenging than the other examples, the fine-tuned model encountered a significant issue. It duplicated its generation, resulting in an answer that lacked reasonability, essentially attempting to mimic the structure it learned during the fine-tuning phase. Conversely, the few-shot model demonstrated a more

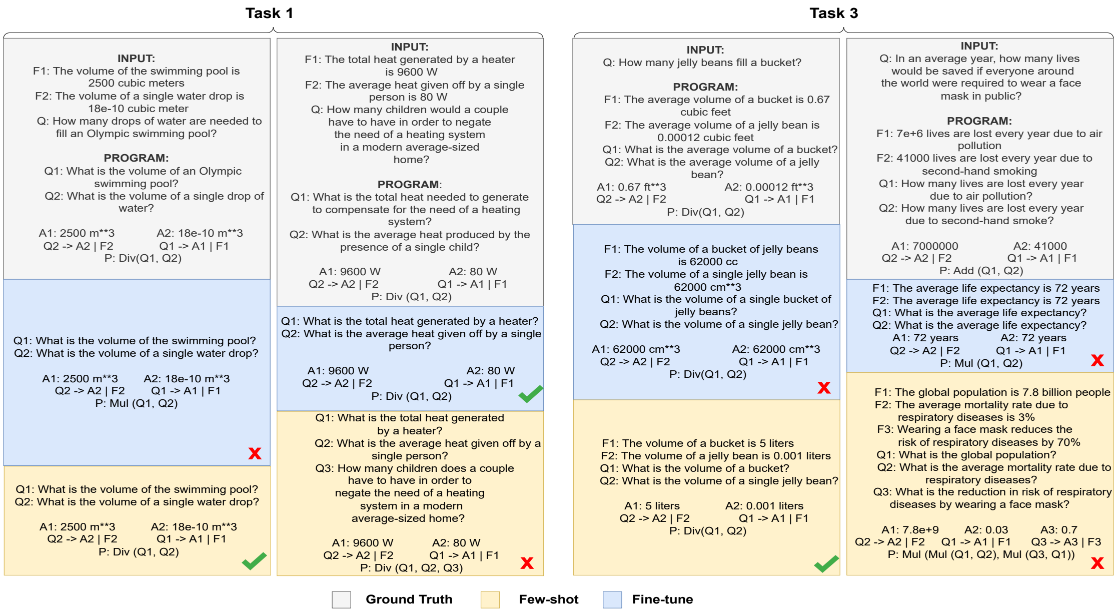
Task 1
INPUT:
F1: The volume of the swimming pool is 2500 cubic meters
F2: The volume of a single water drop is 18e-10 cubic meter
Q: How many drops of water are needed to fill an Olympic swimming pool?
PROGRAM:
Q1: What is the volume of an Olympic swimming pool?
Q2: What is the volume of a single drop of water?
A1: 2500 m**3
A2: 18e-10 m**3
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div(Q1, Q2)
Q1: What is the volume of the swimming pool?
Q2: What is the volume of a single water drop?
A1: 2500 m**3
A2: 18e-10 m**3
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Mul (Q1, Q2)
✗
Q1: What is the volume of the swimming pool?
Q2: What is the volume of a single water drop?
A1: 2500 m**3
A2: 18e-10 m**3
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div (Q1, Q2)
✓
INPUT:
F1: The total heat generated by a heater is 9600 W
F2: The average heat given off by a single person is 80 W
Q: How many children would a couple have to have in order to negate the need of a heating system in a modern average-sized home?
PROGRAM:
Q1: What is the total heat needed to generate to compensate for the need of a heating system?
Q2: What is the average heat produced by the presence of a single child?
A1: 9600 W
A2: 80 W
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div (Q1, Q2)
Q1: What is the total heat generated by a heater?
Q2: What is the average heat given off by a single person?
A1: 9600 W
A2: 80 W
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div (Q1, Q2)
✓
Q1: What is the total heat generated by a heater?
Q2: What is the average heat given off by a single person?
Q3: How many children does a couple have to have in order to negate the need of a heating system in a modern average-sized home?
A1: 9600 W
A2: 80 W
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div (Q1, Q2, Q3)
✗
Task 3
INPUT:
Q: How many jelly beans fill a bucket?
PROGRAM:
F1: The average volume of a bucket is 0.67 cubic feet
F2: The average volume of a jelly bean is 0.00012 cubic feet
Q1: What is the average volume of a bucket?
Q2: What is the average volume of a jelly bean?
A1: 0.67 ft**3
A2: 0.00012 ft**3
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div(Q1, Q2)
F1: The volume of a bucket of jelly beans is 62000 cc
F2: The volume of a single jelly bean is 62000 cm**3
Q1: What is the volume of a single bucket of jelly beans?
Q2: What is the volume of a single jelly bean?
A1: 62000 cm**3
A2: 62000 cm**3
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div(Q1, Q2)
✗
F1: The volume of a bucket is 5 liters
F2: The volume of a jelly bean is 0.001 liters
Q1: What is the volume of a bucket?
Q2: What is the volume of a single jelly bean?
A1: 5 liters
A2: 0.001 liters
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Div(Q1, Q2)
✓
INPUT:
Q: In an average year, how many lives would be saved if everyone around the world were required to wear a face mask in public?
PROGRAM:
F1: 7e+6 lives are lost every year due to air pollution
F2: 41000 lives are lost every year due to second-hand smoking
Q1: How many lives are lost every year due to air pollution?
Q2: How many lives are lost every year due to second-hand smoke?
A1: 7000000
A2: 41000
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Add (Q1, Q2)
F1: The average life expectancy is 72 years
F2: The average life expectancy is 72 years
Q1: What is the average life expectancy?
Q2: What is the average life expectancy?
A1: 72 years
A2: 72 years
Q2 -> A2 | F2
Q1 -> A1 | F1
P: Mul (Q1, Q2)
✗
F1: The global population is 7.8 billion people
F2: The average mortality rate due to respiratory diseases is 3%
F3: Wearing a face mask reduces the risk of respiratory diseases by 70%
Q1: What is the global population?
Q2: What is the average mortality rate due to respiratory diseases?
Q3: What is the reduction in risk of respiratory diseases by wearing a face mask?
A1: 7.8e+9
A2: 0.03
A3: 0.7
Q2 -> A2 | F2
Q1 -> A1 | F1
Q3 -> A3 | F3
P: Mul (Mul (Q1, Q2), Mul (Q3, Q1))
✗
□ Ground Truth
□ Few-shot
□ Fine-tune

Figure 2: Sample generated examples for tasks 1 and 2 using fine-tuning and few-shot Learning. The blue components represent answers obtained from a 5-shot *GPT-3.5-turbo*, and the orange components show the answers generated by the fine-tuned *FLAN-T5-base* model. The ground truth for each example is depicted in gray.

meaningful expansion of its decomposition, even though it failed to identify a correct path to the original question, yielding an incorrect result. Based on these examples, it becomes evident that models, especially larger ones like GPT-based models, have the capacity to retrieve relevant information, but the relationship between such information requires further investigation.

4.3. Fine-tuning vs Few-shot Learning

Figure 3 illustrates the fine-tuned model's behavior in generating program hop counts when compared to the few-shot model across all tasks (denoted as Fp's tasks). In Task 1, depicted as the leftmost image, where all relevant facts are provided, the fine-tuned model generated programs with hop counts that closely matched the two most frequent counts (specifically, 2-hop and 4-hop). This suggests that fine-tuning may be influenced by an unbalanced distribution of examples and may tend to produce the most common strategy learned during the fine-tuning phase. Furthermore, this tendency can explain the high value of the "valid?" score (0.86) for this model. In contrast, the few-shot model appears to mimic the ground-truth behavior more closely. For instance, in Task 1, the few-shot model generates answers with hop counts of 1 and 5, which the fine-tuned model did not produce. Interestingly, in Task 2, where the fine-tuned model displays a similar behavior, the few-shot learning approach generates answers with higher hop counts (e.g., 7, 8, 9) due to the increased number of facts in this task. Furthermore, in Task 3, where models are given the freedom to explore their own reasoning methods, both models exhibit a tendency to generate programs with a lower number of hops. This behavior is

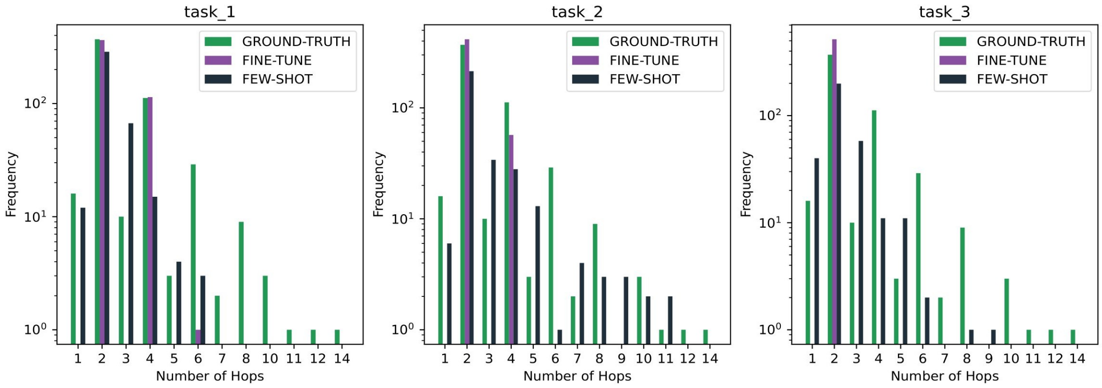

Figure 3: Frequency of hop counts (number of decompositions in program) in both Fine-tune and Few-shot settings on the real test set for tasks 1 to 3. Ground-truth data is represented in green, Fine-tune results correspond to FLAN-T5-base (real), and Few-shot results correspond to GPT-3.5-turbo (5-shot).

noteworthy and may provide insight into why these models face challenges with this task. The inclination towards simpler answers in response to complex questions, such as in the case of Task 3, can potentially lead to suboptimal results.

5. LLMs' Limitations & Possible Solutions

Facts' Units Identifiers: One common issue found in LLMs is the mishandling of FP identifiers. This often results in incorrect responses, where LLMs choose mathematical operations for facts that lack compatibility or fail to yield a correct answer. For example, in Figure 2 leftmost hand side, a fine-tuned model selected multiplication despite the query explicitly requesting a unitless numerical answer. A potential solution involves integrating fact's unit checking into the generation process, using a multi-task approach or a non-sequential procedure, as shown in [19] for addressing equation-based questions with a tree-structured decoder.

Reasoning Deadlock (LLMs' Predicament): A significant challenge in LLMs is their performance in Task 3 especially in the Few-shot setting, where they must provide autonomous reasoning. In this task, LLMs often struggle with their reasoning processes, leading to invalid or incorrect answers. Common issues, including but not limited to, raising sub-questions without answers, assigning numeric values to undecomposed questions, failing to link supportive facts to questions, and condensing multiple mathematical relations into a single equation, such as "Div(Q1, Mul(Q2, Q3), Q4)" which increases the likelihood of encountering impossible equations. An effective approach to address this issue involves the deployment of LLMs within iterative loops, as opposed to requiring them to perform reasoning in a single comprehensive round. Utilizing LLMs in multiple rounds ensures the consistency and reliability of the generated answers. This perspective can yield significant benefits, as the model becomes progressively informed about the components it has expanded upon and those that remain unexplored, resulting in a more coherent and accurate reasoning process.

6. Related Work

Recently, LLMs have emerged as a promising avenue for improving reasoning tasks, particularly in the context of retrieving relevant and supportive information. For instance, [26] demonstrates that LLMs possess the ability to discern implicit relations. They achieve this by decoupling the process of inferring reasoning steps from their execution. This partially aligns with our findings for task 3 (implicit reasoning) of FPs, where LLMs appear to be more successful at retrieving information than conducting reasoning over the retrieved information. Additionally, there have been attempts to enhance existing reasoning and QA tasks by generating intermediate knowledge to facilitate multi-hop reasoning. In their work, [27, 28] employ LLMs to create benchmarks, and they find that this technique can improve both the performance of models and their interpretability. In line with these efforts, other studies [29, 30] have demonstrated that incorporating intermediate supervisory information into the input can enhance the performance of such models. This aligns with our findings, wherein the fine-tuning of task 1, incorporating supportive facts concatenated with the input yielded superior results than the absence of such knowledge. Furthermore, [31] introduced an agent communication mechanism for addressing complex reasoning questions. In this approach, a model engages in a series of QA interactions with agents, such as TextQA and TableQA, to arrive at the final answer. However, this approach has the potential to harness the capabilities of LLMs as agents for solving FPs. Nevertheless, accomplishing this without auxiliary supervision remains a challenging endeavor, necessitating significant dataset modifications to adapt it for FPs. In addition, exploring other directions in the context of complexity is also noteworthy. In terms of complexity, some studies, such as [32], propose that introducing a progressive task complexity framework can yield advantages for LLMs. [33] proposed TELeR, a general guideline for LLMs in prompt designing to perform complex tasks. However, Fermi Problems regard this complexity as comprising three distinct and isolated tasks. The possibility of merging these complex paradigms represents a promising avenue for future research, which we defer to further exploration. Finally, similar to our work, [34] also assessed the capabilities of LLMs in the domain of logical reasoning tasks. They carried out their experiments in a few-shot learning scenario, utilizing methodologies such as prompt engineering and CoT. However, it's important to note that their primary focus was solely on logical reasoning, whereas our study diverges by primarily delving into mathematical reasoning, particularly within the context of Fermi Problems.

7. Conclusion

This paper explored the performance of LLMs in various FP scenarios, including fine-tuning, few-shot, and zero-shot settings. Our findings indicate that despite the advancements in LLMs, there is still a need for further enhancements to enable these models to exhibit creativity at a level comparable to human capabilities

References

[1] E. Bassignana, D. Brunato, M. Polignano, A. Ramponi, Preface to the Seventh Workshop on Natural Language for Artificial Intelligence (NL4AI), in: Proceedings of the Seventh Workshop on Natural Language for Artificial Intelligence (NL4AI 2023) co-located with 22th International Conference of the Italian Association for Artificial Intelligence (AI* IA 2023), 2023.

[2] R. Fagin, J. Y. Halpern, Y. Moses, M. Vardi, Reasoning about knowledge, MIT press, 2004.

[3] J. Huang, K. C.-C. Chang, Towards reasoning in large language models: A survey, arXiv preprint arXiv:2212.10403 (2022).

[4] J. L. Kolodner, An introduction to case-based reasoning, Artificial intelligence review 6 (1992) 3–34.

[5] H. Prakken, G. Sartor, A dialectical model of assessing conflicting arguments in legal reasoning, Logical models of legal argumentation (1997) 175–211.

[6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[7] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, arXiv preprint arXiv:1910.13461 (2019).

[8] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, The Journal of Machine Learning Research 21 (2020) 5485–5551.

[9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[10] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971 (2023).

[11] K. Valmeekam, A. Olmo, S. Sreedharan, S. Kambhampati, Large language models still can't plan (a benchmark for llms on planning and reasoning about change), arXiv preprint arXiv:2206.10498 (2022).

[12] T. Ullman, Large language models fail on trivial alterations to theory-of-mind tasks, arXiv preprint arXiv:2302.08399 (2023).

[13] C. Helwe, C. Clavel, F. Suchanek, Reasoning with transformer-based models: Deep learning, but shallow reasoning, in: International Conference on Automated Knowledge Base Construction (AKBC), 2021.

[14] F. Shi, X. Chen, K. Misra, N. Scales, D. Dohan, E. H. Chi, N. Schärli, D. Zhou, Large language models can be easily distracted by irrelevant context, in: International Conference on Machine Learning, PMLR, 2023, pp. 31210–31227.

[15] T. Mihaylov, P. Clark, T. Khot, A. Sabharwal, Can a suit of armor conduct electricity? a new dataset for open book question answering, arXiv preprint arXiv:1809.02789 (2018).

[16] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, et al., Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, arXiv preprint arXiv:2206.04615 (2022).

[17] A. Kalyan, A. Kumar, A. Chandrasekaran, A. Sabharwal, P. Clark, How much coffee was consumed during emnlp 2019? fermi problems: A new reasoning challenge for ai, arXiv preprint arXiv:2110.14207 (2021).

[18] J. Wei, X. Wang, D. Schuurmans, M. Bosma, brian ichter, F. Xia, E. H. Chi, Q. V. Le, D. Zhou, Chain of thought prompting elicits reasoning in large language models, in: A. H. Oh, A. Agarwal, D. Belgrave, K. Cho (Eds.), Advances in Neural Information Processing Systems, 2022.

[19] Z. Xie, S. Sun, A goal-driven tree-structured neural model for math word problems., in: Ijcai, 2019, pp. 5299–5305.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[21] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, et al., Scaling instruction-finetuned language models, arXiv preprint arXiv:2210.11416 (2022).

[22] T. Schick, H. Schütze, Exploiting cloze questions for few shot text classification and natural language inference, arXiv preprint arXiv:2001.07676 (2020).

[23] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (2019) 9.

[24] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa, Large language models are zero-shot reasoners, Advances in neural information processing systems 35 (2022) 22199–22213.

[25] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[26] U. Katz, M. Geva, J. Berant, Inferring implicit relations with language models, arXiv preprint arXiv:2204.13778 (2022).

[27] E. Zelikman, J. Mu, N. D. Goodman, Y. T. Wu, Star: Self-taught reasoner bootstrapping reasoning with reasoning (2022).

[28] J. Welbl, P. Stenetorp, S. Riedel, Constructing datasets for multi-hop reading comprehension across documents, Transactions of the Association for Computational Linguistics 6 (2018) 287–302.

[29] N. Wies, Y. Levine, A. Shashua, Sub-task decomposition enables learning in sequence to sequence tasks, arXiv preprint arXiv:2204.02892 (2022).

[30] G. Recchia, Teaching autoregressive language models complex tasks by demonstration, arXiv preprint arXiv:2109.02102 (2021).

[31] T. Khot, K. Richardson, D. Khashabi, A. Sabharwal, Hey ai, can you solve complex tasks by talking to agents?, arXiv preprint arXiv:2110.08542 (2021).

[32] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, et al., Show your work: Scratchpads for intermediate computation with language models, november 2021, URL http://arxiv. org/abs/2112.00114 (2021).

[33] S. K. K. Santu, D. Feng, Teler: A general taxonomy of llm prompts for benchmarking complex tasks, arXiv preprint arXiv:2305.11430 (2023).

[34] A. Creswell, M. Shanahan, I. Higgins, Selection-inference: Exploiting large language models for interpretable logical reasoning, arXiv preprint arXiv:2205.09712 (2022).