Improving the Reasoning Abilities of Large Language Models

William G. Marsan
marsa027@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

Abstract

Transformer-based large language models (LLMs) have demonstrated advanced natural language processing capabilities in recent years. However, their abilities to reason through problems involving multiple steps lag behind. Thus, their usefulness for tasks of any significant complexity remains questionable.

Researchers have found that their reasoning abilities are improved if they are prompted to think step by step. However, this "chain of thought" (CoT) prompting strategy only elicits reasoning at a model scale of around 100 billion parameters which is prohibitively expensive to train and do further experiments on. Consequently, researchers have applied a number of optimization techniques to LLMs with the goal of improving their reasoning abilities at a smaller model scale.

This literature review explores two such techniques, fine tuning and reinforcement learning, as well as the initial technique of CoT prompting.

Keywords: natural language processing, language models, reasoning, chain of thought, fine tuning, reinforcement learning

1 Introduction

ChatGPT defines reasoning as "the mental process of deriving conclusions or judgments from facts or premises. It involves the cognitive functions necessary to process information, integrate new knowledge with existing knowledge, and generate logical, rational decisions and understanding from this synthesis." However, when asked, "Which weighs more, a pound of water, two pounds of bricks, a pound of feathers, or three pounds of air?" It responds, "Two pounds of bricks weigh more than a pound of water, a pound of feathers, or three pounds of air. . . "

These two responses illustrate the descriptive strengths and logical weaknesses of this model. It has learned the descriptive facts about reasoning and the lightness of air to a greater extent than it has learned the formal relationship between the numbers two and three. This kind of behavior renders it useless for many use cases which involve "deriving conclusions or judgments from facts or premises."

LLMs like the one underlying ChatGPT have revolutionized natural language processing, demonstrating remarkable capabilities in generating human-like text and processing complex information. However, these models still struggle to reason through multi-step problems. This literature review explores the techniques aimed at enhancing the reasoning capabilities of language models, focusing on three approaches: CoT prompting which improves general reasoning but only at a large model scale, CoT fine tuning which improves reasoning at a smaller model scale but only in specific domains, and CoT reinforcement learning which also improves domain specific reasoning in smaller models, and potentially more-so than CoT fine tuning.

2 Large Language Models

This section will provide an overview of LLMs. A more nuanced and visually descriptive introduction can be found in 3Blue1Brown's YouTube series on Neural Networks and LLMs [1].

At a high level, LLMs take in data, feed it through complex functions defined in part by a large set of numerical parameters, similar to coefficients in a mathematical equation, and output the data in a new form.

These parameters are distributed across a variety of components:

1. Embedding layer
2. Transformer blocks
   a. Attention mechanism
   b. Multi-layer perceptron (MLP)
3. Next word prediction layer

2.1 Embedding Layer

The embedding process begins by converting input text into tokens which can be individual characters, words, or chunks of words (e.g. "?", "cat", and "ology"). Each token is transformed into a high-dimensional vector that captures semantic and syntactic information. These embedding vectors represent the initial encoding of linguistic information, allowing the model to begin processing the input's meaning. The embedding layer's parameters learn to map tokens to vector spaces where semantically similar words are positioned closer together.

2.2 Transformer Blocks: Attention Mechanism

LLMs contain many transformer blocks. The output from one block is the input for the next. The first component within these blocks is the self-attention mechanism which allows the model to update the embedding of each token to reflect the context of the input. For example, if "Queen" was originally embedded as a kind of monarch but "Freddy Mercury" appears earlier in the sentence, the attention mechanism may update the embedding to represent "Queen" as the band instead.

By the time the input passes through the attention mechanisms in all the transformer blocks, the embedding for each token reflects the syntactic and semantic context in which it exists.

2.3 Transformer Blocks: Multi-Layer Perceptron

Following the attention mechanism, each transformer block includes a neural network called a multi-layer perception (MLP). In [1], Sanderson says that the MLP could be thought of as adding facts to each embedding (e.g. given a vector that represents "Michael Jordan", the MLP may add to that vector the representation of the phrase "plays basketball"). It accomplishes this by passing the embedding vector for each individual token through a neural network which has been trained to recognize tokens add relevant information to them in the form of new values.

2.4 Next Word Prediction

The final stage of language model processing involves predicting the next token. This occurs by passing the input through multiple transformer blocks, generating a final embedding vector for the last token, multiplying this vector by the model's vocabulary matrix, applying a softmax function to produce a probability distribution over all possible next tokens and finally selecting one of the next most likely tokens.

The vocabulary matrix is a large parameter matrix where each column represents the embedding of a specific token in the model's vocabulary. By multiplying the final token vector with this matrix, the model generates a score for each possible next token, effectively ranking their likelihood.

2.5 Model Size

The number of parameters directly impacts the model's capacity to understand and generate language. As this paper highlights later, models with around 100 billion parameters demonstrate significantly improved reasoning capabilities compared to smaller models. This is because more parameters allow for more nuanced representation learning, enabling the model to capture more complex linguistic and reasoning patterns.

However, large model size comes with downsides. While larger models can capture more intricate relationships, they also become more challenging to train, interpret, and deploy, thus motivating the ongoing goal of developing smaller, more efficient and interpretable language models.

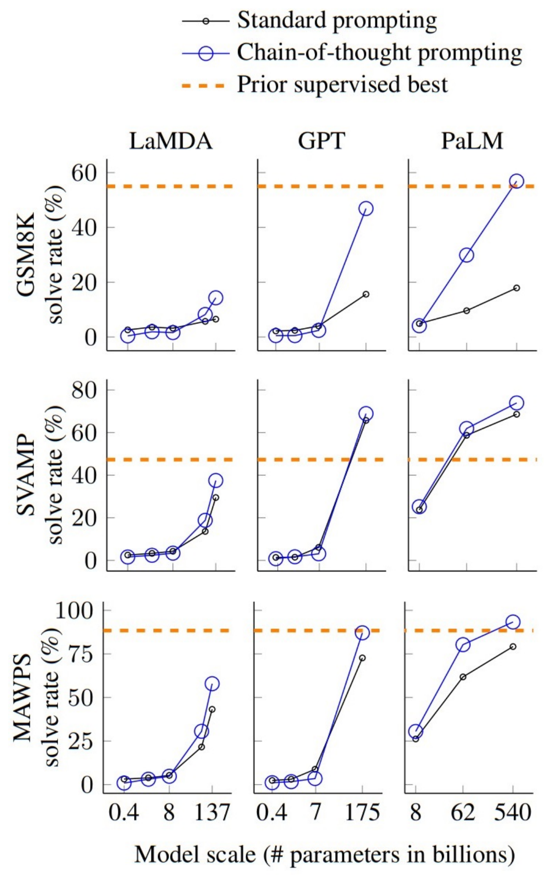

Figure 1. Arithmetic reasoning was tested using three model types (LaMDA, GPT, and PALM) and three different model sizes for each type showing on the x-axis of each graph. Performance was measured by percent of correct answers on three different word problem arithmetic tests (GSM8K, SVAMP, and MAWPS). From [12]

3 Chain of thought

This section provides and overview of a 2022 paper titled "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models" by Wei, et al. [12].

3.1 Introducing CoT prompting

Chain of thought (CoT) prompting enhances reasoning in LLMs by guiding them to generate a structured sequence of intermediate steps during problem-solving. It achieves this by including examples of problems and solutions with these reasoning steps in the prompt before presenting the model with the actual question. This method is contrasted with standard prompting which provides the model with examples of problems and solutions but not the intermediate reasoning steps.

It is often posited that LLMs cannot do complex reasoning since they associate between words using statistical operations rather than deterministic, symbolic operations. However, if they can be prompted to break down a problem into individual steps and these individual steps are simple enough to be solved consistently using statistical operations (e.g. the character string “2+2=” is highly associated with “4”), it is feasible that they could solve complex problems.

The study by Wei et al. provides examples which highlight the differences between standard prompting and CoT prompting. In both cases, an example question and its answer are included in the input prompt as a guide for the model, followed by a real question for the model to answer. The model’s response demonstrates how it processes the information provided.

In standard prompting, the input prompt consists of an example question like: “Roger has 5 tennis balls. He buys 2 more cans of tennis balls, with each can containing 3 tennis balls. How many tennis balls does he have now?” The answer provided in the prompt is simply: “The answer is 11.” The real question given to the model is: “The cafeteria had 23 apples. They used 20 to make lunch and bought 6 more. How many apples do they have now?” When using standard prompting, the model’s response is incorrect, stating: “The answer is 27.” This highlights how the model arrives at an output without reasoning through the steps of the problem.

In contrast, CoT prompting includes an example question and a detailed explanation of how to solve it. For instance, the input prompt begins with: “Roger has 5 tennis balls. He buys 2 more cans of tennis balls, with each can containing 3 tennis balls. How many tennis balls does he have now?” The explanation provided in the prompt walks through the reasoning: “Roger starts with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.” The real question presented to the model is the same as before: “The cafeteria had 23 apples. They used 20 to make lunch and bought 6 more. How many apples do they have now?” With CoT prompting, the model’s response is correct and includes step-by-step reasoning: “The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 6.”

This comparison shows how CoT prompting guides the model to reason through problems systematically, leading to more accurate results.

The Wei et al. study shows that CoT prompting affords significant improvements in arithmetic, common sense, and symbolic reasoning.

3.2 Arithmetic Reasoning

The authors tested CoT prompting across a variety of reasoning benchmarks, starting with arithmetic tasks. They evaluated the model using the following arithmetic word problem benchmarks: GSM8K [3], SVAMP [9] and MAWPS [6]. These contain question and answer couplets very similar in form to the following example taken from [3]:

> [question:] Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?
> [answer:] Natalia sold 48/2 = 24 clips in May. Natalia sold 48+24 = 72 clips altogether in April and May.

To successfully answer questions in these datasets, models need to perform basic addition, subtraction, multiplication and division, often in multiple steps.

The results of the model evaluations using CoT prompting, which can be seen in Figure 1, capture overall trends but it should be noted that the authors tested the model on other datasets too which can be found in their paper [12]. Google’s 540 billion parameter PaLM (Pathways Learning Model) [7], demonstrated the most significant improvements in arithmetic reasoning tasks. On the GSM8K dataset, CoT prompting enabled PaLM 540B to reach the correct answer on approximately 60% of the questions, whereas standard prompting resulted in a much lower performance of roughly 20%. In the MAWPS benchmark, CoT allowed PaLM 540B to solve around 90% of the problems compared to around 75% for standard prompting. It should be noted, however, that CoT performed very similarly to standard prompting on smaller models.

3.3 Common Sense Reasoning

The benefits of CoT extended beyond arithmetic reasoning. Common sense reasoning, which requires applying real-world knowledge and logic, also saw improvement. The authors tested CoT prompting on datasets like StrategyQA [5] and a sports questions dataset [13]. The following is an example from [5]:

> Fact 1: Grey seals have no ear flaps and their ears canals are filled with wax.
> Fact 2: Grey seals hear better underwater when their ears open like a valve.
> Fact 3: Dogs have sensitive ears that can hear as far as a quarter of a mile away.
> Question: Would a dog respond to bell before Grey seal?
> Answer: Yes

Similarly to the tests on arithmetic reasoning, the PaLM’s accuracy when using CoT prompting tended to significantly diverge from standard prompting when model size increased. PaLM 540B, specifically, increased performance from 69.4% to 75.6% on StrategyQA and 84% to 95.4% on the sports questions when using CoT prompting. A full table of these results can be found in [12].

3.4 Symbolic Reasoning

Symbolic reasoning also showed significant improvement with CoT prompting. The authors tested this in two ways: first, by prompting the model to concatenate the last letters of two or four words (e.g. if given "Amy brown" the model should output "yn") and second, by telling to model to say which way a coin is facing after it is flipped or not flipped by a two or four times (e.g. if given "A coin is heads up. Phoebe flips the coin. Osvaldo does not flip the coin. Is the coin still heads up?" the model should answer "No").

CoT prompting enabled PaLM to solve nearly all in-domain problems (these results and other symbolic results can be found in [12]). In-domain tasks refer to problems that have the same structure or complexity as the examples the model has already seen during prompting (e.g. the model is given the CoT for a two word concatenation problem and then tested on a two word concatenation). CoT also helped the model generalize more effectively to out-of-domain tasks, which involve problems that are more complex or require additional steps beyond those in the examples (e.g. the model is given the CoT for a two word concatenation problem and then tested on four word concatenation problem). This improvement in handling out-of-domain tasks demonstrates that CoT not only enables the model to replicate the logical structure and substitute the values of the example problem seen in the prompt, but also enables it to continue to generate similar logical structures as the new problem requires.

3.5 Testing for Causation

It is difficult to explain the behavior of LLMs given that this behavior is determined by complex interactions between billion of parameters. Thus, it cannot be stated with certainty that the CoT prompting itself actually causes improved reasoning. However, by analyzing the reasoning paths of the model the researchers strongly correlated correct answers with correct CoTs. Also, by isolating the effects of variables associated with CoT prompting, the researches have further correlated the improvements in reasoning with the CoTs themselves. Further details on these tests for causation can be found at [12].

3.6 Importance of Model Size

A key finding in the study was the scalability of CoT prompting with model size. As depicted in Figure 1, smaller models saw minimal benefit from CoT prompting, as they struggled to generate coherent chains of thought. However, as model sizes increased—particularly with models like GPT-3 175B and PaLM 540B—the advantages of CoT became more pronounced. This was especially apparent in the GSM8K, where CoT prompting vastly outperformed traditional prompting on large models.

4 Fine Tuning

This section covers the paper "Specializing Smaller Language Models towards Multi-Step Reasoning" [4] by Fu et al. which uses fine tuning, or as the authors call it, specialization, to improve reasoning in smaller language models.

4.1 Background

CoT prompting has shown significant potential in enhancing the reasoning capabilities of LLMs by enabling them to produce structured, step-by-step solutions to problems. However, CoT prompting has its limitations. Primarily, it requires large model size, such as GPT-3 175B and PaLM 540B. Smaller models struggle to generate coherent CoTs which in some cases makes them less likely to reason correctly than if they had not generated the CoT. Consequently, the scalability of CoT prompting alone to achieve the complex reasoning that would be required in real world applications is doubtful.

Fine tuning offers a solution to the limitations of CoT prompting, especially for smaller models that lack the scale required to handle complex reasoning tasks effectively but are still preferred for their lower monetary and energetic costs. Fine tuning involves training smaller models on task-specific data, refining their generic language abilities to specialize in areas such as mathematical reasoning. This targeted training enables smaller models to perform complex reasoning with greater accuracy, though it comes with trade-offs. Fine tuning a model often reduces its flexibility across other kinds of tasks. For example, if a model was fine tuned on a Python code dataset, it would likely be significantly worse at generating Shakespearean sonnets. As will be discussed later, however, fine tuning does not necessarily eliminate all generalizing abilities of models. The closer a task resembles the tasks the model was fine tuned on, the more likely the model is to have specialized abilities for it.

4.2 Methodology

As depicted in Figure 2, the researchers performed two stages of fine tuning after pretraining the T5 model with general language abilities. The first was instruction tuning, which involves training the model on detailed prompts or instructions along with the corresponding responses. By exposing the model to diverse instruction-response pairs, it learns to tailor its responses to the instructions. For example, if it receives a complex instruction, it will respond with a complex solution. The researchers used T5, a raw LLM, as well as FlanT5, an introduction tuned version of T5 as starting points to conduct the second stage of fine tuning on (it should be noted that Figure 2 suggests that this second stage of fine tuning was performed only on the instruction tuned FlanT5. This is in fact the best performing method, however, this stage of fine tuning was also performed on the raw T5 for comparison).

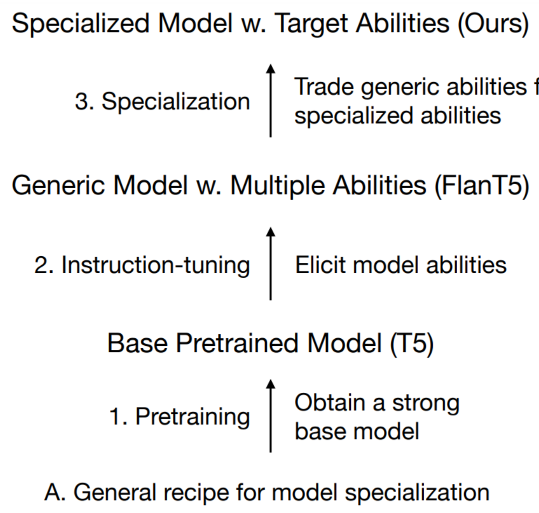
Specialized Model w. Target Abilities (Ours)
3. Specialization
Trade generic abilities for specialized abilities
Generic Model w. Multiple Abilities (FlanT5)
2. Instruction-tuning
Elicit model abilities
Base Pretrained Model (T5)
1. Pretraining
Obtain a strong base model
A. General recipe for model specialization

Figure 2. This figure from Fu et al. shows the process of creating a specialized model showing first pretraining, which gives the model called T5 general language abilities, then instruction-tuning, which makes the model called FlanT5 more responsive to diverse prompts and finally specialization, which gives the model specific reasoning abilities. From [4]

In this second stage, they used a larger model from OpenAI, GPT-3.5 code-davinci-002 [8], to generate 40 solutions with CoTs for questions in the GSM8K word problem dataset. The CoTs that produced correct answers were then used to fine tune the FlanT5 model through a method called distribution matching. This is a technique used in machine learning to help a smaller, student model learn to make predictions similar to a larger, teacher model. As Fu et al. explains in the paper, the goal is to have the student model not just memorize the predictions of the teacher model but learn the criteria by which the teacher makes predictions so that it can generalize or extrapolate to cases outside of the training data.

4.3 Results

Panel A in Figure 3 shows the results of the previously studied CoT prompting, reinforcing the theory that CoT prompting elicits improved reasoning but only at a 100B parameter scale. Panel B shows the results of fine tuning without the initial stage of instruction tuning which primes models to be responsive to problems of varying complexity. The lines with circles show the performance of models which were fine tuned using answer-only (AO) data - that is, data lacking CoT. The lines with triangles show the significantly increased performance of models fine tuned on CoT data. Finally, panel C shows that instruction-tuning combined with CoT fine tuning yields the greatest improvements in models. The researchers point out that the reasoning improvements yielded from instruction-tuning and CoT-tuning on 10B parameter scale models describe a log-linear curve which is only seen in 100B parameter scale models when using CoT prompting alone.

5 Reinforcement Learning

This section covers the paper "WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct" by Luo et al. [2] which uses reinforcement learning (RL) to create near-state-of-the-art models on math reasoning. Models trained with this process have the WizardMath prefix in this paper.

5.1 Background

RL is a general technique in machine learning but in the context of LLM optimization, it focuses on training LLMs to make sequential decisions in an environment to maximize cumulative rewards through trial and error. This process encourages the LLM to improve its behavior according to the standard of the person or machine dispensing the reward. The unique aspect of RL, as opposed to fine tuning, is that it does not rely on labeled data such as the 40 generated chains of thought for each question in the previous paper, but instead learns optimal actions by receiving rewards or punishments from interactions with this outside person or model.

5.2 Methodology

Before RL takes place, there are some preliminary steps. The first, called Math Evol-Instruct, uses GPT-4 [10] to create different versions of each problem ranging from simpler to more complex. For each of these new problems, GPT-4 then assigns a lower score if it is clear and complex and a higher score if it is unclear and simple. This data is then used to train an instruction reward model (IRM) so that it can recreate similar rankings for instructions that the WizardMath encounters during the actual RL phase.

The other preliminary step involves training a process reward model (PRM) to evaluate the CoTs produced by WizardMath models. This PRM is trained as follows: first Llama-2 [11] is used to produce answers with CoTs on questions from the GSM8K. Then these CoTs are evaluated and assigned a score by GPT-4 and finally, the base Llama-2 model is trained with this data to assign similar scores to CoTs.

Now all the pieces are in place to begin the RL. The WizardMath model (the model we are trying to improve the reasoning of) is given questions from the GSM8K and MATH datasets. It generates responses and the associated CoTs. At this point, the problem given to the WizardMath model as well as the CoT response are fed to the IRM and PRM respectively. The IRM assigns varying rewards and punishments based on the clarity and complexity of the instructions (see

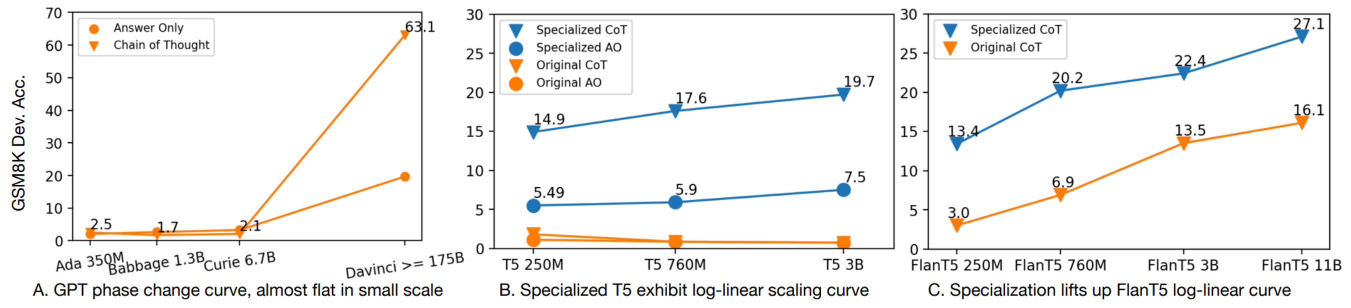

Figure 3. Results from testing the specialized (fine tuned) T5 and Flan5 models on the GSM8K, a math reasoning dataset. While the results for fine tuned models may not appear exponential, their blue lines can be interpreted as an exponential elbow beginning earlier on the plot than the orange lines which lie flat or consistently lag behind. From [4]

[2] for details). The PRM assigns varying rewards based on whether each individual step in the CoT response is correct.

5.3 Results

Model	Params	GSM8k (%)	MATH (%)
ChatGPT-4o	Proprietary	96.1	76.6
Claude 3.5	Proprietary	96.4	71.1
GPT-2-XL	1.5B	15.4	6.9
WizardMath-GPT-2	1.5B	58.9	25.4
Mistral-v0.1	7B	42.9	12.9
WizardMath-Mistral	7B	90.7	55.4

Table 1. Comparison of model performances on GSM8k and MATH benchmarks. Performance is measured as the percentage of correct answers. Based on [2].

The results presented in Table 1 indicate that the WizardMath versions of various models show a substantial improvement in mathematical reasoning performance on the GSM8k and MATH benchmarks compared to their base counterparts. Notably, while these results are not directly comparable to those in the fine tuning section of the previous paper—since the base models here are inherently stronger in math—the performance gains achieved through RL are significant. For each model size, the WizardMath versions approximately double or even triple their base scores. This substantial leap in performance suggests that the RL approach employed in this study is more effective at enhancing mathematical reasoning than the fine tuning methods used previously.

A particularly impressive result is the performance of the 7B-parameter WizardMath-Mistral model, which reaches 90.7% on GSM8k and 55.4% on MATH. This score places it near state-of-the-art performance levels.

6 Conclusion

This paper has explored the progression of reasoning capabilities in LLMs through CoT prompting, CoT fine tuning, and CoT reinforcement learning. CoT prompting has proven instrumental in breaking down complex reasoning tasks into manageable steps, yielding significant improvements in general reasoning, but only at large model scale. CoT fine tuning enables smaller models to handle complex math reasoning tasks effectively, albeit at the cost of generalization to other reasoning domains. Finally, CoT reinforcement learning, exemplified by WizardMath, achieved near-state-of-the-art performance on math reasoning. Though its results are not directly comparable to fine tuning due to differing base models, it shows a much higher ratio of improvement from its base model compared to fine tuning.

While these methods show advancements, reasoning benchmarks alone may not fully capture the depth of a model's reasoning capabilities, as they are often narrowly tailored to specific task structures, and as Zhang et al. show in their paper titled "A Careful Examination of Large Language Model Performance on Grade School Arithmetic" [14], could be inadvertently leaked into training datasets, causing the model, to some extent, to memorize answers to questions rather than develop general reasoning abilities. Future research should address this limitation by developing benchmarks and evaluation strategies that better reflect the dynamic and nuanced nature of real world reasoning and ensure no leakage into training data.

Acknowledgments

Thank you to Dr. Nic McPhee and Dr. Elena Machkasova for advising me on this project, to Dr. Mark Collier for inspiring and enabling my interests in AI, and to Wenkai Guan and Ashlen Plasek for providing feedback on my drafts.

References

[1] 3Blue1Brown. 2023. How large language models work, a visual intro to transformers | Chapter 5, Deep Learning. https://www.youtube.com/watch?v=wjZofJX0v4M [Online; accessed 2024-10-28].

[2] Anonymous. 2024. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. In

Submitted to The Thirteenth International Conference on Learning Representations. https://openreview.net/forum?id=mMPMHWOdOy under review.

[3] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168 [cs.LG] https://arxiv.org/abs/2110.14168

[4] Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing Smaller Language Models towards Multi-Step Reasoning. In *Proceedings of the 40th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 202)*, Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 10421–10430. https://proceedings.mlr.press/v202/fu23d.html

[5] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies. *Transactions of the Association for Computational Linguistics* 9 (04 2021), 346–361. https://doi.org/10.1162/tacl_a_00370

[6] 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*, Kevin Knight, Ani Nenkova, and Owen Rambow (Eds.). Association for Computational Linguistics, San Diego, California, 1152–1157. https://doi.org/10.18653/v1/N16-1136

[7] Sharan Narang and Aakanksha Chowdhery. 2022. https://research.google/blog/pathways-language-model-palm-scaling-to-540-billion-parameters-for-breakthrough-performance/ [Online; accessed 2024-11-24].

[8] OpenAI. [n. d.]. https://platform.openai.com/docs/models

[9] Arkil Patel, S. Bhattamishra, and Navin Goyal. 2021. Are NLP Models really able to Solve Simple Math Word Problems?. In *North American Chapter of the Association for Computational Linguistics*. https://api.semanticscholar.org/CorpusID:232223322

[10] OpenAI staff. 2023. GPT-4 technical report. https://cdn.openai.com/papers/gpt-4.pdf

[11] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL] https://arxiv.org/abs/2307.09288

[12] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '22)*. Curran Associates Inc., Red Hook, NY, USA, Article 1800, 14 pages.

[13] Haotian Xia, Zhengbang Yang, Yuqing Wang, Rhys Tracy, Yun Zhao, Dongdong Huang, Zezhi Chen, Yan Zhu, Yuan-fang Wang, and Weining Shen. 2024. SportQA: A Benchmark for Sports Understanding in Large Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Kevin Duh, Helena Gomez, and Steven Bethard (Eds.). Association for Computational Linguistics, Mexico City, Mexico, 5061–5081. https://doi.org/10.18653/v1/2024.naacl-long.283

[14] Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, William Song, Tiffany Zhao, Pranav Vishnu Raja, Charlotte Zhuang, Dylan Z Slack, Qin Lyu, Sean M. Hendryx, Russell Kaplan, Michele Lunati, and Summer Yue. 2024. A Careful Examination of Large Language Model Performance on Grade School Arithmetic. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. https://openreview.net/forum?id=RJZRhMzZzH