SWER – Midterm Paper

Bowen Yang UNI: by2365

February 2024

Abstract

This paper explores the potential of generative AI in enhancing educational resources, with a focus on its application in mathematics education. Despite its capabilities, the current shortfall in the reasoning abilities of generative AI limits its effectiveness in educational settings. I discuss previous efforts toward the development of an AI-powered math tutor and propose a Retrieval-Augmented Generation (RAG)[13] approach aimed at enhancing the proficiency of Large Language Models (LLMs) in solving middle school level math problems. My objective is to improve the reasoning skills of AI, thereby increasing its accuracy in addressing math challenges, and ultimately, to contribute to the advancement of educational tools that offer personalized learning experiences.

1 Introduction

Generative AI introduces numerous possibilities, with a significant emphasis on the field of education. There exists a shortfall in educational resources, which AI has the potential to mitigate due to its replicability at low cost. Nonetheless, the progression of AI-driven educational solutions is impeded by generative AI's limited reasoning capabilities. This paper primarily focuses on enhancing the ability of LLMs to solve elementary mathematics problems. The structure of the paper is outlined as follows:

1. **Section 2:** Discusses foundational efforts aimed at improving the abilities of LLMs on certain task.
2. **Section 3:** Introduces studies aimed at enhancing the reasoning speed of LLMs.
3. **Section 4:** Proposes a Retrieval-Augmented Generation (RAG) based approach to improve LLMs' proficiency in solving mathematical problems.
4. **Section 5:** Presents experiments to validate the proposed approach.
5. **Section 6:** Conclusion.

1.1 Learning Outcomes for the Reader

The reader is expected to gain understanding of:

1. Current popular methods for improving the qualitative performance of LLMs.
2. A novel RAG-based method that could enhance LLMs' capabilities in solving mathematical problems.
3. Preliminary effects of the proposed method.

2 Background Approaches

Efforts have been made to enhance the performance of LLMs in specific domains. Traditional techniques such as fine-tuning continue to be valuable, as they improve LLMs' performance for particular tasks. In addition to fine-tuning, LLMs have developed the capability for in-context learning, where they improve at tasks by simply analyzing examples, without the need for further training or modifications. This opens up possibilities to incorporate LLMs with other systems.

In this section, I will describe how each of these methods functions and highlight some technologies that I find noteworthy.

2.1 Fine Tuning

Model fine-tuning resembles improving an already intelligent model to excel further in a small domain (specific) task. Imagine you have a robot with a basic understanding of numerous subjects. Fine-tuning involves specializing this robot in a specific field, such as cooking, without erasing its existing knowledge base. This approach is advantageous as it conserves time, effort, and resources that would otherwise be expended in developing a new robot from the ground up. For sophisticated language models such as GPT-3.5 or BERT, fine-tuning means enhancing their expertise in specific domains, like interpreting legal documents, by feeding them additional information on these subjects. As a result, they not only retrain their general intelligence but also become highly proficient in particular tasks with minimal additional training.

However, fine-tuning leads to diminish performance across other areas and demands significant computational power and data for if target is a large language model. Therefore, while fine-tuning can significantly improve the capabilities of these language models in certain roles, it necessary to take cost and generalize ability into consideration.

2.1.1 LoRA

Low-Rank Adaptation (LoRA)[5] is an efficient technique for fine-tuning LLMs by targeting specific model parameters, reducing computational costs. Unlike

conventional fine-tuning that updates extensive model parameters, LoRA modifies a subset using low-rank decomposition, significantly cutting down resources for adaptation.

LoRA embeds trainable (small) weights in the Transformer's attention and feedforward layers, refining the model with minimal parameter adjustments. For example, LoRA first construct a weight matrix W_e with two smaller matrices A and B, with A sized $d \times r$ and B sized $r \times d$ (r is way smaller than d). Thus we have

$$AB = W_e, \ with \ size(W_e) = d \times d$$

W_e is then used to adjusts original weight W (size $d \times d$ as well), updating the weight to $W + AB$ with fewer trainable parameters, preserving pre-trained knowledge while allowing precise, task-specific tuning.

This approach requires significantly lower computational power compare to traditional fine-tuning. However, implementing LoRA demands deep understanding of LLM architecture and careful integration of low-rank matrices, presenting accessibility challenges for outsiders. Besides, LoRA's learning capacity for incorporating new, task-specific knowledge is limited. They perform worse than traditional fine-tuning in average.

2.1.2 Adapter Layers

Adapter models, introduced by Houlsby et al.[11], enhance the fine-tuning process in NLP and are effective for LLMs. These models incorporate small, trainable "adapters" between the layers of a pre-trained model (like GPT3.5), enabling the model to acquire new capabilities without altering its foundational structure. Adapters are particularly compatible with transformer models due to their modular design.

Same as LoRA, the training of adapter models requires updating fewer parameters compared to full model retraining, thus making the fine-tuning process much cheaper.

Using adapters offers the benefits of lower costs and rapid deployment for models for specific tasks. Since this approach maintains the model's original structure, simply swapping adapters is all that's needed to adapt the model for various tasks.

Nonetheless, employing adapters presents certain challenges. They need to be meticulously engineered to assemble task-specific knowledge. Adapters introduce an additional complexity to the and may decelerate the model's performance.

2.2 In Context Learning

In-context learning represents a novel approach in the field of AI, particularly for LLMs. Unlike traditional methods, which require extensive data and long training periods, in-context learning enables a model to acquire new skills from a limited number of examples provided in its input.

The fundamental principle behind in-context learning is leveraging the model's existing in prompt learning ability to adapt new tasks without undergoing retraining. This capability allows the model to understand rules or patterns from a minimal set of examples, thereby reducing the dependency on large datasets and significant computational resources. The adaptability afforded by in-context learning is the bedrock of LLMs' versatility.

In this section, I will explore two pivotal studies that have significantly contributed to the understanding and development of in-context learning:Few-Shot Prompting[1] and COT[2].

2.2.1 Few-Shot Prompting

At the heart of in-context learning lies the concept of Few-Shot Prompting[1]. This method involves providing the model with a handful of examples directly in the prompt, enabling it to understand the task without needing retraining.

Consider the task of sentiment analysis, where the goal is to determine whether a piece of text is positive, negative, or neutral. Instead of relying on a large dataset, few-shot prompting introduces a few examples right away:

Question: "Please classify the sentiment of the following sentences as Positive, Negative, or Neutral.

1. 'I love sunny days.' Sentiment: Positive
2. 'It's raining again. So depressing.' Sentiment: Negative
3. 'The book was okay, nothing special.' Sentiment: Neutral

Now, classify the sentiment of this sentence: 'The movie was a groundbreaking achievement in cinema.'"

Through this approach, the model quickly grasps the task at hand using just a few examples. Few-shot prompting leverages the model's existing knowledge and its ability to infer from limited information, allowing it to accurately perform tasks it wasn't explicitly trained for.

2.2.2 Chain of Thought

Building on the concept of few-shot prompting, Chain of Thought (CoT)[2] prompting enhances this approach by guiding models not just to produce an output, but to reveal the reasoning process that leads to that output. For example to let LLM solve the following problem from GSM8K[10] benchmark:

Question: "Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?"

We provide few shot prompts like:

1. Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls.
2. $5 + 6 = 11$.

3. The answer is 11.

By requiring the model to articulate the logic behind its classification, CoT prompting not only aids in understanding the model's decision-making process but also improves its ability to handle complex tasks.

2.3 LLM Hybrid System

The capability for in-context learning allows LLM to utilize external resources, such as search engines, to improve their functionality.

This section I will detail several influential papers that I feel inspiring.

2.3.1 ReAct

The ReAct Framework[9] for LLMs is a conceptual and operational framework designed to enhance the interaction and performance of LLMs in dynamic environments. The ReAct Framework aims to improve the responsiveness, relevance, and overall effectiveness of LLMs and to generate more accurate and contextually appropriate responses.

For example, if you provide the system with question:"Aside from the apple remote, what other device can control the program that Apple Remote was original designed to interact with?"

Initially, the model engages in a reasoning process (Thought), contemplating the nature of the Apple Remote and its primary function with the Apple TV. Then, the model takes an action (Act), which, in this case, is performing a search with the term *Apple Remote* to gather relevant information. Upon observing (Obs) that the Apple Remote is linked to the Front Row media center program, the model identifies a gap in its knowledge regarding other control methods for Front Row and thus decides to conduct another search.

The adaptive part of the ReAct Framework comes into play when the model, after not finding Front Row in its initial search, changes its approach to search specifically for *Front Row (software)*. Through this iterative process, the model discovers that Front Row is discontinued software and refines its reasoning to conclude that, aside from the Apple Remote, Front Row can be controlled by Apple keyboard function keys.

The final step, marked as "Finish," is where the model consolidates all the gathered information, confirming the keyboard function keys as the answer.

2.3.2 LPML

Research [3] presents an innovative approach to enhancing mathematical reasoning through a structured interaction between a system and an assistant, using a specially designed markup language and reasoning rules. The process unfolds as follows:

1. **Problem Presentation:** The system initially presents a mathematical problem, specifying the markup language's syntax and the established

reasoning rules. A critical rule force the assistant to produce Python code and Chain of Thought reasoning as separate messages.

2. **Assistant's Response:** Upon analyzing the problem, the assistant returns multiple responses. If any response include Python code, the system executes this code and relays the results back to the assistant.

3. **Output Filtering:** The system examines the assistant's outputs to remove any undesirable elements, such as buggy Python execution results. Moreover, if the assistant submits both the Python code and its solution within a single message. the system will discard the whole message and regenerate the solution.

4. **Iterative Process:** The system and assistant engage in a cyclic interaction, persisting until the assistant furnishes the correct answer.

The essence of this paper lies in the separation of code generation and reasoning. This technique prompts LLMs to view the question from multiple view points and verify its own answers.

2.3.3 Math Prompter

MathPrompter, introduced by Imani et al.[4], works a lot like LPML with special feature. It possess an extra pipeline that replace numerical value in math questions with variables to assist LLMs' understanding on how to write Python code.

For instance, for the following question:

```
At a restaurant, each adult meal costs $5 and kids eat free.
If a group of 15 people came in and 8 were kids,
how much would it cost for the group to eat?
```

MathPrompter will transform it into:

```
At a restaurant, each adult meal costs A and kids eat free.
If a group of B people came in and C were kids,
how much would it cost for the group to eat?
Mapping: {A:5, B:15, C:18}
```

They claimed that conducting this extra step would assist LLMs in understanding the general structure of the question and generate better python code.

2.3.4 RAG

The Retrieval-Augmented Generation (RAG)[13] model mixes retrieval-based and generative approaches for language tasks. It boosts text creation using external knowledge, ideal for tasks needing context or detailed info, like question answering, summarizing, and chatting.

RAG works in two steps: retrieval and generation.

1. **Retrieval Phase:** It first finds relevant text or documents from a big collection based on the input. This usually uses dense vector search, which turns both query and documents into vectors in a high-dimensional space. It then picks the most relevant documents based on vector similarity, using models like Dense Passage Retrieval (DPR)[6] for efficient information finding.

2. **Generation Phase:** After getting the relevant documents, the model creates a response. It combines the found documents and the input query in a generative model, such as GPTs, to make a relevant and coherent reply. This step lets the model use specific info from the documents to make better responses.

3 Other Related Researches

The previous section has majorly focused on methodologies that were implemented to improve the qualitative performance of LLMs, i.e. make LLMs more powerful for accomplishing certain tasks. This section will quickly introduce some other topics focusing on how to improve LLMs' speed.

3.1 GPTCache

Introduced by Bang et al., GPTCache[7], is a semantic caching system designed specifically for LLM applications.

Similar to other caching systems, GPTCache stores answers from LLMs in a cache. The system determines the similarity between new queries and the data in the cache using an NLP classifier, and it uses previous answers as responses if a similar query is found.

3.2 Distilling Model

The paper "Distilling the Knowledge in a Neural Network" by Hinton et al.[8], introduces "knowledge distillation." This method transfers knowledge from a big, complex model (teacher) to a smaller, simpler one (student). The student learns by copying the teacher's outputs (soft targets) instead of from the original data (hard targets). This way, the student picks up the teacher's broad understanding, including the deeper insights. Knowledge distillation is a smart way to make smaller neural networks that still perform well, making it easier to use advanced models where there's not much computing power. This idea also applies to big LLMs, allowing for the creation of smaller models focused on specific areas using LLMs as teachers. This shows how knowledge distillation can make efficient, specialized models by using the rich knowledge in LLMs.

4 Building AI with Robust Mathematical skill

Given my focus on math problems at the primary/middle school level, the topics will be limited in scope. My theory posits that for small areas such as solving elementary math problems, instead of only relying on the capabilities of LLMs, it is possible to also enhance LLM performance by supplementing them with highly relevant information for in-context learning. For instance, when LLMs are presented with a problem about calculating the total area of a shape formed by combining a triangle and a circle, we would identify and retrieve similar problems from datasets or online sources. This information would then be integrated into the prompt as context to aid the LLM in solving the problem more effectively.

Incorporating the concept, I want to build a RAG[13] based system, which initially retrieves related information based on the input and then constructs a prompt based on the retrieved information. Consider we have datasets $D = [d_1, d_2, \ldots, d_n]$, where each d_i represents a dataset accessible online, such as GSM8K[10]. I will define a categorization tool C, where $c = C(x)$, with x being the input natural language question and c the resulting category. Such pipeline can be obtained by first categorizing a dataset using both question and solution, then training a machine learning model on the categorized dataset to learn the connection between question and categories. The trained model then can be used as the categorization tool. The tool C is then applied to D to yield $D_c = [dc_1, dc_2, \ldots, dc_n]$, where each dc_i corresponds to the categorized form of d_i.

Upon receiving a natural language question q from a user, we apply C to determine the question's category c_q via $c_q = C(q)$. We then use c_q as a key to retrieve data d in D_c categorized under c, where $c_q = c$. Subsequently, d is structured into COT[2] prompts and fed into LLMs to enhance performance.

4.1 Proposed Categorization Method

How to define the categories for categorization is the pivot, as they must closely align with the semantic essence of the problems to boost the relevance of related problems identified. There are plenty of studies aimed at finding the optimal categorization of mathematical problems. For instance, Cetintas et al.[12] utilize NLP techniques to distill features from mathematical word problem texts, and then employ machine learning algorithms to segregate these problems into pre-established categories.

Nonetheless, despite achieving high accuracy in their experiments, such precision often benefits from those broadly defined categories. Many studies opt for expansive categories like 'geometry', which may not significantly enhance LLMs' capability to solve math problems using my proposed method.

Due to the limited learning capacity of in-context learning, it's essential to highlight only the most important information. A categorizer returning a voluminous array of questions might distract the LLM's focus on the specific learning required for solving a given problem. Moreover, LLMs can only accept a limited amount of tokens for each prompt.

Therefore, my objective is to find methods that categorize problems into more finely detailed categories, aiding LLMs in identifying and focusing on the most analogous problems to enhance problem-solving efficiency. The methodologies I currently propose are outlined below.

1. **Structural Equation Categorization:** Utilizing the structure of equations within solutions as categories. By abstracting numerical solutions to variable-based representations, we can classify problems by their solution structure. For instance, equations solved in steps like $50 + 100 + 40 + 60 = 250$ leading to $A + B + C + D = F$ could represent a category. Further, to make retrieving similar problems possible, it could be feasible to transform these equations into a tree structure (with each number and operator as nodes) and judge similarity based on the minimal edit distance between trees.

2. **Operator-Based Categorization:** A simpler method categorizing problems by the count of operators (addition, subtraction, multiplication, division) used in their solutions. This approach focuses on the quantitative aspect of the solution process.

4.2 Challenges of Each Proposed Categorized Method

Each proposed method contains specific draw backs.

Categorization via Structural Equations sounds like a highly promising approach since equations directly encapsulate the problem's essence. Nonetheless, capturing such equations poses a significant challenge. Given that the answers within datasets are exclusively numerical, deriving a variable-based representation without grasping the problem's semantics seems impossible. I tried to implement a naive categorization predicated on the input equation, but I encountered an endless array of edge cases. For details, please refer to the section 6.

Operator-Based Categorization This categorization approach is tailored to explore the concept of detailed, fine-grained prompting. While counting the number of mathematical operations can offer some insights into the targeted question, like its complexity (with more complex problems generally involving more operations), this method of categorization remains imprecise. For instance, a problem formulated as:

$$A + B = C,\ D + C = E$$

differs significantly from:

$$A + B = C,\ A + C = D$$

despite both featuring two addition operations. This highlights the limitations of relying solely on the number of operations to distinguish between mathematical problems.

5 Experiment

I carried out multiple experiments to see if providing additional related information would improve the performance of LLMs in solving math problems. For these experiments, I chose the GSM8K[10] dataset as the standard for comparison.

5.1 Categorization Experiments

Firstly, I tested the categorization ability of my purposed method. A good categorization tool should have sufficient examples per category, easy to use, and capture some aspect of the problem semantics.

I start with building a naive equation-based categorization tool that operates in the following order

1. Extract the solution equation from the answer
2. Scan through the numbers and replace each unique number appeared a letter starting from 'A'
3. Concatenating the equations with special sequence '###'

In the training set of 7473 problems, there are 4504 unique categories. However, 3975 of these categories consist of just a single problem. Such categories are not useful because over fifty percent of the questions lack accompanying problems for meaningful categorization.

"After reviewing the categories manually, I've concluded that an effective equation-based categorization tool needs to have:

1. **The ability to understand semantics:** For example, consider the equation
$$6 \times 6 = 36$$
. This can be seen in two different ways: either as $A \times A = B$ or $A \times C = B$, with each view representing a different mathematical meaning.
2. **The ability to consolidate functions:** Since answers can be expressed in various forms, the same problem might appear differently. Take, for instance, $A + B = C$, $C + D = E$, and $A + B + D = E$ - essentially, they convey the same idea. The tool should be able to merge these equations into one standardized format.

The need to grasp the underlying meaning of problems means we can't rely on a fixed, method-based tool for categorization. Nonetheless, I believe it's still possible to develop a dataset categorized by equations with the assistance of LLMs. In theory, LLMs should be capable of comprehending the answers to questions and converting them into mathematical equations without much difficulty.

To further explore whether more specific prompts could enhance the reasoning abilities of Large Language Models (LLMs), I developed a new tool that organizes problems by counting the specific types of operators used, such as how many '+'s, '-'s, '/'s, and '*'s there are.

This approach turned out to be quite effective at categorizing the problems. Out of 7473 problems, it created 432 distinct categories, with only 147 categories containing a single problem. Although this method doesn't fully grasp the underlying meaning of the problems, it's sufficient for proceeding with my experiment.

5.2 Correctness Experiment

In this experiment, I explored the hypothesis that providing LLMs with finely tuned prompts, which include examples closely related to the target, results in better performance compared to using generic Chain of Thought (COT) prompts.

The study involved constructing the following processes:

5.2.1 Related COT Process

Initially, I developed a tool for categorizing based on operators, denoted as C, capable of analyzing a problem's solution and identifying its operator class c. For instance, a problem solved by the steps

$$6 + 4 = 10,\ 10/2 = 5$$

would be categorized as

$$1001$$

, with each digit representing the count of $+, -, *, /$ operators, respectively.

Using C, I categorized all problems in the GSM8K[10] training set to form a new dataset D. For a given input question q, C determines its class c_q, which is then used to find related questions within D. Up to ten random questions from the same category as c_q are selected from D and included in the prompt as COT examples. If no related question is found, a standard COT prompt P is used instead. This standard prompt P is also utilized in the subsequent process.

The answers are then derived from the LLM's responses. For more details, please refer to the appendix.

5.2.2 Regular COT Process

For any input question q, it is submitted to the LLM along with the standard COT prompt P.

The answers are again obtained from the LLM's responses. For further information, please consult the appendix.

5.2.3 Experiment, Results, and Analysis

I selected 500 problems at random from the GSM8K[10] test set and processed them through both pipelines using GPT-3.5-turbo to compare their performance.

The results showed that the regular COT pipeline correctly answered 408 questions, while the related COT pipeline correctly answered 407 questions. There were 51 different mistakes made between the two approaches.

From these results, I propose the following hypotheses:

1. The operator-based classifier cannot capture extremely fine-grained details and may not differ from regular COT.
2. Variability in COT examples could be beneficial in providing hints to the model.
3. The results may not be robust, as I did not account for scenarios where there are only a few examples under target categories, leading to an insufficient number of related COT prompts.

Based on these insights, I plan to

1. Develop an equation-based categorization tool with the assistance of LLMs.
2. Construct a multi-level based RAG system that incorporates not only the most relevant information but also includes some less related examples.

6 Conclusion

Despite the outcomes of the experiment not being particularly encouraging, the occurrence of 51 different mistakes between the two systems highlights some intriguing potentials. Firstly, this suggests that LLMs, such as GPT-3.5-turbo (and not even considering more advanced versions like GPT-4 or the upcoming GPT-5), have the capability to correctly solve at least 459 out of 500 questions when given proper instructions. Secondly, although the experimental setup was not highly robust due to limitations in resources preventing repeated trials, it still demonstrates the potential that varying COT prompts can lead to different behaviors in LLMs. This is an aspect that could be easily exploited for better performance.

6.1 What I learned

1. Sadly (despite not surprisingly), RAG has been proposed.
2. It is actually pretty hard to categorize math problems without understanding the semantics even with the solution.
3. All papers listed are inspiring.
4. Research is actually fun!

Reference

1. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language Models are Few-Shot Learners. Retrieved from https://arxiv.org/abs/2005.14165

2. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., & Zhou, D. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. Retrieved from https://arxiv.org/abs/2201.11903

3. Yamauchi, R., Sonoda, S., Sannai, A., & Kumagai, W. (2023). LPML: LLM-Prompting Markup Language for Mathematical Reasoning. Retrieved from https://arxiv.org/abs/2309.13078

4. Imani, S., Du, L., & Shrivastava, H. (2023). MathPrompter: Mathematical Reasoning using Large Language Models. Retrieved from https://arxiv.org/abs/2303.05398

5. Hu, H., Peng, J., Wang, R., & Liang, P. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv. https://arxiv.org/abs/2106.09685

6. Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-t. (2020). Dense Passage Retrieval for Open-Domain Question Answering. Retrieved from https://arxiv.org/abs/2004.04906

7. Bang, F., Tan, L., Milajevs, D., Chauhan, G., Gwinnup, J., & Rippeth, E. (2023). GPTCache: An Open-Source Semantic Cache for LLM Applications Enabling Faster Answers and Cost Savings. In Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023), December 2023. Association for Computational Linguistics.

8. Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

9. Wu, Y., Lelkes, A. D., Chen, X., et al. (2023). REACT: Synergizing Reasoning and Acting in Language Models. arXiv. arXiv:2302.06976. Available at https://arxiv.org/abs/2302.06976

10. Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., & Schulman, J. (2021). Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.

11. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., & Gelly, S. (2019). Parameter-Efficient Transfer Learning for NLP. Proceedings of the 36th International Conference on Machine Learning, PMLR 97:2962-2971.

12. Cetintas, S., Si, L., Xin, Y. P., Zhang, D., & Park, J. Y. (2023). Automatic Text Categorization of Mathematical Word Problems. Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference (FLAIRS 2009). AAAI.

13. Lewis, P., Oguz, B., Rinott, R., Riedel, S., & Schwenk, H. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv. https://arxiv.org/abs/2005.11401

7 Appendix

7.1 Fixed COT Examples

```
"question": "At 30, Anika is 4/3 the age of Maddie.
What would be their average age in 15 years?", "answer": "#### 50"
 \n
 "question": "Janet, a third grade teacher, is picking up the sack lunch order from
 local deli for the field trip she is taking her class on.
 There are 35 children in her class, 5 volunteer chaperones, and herself.
 She she also ordered three additional sack lunches, just in case there was a proble
 Each sack lunch costs $7.
 How much do all the lunches cost in total?", "answer": "#### 308"
 \n
 "question": "Colin can skip at six times the speed that Brandon can.
 Brandon can skip at one-third the speed that Tony can.
 And Tony can skip at twice the speed that Bruce can.
 At what speed, in miles per hour, can Colin skip if Bruce skips at
 1 mile per hour?", "answer": "#### 4"

 \n
 "question": "Josh is saving up for a box of cookies.
 To raise the money, he is going to make bracelets and sell them.
 It costs $1 for supplies for each bracelet and he sells each one for $1.5.
 If he makes 12 bracelets and after buying the cookies still
 has $3, how much did the box of cookies cost?", "answer": "#### 3"
 \n

 "question": "Very early this morning, Elise left home in
 a cab headed for the hospital.
 Fortunately, the roads were clear, and the cab company only charged her a
 base price of $3, and $4 for every mile she traveled.
 If Elise paid a total of $23, how far is the hospital from her house?",
 "answer": "#### 5"
 \n
 "question": "Roy owns a refrigerated warehouse where he
 stores produce before selling it at the farmer\u2019s market.
 The fruits and vegetables he stores are very sensitive to temperature,
```

```
and he must keep them all cold or they will spoil.
One day, the power went out and the air conditioner was turned off for three hours,
during which time the temperature rose by 8 degrees per hour.
If Roy gets the power back on, it will activate the air conditioner
to lower the temperature at the rate of 4 degrees F per hour.
What is the amount of time, in hours, it will take for the
air conditioner to restore the warehouse to 43 degrees F?",
"answer": "#### 6"
\n
"question": "Janet pays $40/hour for 3 hours per week of clarinet lessons and
$28/hour for 5 hours a week of piano lessons. How much more does she spend on piano
lessons than clarinet lessons in a year?", "answer": "#### 1040"
```

7.2 COT Prompt

```
"""
    You are given a math question {information}, think through it step by step and prov

    1. step by step reasoning
    2. Provide the answer as a integer at the end with four '#' and one space character
    Don't put unit or '%' sign in the final answer. one number only
   For example:
    ```It takes 2/2=<<2/2=1>>1 bolt of white fiber\nSo the total amount
    of fabric is 2+1=<<2+1=3>>3 bolts of fabric\n#### 3```
    \n
    Here are few shot example:
   {COT Prompt}
 """
```

7.3 Related COT Prompt

```
"""
    You are given a math question {information}, think through it
    step by step and provide me
    1. step by step reasoning
    2. Check your reasoning against the question sentence by sentence
    3. Provide the answer as a integer at the end with four '#'
    and one space character
    before it. Don't put unit or '%' sign in the final answer. one number only
    For example:
    ```It takes 2/2=<<2/2=1>>1 bolt of white fiber\nSo the total amount
    of fabric is 2+1=<<2+1=3>>3 bolts of fabric\n#### 3```
    \n
    Here are some similar problems and answers for your reference (Could be none):
    {relatedProblems}
```

```
    \n
"""
```

7.4 Venn Diagram

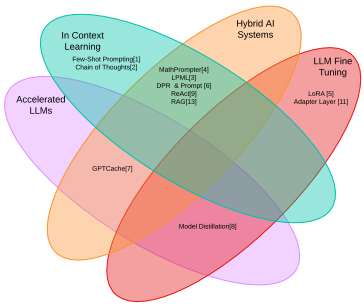

Figure 1: Venn Diagram

Changes

1. Add references
2. Change ReAct & Rest Agent to ReAct Agent as it is more related
3. Add RAG
4. Not mentioning research [12] and GSM8K[10] as they are not the core of this study.