Paper Survey

Team 16
r11922008 吳庭維
r09922046 陳奕均
t11902210 張一凡

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

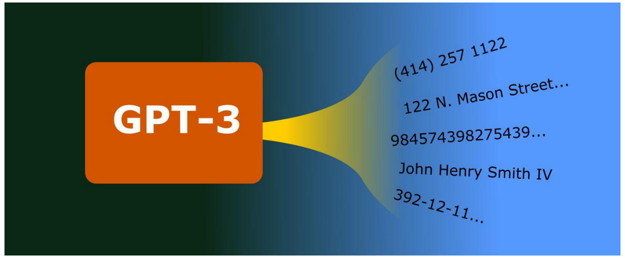
預訓練語言模型之思維鏈

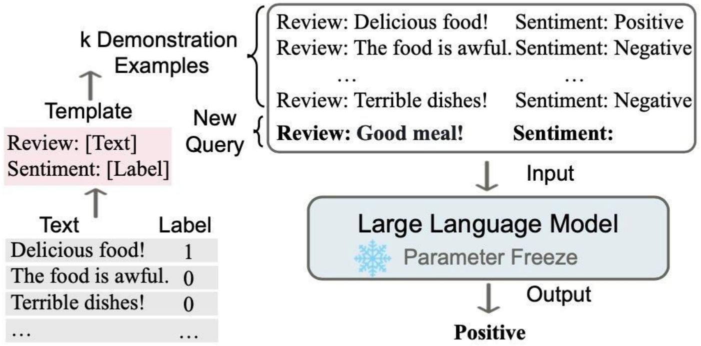
Author:
Jason Wei
Xuezhi Wang
Dale Schuurmans
Maarten Bosma
Brian Ichter
Fei Xia
Ed H. Chi
Quoc V. Le
Denny Zhou

Institution:
Google

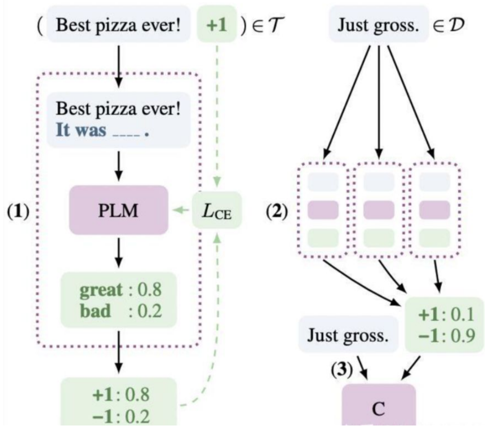
Background

1. Example
The scale of models has been increasing recently
GPT3: 175B -> a large model
Advantages: strong reasoning ability & high sampling efficiency
As the model parameters are large and can store a lot of knowledge





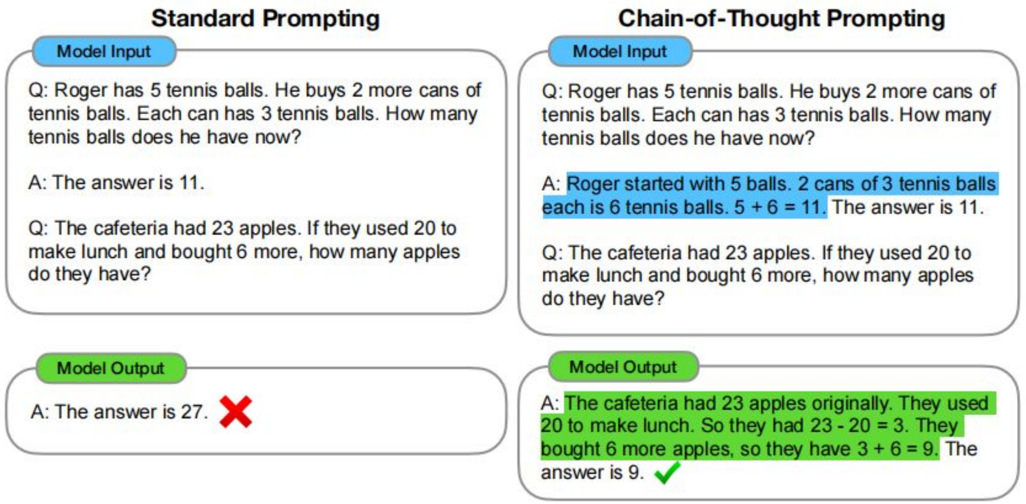
2. Development
Commonsense, and symbolic reasoning is not good enough
Good results in simplifying reasoning by constructing this intermediate process
Such as the strong in context feed shot ability of large models



3.Problem
Creating many intermediate steps for supervised fine tunes -> time-consuming
Traditional prompt methods do not perform well in mathematical calculations, common sense reasoning
How to combine in-context feed shots and intermediate steps to improve reasoning ability is a problem.

Abstract / Motivation

1. Abstract
(1)How to generate a thought chain - a series of intermediate reasoning steps - to significantly improve the ability of large language models to perform complex reasoning.
(2)Especially in sufficiently large language models, how does this reasoning ability naturally emerge through a simple method called thought chain prompts.
(3)Through experiments, it has been shown that thinking chain prompts can improve the performance of a range of arithmetic, common sense, and symbolic reasoning tasks.



Standard Prompting

Model Input

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?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

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?

Model Output

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. ✓

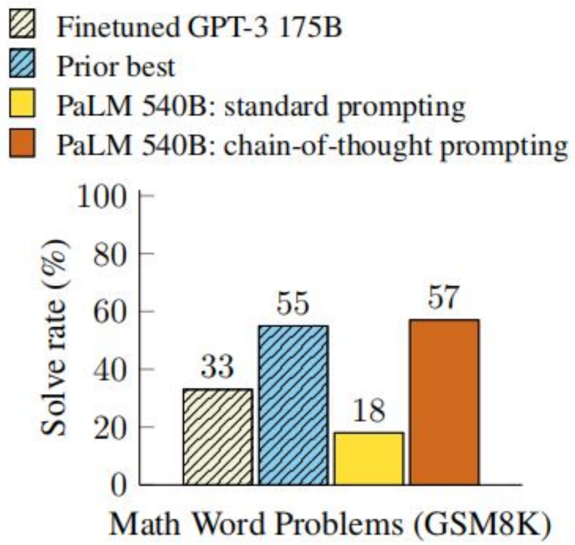
COT thinking chain example
(1) Jane has 12 flowers
(2) After Jane gives 2 flowers to her mom she has 10
(3) Then after she gives 3 to her dad she will have 7
(4) So the answer is 7

2. Purpose
(1)Improve the effectiveness of mathematical calculations and common sense reasoning through a large model by combining in-context feed shot promotion and multi-step intermediate reasoning.
(2)Compared to traditional prompt, the difference of COT lies in the addition of a logical inference process that is pushed in the middle between the answers.
(3)Format:from x ->y to x chain of thought ->y

Introduction

Expanding the scale of language models have benefits, such as improved performance and sample efficiency.

However, simply expanding the model size is not enough to achieve high performance in challenging tasks such as (arithmetic, common sense, and symbolic reasoning).



Two ideas:

(1) Arithmetic reasoning technology can benefit from generating natural language principles that lead to the final answer

(2) Large language models provide exciting prospects for small sample learning in context through prompts

Function

(1)Combines the advantages of these two ideas in a way that avoids their limitations.

(2)Explores the ability of language models to perform small sample prompts for inference tasks

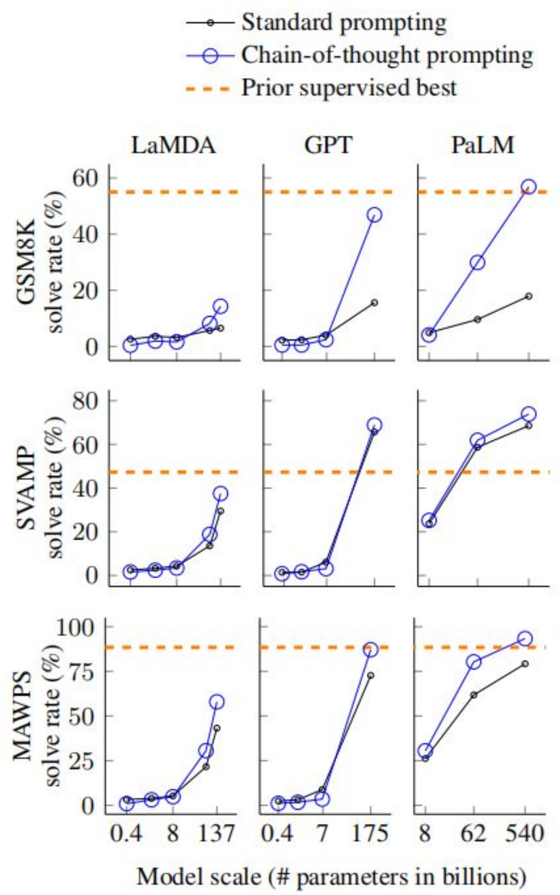
(3)Prompt format:(triples)

<input, chain of thought, output>

Arithmetic Reasoning計算推理

Language models for comparison
•GPT-3 (OpenAI) text-ada-001, text-babbage-001, text-curie-001, and text-davinci-002 350M, 1.3B, 6.7B, and 175B parameters
•LaMDA (Google) 422M, 2B, 8B, 68B, and 137B parameters.
•PaLM (Google) 8B, 62B, and 540B parameters.
•UL2 (Google) 20B parameters.
•Codex(OpenAI)code-davinci-002175B parameters.

Empirical conclusion
(1)Models with parameters below 100B (100 billion parameters) have poor performance, which indirectly reflects their insufficient instrument fine-tune, making it difficult for COT to stimulate their in context reasoning ability.
(2)The performance of models above 100B is very good, even surpassing the previous SOTA model based on supervised training.



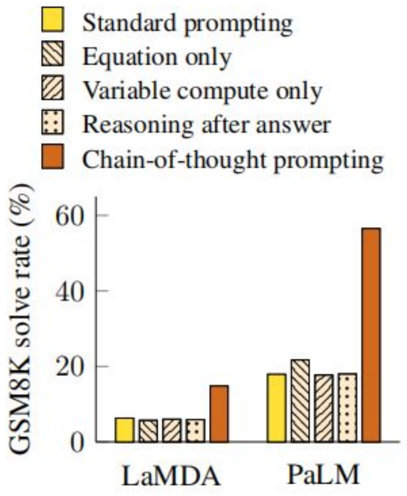
Ablation Study消融研究 + COT Robustness鲁棒性

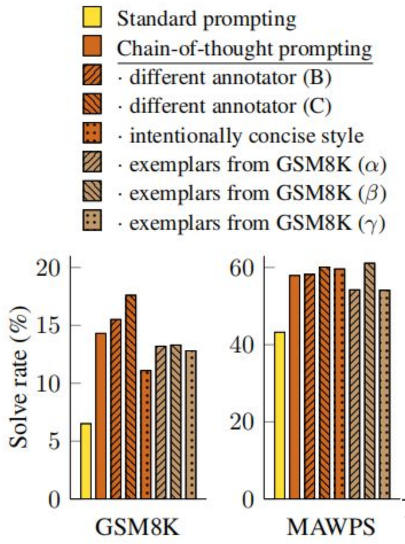
1. Ablation Study1:Equation only
(1)Factor:Equation better?
(2)Experiment: simply changed the logical reasoning process into equations
(3)Result:the effect was not better than traditional prompt, but worse than COT, indicating the importance of COT reasoning

2.Ablation Study2:Variable compute only
(1)Factor:spending more energy on complex reasoning?
(2)Experiment: adding dots with the same length as those equations to the prompt.
(3)Result:the effect was not good either, indicating that it was not as expected. Increasing the length would not improve the effect.language seems useful.

2.Ablation Study3:Chain of thought after answer
(1)Factor:order of ntermediate reasoning process important?
(2)Experiment: changing the order of the answers and reasoning process
(3)Result:entering the results first, followed by the intermediate reasoning process, will not be helpful for the effect

4. Robustness
(1)Factor:writing style ?
(2)Experiment:write a paradigm for COT (randomly selected 8 samples from GSM8K)
(3)Result:(writing style has little impact on the effectiveness)although different samples have a significant impact on the effect, they are not too significant and still significantly better than traditional prompts.
(4)Other:diifferent models also have different reactions to COT prompts, such as Palm and GPT3



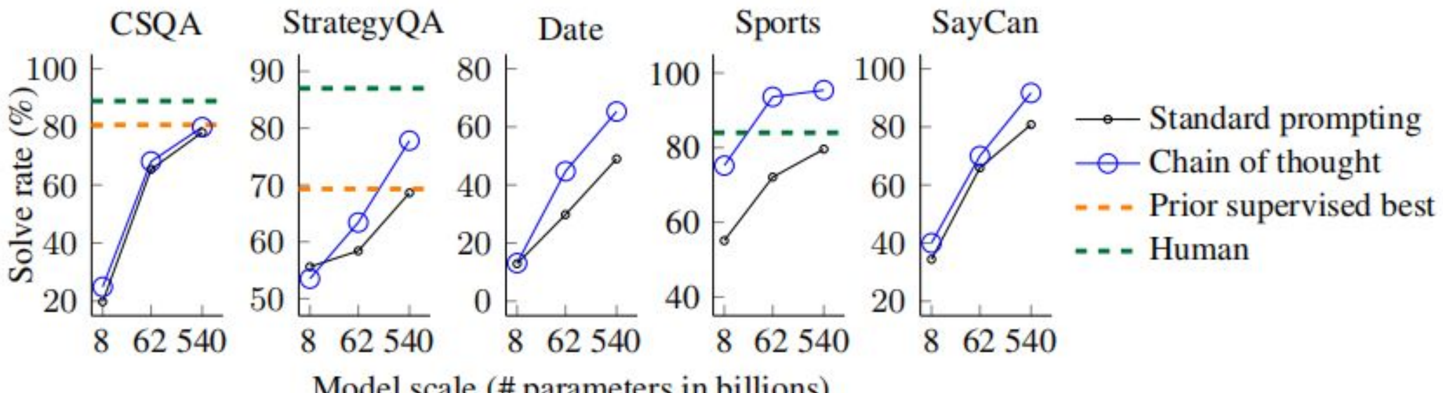


Commonsense Reasoning常識性推理

1.Introduce
Common sense reasoning is the key to interacting with the world, but it still exceeds the capabilities of current natural language comprehension systems

2.Datasets(covering different types of common sense):
the popular CSQA(not obvious)->Structure/Form
SraticyQA
Date Understanding
Sports Understanding
SayCan datasetand symbolic reasoning tasks.

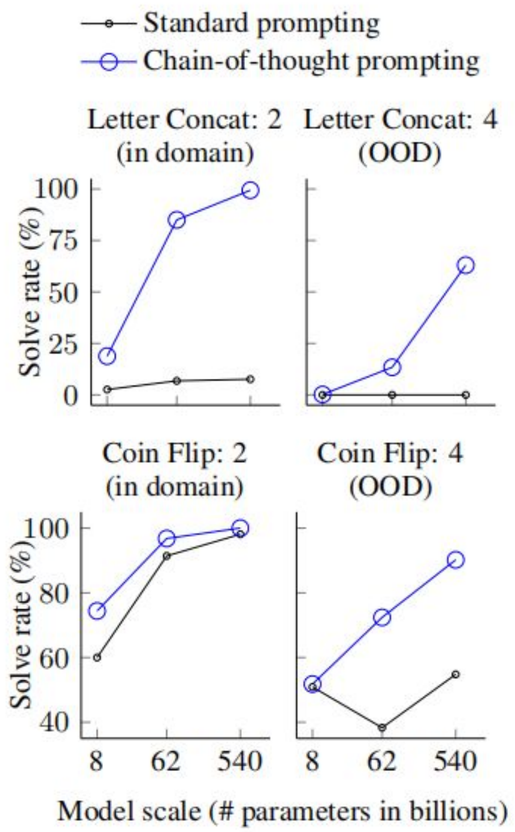
Empirical conclusion
(1)For all tasks, expanding the model size improves the performance of standard prompts;
(2)Thinking chain prompts bring further benefits
(3)Thinking chain prompts improve the performance of tasks that require a range of common sense reasoning abilities



Symbolic Reasoning符號推理

1.First task :Last letter concatenation
Involving concatenating the abbreviations of a person's name
“Amy Brown” → “yn”

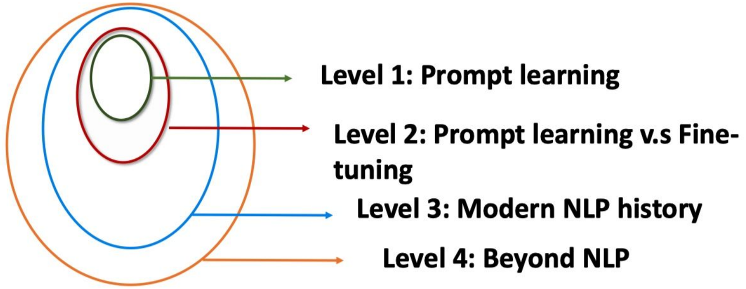
2.Second task :Coin flip
Involving tossing coins multiple times and inferring the pros and cons
A coin is heads up. Phoebe flips the coin. Osvaldo does not flip the coin. Is the coin still heads up?



Empirical conclusion
(1)COT can easily defeat the traditional Prompt form
(2)Thinking chain method is superior to the standard prompt method

Conclusion / Discussion

1.Advantage
The performance of arithmetic inference tasks and has strong robustness.
CoT's nature based on natural language makes it universally applicable to many reasoning tasks.
CoT helps to generalize the model's inference ability to a longer inference sequence length.


Level 1: Prompt learning
Level 2: Prompt learning v.s Fine-tuning
Level 3: Modern NLP history
Level 4: Beyond NLP

2.Limitations
Although manual construction of CoT does not incur much cost for feed shot in context learning, the cost of fine tuning is unacceptable.

3.CoT inference
Only applicable in large-scale models
NOT widely used in practical applications: the deployment of large models is very difficult.

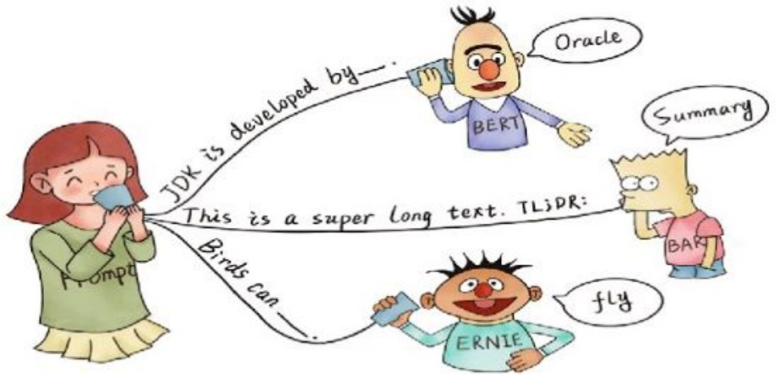
Further research has achieved good inference ability by using CoT data fine tune small models.

Questions / Connections

1.Premise
The experiment in this paper only relies on feed shot promotion without fine-tuning.

COT can stimulate the reasoning ability of models above 100B.

But there are also several issues left



2.Question
(1)If we continue to expand the model, will the effect still improve?
(2) Are there any other better prompt methods
(3)How to demonstrate that the model is indeed reasoning
(4)Is there a better way than manually writing a prompt
(5)How to ensure the correctness of the inference path
(6)How to achieve similar effects on small models

3.Expand and connect
Based on these issues, we have selected the next paper that can more profoundly enhance the prompt effect !

Successive Prompting for Decomposing Complex Questions

(ACL 2022)

Dheeru Dua, Shivanshu Gupta, Sameer Singh, Matt Gardner

Table of Contents

Main Contributions

1. Introduce “Successive Prompting”, iteratively **breaking down** a complex task into simpler tasks, solve them, and repeat until we get the final solution.

2. **Decouple** the supervision for decomposing complex questions from the supervision for answering simple questions.

3. Introduce a way to generate a **synthetic** dataset, used to **bootstrap** a model’s ability to decompose and answer intermediate questions.

How Does It Works ?

1. Iteratively **alternate** between 2 kinds of sessions :
    - QD (Question Decomposition)
        - Index I_D contains partially decomposed chains at each step k, demonstrating the next question q_k to be produced for every complex question in training data.
        - Query I_D with the **complex test question q** and **current step number k**, to select demonstrations regarding how to generate the next question for the held-out example
    - QA (Question Answering)
        - Index I_A contains all the simple QA pairs in the training data from all the complex questions.
        - Query I_A with the **simple question q_k generated during QD** to select relevant simple QA pairs.

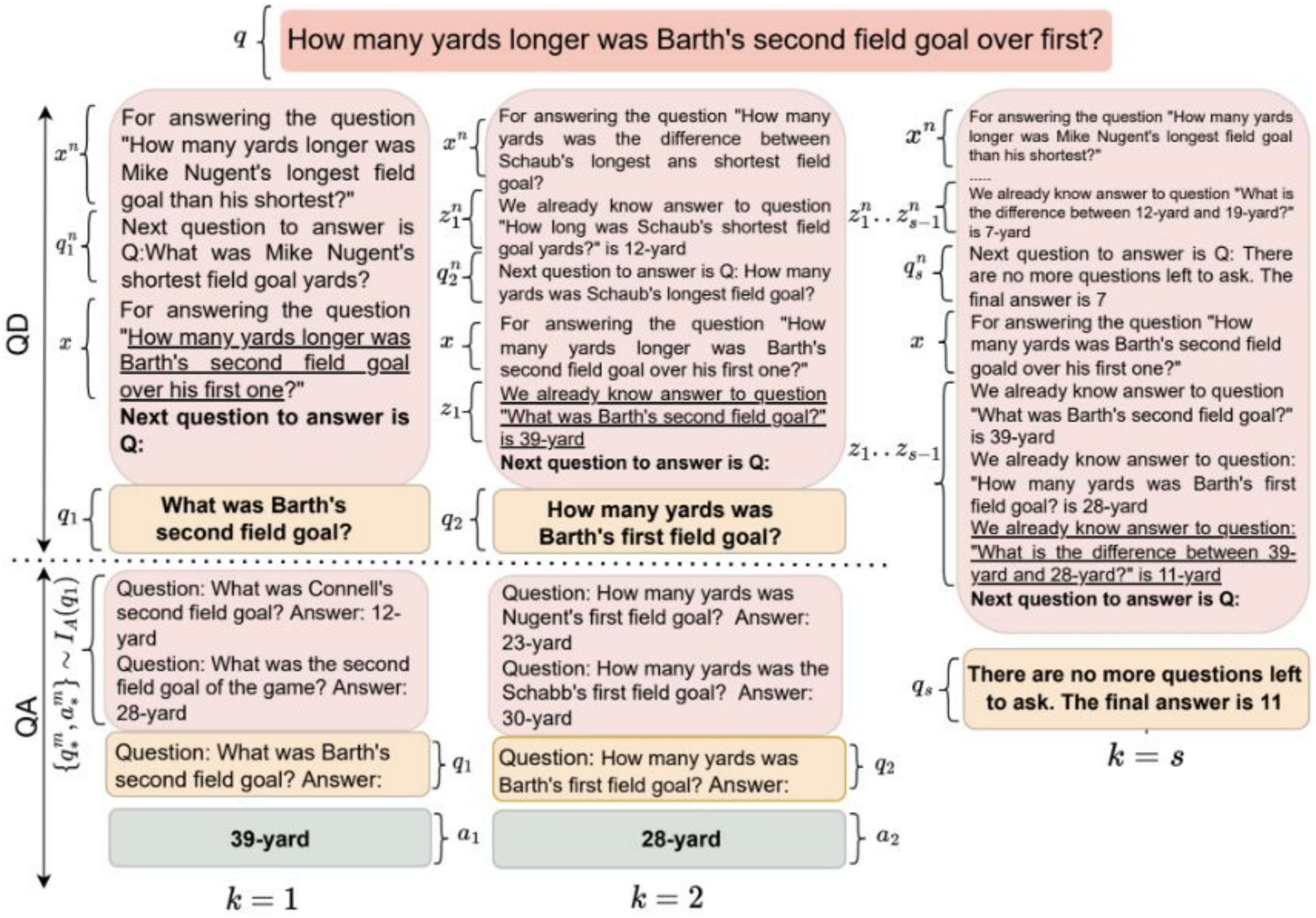
How Does It Works ?

2. So, the process is QD ⇒ QA ⇒ … ⇒ QD (outputs the **special phrase** “There are no more questions left to ask”, along with a final answer).

3. Also, the QD may stop because of a hardcoded **step limit**.

4. Utilize specialized **sub-modules** for solving different QA tasks :
   - Construct a simple mathematical sub-module for QA which **parses** the generated simple question for **symbolic operation** type, then executes in a deterministic way

   - If the generated simple question cannot be parsed as a mathematical operation, apply the language model to solve it.



q
How many yards longer was Barth's second field goal over first?
QD
x^n
For answering the question "How many yards longer was Mike Nugent's longest field goal than his shortest?"
q_1^n
Next question to answer is Q:What was Mike Nugent's shortest field goal yards?
x
For answering the question "How many yards longer was Barth's second field goal over his first one?"
Next question to answer is Q:
q_1
What was Barth's second field goal?
x^n
For answering the question "How many yards was the difference between Schaub's longest ans shortest field goal?
z_1^n
We already know answer to question "How long was Schaub's shortest field goal yards?" is 12-yard
q_2^n
Next question to answer is Q: How many yards was Schaub's longest field goal?
x
For answering the question "How many yards longer was Barth's second field goal over his first one?"
z_1
We already know answer to question "What was Barth's second field goal?" is 39-yard
Next question to answer is Q:
q_2
How many yards was Barth's first field goal?
x^n
For answering the question "How many yards longer was Mike Nugent's longest field goal than his shortest?"
.....
$z_1^n..z_{s-1}^n$
We already know answer to question "What is the difference between 12-yard and 19-yard?" is 7-yard
q_s^n
Next question to answer is Q: There are no more questions left to ask. The final answer is 7
x
For answering the question "How many yards was Barth's second field goald over his first one?"
$z_1..z_{s-1}$
We already know answer to question "What was Barth's second field goal?" is 39-yard
We already know answer to question: "How many yards was Barth's first field goal? is 28-yard
We already know answer to question: "What is the difference between 39-yard and 28-yard?" is 11-yard
Next question to answer is Q:
q_s
There are no more questions left to ask. The final answer is 11
$k = s$
QA
$\{q_*^m, a_*^m\} \sim I_A(q_1)$
Question: What was Connell's second field goal? Answer: 12-yard
Question: What was the second field goal of the game? Answer: 28-yard
Question: What was Barth's second field goal? Answer:
q_1
39-yard
a_1
$k = 1$
Question: How many yards was Nugent's first field goal? Answer: 23-yard
Question: How many yards was the Schabb's first field goal? Answer: 30-yard
Question: How many yards was Barth's first field goal? Answer:
q_2
28-yard
a_2
$k = 2$

Comparison between Chain-of-Thought

1. The general form of CoT :

$$y, z \leftarrow \mathbb{L}\left(x, \{\, (x^n, y^n, \mathbf{z}^n) \mid n \in [1, N] \,\}\right).$$

- Given an input x, a language model L and N in-context examples obtained from querying an index I, each containing a triplet of passage with complex question ***x***, latent steps ***z*** and final answer ***y***. That is, CoT **prompts L only once** !!!

- Also, the training data of CoT is a small handful of **manually-written** example decomposition, which is expensive and requires expert knowledge, and lead to poor performance on a dataset as varied as DROP.

Comparison between Chain-of-Thought

2. The general form of Successive Prompting :

$$q_1 \leftarrow \mathbb{L}\left(x, \left\{(x^n, q_1^n) \mid n \in [1, N]\right\}\right)$$
$$a_1 \leftarrow \mathbb{L}\left(p, q_1, \{(p_*^m, q_*^m, a_*^m) \mid m \in [1, M]\}\right)$$
$$q_2 \leftarrow \mathbb{L}\left(x, q_1, a_1, \left\{(x^n, q_1^n, a_1^n, q_2^n) \mid n \in [1, N]\right\}\right)$$
$$a_2 \leftarrow \mathbb{L}\left(p, q_2, \{(p_*^m, q_*^m, a_*^m) \mid m \in [1, M]\}\right)$$
$$\ldots$$
$$y \leftarrow \mathbb{L}\left(x, \mathbf{z}, \{(x^n, y^n, \mathbf{z}^n) \mid n \in [1, N]\}\right)$$

- Separates z into multiple question and answering **sessions**, enabling us to prompt L many times, with potentially different in-context examples that are more **tailored** to the simple question at each step.

- It **bootstraps** model learning with out-of-domain, **synthetically** generated data.

- Can be used with **fine-tuning**, where each intermediate output is treated as a training example.

How to Generate the Decomposition Dataset ?

1. Use **semi-structured** data from tables in English Wikipedia.

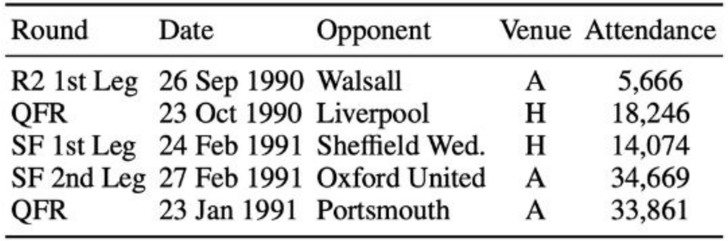
2. Convert the rows in the tables into paragraphs, and use single column headers to create first order simple questions and a combination of columns for higher order complex questions.

3. There are 10 simple **operations** :
   COUNT, TOP(k), BOTTOM(k), FILTER, SUM, COMPARISON, DIFFERENCE, NEGATION, GATHER, and INTERSECTION

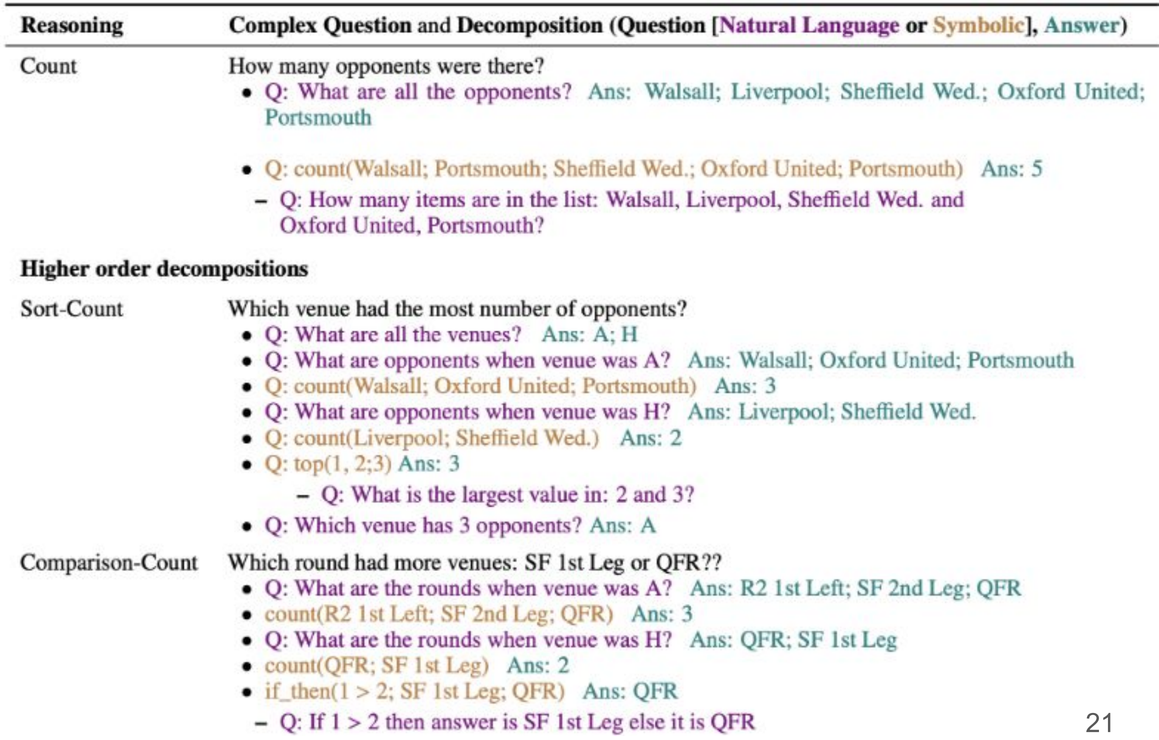
4. Depending on the model (language model or specialized sub-module), use either **symbolic or natural language** version of the arithmetic operations.

How to Generate the Decomposition Dataset ?

5. For example, we can use “COUNT” operation to convert the table below



Round	Date	Opponent	Venue	Attendance
R2 1st Leg	26 Sep 1990	Walsall	A	5,666
QFR	23 Oct 1990	Liverpool	H	18,246
SF 1st Leg	24 Feb 1991	Sheffield Wed.	H	14,074
SF 2nd Leg	27 Feb 1991	Oxford United	A	34,669
QFR	23 Jan 1991	Portsmouth	A	33,861



Reasoning	**Complex Question** and **Decomposition** (**Question** [**Natural Language** or **Symbolic**], **Answer**)
Count	How many opponents were there? • Q: What are all the opponents? Ans: Walsall; Liverpool; Sheffield Wed.; Oxford United; Portsmouth • Q: count(Walsall; Portsmouth; Sheffield Wed.; Oxford United; Portsmouth) Ans: 5 – Q: How many items are in the list: Walsall, Liverpool, Sheffield Wed. and Oxford United, Portsmouth?
Higher order decompositions	
Sort-Count	Which venue had the most number of opponents? • Q: What are all the venues? Ans: A; H • Q: What are opponents when venue was A? Ans: Walsall; Oxford United; Portsmouth • Q: count(Walsall; Oxford United; Portsmouth) Ans: 3 • Q: What are opponents when venue was H? Ans: Liverpool; Sheffield Wed. • Q: count(Liverpool; Sheffield Wed.) Ans: 2 • Q: top(1, 2;3) Ans: 3 – Q: What is the largest value in: 2 and 3? • Q: Which venue has 3 opponents? Ans: A
Comparison-Count	Which round had more venues: SF 1st Leg or QFR?? • Q: What are the rounds when venue was A? Ans: R2 1st Left; SF 2nd Leg; QFR • count(R2 1st Left; SF 2nd Leg; QFR) Ans: 3 • Q: What are the rounds when venue was H? Ans: QFR; SF 1st Leg • count(QFR; SF 1st Leg) Ans: 2 • if_then(1 > 2; SF 1st Leg; QFR) Ans: QFR – Q: If 1 > 2 then answer is SF 1st Leg else it is QFR

Experiment Details

1. DROP dataset is **not uniformly** distributed, so the author turns them into sentence embeddings, and use **cosine similarity** to get the top-50 nearest neighbor questions for each training example.

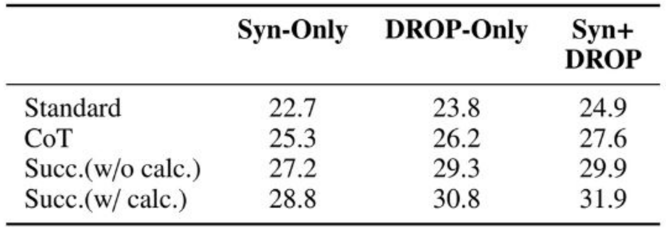
2. By the vertex cover algorithm, 300 complex questions that cover the majority of the dataset were selected as the training set and **manually annotated** with decomposed QA pairs in the same format as the synthetic data.

3. Use the **FAISS** package developed by Meta to query embeddings in QD & QA.

4. In In-context Learning, the model is **GPT-J (6B)**, which is the largest freely available model we could use; In fine-tuning, the model is **T5-large** version of UnifiedQA trained in a multi-task manner.

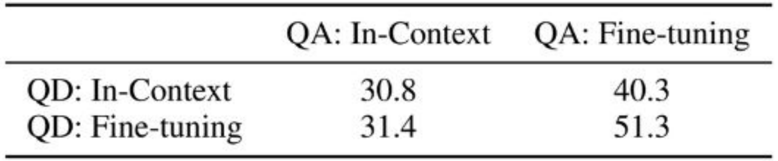
Experiment Details

5. (In-context Learning) The table below shows that :
    - In each dataset setting, Successive Prompting outperforms CoT.
    - Modular approach is better over a single model that tries to solve all the tasks.



	Syn-Only	DROP-Only	Syn+ DROP
Standard	22.7	23.8	24.9
CoT	25.3	26.2	27.6
Succ.(w/o calc.)	27.2	29.3	29.9
Succ.(w/ calc.)	28.8	30.8	31.9

6. In-context v.s. Fine-tuning :
    - In-context learning is unable to do well on answering simple questions that result in a list of answers—which is especially important for DROP as symbolic aggregations are generally applied on a list of answers
    - A better QD model (fine-tuned) **does not help** the overall performance much unless the QA model can handle the produced decompositions. (30.8 → 31.4)



	QA: In-Context	QA: Fine-tuning
QD: In-Context	30.8	40.3
QD: Fine-tuning	31.4	51.3

Thank you for listening