TheoremQA: A Theorem-driven Question Answering Dataset

Wenhu Chen∗, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, Tony Xia
University of Waterloo, Canada
University of California, Santa Barbara, United States
University of California, Los Angeles, United States

Abstract

The recent LLMs like GPT-4 and PaLM-2 have made tremendous progress in solving fundamental math problems like GSM8K by achieving over 90% accuracy. However, their capabilities to solve more challenging math problems which require domain-specific knowledge (i.e., theorem) have yet to be investigated. In this paper, we introduce TheoremQA, the first theorem-driven question-answering dataset designed to evaluate AI models’ capabilities to apply theorems to solve challenging science problems. TheoremQA is curated by domain experts containing 800 high-quality questions covering 350 theorems1 from Math, Physics, EE&CS, and Finance. We evaluate a wide spectrum of instruction-finetuned language and code models including GPT (Brown et al., 2020), Claude (Bai et al., 2022), LLaMA (Touvron et al., 2023), Pythia (Biderman et al., 2023), CodeGen (Nijkamp et al., 2022), GLM (Zeng et al., 2022), StarCoder (Li et al., 2023), and CodeT5+ (Wang et al., 2023) a narrow subject. On the other hand, these datasets do not involve much domain-specific knowledge, aka theorem. Due to these two deficiencies, we believe that these datasets are not ideal to benchmark the existing powerful LLMs (Brown et al., 2020; Tamkin et al., 2022; Chen et al., 2021b; Chowdhery et al., 2022; Hoffmann et al., 2022; Taylor et al., 2022) due to their simplicity. In fact, on the popular GSM8K dataset (Cobbe et al., 2021), GPT-4 (OpenAI, 2023) and PaLM-2 (Google, 2023) both already achieved 92% accuracy. Similarly, we tested GPT-4 (OpenAI, 2023) on the subsets of several other listed datasets in Table 1 and observed 90+% accuracy in most cases. The only exception is MATH (Hendrycks et al., 2021) containing high-school math competition problems with SoTA performance around 50% (Zheng et al., 2023). However, MATH (Hendrycks et al., 2021) is focused on math skills rather than theorem.

In this paper, we propose the first theorem-driven QA dataset built on university-level theorems across Math, Physics, EE&CS, and Finance. The whole collection process takes two steps: (1) we first enumerate roughly 400 theorems in different subfields like algebra, number theory, graph theory, information theory, etc, (2) we ask domain experts to search for questions regarding these theorems from different sources like Internet and Textbooks. The domain experts will adjust these questions to ensure the answers follow the desired format for the ease of automatic evaluation. Through the careful construction process, we collected 800 high-quality question-theorem-answer triples as our final release version.

We evaluate a wide spectrum of instruction-finetuned language and code models including GPT (Brown et al., 2020), Claude (Bai et al., 2022), LLaMA (Touvron et al., 2023), Pythia (Biderman et al., 2023), CodeGen (Nijkamp et al., 2022), GLM (Zeng et al., 2022), StarCoder (Li et al., 2023), and CodeT5+ (Wang et al., 2023).
on our dataset. We adopt two prompting methods: Chain-of-Thoughts (CoT) (Wei et al., 2022b) and Program-of-Thoughts (PoT) (Chen et al., 2022b) to prompt the large language models. We also investigate how to infuse the theorem into the thought process of LLMs and how to present the multimodal inputs to the LLMs.

In the course of our experiments, several notable observations were made. First, GPT-4 (OpenAI, 2023) significantly outperformed all existing models, reaching an accuracy level of 51% when combined with Program-of-Thoughts prompting. Trailing behind GPT-4, the second most effective model was ChatGPT, achieving an accuracy of 35% through the same prompting method. Additionally, our human evaluation determined that half of GPT-4’s errors are caused by minor mistakes like calculation errors, rounding errors, etc. We believe these errors could be easily rectified with a more deliberate prompting strategy or human intervention. This suggests that there is still significant headroom for GPT-4 to achieve with more deliberate prompting strategies. Secondly, we found that all open-source, instruction-tuned language and code models scored below 15% in accuracy, barely exceeding the random guess baseline of 10%. Our human evaluation reveals that open-source models like Alpaca are making errors mainly due to their ignorance of the theorem, where 90% of the errors are not rectifiable. This stark gap between GPT and open-source models suggests that further enhancement strategies, such as science-focused pre-training or fine-tuning, should be considered to narrow the performance disparity. Thirdly, we explored the potential to do theorem-augmented generation. However, the simple strategy of concatenation did not yield a significant improvement. We conjecture that a more complex integration strategy may be needed to achieve more gains. Lastly, we examined the performance of various multi-modal instruction-tuned models on the multimodal subset of the TheoremQA dataset. Surprisingly, these models did not demonstrate significant performance gains over their text-only counterparts. This is mainly due to the unnaturalness of the image, which consists of lots of diagrams and text. Such images are not well

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Domain</th>
<th>Level</th>
<th>Source</th>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRAW</td>
<td>Algebra</td>
<td>Elementary School</td>
<td>Generated</td>
<td>-</td>
</tr>
<tr>
<td>MAWPS (Koncel-Kedzierski et al., 2016)</td>
<td>Arithmetic</td>
<td>Elementary School</td>
<td>Generated</td>
<td>-</td>
</tr>
<tr>
<td>DRAW1K</td>
<td>Algebra</td>
<td>Elementary School</td>
<td>Generated</td>
<td>-</td>
</tr>
<tr>
<td>ASDiv (Miao et al., 2020)</td>
<td>Arithm/Algebra</td>
<td>Elementary School</td>
<td>Internet</td>
<td>-</td>
</tr>
<tr>
<td>SVAMP (Patel et al., 2021a)</td>
<td>Arithm/Algebra</td>
<td>Elementary School</td>
<td>ASDiv</td>
<td>-</td>
</tr>
<tr>
<td>Math23K (Wang et al., 2017)</td>
<td>Algebra</td>
<td>Elementary School</td>
<td>Internet</td>
<td>-</td>
</tr>
<tr>
<td>TabMWP (Lu et al., 2023b)</td>
<td>Arithm/Algebra</td>
<td>Elem./Middle School</td>
<td>Textbooks</td>
<td>NO</td>
</tr>
<tr>
<td>GSM8K (Cobbe et al., 2021)</td>
<td>Arithm/Algebra</td>
<td>Middle School</td>
<td>Annotated</td>
<td>NO</td>
</tr>
<tr>
<td>GEOS (Seo et al., 2015)</td>
<td>Geometry</td>
<td>Middle School</td>
<td>SAT</td>
<td>NO</td>
</tr>
<tr>
<td>Geometry3K (Lu et al., 2021a)</td>
<td>Geometry</td>
<td>Middle/High School</td>
<td>Textbooks</td>
<td>NO</td>
</tr>
<tr>
<td>GeoQA (Chen et al., 2021a)</td>
<td>Geometry</td>
<td>Middle/High School</td>
<td>Exam</td>
<td>NO</td>
</tr>
<tr>
<td>UniGeo (Chen et al., 2022a)</td>
<td>Geometry</td>
<td>Middle/High School</td>
<td>Textbooks</td>
<td>NO</td>
</tr>
<tr>
<td>ScienceQA (Lu et al., 2022)</td>
<td>Science</td>
<td>Middle/High School</td>
<td>Textbooks</td>
<td>NO</td>
</tr>
<tr>
<td>MATH (Hendryckx et al., 2021)</td>
<td>Math</td>
<td>High School</td>
<td>Competition</td>
<td>YES</td>
</tr>
<tr>
<td>AQuA (Ling et al., 2017)</td>
<td>Arithm/Algebra</td>
<td>University</td>
<td>GMAT/GRE</td>
<td>NO</td>
</tr>
<tr>
<td>MathQA (Amini et al., 2019)</td>
<td>Arithm/Algebra</td>
<td>University</td>
<td>AQuA</td>
<td>NO</td>
</tr>
<tr>
<td>MathQA-Python (Austin et al., 2021)</td>
<td>Arithm/Algebra</td>
<td>University</td>
<td>AQuA</td>
<td>NO</td>
</tr>
<tr>
<td>FinQA (Chen et al., 2021c)</td>
<td>Finance</td>
<td>University</td>
<td>CrowdSource</td>
<td>NO</td>
</tr>
<tr>
<td>TAT-QA (Zhu et al., 2021)</td>
<td>Finance</td>
<td>University</td>
<td>CrowdSource</td>
<td>NO</td>
</tr>
<tr>
<td>TheoremQA (Ours)</td>
<td>STEM</td>
<td>Internet+Expert</td>
<td>350+</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: List of existing Math and STEM QA datasets.
captured by existing visual encoder models. To sum up, our contributions are three folds:

- We propose the first theorem-driven question-answering dataset to understand LLMs’ capabilities to apply science theorems.

- We comprehensively evaluate a wide spectrum of 16 LLMs on TheoremQA.

- We perform different analyses in the theorem integration and multimodal understanding aspects to provide detailed insights.

2 Related Work

2.1 Math Word Problems

Mathematical reasoning skills are crucial for general-purpose intelligent systems, garnering significant interest from the research community. In the past, studies have explored the ability of NLP models to solve arithmetic and algebraic problems (Hosseini et al., 2014; Koncel-Kedziorski et al., 2015; Roy and Roth, 2015; Ling et al., 2017). More recently, researchers have introduced increasingly challenging datasets (Saxton et al., 2019; Miao et al., 2020; Amini et al., 2019; Hendrycks et al., 2021; Lu et al., 2021b; Patel et al., 2021b) aimed at enhancing difficulty, diversity, and adversarial robustness. LiLA (Mishra et al., 2022) proposes to assemble a vast collection of mathematical datasets into a single, unified dataset. LiLA also annotates Python programs as target outputs for solving mathematical problems. However, the existing datasets were mostly focused on grade school simple mathematics. To further investigate the LLMs’ capabilities to assist humans to solve challenging math problems, we propose TheoremQA as the first benchmark to enable research in this direction.

2.2 Large Language Models

In recent years, there has been a surge of research and development in the area of large language models (LLMs) that have significantly advanced the field of natural language processing. GPT-3 (Brown et al., 2020) demonstrated a strong capability to perform few-shot predictions, where the model is given a description of the task in natural language with few examples. By using human-feedback reinforcement learning, InstructGPT (Ouyang et al., 2022) has shown its unprecedented capabilities to follow human instructions. Scaling model size, data, and computing are crucial to enable this learning ability. Later, Rae et al. (2021); Chowdhery et al. (2022); Zhang et al. (2022); Touvron et al. (2023); Chen et al. (2021b) have proposed to train different types of LLMs with different training recipes. The capability to follow few-shot exemplars to solve unseen tasks is not existent on smaller LMs, but only emerges as the model scales up (Wei et al., 2022a). More recently, GPT-4 (OpenAI, 2023) shows tremendous progress on lots of complex reasoning tasks spanning mathematics, coding, vision, medicine, law, psychology, and more. Bubeck et al. (2023) shows that GPT-4 is already demonstrating more general intelligence than previous AI models. To further validate GPT-4’s capability to solve challenging reasoning tasks, we propose TheoremQA as the new benchmark to further understand LLMs’ upper limit.

2.3 Reasoning with Large Language Model

To better unleash large language models’ capabilities to solve complex reasoning tasks. Chain-of-Thought Prompting (Wei et al., 2022b; Kojima et al., 2022; Wang et al., 2022) was proposed, which aims at prompting the large language models to generate the ‘thought process’ before outputting the answer. Later on, several other works (Drozdov et al., 2022; Zhou et al., 2022; Nye et al., 2021) also propose different approaches to utilize LLMs to solve reasoning tasks by allowing intermediate steps. Our method can be seen as an extension to CoT by leveraging an extra step of symbolic execution. Another line of work (Gao et al., 2022; Chen et al., 2022b) was proposed to adopt Python programs as the demonstration for the ‘thought process’ to solve different reasoning tasks.

3 Dataset

Our dataset collection pipeline contains two steps:

Theorem Enumeration Our aim was to encompass a wide range of theorems. To this end, we began by prompting Large Language Models (LLMs), specifically GPT-4 (OpenAI, 2023), to enumerate popular subfields in Mathematics, Physics, Finance, and Electrical Engineering & Computer Science. The covered subfields are listed in Figure 4. Subsequently, we prompted GPT-4 to propose plausible university-level theorems relevant to these subfields. For instance, within the ’Calculus’ subfield, GPT-4 might suggest the ’Intermediate Value Theorem’, ’Rolle’s Theorem’, and so on. After gath-
ering an extensive list of theorems, we assembled a team of domain experts (holders of Masters and PhDs in Statistics, Electrical Engineering, Computer Science, and Finance) to refine the theorem inventory and supplement any omitted theorems. Ultimately, we collected approximately 400 theorems, encapsulating a diverse range of topics within these fields. We then delegated these theorems to nine domain experts, instructing them to locate question/answer pairs from varied sources. During the annotation process, a small number of theorems were discarded due to their evaluation complexity.

**Question Annotation**

Our problems were sourced from websites, books, or devised by the experts themselves. One challenge we encountered was the potential for questions found online to have been included in the training data. To mitigate this ‘data contamination’ issue, we encouraged domain experts to modify these questions. Another challenge arose from questions with answers in symbolic form, matrix form, figure form, etc. These presented significant obstacles for our automatic evaluation. To overcome this, we instructed domain experts to alter the question so the answer would be limited to the following forms: (1) integer, (2) float, (3) list of integers/floats, (4) boolean, and (5) multiple-choice options. For instance, if the original question concerned a matrix, we would revise it to ask about the trace of the answer matrix. This modification significantly streamlined the evaluation process. An example of this can be found in Figure 2.

**Dataset Statistics**

Finally, we collected a total of 800 questions over 354 theorems. Specifically, there are 199 Math theorems, 52 Physics theorems, 55 Finance theorems, and 48 CS&EE theorems. There are 442 Math questions, 146 CS&EE questions, 131 physics questions, and 81 Finance questions. We show the answer-type distribution in Figure 3. To further enhance the multimodality aspect of TheoremQA, we also include 51 questions with image input (diagrams), where the model needs to understand the visual input to answer questions.

The majority of the questions in TheoremQA have float and integer as the answers, which is more realistic than the existing multi-choice datasets like ScienceQA (Lu et al., 2022) or AQuA QA (Ling et al., 2017). Therefore, the models are unlikely to take shortcuts to achieve high accuracy.

**Human-Level Performance**

To provide a rough but informative estimate of human-level performance. We randomly select 20 questions and assign these questions to the 4 Math&CS undergraduate students (average GPA) who have taken the required courses regarding these questions. The participants are given 24 hours with internet access to solve these questions. The four undergraduate students achieve 12/20, 15/20, 18/20, and 19/20 scores on these randomly sampled questions. From this experiment, we are more confident that an expert-level performance should be 100%.

**4 Method**

Our method for addressing these demanding questions in the TheoremQA is comprised of several distinct modules, as outlined in Figure 1:

**Prompting**

We utilize two established prompting strategies:

- Chain-of-Thought Prompting (Wei et al., 2022)
Figure 3: Answer type distribution in TheoremQA.

2022b): This strategy prompts the language model to initially generate a step-by-step thought process, eventually leading to the final answer.

• Program-of-Thought Prompting (Chen et al., 2022b; Gao et al., 2022): This strategy prompts the language model to progressively generate a program. The final answer is then derived by executing this program.

By delegating computational tasks to an external executor, the problem-solving process is considerably enhanced in its reliability. This improvement results in remarkable advancements in existing math datasets being reported in (Chen et al., 2022b).

Answer Extraction We observed that parsing the output from Large Language Models (LLMs) can be challenging due to two main issues: (1) The answer is often embedded within a sentence, making it difficult to extract using regular expressions, and (2) The answer may not be normalized, such as '\pi / 3' or '2*10 - e', which complicates comparison with the ground truth. To tackle these problems, we initially employ ChatGPT to identify the answer span within the model’s output, then forward this string to WolframAlpha (Inc.) for normalization into a float, integer, or list.

Theorem Augmentation We explored the potential of enhancing large language models with retrieved theorem descriptions to assess their effect on performance. One approach is to retrieve descriptions of the given theorems from the Internet to supplement the LLMs’ output. Another experiment involved prompting GPT-4 to generate text descriptions of the theorem, which are then used as an additional augmentation signal.

Multimodal Input A small portion of our data (50 instances) includes images, such as diagrams, as supplemental input, particularly in geometry questions. Since current LLMs don’t support such multimodal inputs, we propose a solution: to employ captions like Chameleon (Lu et al., 2023a). These captions describe the image and are then appended to the LLMs’ output as an additional signal.

5 Experiments

5.1 Model Descriptions

In our experiments, we mainly investigate the following models:

• GPT3/3.5/ChatGPT/GPT4: These are instruction-tuned models from OpenAI.

• PaLM-2: This is the instruction-tuned model from Google (Google, 2023).

• Claude-v1/Claude-instant: These are instruction-tuned models from AnthropicAI.

• Alpaca-13B: This model is based on the LLaMA (Touvron et al., 2023). Alpaca is instruction-tuned by the 52K data generated from GPT-4.

• Vicuna-13B: This model is based on the LLaMA (Touvron et al., 2023). Vicuna is instruction-tuned by the 100K ShareGPT data generated by different GPT-based models.

• OpenAssistant-12B: This model is based on Pythia (Biderman et al., 2023). The model is instruction-tuned by OpenAssistant data.

• MOSS-instruct-16B: This model is based onCodeGen (Nijkamp et al., 2022), which is further instruction-tuned with instruction following dataset distilled from GPT.

• StarChat-16B: This model is based on StarCoder (Li et al., 2023). StarChat is being instruction-tuned on OpenAssistant data and ShareGPT data.

---

• InstructCodeT5+: This model is based on CodeT5+ (Wang et al., 2023). InstructCodeT5+ is further instruction-tuned on Code Alpaca data to follow instructions.

5.2 Main Results

We demonstrate our main results on Table 2. We will summarize different findings in the following:

Closed-source Models For GPT-3 (text-davinci-002) and GPT-3.5 model, since these two models are not Chat-based models, we need to demonstrate one example ensure to help them generate outputs of the desired format. With CoT prompting, GPT-3 (text-davinci-002) and GPT-3.5 models are only achieving 16.6% and 22.8% accuracy. By adopting the program as the intermediate reasoning form, both models can gain reasonable improvements. For Claude-v1, we found that it is matching the performance of GPT-3.5. ChatGPT outperforms GPT-3.5 and Claude-v1 significantly by 8%, which indicates ChatGPT’s capabilities to perform complex numerical reasoning. GPT-4 is the strongest model being evaluated, which beats all the rest models by a huge margin. With Chain-of-Thoughts prompting, GPT-4 can outperform ChatGPT by 13%. With Program-of-Thoughts prompting, GPT-4 can outperform ChatGPT by 16%. Though some other models have shown to match GPT-4 on simple tasks, GPT-4’s capability to solve challenging tasks seems unparalleled.

Open-source Models For the open-source models, we found that their performance is much behind. To better understand their accuracy, we also provide the random-guess baseline of 10%. We test both prompting strategies, however, their results consistently lie in the range of 10-14%. The results indicate that these open-source LMs are still struggling with more complex mathematical reasoning tasks in TheoremQA. Given that ChatGPT of a similar size is able to achieve much higher performance, we believe the parameter size is not the only cause. There is still a significant amount of effort during pre-training or supervised fine-tuning to install enough scientific knowledge into the models’ parameters to close the gap.

Program of Thoughts Analysis From Table 2, we observe that PoT brings consistent improvement over CoT on GPT-* models. Different GPT-* models can normally yield a gain of 5-8% accuracy. In contrast, Claude-v1 and StarChat are almost obtaining the same accuracy. To better analyze where the gains are coming from, we plot Figure 5 to understand how many of generated Python programs are actually ‘executable’. As can be seen, both StarChat and CodeT5+ are having trouble generating ‘runnable’ programs with only 40% programs being executable. Claude-v1 is able to increase the validity of the generated programs to 60%. In contrast, GPT3.5 and ChatGPT can further increase the ratio to around 80%. GPT-4 is extremely accurate in generating programs, where 92% of the generated programs are runnable. Such a high executable ratio explains why the gain brought to GPT-* model

<table>
<thead>
<tr>
<th>Mathematics</th>
<th>Physics</th>
<th>Finance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculus 103</td>
<td>Kinematics 30</td>
<td>Economics 22</td>
</tr>
<tr>
<td>Combinatorics 57</td>
<td>Electromagnetism 21</td>
<td>Quantitative methods 14</td>
</tr>
<tr>
<td>Algebra 52</td>
<td>Atomic physics 11</td>
<td>Derivatives 14</td>
</tr>
<tr>
<td>Mathematical analysis 42</td>
<td>Wave 8</td>
<td>Fixed income 11</td>
</tr>
<tr>
<td>Number theory 29</td>
<td>Optics 8</td>
<td>Management 10</td>
</tr>
<tr>
<td>Geometry 27</td>
<td>Condensed matter 8</td>
<td>Investments 10</td>
</tr>
<tr>
<td>Numerical analysis 24</td>
<td>Particle 6</td>
<td>Total 81</td>
</tr>
<tr>
<td>Statistics 24</td>
<td>Statistical physics 6</td>
<td></td>
</tr>
<tr>
<td>Complex analysis 21</td>
<td>Relativity 7</td>
<td></td>
</tr>
<tr>
<td>Probability theory 19</td>
<td>Celestial mechanics 6</td>
<td>Signal processing 47</td>
</tr>
<tr>
<td>Stochastic process 16</td>
<td>Thermodynamics 5</td>
<td>Graph theory 34</td>
</tr>
<tr>
<td>Group theory 11</td>
<td>Quantum 5</td>
<td>Information theory 29</td>
</tr>
<tr>
<td>Functional analysis 10</td>
<td>Classic mechanics 5</td>
<td>Computer networking 23</td>
</tr>
<tr>
<td>Real analysis 7</td>
<td>Fluid mechanics 5</td>
<td>Machine learning 13</td>
</tr>
<tr>
<td>Total 442</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CS & EE

Figure 4: Subfields of TheoremQA under Math, Physics, Engineering, and Finance.

7894
is much higher than Claude-v1 and StarChat.

5.3 Additional Result

Theorem Augmentation We also investigate whether feeding theorem as an additional text condition would help the model better solve the problem. Specifically, we ask GPT-4 to generate a paragraph to describe the theorem, which we postprocessed to ensure correctness. We feed the theorem in the prompt to different language models to see the performance change and plot our findings in Table 3. For all the evaluated scenarios, we found that the improvement is limited to within 1%. Unlike the Text or KB knowledge, theorem knowledge is more abstract and symbolic, simply concatenating the theorem definition is not enough. We believe a more sophisticated augmentation scheme is needed to truly help the model understand and apply the theorems to solve problems.

Multimodal Questions Our aim was to assess how effectively the current method could tackle multimodal questions (those with image inputs) in the TheoremQA dataset. An example is illustrated in Figure 6, where an image is converted into 'captions' by BLIP (Li et al., 2022). We graphed the results from over 50 multimodal question subsets in Figure 7. Notably, this subset posed substantial challenges; none of the models were able to achieve an accuracy rate of 10%. This is primarily due to information loss during the captioning process.

In light of this, we conducted further evaluations on two multimodal instruction-tuned models, LLaVA-13B (Liu et al., 2023) and VisualGLM-6B (Zeng et al., 2022). These models utilize a visual encoder (either CLIP (Radford et al., 2021) or BLIP (Li et al., 2022)) to encode image input, which is then integrated with language models for multimodal conversation. However, these models demonstrated performance similar to their text-only equivalent, Alpaca, with the addition of a visual encoder not significantly enhancing the results. We hypothesize that the current visual encoding modules may not be suited for representing these diagrammatic images, resulting in these less than ideal outcomes. We believe these multimodal questions remain a challenge for the research community, and we eagerly anticipate further advancements in addressing these multimodal scientific questions.

Error Analysis We conduct detailed error analysis on 200 erroneous cases from different models to analyze their error distribution. Specifically, we pick GPT4, ChatGPT and Alpaca to understand their error sources. We include the following error types: (E1) the model does not even know this theorem, (E2) the model does know the theorem, but uses the wrong formula or algorithm, (E3) the model knows the theorem and the formula, the error is only caused by minor calculation mistakes. The severity of the error decrease as the error number increases. We plot our findings in Figure 8, where the bar indicate the percentage of specific error types. We can observe that almost half of the errors made by GPT4 are non-critical with caused by minor calculation mistakes. This error analysis suggests that there is a still a significant headroom for GPT4 to improve with more deliberate prompting strategies or human intervention to mitigate these minor errors. In contrast, Alpaca’s errors are mainly caused by not knowing the theorem at all.

8https://github.com/THUDM/VisualGLM-6B
### Table 2: Results for CoT and PoT prompting on TheoremQA. We report the accuracy over different fine-grained question types and scientific fields.

<table>
<thead>
<tr>
<th>Model</th>
<th>Method</th>
<th>Theorem</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChatGPT</td>
<td>CoT</td>
<td>-</td>
<td>30.2</td>
</tr>
<tr>
<td>ChatGPT</td>
<td>CoT</td>
<td>+</td>
<td>30.8</td>
</tr>
<tr>
<td>Claude-v1</td>
<td>CoT</td>
<td>-</td>
<td>24.9</td>
</tr>
<tr>
<td>Claude-v1</td>
<td>CoT</td>
<td>+</td>
<td>25.4</td>
</tr>
<tr>
<td>ChatGPT</td>
<td>PoT</td>
<td>-</td>
<td>35.6</td>
</tr>
<tr>
<td>ChatGPT</td>
<td>PoT</td>
<td>+</td>
<td>35.8</td>
</tr>
<tr>
<td>Alpaca (13B)</td>
<td>CoT</td>
<td>-</td>
<td>13.5</td>
</tr>
<tr>
<td>Alpaca (13B)</td>
<td>CoT</td>
<td>+</td>
<td>14.2</td>
</tr>
</tbody>
</table>

### Table 3: Results for CoT and PoT prompting with additional theorem conditions.

<table>
<thead>
<tr>
<th>Model</th>
<th>Method</th>
<th>Theorem</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPT-3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPT-3.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ChatGPT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPT-4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PaLM-2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Claude-v1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Claude-instant</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alpaca (13B)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vicuna (13B)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenAssistant (12B)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOSS (16B)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StarChat (16B)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Case Study
We list a few successful and failed examples generated by GPT-4 in Figure 9 to do a side-by-side comparison between chain-of-thoughts prompting and program-of-thoughts prompting. In the first example, the question is regarding ‘orthogonal projection theorem’. As can be seen, Chain-of-Thoughts prompting requires a very long paragraph to generate the results. We prompted GPT-4 a few times with the same input and the results seem unstable. Sometimes the model will make tiny computation mistakes in the middle to derive the wrong answer. In contrast, the program solution is brief and concise, which leads to rather stable outputs. For the second example, the computation requires ‘for loop’ to iteratively compute delta values for Riemann Sum. We found that such problems are also more natural for programs to solve. Through these examples, we can see GPT-4’s unprecedented capabilities to solve these difficult math problems even without any demonstration or hints.

We also show some examples in Figure 9 to compare the results of CoT and PoT prompting. We can see that the PoT can significantly shorten the output sequence length. By leveraging the additional tool, PoT is able to significantly lower the task difficulty.
6 Conclusion

In this paper, we propose the first theorem-driven science question-answering dataset and evaluate different LLMs on it. Though GPT-4 can achieve strong performance on our new dataset, the existing open-source LLMs are still struggling to achieve reasonable performance. We conjecture it is essential to leverage more science-related pre-training or fine-tuning to close the gap. On the hand, we found that the multimodal science questions are still extremely challenging for the existing visual LLMs. We believe more specialized visual encoding models are needed to better represent diagrams in these science questions.
Limitations

In this work, we explore the possibilities to utilize different large language models to solve challenging theorem-driven questions. There are some limitations: (1) our answer extraction is still not perfect. There are some cases where our answer extractor is not able to locate the answer. Thus the final accuracy is still an approximate lower bound. (2) in our dataset collection, we specifically avoid the hard-to-evaluate cases where the answer is a formula, figure, or a matrix. Our choice of the questions can be biased in terms of evaluating the overall ability. (3) in the multimodal questions in TheoremQA, we have investigated different existing models but none of them succeed in achieving reasonable performance.

References


7898


