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 theorems[1] from Math, Physics, EE&CS, and Finance. We evaluate a wide spectrum of 16 large language and code models with different prompting strategies like Chain-of-Thoughts and Program-of-Thoughts. We found that GPT-4's capabilities to solve these problems are unparalleled, achieving an accuracy of 51% with Program-of-Thoughts Prompting. All the existing open-sourced models are below 15%, barely surpassing the random-guess baseline. Given the diversity and broad coverage of TheoremQA, we believe it can be used as a better benchmark to evaluate LLMs' capabilities to solve challenging science problems.

1 Introduction

A long-standing goal of AI systems is to help human beings solve challenging problems, especially more domain-specific problems. To benchmark the progress towards this goal, researchers propose to evaluate AI systems' performance on different math word problem (WMP) datasets. In recent years, there has been a plethora of WMP datasets (Lu et al., 2023c), which we include in Table 1. Most of these datasets are meant for fundamental questions aimed at Grade 1-12 students on 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)

* Authors ordered by contribution. Corresponding author email: wenhuchen@uwaterloo.ca

[1] e.g. Taylor's theorem, Lagrange's theorem, Huffman coding, Quantum Theorem, Elasticity Theorem, etc

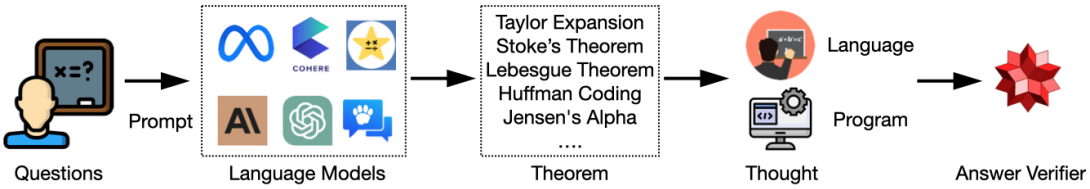


Figure 1: The overview of TheoremQA and the prompting strategies adopted.

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

Table 1: List of existing Math and STEM QA datasets.

on our dataset. We adopt two prompting methods: Chain-of-Thoughts (CoT) (Wei et al., 2022b) and Program-of-Thoughs (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 multi-modal 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

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.

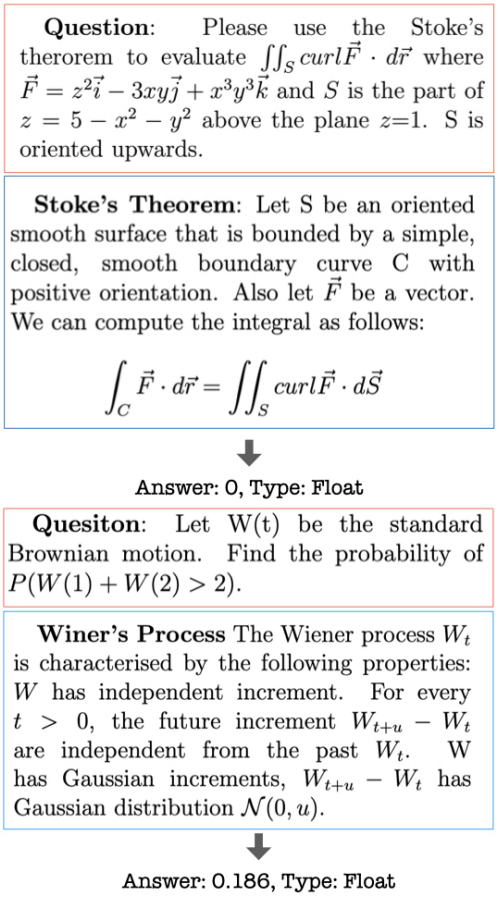


Figure 2: Examples from `TheoremQA`. The first question requires the usage of Stoke's theorem to transform the double integral into a line integral. The second question requires knowing the properties of Wiener's process.

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.,

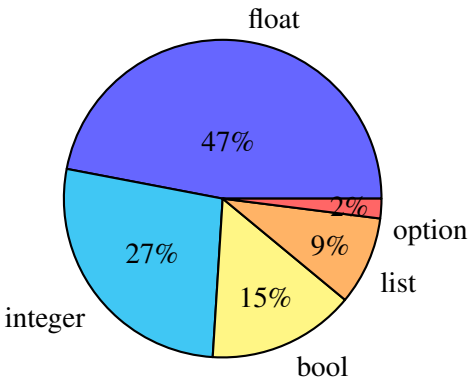


Figure 3: Answer type distribution in TheoremQA.

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[2].
- PaLM-2: This is the instruction-tuned model from Google (Google, 2023).
- Claude-v1/Claude-instant: These are instruction-tuned models from AnthropicAI[3].
- Alpaca-13B: This model is based on the LLaMA (Touvron et al., 2023). Alapaca 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[4].
- MOSS-instruct-16B: This model is based on CodeGen (Nijkamp et al., 2022), which is further instruction-tuned with instruction following dataset distilled from GPT.[5].
- StarChat-16B: This model is based on StarCoder (Li et al., 2023). StartChat is being instruction-tuned on OpenAssistant data[6] and ShareGPT data.

[2]https://openai.com/
[3]https://www.anthropic.com/index/introducing-claude
[4]https://open-assistant.io/
[5]https://txsun1997.github.io/blogs/moss.html
[6]https://open-assistant.io/

Mathematics		Physics		Finance	
Calculus	103	Kinetics	30	Economics	22
Combinatorics	57	Electromagnetism	21	Quantitive methods	14
Algebra	52	Atomic physics	11	Derivatives	14
Mathematical analysis	42	Wave	8	Fixed income	11
Number theory	29	Optics	8	management	10
Geometry	27	Condensed matter	8	Investments	10
Numerical analysis	24	Particle	6	Total	81
Statistics	24	Statistical physics	6		
Complex analysis	21	Relativity	7	**CS & EE**	
Probability theory	19	Celestial mechanics	6	Signal processing	47
Stochastic process	16	Thermodynamics	5	Graph theory	34
Group theory	11	Quantum	5	Information theory	29
Functional analysis	10	Classic mechanics	5	Computer networking	23
Real analysis	7	Fluid mechanics	5	Machine learning	13
Total	442	Total	131	Total	146

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

- InstructCodeT5+: This model is based on CodeT5+ (Wang et al., 2023). InstructCodeT5+ is further instruction-tuned on Code Alpaca data[7] 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.

[7] https://github.com/sahil280114/codealpaca

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 instill 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

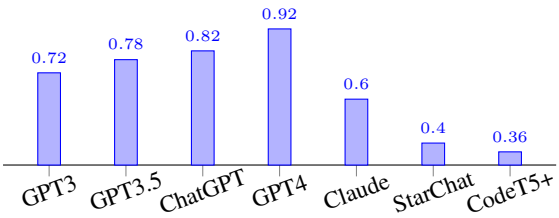


Figure 5: Ratio of Executable Python Program of different models with PoT prompting.

is much higher than Claude-v1 and StarChat.

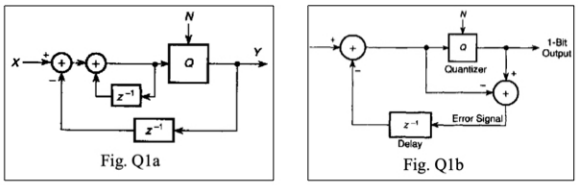
5.3 Additional Result

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 post-processed 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)[8]. 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

[8]https://github.com/THUDM/VisualGLM-6B



Question: Are the circuits shown in Fig. Qla and Fig. Q1b are identical in terms of the Tranfer functions.

BLIP Caption: a diagram of a block diagram with a block diagram and a block diagram.

Figure 6: An example of Multimodal question.

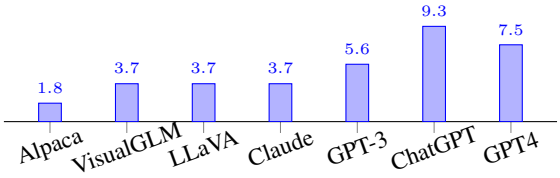


Figure 7: Accuracy on the Multimodal Question Subset

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.

Model	Integer	Float	Option	List	Bool	Math	CS&EE	Physics	Finance	All
Random Guess	0	0	38.9	0	65.5	10.0	24.7	0	4.9	10.5
Chain of Thoughts (CoT)										
GPT-3	11.6	11.7	27.8	6.8	46.6	15.8	34.2	2.3	12.3	16.6
GPT-3.5	13.0	14.3	50.0	13.7	69.8	22.6	36.3	7.6	23.5	22.8
ChatGPT	32.4	22.3	50.0	20.5	55.2	31.0	41.1	16.8	28.4	30.2
GPT-4	40.3	36.7	66.7	37.0	74.6	43.9	50.6	30.5	51.4	**43.8**
PaLM-2	26.4	22.8	61.1	23.3	71.6	31.0	47.3	19.8	27.2	31.8
Claude-v1	18.1	19.4	27.8	15.1	61.2	21.7	42.5	13.7	28.4	24.9
Cluade-instant	19.9	16.7	44.4	17.8	53.4	21.5	36.3	14.5	27.2	23.6
Alpaca (13B)	11.1	6.9	27.8	2.7	45.7	12.9	27.4	3.8	9.9	13.5
Vicuna (13B)	8.8	6.9	16.7	2.7	45.7	12.2	24.0	3.1	12.3	12.9
OpenAssistant (12B)	8.3	5.0	22.2	1.4	37.9	10.2	25.0	0	4.9	10.7
MOSS (16B)	8.8	5.4	24.2	2.4	44.2	11.3	28.4	1.6	8.9	12.2
StarChat (16B)	7.9	4.9	22.3	1.9	44.1	10.7	23.5	0.6	6.8	11.6
Program of Thoughts (PoT)										
GPT-3	17.1	15.9	22.2	9.6	49.1	23.3	25.4	8.4	17.3	20.6
GPT-3.5	23.6	19.9	50.0	21.9	61.2	26.7	41.1	14.5	30.9	27.8
ChatGPT	31.0	35.0	38.9	21.9	54.3	35.7	35.6	26.7	49.4	35.6
GPT-4	44.4	50.7	66.7	39.7	78.4	52.0	51.4	45.8	66.7	**52.4**
Claude-v1	17.1	21.8	33.3	6.9	62.5	23.1	37.5	17.1	28.4	25.9
StarChat (16B)	7.7	6.1	0.0	3.0	43.5	13.6	17.6	5.1	5.1	11.3
InstructCodeT5+ (16B)	8.9	6.3	0.0	6.9	45.2	13.8	17.9	4.2	5.1	11.6

Table 2: Results for CoT and PoT prompting on `TheoremQA`. We report the accuracy over different fine-grained question types and scientific fields.

Model	Method	Theorem	All
ChatGPT	CoT	-	30.2
ChatGPT	CoT	+	30.8
Claude-v1	CoT	-	24.9
Claude-v1	CoT	+	25.4
ChatGPT	PoT	-	35.6
ChatGPT	PoT	+	35.8
Alpaca-13B	CoT	-	13.5
Alpaca-13B	CoT	+	14.2

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

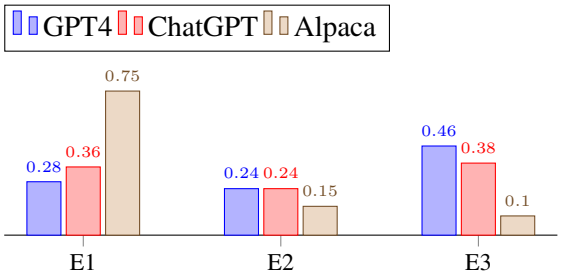


Figure 8: Error Analysis for GPT4, ChatGPT and Alpaca. Error severity: E1 > E2 > E3

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.

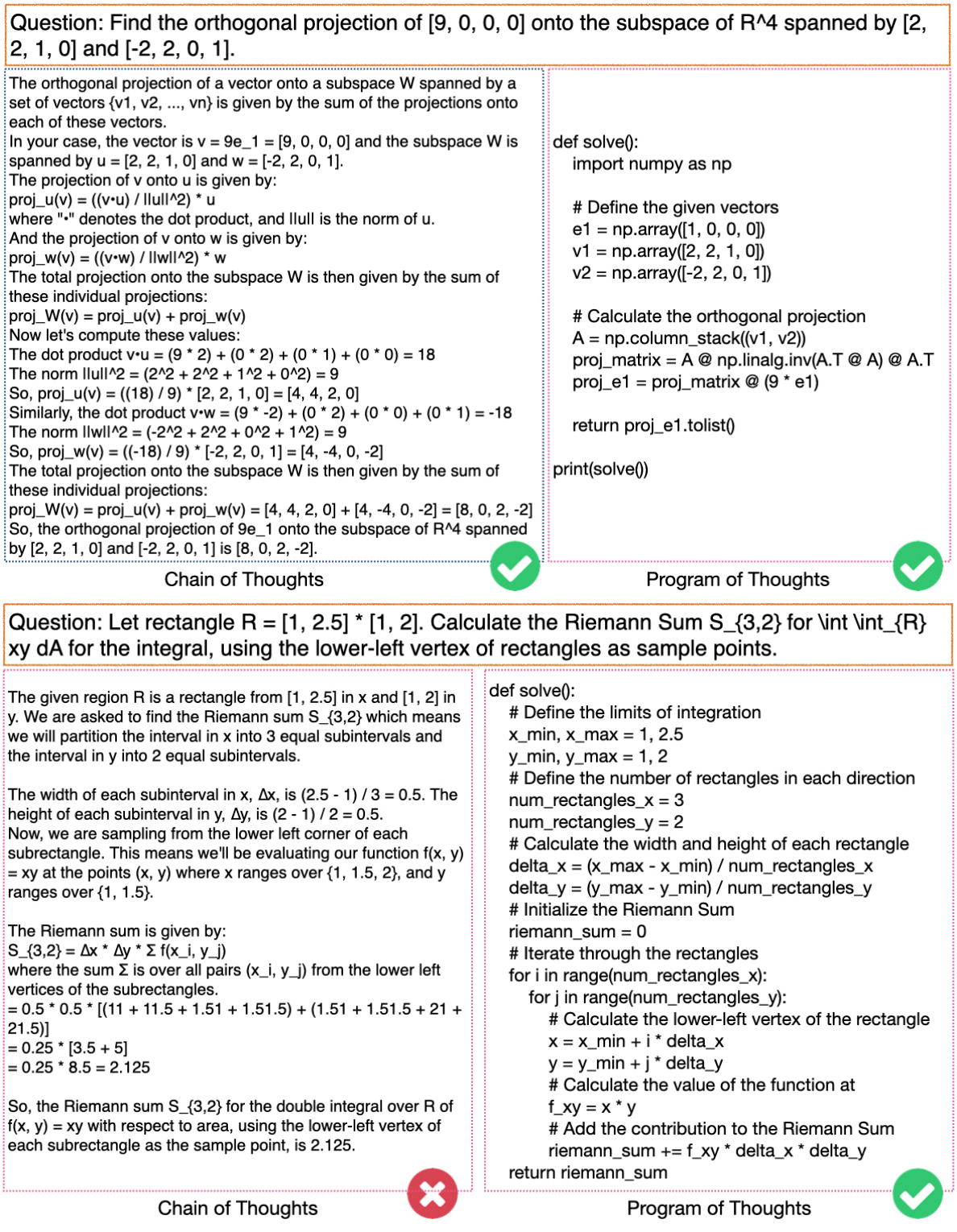


Figure 9: Case Study of GPT-4 generation with both prompting strategies.

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 still 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

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*.

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

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Jiaqi Chen, Tong Li, Jinghui Qin, Pan Lu, Liang Lin, Chongyu Chen, and Xiaodan Liang. 2022a. UniGeo: Unifying geometry logical reasoning via reformulating mathematical expression. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3313–3323, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric Xing, and Liang Lin. 2021a. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 513–523.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021b. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022b. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, et al. 2021c. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

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

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2022. Compositional semantic parsing with large language models. *arXiv preprint arXiv:2209.15003*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*.

Google. 2023. Palm 2 technical report. *https://ai.google/static/documents/palm2techreport.pdf*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *Conference on Neural Information Processing Systems*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *EMNLP*, pages 523–533.

Wolfram Research, Inc. Mathematica, Version 13.2. Champaign, IL, 2022.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. 2021a. Inter-GPS: Interpretable geometry problem solving with formal language and symbolic reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6774–6786, Online. Association for Computational Linguistics.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023a. Chameleon: Plug-and-play compositional reasoning with large language models. *arXiv preprint arXiv:2304.09842*.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023b. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *International Conference on Learning Representations (ICLR)*.

Pan Lu, Liang Qiu, Jiaqi Chen, Tony Xia, Yizhou Zhao, Wei Zhang, Zhou Yu, Xiaodan Liang, and Song-Chun Zhu. 2021b. Iconqa: A new benchmark for abstract diagram understanding and visual language reasoning. In *The 35th Conference on Neural Information Processing Systems (NeurIPS) Track on Datasets and Benchmarks*.

Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. 2023c. A survey of deep learning for mathematical reasoning. In *The 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984.

Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Oyvind Tafjord, Ashish Sabharwal, Peter Clark, and Ashwin Kalyan. 2022. Lila: A unified benchmark for mathematical reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. 2022. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. In *Deep Learning for Code Workshop*.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021a. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021b. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. *arXiv preprint arXiv:1904.01557*.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1476, Lisbon, Portugal. Association for Computational Linguistics.

Alex Tamkin, Kunal Handa, Avash Shrestha, and Noah Goodman. 2022. Task ambiguity in humans and language models. *arXiv preprint arXiv:2212.10711*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*.

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

Shyam Upadhyay and Ming-Wei Chang. 2015. Draw: A challenging and diverse algebra word problem set. Technical report, Citeseer.

Shyam Upadhyay and Ming-Wei Chang. 2017. Annotating derivations: A new evaluation strategy and dataset for algebra word problems. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 494–504, Valencia, Spain. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 845–854.

Yue Wang, Hung Le, Akhilesh Deepak Gotmare, Nghi DQ Bui, Junnan Li, and Steven CH Hoi. 2023. Codet5+: Open code large language models for code understanding and generation. *arXiv preprint arXiv:2305.07922*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. 2023. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287.