CS447 Literature Review: Effectiveness of Prompting Techniques for LLM-Based Complex Reasoning

Anikesh Haran,
anikesh2@illinois.edu

December 9, 2023

Abstract

Prompting techniques have emerged as a powerful tool for enhancing the performance of large language models (LLMs) in complex reasoning tasks. These techniques provide LLMs with additional information or instructions to guide their reasoning process, leading to improved accuracy and generalization capabilities. In this paper, we compare the effectiveness of three different prompting techniques: chain-of-thought prompting, retrieval-augmented generation, and Tree of Thoughts. We apply these techniques to a variety of complex reasoning tasks and evaluate their performance using a range of metrics. Our results show that all three techniques can improve the performance of LLMs on complex reasoning tasks, but their effectiveness varies depending on the task and the specific LLM being used. Chain-of-thought prompting is generally the most effective technique for tasks that require explicit reasoning steps, while retrieval-augmented generation is more effective for tasks that require access to external knowledge. Tree of Thoughts is a promising new technique that has the potential to be even more effective than the other two techniques for a wider range of tasks.

Keywords: Large language models, complex reasoning, prompting techniques, chain-of-thought prompting, retrieval-augmented generation, Tree of Thoughts

1 Introduction

Large pre-trained language models have been shown to store factual knowledge in their parameters, and achieve state-of-the-art results when fine-tuned on downstream NLP tasks. However, they can not easily expand or revise their memory, For example if we give a new information such as ”Incident x happen in 1990” to the model, actually does not change anything into the existing knowledge of the model. Secondly, the Large pre-trained models can not provide insight into their prediction, its really hard to explain **[lack of explainability / reasoning]** why model predicted a particular sequence of words. Also, they may ”Hallucinate” factual knowledge. they may generate false factual knowledge which actually does not exists. This leads to fake news and articles. Large language models also face difficulties in performing intermediate reasoning task such as range of arithmetic, commonsense, and symbolic reasoning tasks.

This paper embarks on a journey to explore and evaluate the effectiveness of various prompting techniques tailored for LLMs, with a specific focus on enhancing their capabilities in intricate reasoning tasks. Our goal is to delve into innovative prompting techniques— retrieval-augmented generation Lewis et al. (2020), chain-of-thought prompting Wei et al. (2022), and Tree-of-Thoughts Yao et al. (2023), ART Paranjape et al. (2023) and examine their comparative effectiveness in addressing the limitations of LLMs in the context of complex reasoning task.

As we progress, this paper will briefly cover each paper i am reviewing and the strengths of each prompting technique, providing valuable insights into their applicability across diverse reasoning tasks. The journey is not only about assessing effectiveness but also understanding the specific contexts in which these techniques excel.

2 Motivation

Before we deep dive into the specifics of prompting techniques for Large Language Models (LLMs), it's crucial to understand the motivation behind choosing these four papers. Each of the papers being discussed in the next section makes a significant contribution to the field of natural language processing (NLP) and large language models. Let's break down the importance of each paper:

#1 Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

- Innovation in NLP: The paper introduces a novel approach to NLP tasks by combining retrieval-based models with generative models. This fusion enhances the model's ability to perform knowledge-intensive tasks.
- Practical Applications: The proposed method has applications in various NLP tasks that require a deep understanding of context and relevant knowledge.
- Addressing Limitations: The paper addresses limitations in purely generative or purely retrieval-based models, providing a more holistic solution.

#2 Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

- Prompting for Reasoning: The paper introduces a method of prompting language models to elicit reasoning, contributing to the development of more explainable and interpretable AI systems.
- Insights into Model Behavior: Understanding how specific prompts lead to reasoning can provide insights into the inner workings of large language models.
- Applications in Explainable AI: The research has implications for improving the transparency and interpretability of AI systems, making them more accountable and trustworthy.

#3 Tree of Thoughts: Deliberate Problem Solving with Large Language Models

- Deliberate Problem Solving: The paper explores how large language models can engage in deliberate problem-solving, shedding light on their cognitive processes.
- Understanding Thought Processes: By examining the ”tree of thoughts,” the paper contributes to understanding how language models approach and solve complex problems.
- Implications for AI Education: Insights from this research could influence the design of educational tools and approaches that leverage large language models for problem-solving and learning.

#4 ART: Automatic Multi-Step Reasoning and Tool-Use for Large Language Models

- Advancing Reasoning Capabilities: The paper focuses on enhancing the reasoning abilities of large language models by introducing automatic multi-step reasoning and tool-use. This is crucial for more complex problem-solving tasks.

- Practical AI Applications: The proposed techniques have potential applications in real-world scenarios where complex decision-making and multi-step reasoning are required.
- Model Interpretability: The paper may contribute to understanding and interpreting the decision-making process of large language models, which is essential for building trust in AI systems.

The effectiveness of these techniques lies in their ability to tailor the guidance based on the task at hand. Whether it's understanding complex reasoning, recalling facts, or making predictions, prompting techniques strive to make the LLM's more capable in these areas. In summary, each of these papers tackles different aspects of improving language models, ranging from knowledge integration and reasoning to deliberate problem-solving and explainability.

3 Various Prompting Techniques

In the following section, I will furnish comprehensive details on each of the four papers, which will be the focus of my review.

3.1 Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

The Retrieval-Augmented Generation [RAG] models are Hybrid models that combine parametric memory with non-parametric retrieval-based memories and address some of the issues with large pre-trained models such as expansion or revision of the memory, model explainability and knowledge hallucination issues.

As you can see in Figure 1, On a high-level they have build a probabilistic model trained end-to-end by combining pre-trained retriever (Query Encoder + Document Index) with a pre-trained seq2seq model (Generator). In RAG models the parametric memory is a pre-trained seq2seq transformer, and the non-parametric memory is a dense vector index of Wikipedia, accessed with a pre-trained neural retriever.

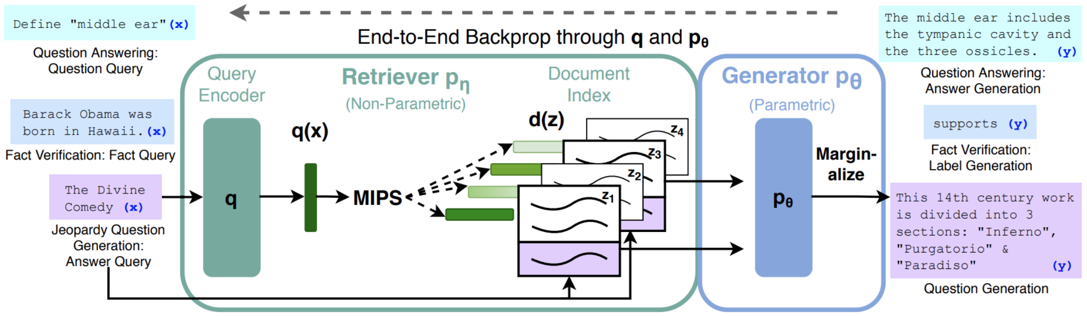

Define "middle ear" (x)
Question Answering: Question Query
Barack Obama was born in Hawaii. (x)
Fact Verification: Fact Query
The Divine Comedy (x)
Jeopardy Question Generation: Answer Query
End-to-End Backprop through q and p_θ
Query Encoder
Retriever p_η
(Non-Parametric)
Document Index
q(x)
q
MIPS
d(z)
z_4
z_3
z_2
z_1
Generator p_θ
(Parametric)
p_θ
Marginalize
The middle ear includes the tympanic cavity and the three ossicles. (y)
Question Answering: Answer Generation
supports (y)
Fact Verification: Label Generation
This 14th century work is divided into 3 sections: "Inferno", "Purgatorio" & "Paradiso" (y)
Question Generation

Figure 1: Overview of RAG model for Knowledge-Intensive NLP Tasks

3.1.1 Retrieval Component:

The model has access to a large database or knowledge base containing information, like articles, documents, or any structured data. It retrieves relevant information from this knowledge base based on the input query.

3.1.2 Generation Component:

Once it retrieves information, the model doesn't just copy-paste the answer. It also generates new content to ensure the response is coherent and fits well in the given context. This generation aspect makes the model more flexible and capable of producing human-like responses.

3.1.3 Method

The RAG model which use the input sequence x to retrieve text documents z and use them as additional context when generating the target sequence y. As shown in Figure 1, the model leverage two components:

1. Retriever $p - eta(z|x)$ with parameters eta that returns (top-K truncated - ex. 100 words) distributions over text passages given a query x

2. A Generator $p - theta(yi|x,z,y1 : i-1)$ parameterized by $theta$ that generates a current token based on a context of the previous $i-1$ tokens $y1 : i-1$, the original input x and a retrieved passage z.

There are 2 RAG model variants: RAG-Sequence Model and RAG-Token Model.

RAG-Sequence Model: The RAG-Sequence model uses the same retrieved document to generate the complete sequence. Technically, it treats the retrieved document as a single latent variable that is marginalized to get the seq2seq probability $p(y|x)$ via a top-K approximation.Concretely, the top K documents are retrieved using the retriever, and the generator produces the output sequence probability for each document, which are then marginalized as -

$$p_{\text{RAG-Sequence}}(y|x) \approx \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y|x,z) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) \prod_i^N p_\theta(y_i|x,z,y_{1:i-1})$$

Figure 2: RAG-Sequence Model

RAG-Token Model: In the RAG-Token model, a different latent document is drawn for each target token and marginalized accordingly. This allows the generator to choose content from several documents when producing an answer. Concretely, the top K documents are retrieved using the retriever, and then the generator produces a distribution for the next output token for each document, before marginalizing, and repeating the process with the following output token, Formally:

$$p_{\text{RAG-Token}}(y|x) \approx \prod_i^N \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y_i|x,z,y_{1:i-1})$$

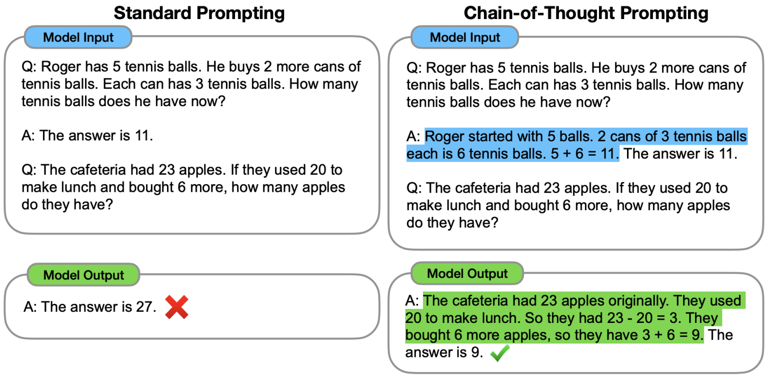
Figure 3: RAG-Token Model

The RAG model is suitable for tasks where humans could not reasonably be expected to perform without access to an external knowledge source. RAG models achieve state-of-the-art results on Open Natural Questions, Web Questions and strongly outperform recent approaches that use specialised pre-training objectives. The Retrieval-Augmented Generation (RAG) model is a type of natural language processing model designed to answer questions or generate text based on a combination of retrieving information and generating new content.

3.2 Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Large Language Models (LLMs) have revolutionized the field of artificial intelligence, offering unprecedented capabilities in natural language understanding and generation. Scaling up the size of language models provide lot of benefits, such as improved performance and sample efficiency. However, scaling up model size alone has not proved sufficient for achieving high performance on challenging tasks such as arithmetic, commonsense, and symbolic reasoning. Their ability to perform complex reasoning tasks has been a subject of intense research. One technique that has shown promise in this regard is Chain-of-Thought (CoT) prompting.

The standard prompting Brown et al. (2020) asks the model to directly give the answer to a multi-step reasoning problem, chain of thought prompting induces the model to decompose the problem into intermediate reasoning steps and produce more accurate results. CoT prompting, as introduced in this paper, is a method that encourages LLMs to explain their reasoning process. This is achieved by providing the model with a few-shot exemplars where the reasoning process is explicitly outlined. The LLM is then expected to follow a similar reasoning process when answering the prompt. This technique has been found to significantly improve the model's performance on tasks that require complex reasoning.



Q: 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?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

A: The answer is 27. ✗

Q: 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?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

A: 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 9. ✓

Figure 4: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

Initially there were two ideas for Arithmetic Reasoning task, the first one was to train a model or fine-tune a pre-trained model to generate natural language intermediate steps and second was using in-context few-shot learning via prompting that means simply "prompt" the model with a few input–output exemplars demonstrating the task. Both ideas works very well on simple question-answering tasks. but both of them have limitations. For rationale-augmented training and fine-tuning methods, it is costly to create a large set of high quality rationales, which is much more complicated than simple input–output pairs used in normal machine learning. For the traditional few-shot prompting method used in Brown et al. (2020), it works poorly on tasks that require reasoning abilities, and often does not improve substantially with increasing language model scale.

Chain-Of-thought combines the strengths of these two ideas without adhering their limitation.

3.2.1 Chain-of-Thought Prompting

Figure 4 shows an example of a model producing a chain of thought to solve a math word problem. Chain-of-thought prompting has several attractive properties as an approach for facilitating reasoning in language models.

1. COT allows models to decompose multi-step problems into intermediate steps, which means that additional computation can be allocated to problems that require more reasoning steps.

2. COT provides an interpretability which helps in debugging where the reasoning path went wrong.

3. COT reasoning can be used for tasks such as math word problems, commonsense reasoning, and symbolic manipulation, and is potentially applicable.

4. COT reasoning can be readily elicited in sufficiently large off-the-shelf language models simply by including examples of chain of thought sequences into the exemplars of few-shot prompting.

At its core, CoT prompting is about guiding the LLM to think step by step. This is achieved by providing the model with a few-shot exemplar that outlines the reasoning process. The model is then expected to follow a similar chain of thought when answering the prompt. This approach is particularly effective for complex tasks that require a series of reasoning steps before a response can be generated.

3.3 Tree of Thoughts: Deliberate Problem Solving with Large Language Models

Language models are increasingly being deployed for general problem solving across a wide range of tasks, but are still confined to token-level, left-to-right decision-making processes during inference. This means they can fall short in tasks that require exploration, strategic look-ahead and backtracking where initial decisions play a pivotal role.

> ***A genuine problem-solving process involves the repeated use of available information to initiate exploration, which discloses, in turn, more information until a way to attain the solution is finally discovered.—— Newell et al. [18]***

Figure 5: Newell et al.

As i mentioned earlier, in COT we guide the LLM to think step by step by providing some few-shot examples that helps outlining the reasoning, but this is not exactly same the way human brain solves the problem. As we know, there are N number of ways to solve a problem and at each step we take a decision on the next action or step. The ”Tree-of-Thought” [TOT] Yao et al. (2023) is a new framework for large language model inference which generalize over the ”Chain-of-Thought [COT]” approach to prompting language models, and enables exploration over coherent units of text (“thoughts”) that serve as intermediate steps toward problem solving. ToT allows LMs to perform deliberate decision making by considering multiple different reasoning paths and self-evaluating choices to decide the next course of action, as well as looking ahead or backtracking when necessary to make global choices.

Experiments show that ToT significantly enhances language model's problem-solving abilities. For instance, in Game of 24, while GPT-4 with chain-of-thought prompting only solved 4% percent of tasks, The TOT prompting approach was able to solve 74% of problems successfully.

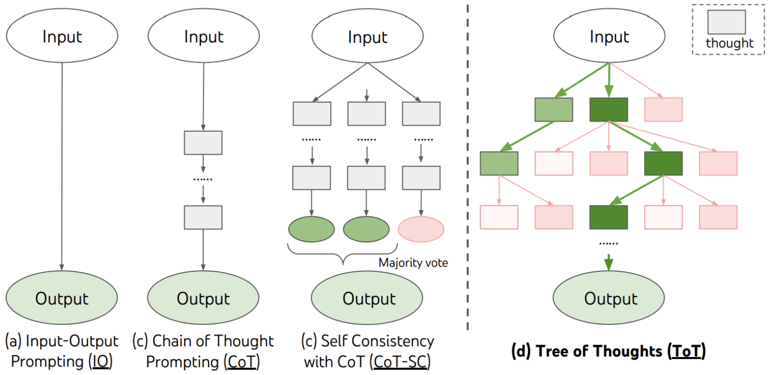

Input
Input
Input
Input
thought
Majority vote
Output
Output
Output
Output
(a) Input-Output Prompting (IO)
(c) Chain of Thought Prompting (CoT)
(c) Self Consistency with CoT (CoT-SC)
(d) Tree of Thoughts (ToT)

Figure 6: Schematic illustrating various approaches to problem solving with LLMs. Each rectangle box represents a thought, which is a coherent language sequence that serves as an intermediate step toward problem solving

Research on human problem-solving suggests that people search through a combinatorial problem-space – a tree where the nodes represent partial solutions, and the branches correspond to operators that modify them. Which branch to take is determined by heuristics that help to navigate the problem-space and guide the problem-solver towards a solution. This perspective highlights two key shortcomings of existing approaches that use LMs to solve general problems: 1) Locally, they do not explore different continuations within a thought process – the branches of the tree. 2) Globally, they do not incorporate any type of planning, look-ahead, or backtracking to help evaluate these different options – this kind of heuristic-guided search that seems characteristic of human problem-solving.

The Tree of Thoughts (ToT) prompting approach addresses these shortcomings and allows LMs to explore multiple reasoning paths over thoughts. ToT frames any problem as a search over a tree, where each node is a state s = [x, z1···i] representing a partial solution with the input and the sequence of thoughts so far. A specific instantiation of ToT involves answering four questions: 1. How to decompose the intermediate process into thought steps; 2. How to generate potential thoughts from each state; 3. How to heuristically evaluate states; 4. What search algorithm to use.

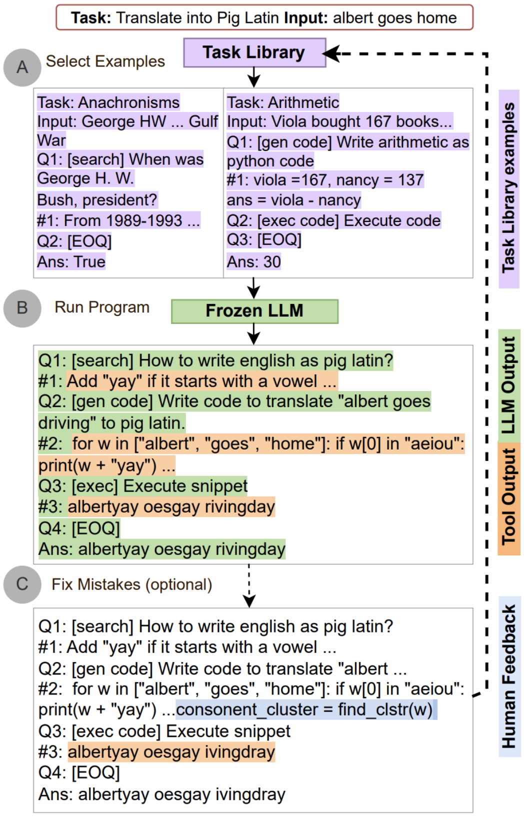
More details can be found on how TOT works, in this paper my focus will be on how effective TOT is when compare to other prompting approaches.

3.4 ART: Automatic multi-step reasoning and tool-use for large language models

Large language models (LLMs) can perform complex reasoning in few-shot and zero-shot settings by generating intermediate chain of thought (CoT) reasoning steps but Chain-of-Thought prompting typically requires hand-crafting task-specific demonstrations and carefully scripted interleaving of model generations to use.

In this section i will talk about Automatic Reasoning and Tool-use (ART), a framework that uses frozen LLMs to automatically generate intermediate reasoning steps as a program.

For given new task to solve, ART selects demonstrations of multistep reasoning and tool use from a **task library**. ART is capable of pausing generation whenever external tools are called, and integrates the output before resuming generation. ART achieves a substantial improvement over few-shot prompting and automatic CoT on unseen tasks in the BigBench and MMLU benchmarks, and matches performance of hand-crafted CoT prompts on a majority of these tasks. ART is also extensible, and makes it easy for humans to improve performance by correcting errors in task-specific programs or incorporating new tools.



Task: Anachronisms
Input: George HW ... Gulf War
Q1: [search] When was George H. W. Bush, president?
#1: From 1989-1993 ...
Q2: [EOQ]
Ans: True

Task: Arithmetic
Input: Viola bought 167 books...
Q1: [gen code] Write arithmetic as python code
#1: viola =167, nancy = 137
ans = viola - nancy
Q2: [exec code] Execute code
Q3: [EOQ]
Ans: 30

Q1: [search] How to write english as pig latin?
#1: Add "yay" if it starts with a vowel ...
Q2: [gen code] Write code to translate "albert goes driving" to pig latin.
#2: for w in ["albert", "goes", "home"]: if w[0] in "aeiou":
print(w + "yay") ...
Q3: [exec] Execute snippet
#3: albertyay oesgay rivingday
Q4: [EOQ]
Ans: albertyay oesgay rivingday

Q1: [search] How to write english as pig latin?
#1: Add "yay" if it starts with a vowel ...
Q2: [gen code] Write code to translate "albert ...
#2: for w in ["albert", "goes", "home"]: if w[0] in "aeiou":
print(w + "yay") ...consonent_cluster = find_clstr(w)
Q3: [exec code] Execute snippet
#3: albertyay oesgay ivingdray
Q4: [EOQ]
Ans: albertyay oesgay ivingdray

Figure 7: ART generates automatic multi-step decomposition's for new tasks by selecting decomposition's of related tasks in the task libray (A) and selecting and using tools in the tool library alongside LLM generation (B). Humans can optionally edit decomposition's (eg. correcting and editing code) to improve performance (C).

4 Discussion

Complex reasoning tasks in Natural Language Processing (NLP) refer to challenges that go beyond simple language understanding and require the model to perform intricate cognitive operations. These tasks often demand a higher level of reasoning, abstraction, and comprehension, making them more sophisticated than routine language processing.

In the subsequent section of this review, we delve into the intricacies of several complex reasoning tasks [as shown in Table 1] in NLP. Our aim is to evaluate effectiveness of various prompting techniques [Lewis et al. (2020), Paranjape et al. (2023), Wei et al. (2022), Yao et al. (2023)], can address the demands posed by these intricate tasks. By evaluating the performance of prompting methods such as retrieval-augmented generation, chain-of-thought prompting, and Tree of Thoughts, we seek to discern their impact on the model's ability to handle the complexities inherent in tasks that demand advanced reasoning abilities. This examination will shed light on the practical applicability of these techniques in enhancing the model's performance across a spectrum of complex reasoning challenges in the domain of NLP.

4.1 Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

There has been lot of work done previously to solve complex reasoning task such as - Open-domain Question Answering, Abstractive Question Answering, Jeopardy Question Generation and Fact Verification (FEVER). Previous techniques such as Single-Task Retrieval, General-Purpose Architectures such as GPT-2 Radford et al. (2018), BERT, and T5 Raffel et al. (2019), have shown remarkable success on diverse tasks without relying on retrieval. Also other techniques such as Learned Retrieval, Memory-based Architectures and Retrieve-and-Edit Approaches improves performance across a variety of NLP tasks when considered in isolation.

As described in the paper, here are the summary of related work and how this paper is contributing in solving/enhancing some specific reasoning tasks -

4.1.1 Single-Task Retrieval:

- Previous studies have demonstrated the effectiveness of retrieval in enhancing performance across various NLP tasks, such as question answering, fact checking, dialogue, translation, and language modeling.
- The paper contributes by unifying these successes, showcasing that a single retrieval-based architecture can achieve strong performance across multiple tasks.

4.1.2 General-Purpose Architectures for NLP:

- Traditional pre-trained language models, like GPT-2, BERT, and T5, have shown remarkable success on diverse tasks without relying on retrieval.
- This paper expands the capabilities of general-purpose architectures by incorporating a retrieval module to augment pre-trained, generative language models, aiming to cover a broader spectrum of tasks within a unified framework.

The paper introduces a groundbreaking approach, outlined in Figure 1, that fuses a pre-trained retriever with a pre-trained seq2seq model, leveraging both parametric and non-parametric memory for end-to-end fine-tuning. The authors term this innovative methodology as retrieval-augmented generation (RAG). Notably, this method differs from previous work by incorporating pre-trained components, eliminating the need for task-specific training of non-parametric memory.

The combination of a pre-trained seq2seq transformer as parametric memory and a dense vector index of Wikipedia as non-parametric memory sets the stage for a powerful probabilistic model, trained end-to-end. The Dense Passage Retriever (DPR) provides latent documents, while the seq2seq model (BART) conditions on these documents along with the input to generate the output.

The results presented in the paper underscore the efficacy of RAG models, showcasing state-of-the-art performance on various tasks, including open Natural Questions, WebQuestions, and CuratedTrec. Notably, RAG outperforms recent approaches using specialized pre-training objectives on

Commonsense Reasoning	Understanding and applying everyday knowledge and reasoning abilities that go beyond explicit information in the text. For example, comprehending jokes, identifying implied meanings, or resolving ambiguous statements.
Arithmetic Reasoning	Performing mathematical operations expressed in natural language. This involves interpreting and solving word problems, equations, or mathematical expressions within a given context.
Temporal Reasoning	Grasping and reasoning about temporal relationships expressed in text, such as understanding the order of events, durations, or intervals.
Causal Reasoning	Determining cause-and-effect relationships between events or statements. This task involves identifying the reasons behind certain occurrences or predicting the consequences of specific actions.
Logical Reasoning	Applying logical operations to draw conclusions from information provided in text. This includes tasks like identifying contradictions, validating arguments, or making deductive inferences.
Analogical Reasoning	Recognizing and applying analogies within language. This involves understanding relationships between words or concepts and extending that understanding to solve new problems.
Abstractive Summarization or Abstractive Question Answering	Generating concise and coherent summaries that require not just copying parts of the input but understanding the context and generating new, informative content.
Machine Reading Comprehension	Going beyond basic question-answering tasks, machine reading comprehension involves understanding and synthesizing information from longer passages, often requiring multiple steps of reasoning.
Quantitative Reasoning	Dealing with quantitative information and performing operations like comparison, aggregation, or computation based on numerical data provided in the text.
Symbolic Reasoning	Manipulating symbols and understanding symbolic relationships expressed in language, often involving tasks related to programming languages or formal systems.

Table 1: Various Reasoning Tasks

TriviaQA. Even in extractive tasks, RAG's unconstrained generation demonstrates superior performance.

The paper goes beyond extractive tasks, experimenting with knowledge-intensive generation tasks like MS-MARCO and Jeopardy question generation. The results show that RAG models generate responses that are more factual, specific, and diverse compared to a BART baseline. Additionally, in FEVER fact verification, RAG achieves results within 4.3% of state-of-the-art pipeline models, which rely on strong retrieval supervision.

An intriguing aspect of the paper is its demonstration of the adaptability of non-parametric memory. The authors show that this memory component can be replaced to update the model's knowledge as the world changes, highlighting the potential for real-time knowledge enhancement.

In conclusion, the paper introduces a transformative approach to NLP, pushing the boundaries of what is achievable with the combination of parametric and non-parametric memory. The comprehensive experiments and impressive results make a compelling case for the effectiveness of retrieval-augmented generation in handling knowledge-intensive tasks, setting a new standard in the field.

4.2 Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

In this paper authors majorly focus another set of complex reasoning task such as arithmetic reasoning where aim is to make the model solve a mathematical problem and express in natural language and the commonsense reasoning where aim is to make model to apply everyday knowledge to identify meaning of the text or summarizing the text using reasoning abilities. Also this paper focuses on symbolic reasoning which is very famous now days related to understand and explain symbols in programming languages and format math syntax and expression.

This paper is inspired by many previous researches, notably emphasizing the use of intermediate steps. This paper further improvise by Building on Ling et al. (2017) concept of employing natural language rationales for math word problems, Cobbe et al. (2021) extend this by creating a larger dataset and fine-tuning a pre-trained language model. In the realm of program synthesis, Nye et al. (2021) leverage language models to predict Python program outputs by first predicting intermediate computational steps. The study is aligned with recent prompting advancements, contrasting with approaches that enhance input prompts, as it uniquely focuses on augmenting language model outputs with a chain of thought, setting it apart from prevalent methods like few-shot prompting.

4.2.1 Arithmetic Reasoning

As mentioned in the paper, there has been lot of work done previously to perform math operations using Neural Calculator using transformer but these approaches struggle in solving problems which are very simple for humans.

Author did the baseline comparison which involves standard few-shot prompting, where language models receive in-context exemplars of input-output pairs before making predictions. The proposed chain-of-thought prompting augments each exemplar with a reasoning chain for the associated answer. Further evaluation involves five large language models, including GPT-3, LaMDA, PaLM, UL2 20B, and Codex. The results of the chain-of-thought prompting experiments are compelling and provide valuable insights into its impact on large language models across challenging math problem benchmarks.

Three key takeaways emerge from the findings.

Firstly, the effectiveness of chain-of-thought prompting is contingent on the model scale, with significant performance gains observed for models of approximately 100 billion parameters. Smaller-scale models demonstrated fluent but illogical chains of thought, leading to inferior performance compared to standard prompting.

Secondly, the performance gains of chain-of-thought prompting are more pronounced for complex problems, as evidenced by substantial improvements for datasets with lower baseline performance. Notably, the largest models exhibit more than doubled performance for the GSM8K dataset, which represents the most challenging problems.

Thirdly, when employing chain-of-thought prompting, particularly with GPT-3 175B and PaLM 540B, the models compare favorably to prior state-of-the-art approaches that typically involve fine-tuning on labeled training datasets. PaLM 540B, in particular, achieves new state-of-the-art results on GSM8K, SVAMP, and MAWPS, while approaching within 2% of the state of the art on AQuA and ASDiv datasets.

4.2.2 Commonsense Reasoning

The adaptability of the chain-of-thought approach extends beyond its effectiveness in solving math word problems. Its language-centric nature allows it to be applied to a diverse range of commonsense reasoning challenges, encompassing the deduction of conclusions related to both physical and human interactions, grounded in general background knowledge. Commonsense reasoning, crucial for meaningful engagement with the world, remains a formidable challenge for contemporary natural language understanding systems, as emphasized by Talmor et al. (2022)

As per the results, depicted in the paper for PaLM (with comprehensive data for LaMDA, GPT-3, and varying model scales), reveal that increasing the model size enhances standard prompting performance. The introduction of chain-of-thought prompting further amplifies these gains, with PaLM 540B achieving notable success, surpassing prior benchmarks on StrategyQA Geva et al. (2021) and sports understanding. This indicates that chain-of-thought prompting enhances commonsense reasoning across various tasks, showcasing significant improvements, particularly for larger model scales. Notably, this approach's impact was marginal on CSQA Talmor et al. (2019) but demonstrated substantial gains in other commonsense reasoning domains.

In conclusion, the results underscore the effectiveness of chain-of-thought prompting in enabling large language models to tackle complex math problems. The findings contribute to our understanding of the interplay between model scale, problem complexity, and the success of novel prompting techniques, offering valuable implications for future advancements in natural language processing research.

As authors mentioned in the paper, there are few limitations as well with COT approach. Firstly, while chain of thought mimics human reasoning processes, it does not confirm whether neural networks are genuinely engaging in ”reasoning,” Secondly, while manually augmenting exemplars with chains of thought is feasible in few-shot scenarios, the associated annotation costs may be prohibitive for fine-tuning, although potential solutions like synthetic data generation or zero-shot generalization are mentioned. Thirdly, the absence of a guaranteed correct reasoning path may lead to both accurate and inaccurate answers, prompting the need for future work in improving the factual accuracy of language model generations. Lastly, the emergence of chain-of-thought reasoning only in large-scale models raises concerns about its practical applicability in real-world scenarios, prompting further research into inducing reasoning in smaller models.

4.3 Tree of Thoughts: Deliberate Problem Solving with Large Language Models

The paper ”Tree of Thoughts: Deliberate Problem Solving with Large Language Models” Yao et al. (2023) proposes a novel approach to solving challenging problems using large language models, specifically GPT-4. The authors introduce three tasks that are difficult even for state-of-the-art language models when employing standard input-output prompting or chain-of-thought prompting. The focus is on deliberate search in ”trees of thoughts” (ToT) to yield improved and intriguing solutions to problems requiring search or planning.

One of the tasks explored in the paper is the ”Game of 24,” a mathematical reasoning challenge where the goal is to use four numbers and basic arithmetic operations to obtain a result of 24. The authors compare the performance of various prompting methods, including standard input-output, chain-of-thought, and their proposed Tree of Thoughts approach. They present results indicating that deliberate search in ToT outperforms other methods in terms of success rate.

The Game of 24 results table shows that the ToT method achieves a success rate of 45% with a breadth of 1 and 74% with a breadth of 5, outperforming input-output and chain-of-thought prompting methods. The authors also conduct scale and error analyses, demonstrating the advantages of the ToT approach over traditional prompting methods.

Overall, ToT exhibits a unique blend of advantages that makes it a powerful tool for enhancing the problem-solving capabilities of LLMs. While other techniques offer specific strengths, ToT’s combined performance, explainability, and efficiency advantages position it as a front-runner in the field of advanced prompting techniques.

It’s important to note that this is an ongoing area of research, and further comparisons are needed to assess the effectiveness of ToT across diverse task domains and LLM architectures.

4.4 ART: Automatic multi-step reasoning and tool-use for large language models

This paper discuss briefly about many prompting techniques their evolution and and their limitations. The major breakthrough in prompting was with intermediate reasoning steps discussed in Chain-of-Thoughts [COT] prompting. While such prompts were initially hand-crafted, the recent work Kojima et al. (2022) showed that LLMs can generate CoT-style multi-step reasoning in a zero-shot manner, when prompted with the prefix. We can use LLMs to automatically generate such CoT-style prompts AutoCoT which are competitive with hand-crafted prompts in their performance on arithmetic and commonsense reasoning tasks. The authors of the paper did a detailed comparison of ART with related approaches for multi-step reasoning and tools such as COT, Auto-COT and Tool-former.

This review will assess ART’s effectiveness compared to other prompting techniques, focusing on its advantages and limitations.

In a nutshell the strengths of ART are as follow -

- **Automatic generation of intermediate reasoning steps:** This eliminates the need for manually crafting complex prompts, making it more efficient and scalable.
- **Improved performance on unseen tasks:** ART achieves substantial improvements over few-shot prompting and automatic CoT on unseen tasks in the BigBench and MMLU benchmarks.
- **Matches hand-crafted CoT prompts:** ART performs as well as hand-crafted CoT prompts on most tasks, demonstrating its ability to capture complex reasoning chains.

- **Programmatic representation of reasoning steps:** Converting reasoning steps into programs allows for easier interpretation and debugging.
- **Potential for wider applicability:** The framework can be applied to various tasks involving multi-step reasoning and tool-use, beyond the evaluated benchmarks.

As authors mentioned in the paper, ART has some limitations as well such as -

- **Black-box nature:** The generated programs might be difficult to interpret and understand, limiting its transparency and controllability.
- **Computational cost:** Generating intermediate reasoning steps can be computationally expensive, especially for complex tasks.
- **Limited to frozen LLMs:** ART currently only works with frozen LLMs, which restricts its ability to adapt and learn from new information.
- **Potential for bias and errors:** Errors in the generated program or biases in the LLM can lead to incorrect results.

This paper shows promising results on comparison with other prompting techniques:

- **Few-shot prompting:** ART offers a more automated and efficient way to generate intermediate reasoning steps compared to manually crafting few-shot prompts.
- **Automatic CoT:** ART improves upon automatic CoT by generating more accurate and complete reasoning chains.
- **Hand-crafted CoT prompts:** ART can achieve similar or better performance than hand-crafted CoT prompts, but it is more efficient and less prone to human error.

ART presents a significant advancement in prompting techniques for large language models. It automates the generation of intermediate reasoning steps, leading to improved performance on various tasks. However, certain limitations like computational cost and black-box nature need to be addressed for wider adoption. Overall, ART holds great potential for further development and application in NLP research.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are Few-Shot learners.

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.

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. volume 9, pages 346–361.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are Zero-Shot reasoners.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-Tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks.

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, Vancouver, Canada. Association for Computational Linguistics.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show your work: Scratchpads for intermediate computation with language models.

Bhargavi Paranjape, Scott Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Marco Tulio Ribeiro. 2023. ART: Automatic multi-step reasoning and tool-use for large language models.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2022. CommonsenseQA 2.0: Exposing the limits of AI through gamification.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models.