Carnegie Mellon University
Language Technologies Institute

61 ACL 2023

Complex Few-shot Reasoning with LLMs

Aman Madaan
https://madaan.github.io/

Part 5/7 of the ACL 2023 Tutorial
Complex Reasoning in Natural Language

Questions @ Rocket Chat

Complex Few-shot Reasoning with LLMs

Goal Today:

- Try to cover the details of all possible exciting recent works ✗
- Provide a high-level summary of a *class* of few-shot prompting techniques ✓

I will definitely miss interesting works:

- Please reach out! We will add it to our website

Complex Few-shot Reasoning with LLMs: Key Techniques

Reasoning Elaboration

Instructions

Tool Augmentation

Structured Generation

Feedback

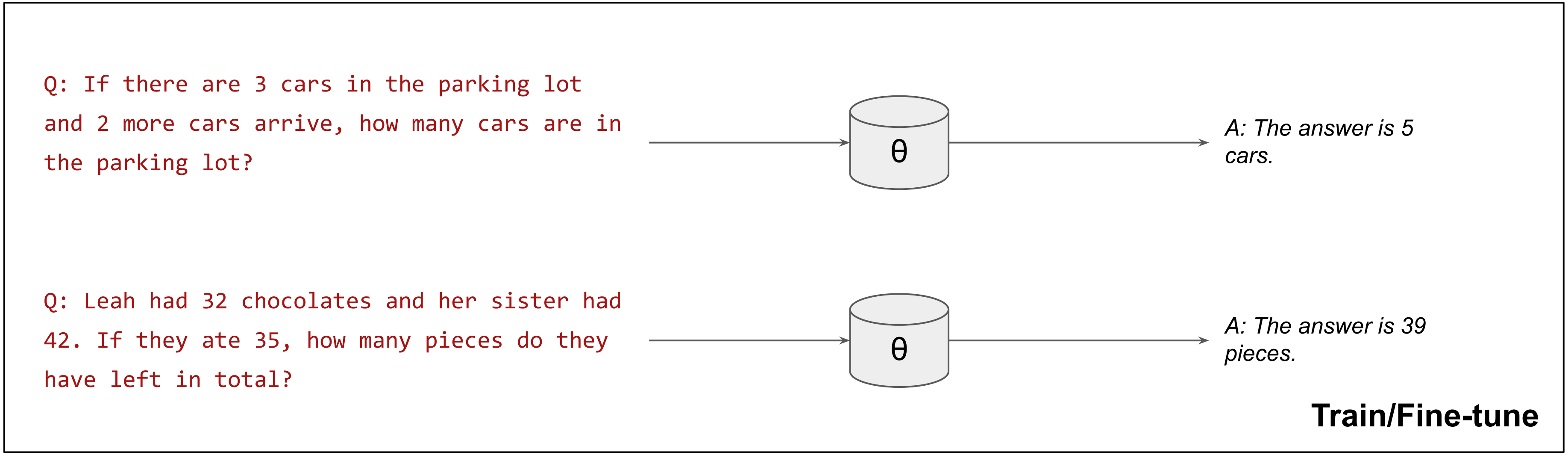
Memory

Quick Detour: Few-shot Prompting

Q: Shawn has 5 toys. For Christmas, he got 2 toys each from his mom and dad. How many toys does he have now?

A: The answer is 9 toys

Fine-tuning



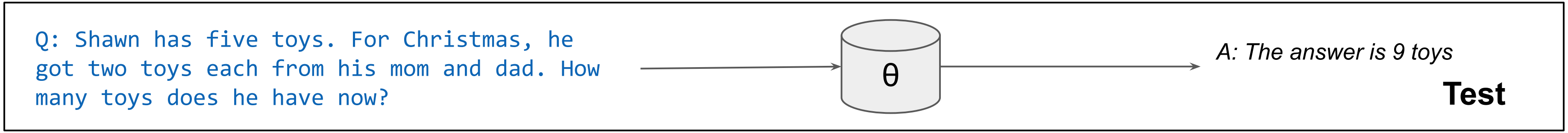
`Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?`

A: The answer is 5 cars.

`Q: Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total?`

A: The answer is 39 pieces.

Train/Fine-tune



`Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now?`

A: The answer is 9 toys

Test

Few-shot prompting

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

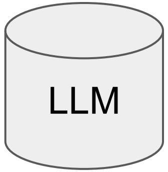
A: *The answer is 5 cars.*

Prompt

Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now?

A:

Test Example



The answer is 9 toys

Complex Few-shot Reasoning with LLMs: Key Techniques

Reasoning Elaboration

Instructions

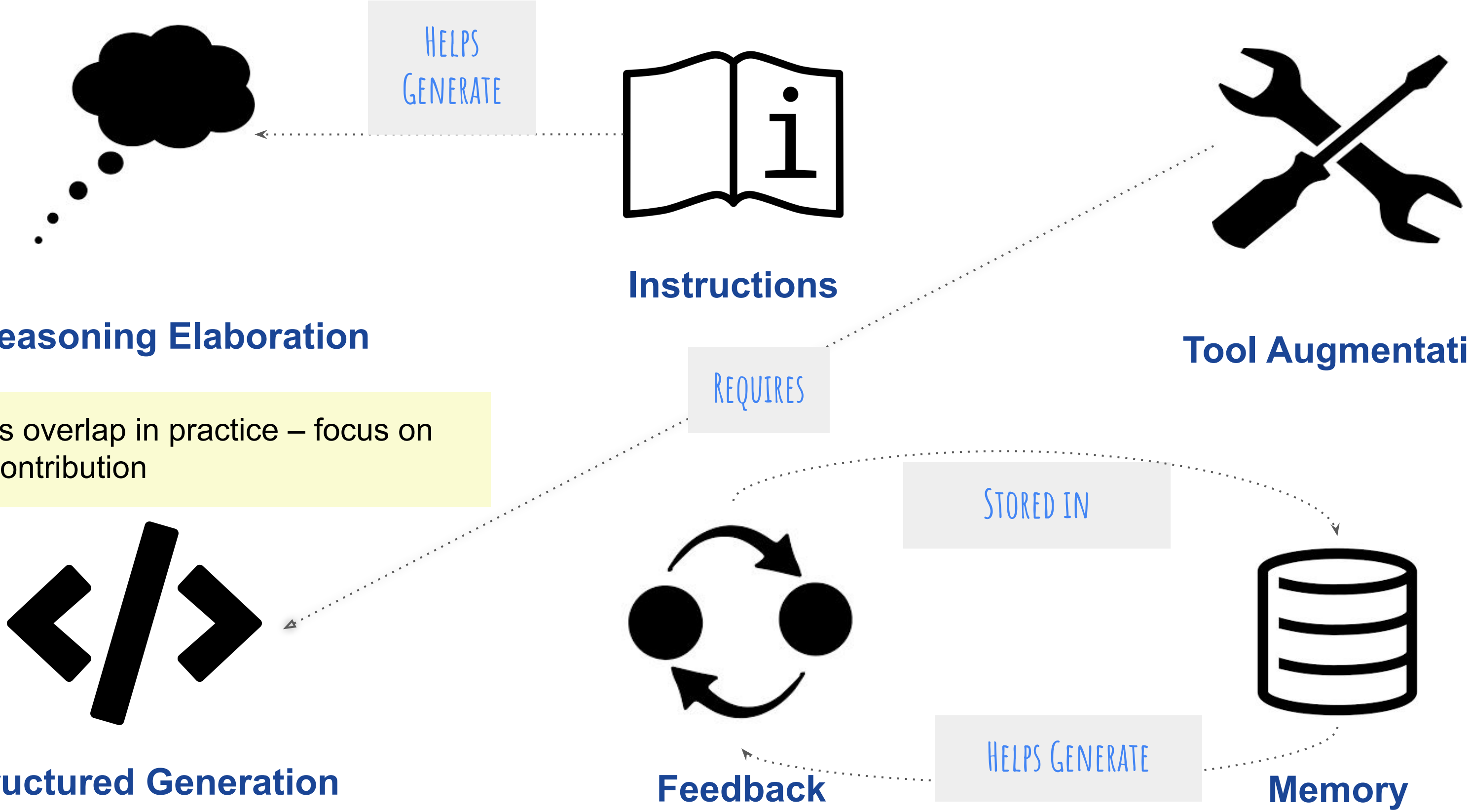
Tool Augmentation

Structured Generation

Feedback

Memory

Complex Few-shot Reasoning with LLMs: Key Techniques



Reasoning Elaboration

HELPS GENERATE

Instructions

Tool Augmentation

REQUIRES

Structured Generation

STORED IN

Feedback

HELPS GENERATE

Memory

Techniques overlap in practice – focus on the main contribution

Rest of the slides

- General idea of the technique
- Representative Work
- Goal/Hope:
  - A *checklist* of techniques for complex reasoning with LLMs

Progress Bar

Complex Few-shot Reasoning with LLMs: Key Techniques

Reasoning Elaboration

Instructions

Tool Augmentation

Structured Generation

Feedback

Memory

Few-shot prompting

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

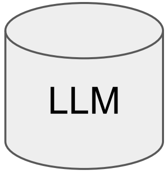
A: *The answer is 5 cars.*

Prompt

Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now?

A:

Test Example



The answer is 9 toys

Few-shot prompting

Direct Prompt

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

***A:** The answer is 5 cars.*

Chain-of-thought Prompting (Wei et al. 2022)

Direct Prompt

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

A: *The answer is 5 cars.*

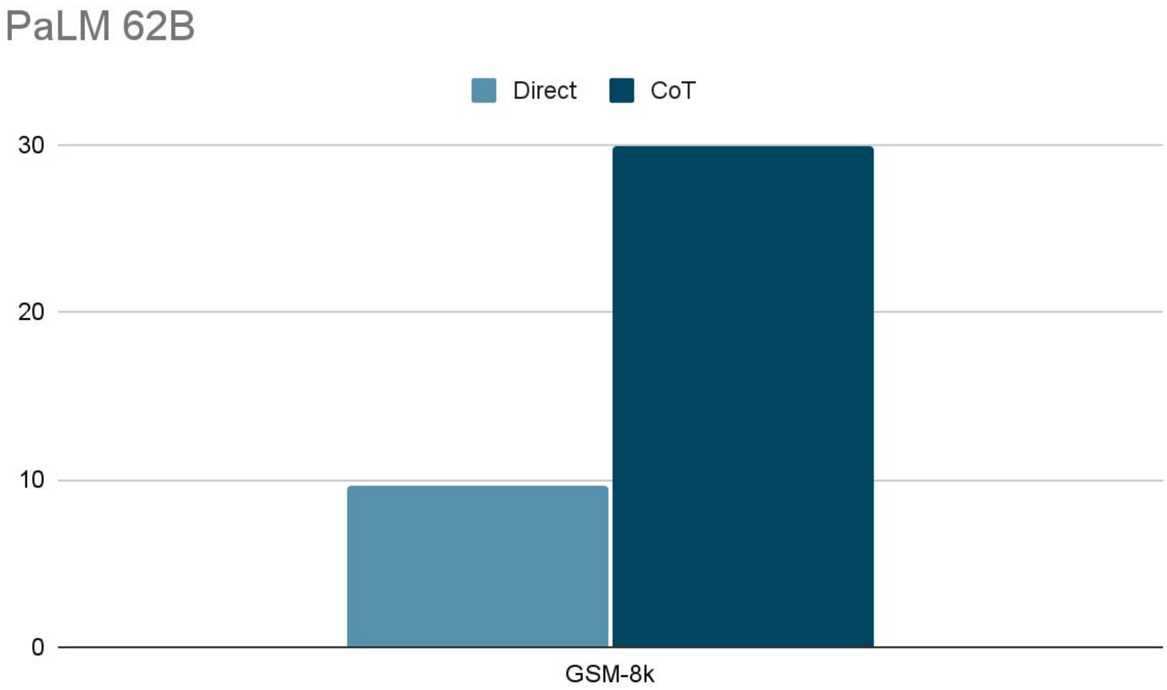
Chain-of-Thought Prompt

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

Thought (T): There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5.

A: *The answer is 5 cars.*

Chain-of-thought prompting is extremely effective



Chain-of-thought Prompting (CoT)

- General idea:
  - Standard prompt:
    - Q → A
  - Chain-of-thought prompt:
    - Q → Reasoning Process, A
- Many existing prompting techniques can be seen as an improvement over the general CoT strategy
- Similar ideas:
  - Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems (Ling et al. 2017)
  - Think about it! Improving defeasible reasoning by first modeling the question scenario (Madaan et al. 2021).
  - Show your work: Scratchpads for Intermediate Computation with Language Models (Nye et al. 2021)

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

Thought (T): There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5.

***A:** The answer is 5 cars.*

Least-to-Most Prompting (Zhou et al. 2022)

- Breakdown the reasoning process into steps
  - Decompose the problem into simpler sub-problems
  - Solve simpler sub-problems

Least-to-Most Prompting

Question

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

Question Decomposition

Q: How many cars are in the parking lot after the first car arrives?

Q: How many cars are in the parking lot after the second car arrives?

Subproblem Solving

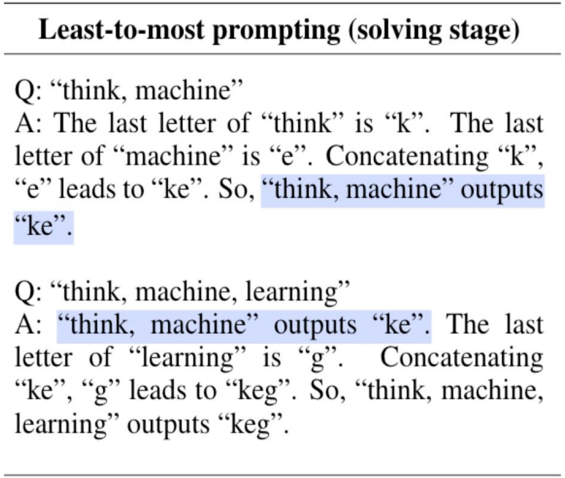
Q: How many cars are in the parking lot after the first car arrives?

A: There are originally 3 cars. After the first car arrives, we have 3 + 1 = 4 cars.

Q: How many cars are in the parking lot after the second car arrives?

A: After the first car arrives, we have 3 + 1 = 4 cars. After the second car arrives, we have 4 + 1 = 5 cars.

Least-to-Most Prompting

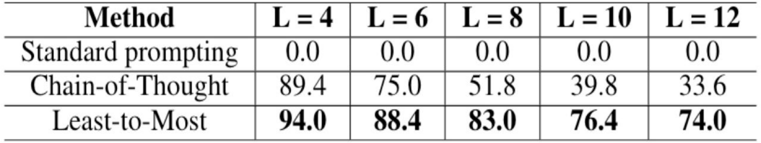


Least-to-most prompting (solving stage)

Q: “think, machine”
A: The last letter of “think” is “k”. The last letter of “machine” is “e”. Concatenating “k”, “e” leads to “ke”. So, “think, machine” outputs “ke”.

Q: “think, machine, learning”
A: “think, machine” outputs “ke”. The last letter of “learning” is “g”. Concatenating “ke”, “g” leads to “keg”. So, “think, machine, learning” outputs “keg”.

Table 2: A test case of least-to-most prompting for the last-letter-concatenation task. Generated with `code-davinci-002` in GPT-3. The prompt context is shown on the right column of Table 1.



Method	L = 4	L = 6	L = 8	L = 10	L = 12
Standard prompting	0.0	0.0	0.0	0.0	0.0
Chain-of-Thought	89.4	75.0	51.8	39.8	33.6
Least-to-Most	**94.0**	**88.4**	**83.0**	**76.4**	**74.0**

When do you need more steps? Complexity-Based Prompting for Multi-Step Reasoning, Fu et al. 2023

Decomposed Prompting: A Modular Approach for Solving Complex Tasks

Complex Few-shot Reasoning with LLMs: Key Techniques

✓ **Reasoning Elaboration: Spell out the reasoning process before generating the answer**

Instructions

Tool Augmentation

Structured Generation

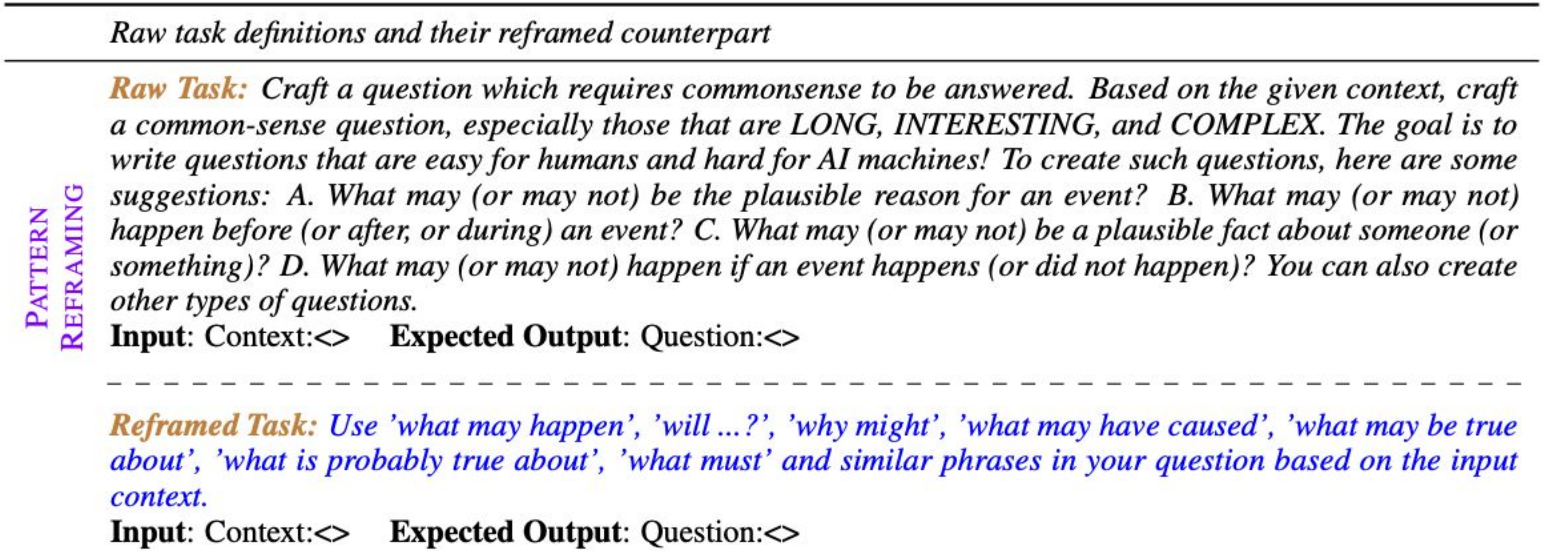
Feedback

Memory

Instructions

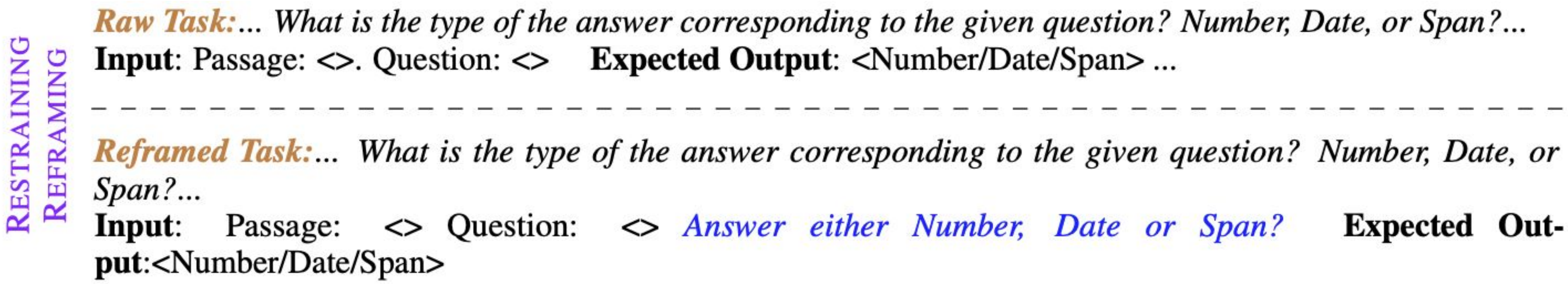
- Many ways to communicate the same intent:
  - Write a summary of the history of computer science
  - Summarize the history of computer science, highlighting key developments.
  - Provide a summary of the history of computer science in this format:
    - Inventors and innovations
    - Evolution of programming languages
    - Impact of internet
  - Summarize the history of computer science in no more than 200 words, keeping the language simple and easy to understand
- Different levels of abstract, details, requirements

Reframing Instructional Prompts to GPTk's Language (Mishra et al., 2022)



	Raw task definitions and their reframed counterpart
PATTERN REFRAMING	**Raw Task:** *Craft a question which requires commonsense to be answered. Based on the given context, craft a common-sense question, especially those that are LONG, INTERESTING, and COMPLEX. The goal is to write questions that are easy for humans and hard for AI machines! To create such questions, here are some suggestions: A. What may (or may not) be the plausible reason for an event? B. What may (or may not) happen before (or after, or during) an event? C. What may (or may not) be a plausible fact about someone (or something)? D. What may (or may not) happen if an event happens (or did not happen)? You can also create other types of questions.* **Input**: Context:<> **Expected Output**: Question:<>
	Reframed Task: *Use 'what may happen', 'will ...?', 'why might', 'what may have caused', 'what may be true about', 'what is probably true about', 'what must' and similar phrases in your question based on the input context.* **Input**: Context:<> **Expected Output**: Question:<>

Instructions


RESTRAINING
REFRAMING

Raw Task:... What is the type of the answer corresponding to the given question? Number, Date, or Span?...
Input: Passage: <>. Question: <> **Expected Output**: <Number/Date/Span> ...

Reframed Task:... What is the type of the answer corresponding to the given question? Number, Date, or Span?...
Input: Passage: <> Question: <> *Answer either Number, Date or Span?* **Expected Output**:<Number/Date/Span>

Mishra, Swaroop, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. "Reframing Instructional Prompts to GPTk's Language." *ACL 2022 Findings*.

Instructions

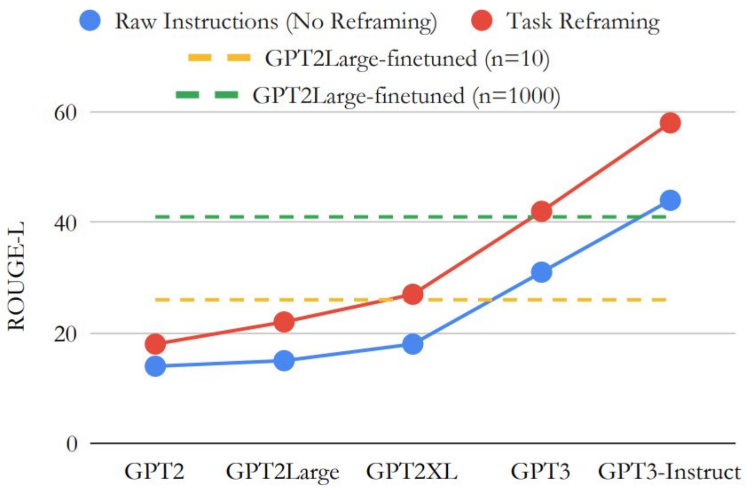


Figure 2: Across a variety of model sizes, reframed prompts consistently show considerable performance gain over raw task instructions (no reframing) in a few-shot learning setup. Since fine-tuning GPT3 is

- Explicit is better than implicit
- Be aware of instances where you might be expecting the model to read your mind!

Mishra, Swaroop, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. "Reframing Instructional Prompts to GPTk's Language." *ACL 2022 Findings*.

Complex Few-shot Reasoning with LLMs: Key Techniques

✓ **Reasoning Elaboration: Spell out the reasoning process before generating the answer**

✓ **Instructions: Provide explicit instructions to the LLM, capture all expectations**

Tool Augmentation

Structured Generation

Feedback

Memory

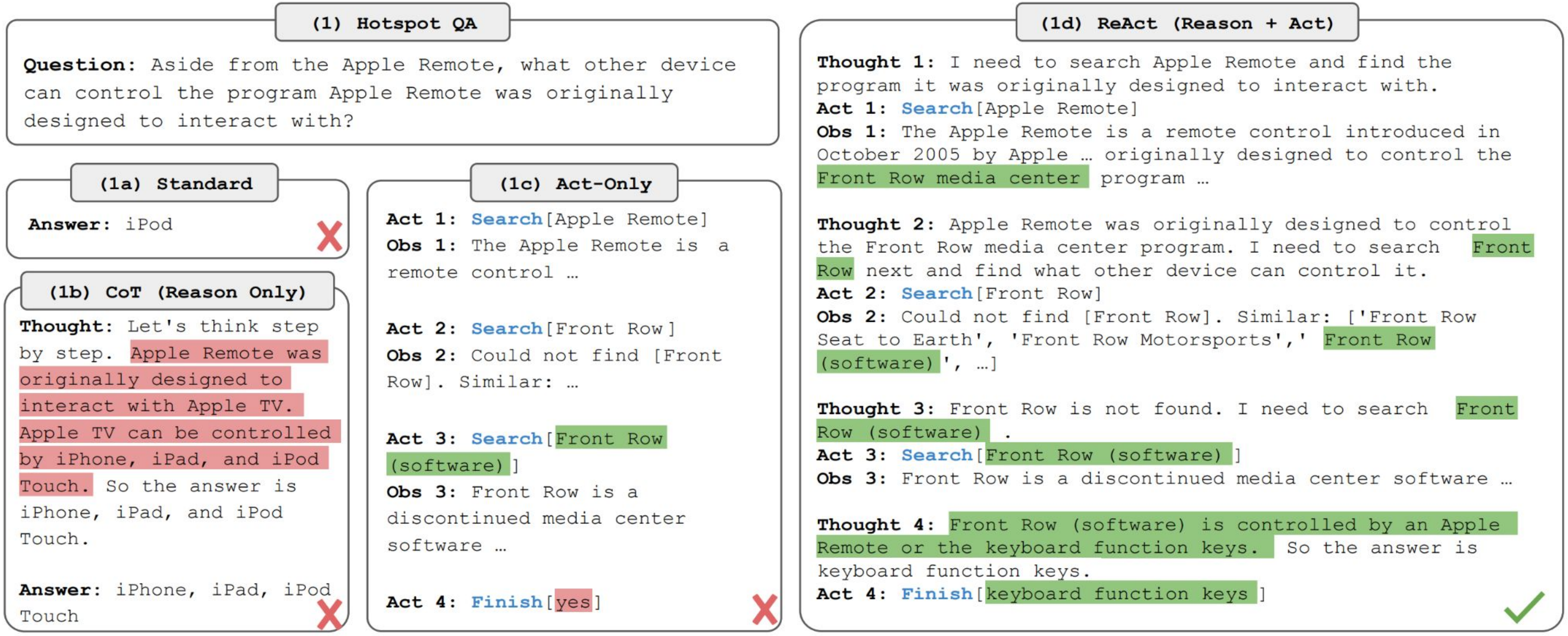
Augmenting LLMs with Tools

- Typical language generation setup: LLMs handle *everything*
    - `There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5.`
        - *What has to be done?*
            - Add `3 + 2`
        - *How to do it?*
            - Perform the calculation and produce the answer `= 5`

- Leverage external tools to separate 'what' needs to be done from 'how' to do it.

- Calculator as a tool (T):
    - `There are originally 3 cars.`
    - `2 more cars arrive.`
    - `Number of cars after the first car comes = calculator(3, 1) = 4.`
    - `Number of cars after the second car comes = calculator(4, 1) = 5.`
    - `The answer is 5.`

React (Yao et al. 2022)



(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1a) Standard

Answer: iPod ✗

(1b) CoT (Reason Only)

Thought: Let's think step by step. Apple Remote was originally designed to interact with Apple TV. Apple TV can be controlled by iPhone, iPad, and iPod Touch. So the answer is iPhone, iPad, and iPod Touch.

Answer: iPhone, iPad, iPod Touch ✗

(1c) Act-Only

Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control …

Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: …

Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Act 4: Finish[yes] ✗

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports',' Front Row (software) ', …]

Thought 3: Front Row is not found. I need to search Front Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.
Act 4: Finish[keyboard function keys] ✓

React (Yao et al. 2022)

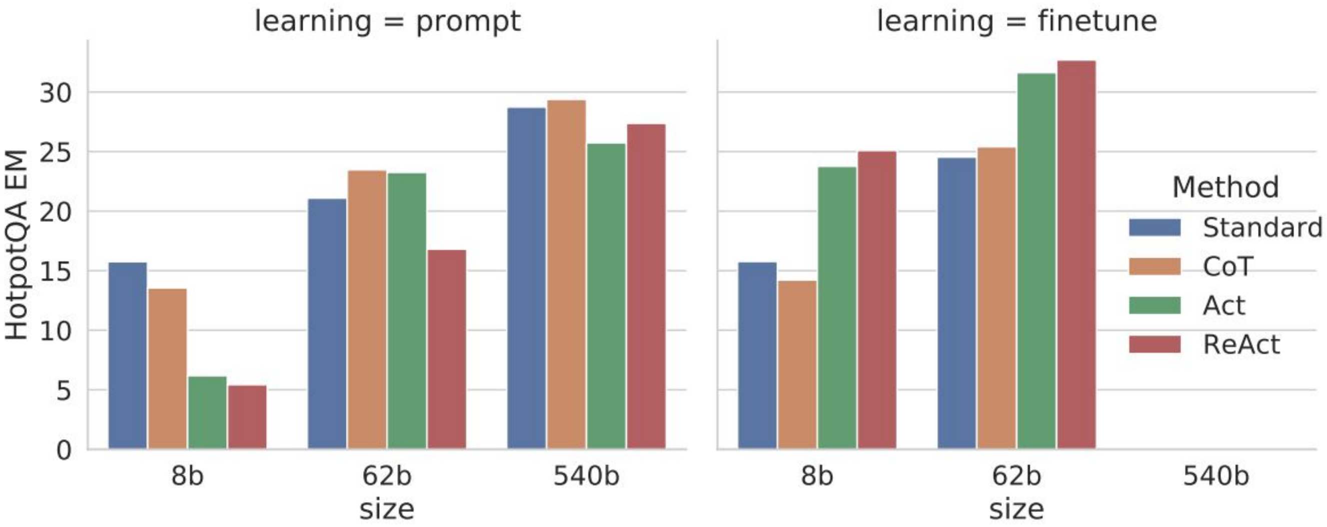


Figure 3: Scaling results for prompting and finetuning on HotPotQA with `ReAct` (ours) and baselines.

Augmenting LLMs with Tools

- Obvious use cases
  - **Real-time information:** Stock market updates, Temperature monitoring, Traffic reports.
  - **Specialization:** Solving Ordinary Differential Equations (ODEs)
  - **Multimodal**: Performing clicks, Generating image captions.

- Future Directions
  - Combine multiple tools within the same language model framework for increased functionality and adaptability.
  - [2304.09842] Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models
  - [2304.08354] Tool Learning with Foundation Models
  - [2302.04761] Toolformer: Language Models Can Teach Themselves to Use Tools
  - [2303.09014] ART: Automatic multi-step reasoning and tool-use for large language models

Complex Few-shot Reasoning with LLMs: Key Techniques

✓ **Reasoning Elaboration: Spell out the reasoning process before generating the answer**

✓ **Instructions Provide explicit instructions to the LLM, capture all expectations**

✓ **Tool Augmentation: Enhance LLMs for specialized tasks**

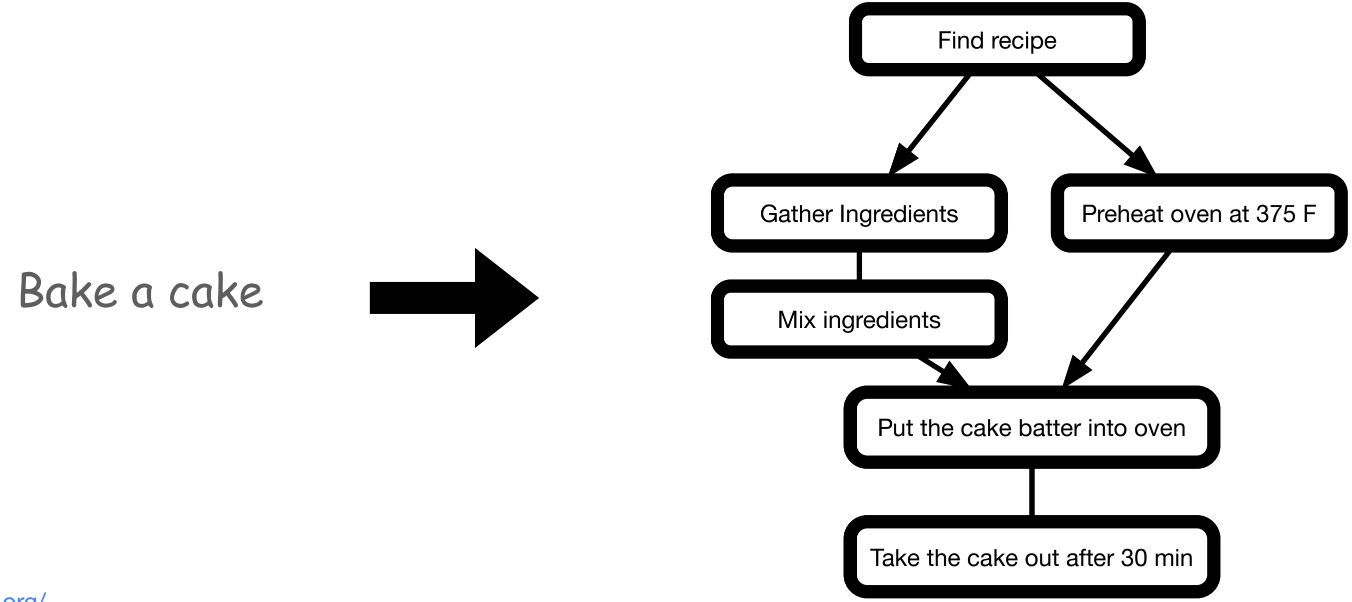
Structured Generation

Feedback

Memory

Structured Commonsense Reasoning

- Natural language input (e.g., scenario)
- Structured output (e.g., plan graph, reasoning graph)



Bake a cake

Find recipe

Gather Ingredients

Preheat oven at 375 F

Mix ingredients

Put the cake batter into oven

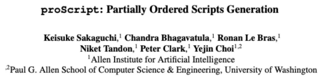
Take the cake out after 30 min

https://proscript.allenai.org/

Leveraging Language Models for Structured commonsense Reasoning

Neural Language Modeling for Contextualized Temporal Graph Generation

Aman Madaan, Yiming Yang


`proScript`: Partially Ordered Scripts Generation

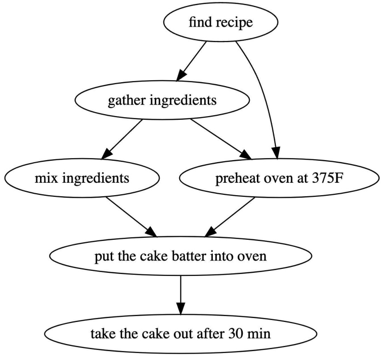
Keisuke Sakaguchi,[1] Chandra Bhagavatula,[1] Ronan Le Bras,[1]
Niket Tandon,[1] Peter Clark,[1] Yejin Choi[1,2]
[1]Allen Institute for Artificial Intelligence
[2]Paul G. Allen School of Computer Science & Engineering, University of Washington

- Need to generate a graph but … language models can only generate strings
- Workaround
  - *Flatten* the graph as a string
  - Train a seq2seq model

Goal: Bake a cake

"find recipe" -> "gather ingredients”;
"gather ingredients" -> "mix ingredients”;
"gather ingredients" -> "preheat oven at 375F”;
"find recipe" -> "preheat oven at 375F”;
"preheat oven at 375F" -> "put the cake batter into oven";
"mix ingredients" -> "put the cake batter into oven";
"put the cake batter into oven" -> "take the cake out after 30 min"

Intermediate Representation

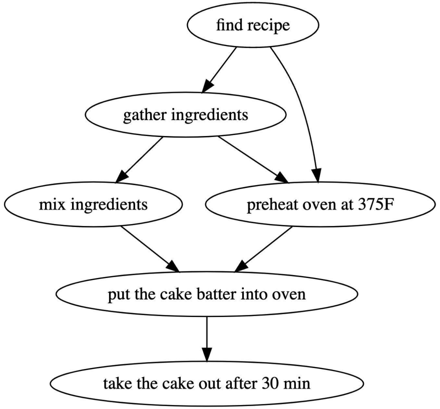


Recovered Graph

Leveraging Language Models for Structured commonsense Reasoning

- Issues with the workaround

```
"find recipe" → "gather ingredients";
“gather ingredients" → "mix ingredients”;
"find recipe" → "preheat oven at 375F”;
"preheat oven at 375F" → "put the cake batter into oven";
"mix ingredients" → "put the cake batter into oven";
"put the cake batter into oven" → "take the cake out after 30
min"
```

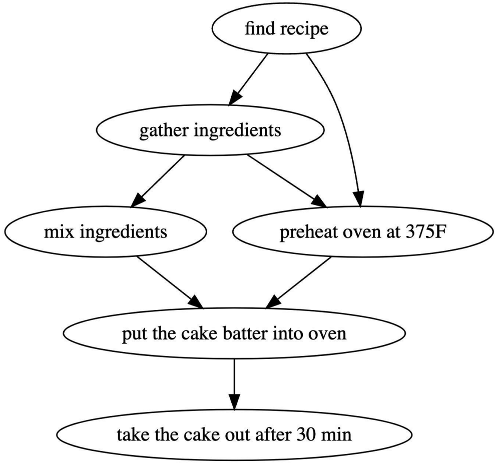


Are the two
mix ingredients the same?

- We want ***structures***, ***not strings***

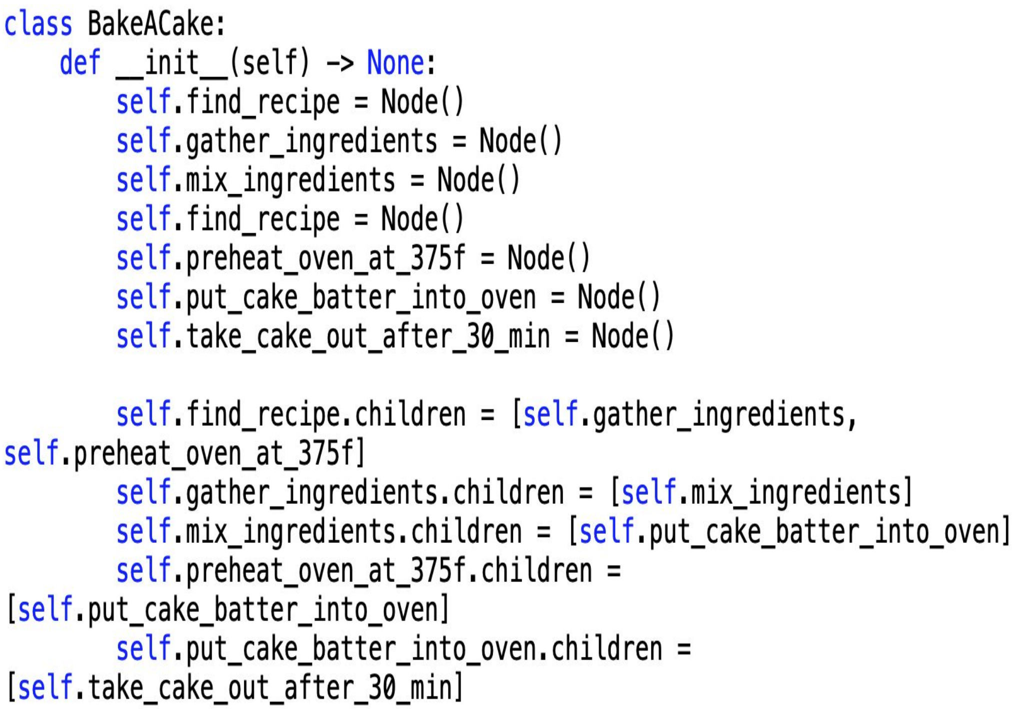
Language Models of Code Are Few-shot Commonsense Learners (Madaan et al. 2022)

```
class BakeACake:
    def __init__(self) -> None:
        self.find_recipe = Node()
        self.gather_ingredients = Node()
        self.mix_ingredients = Node()
        self.find_recipe = Node()
        self.preheat_oven_at_375f = Node()
        self.put_cake_batter_into_oven = Node()
        self.take_cake_out_after_30_min = Node()

        self.find_recipe.children = [self.gather_ingredients,
self.preheat_oven_at_375f]
        self.gather_ingredients.children = [self.mix_ingredients]
        self.mix_ingredients.children = [self.put_cake_batter_into_oven]
        self.preheat_oven_at_375f.children =
[self.put_cake_batter_into_oven]
        self.put_cake_batter_into_oven.children =
[self.take_cake_out_after_30_min]
```

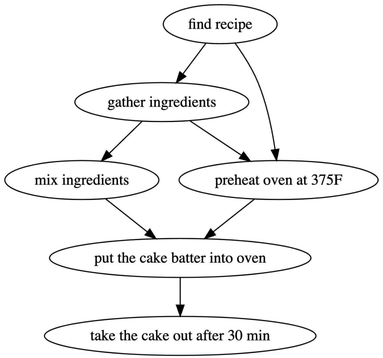


Language Models of Code Are Few-shot Commonsense Learners (Madaan et al. 2022)

Goal: Bake a cake



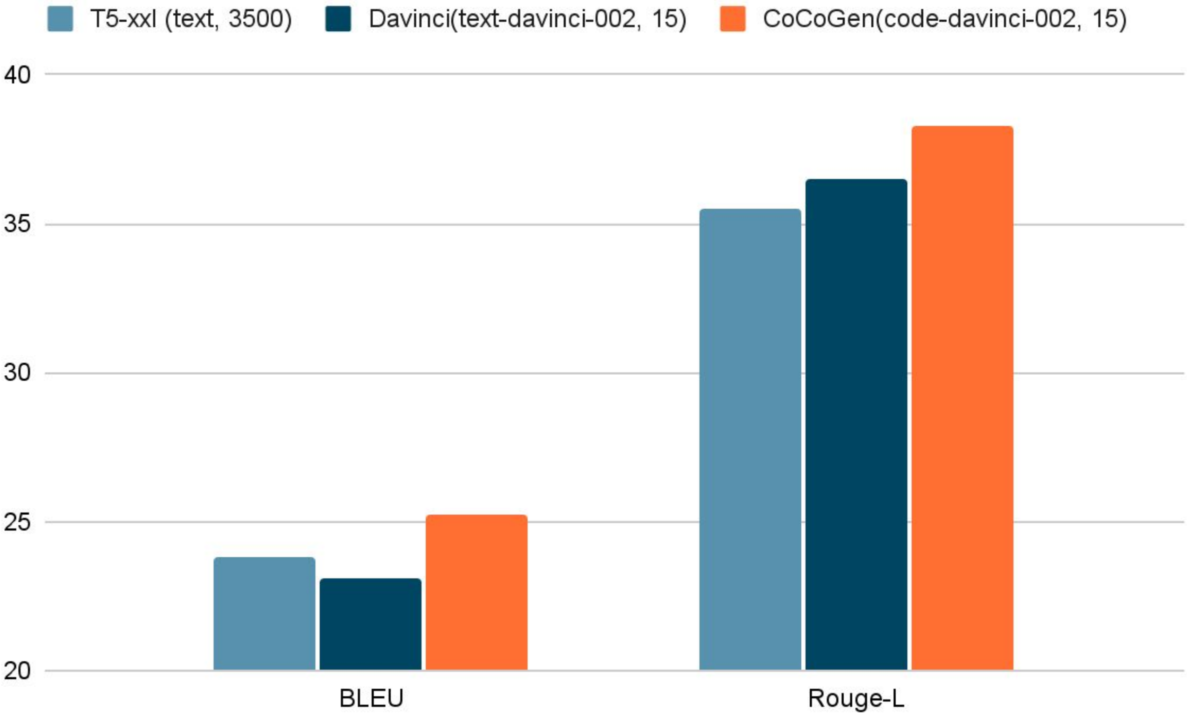
```
class BakeACake:
    def __init__(self) -> None:
        self.find_recipe = Node()
        self.gather_ingredients = Node()
        self.mix_ingredients = Node()
        self.find_recipe = Node()
        self.preheat_oven_at_375f = Node()
        self.put_cake_batter_into_oven = Node()
        self.take_cake_out_after_30_min = Node()

        self.find_recipe.children = [self.gather_ingredients,
self.preheat_oven_at_375f]
        self.gather_ingredients.children = [self.mix_ingredients]
        self.mix_ingredients.children = [self.put_cake_batter_into_oven]
        self.preheat_oven_at_375f.children =
[self.put_cake_batter_into_oven]
        self.put_cake_batter_into_oven.children =
[self.take_cake_out_after_30_min]
```



Recovered Graph

Intermediate Representation

Script Generation Results on ProScript



PAL: Program-aided Language Models (Gao*, Madaan*, Zhou* et al. 2023)

- Olivia has $23. She bought five bagels for $3 each. How much money does she have left?

Olivia had 23 dollars. 5 bagels for 3 dollars each will be dollars. So she has dollars left.

CoT

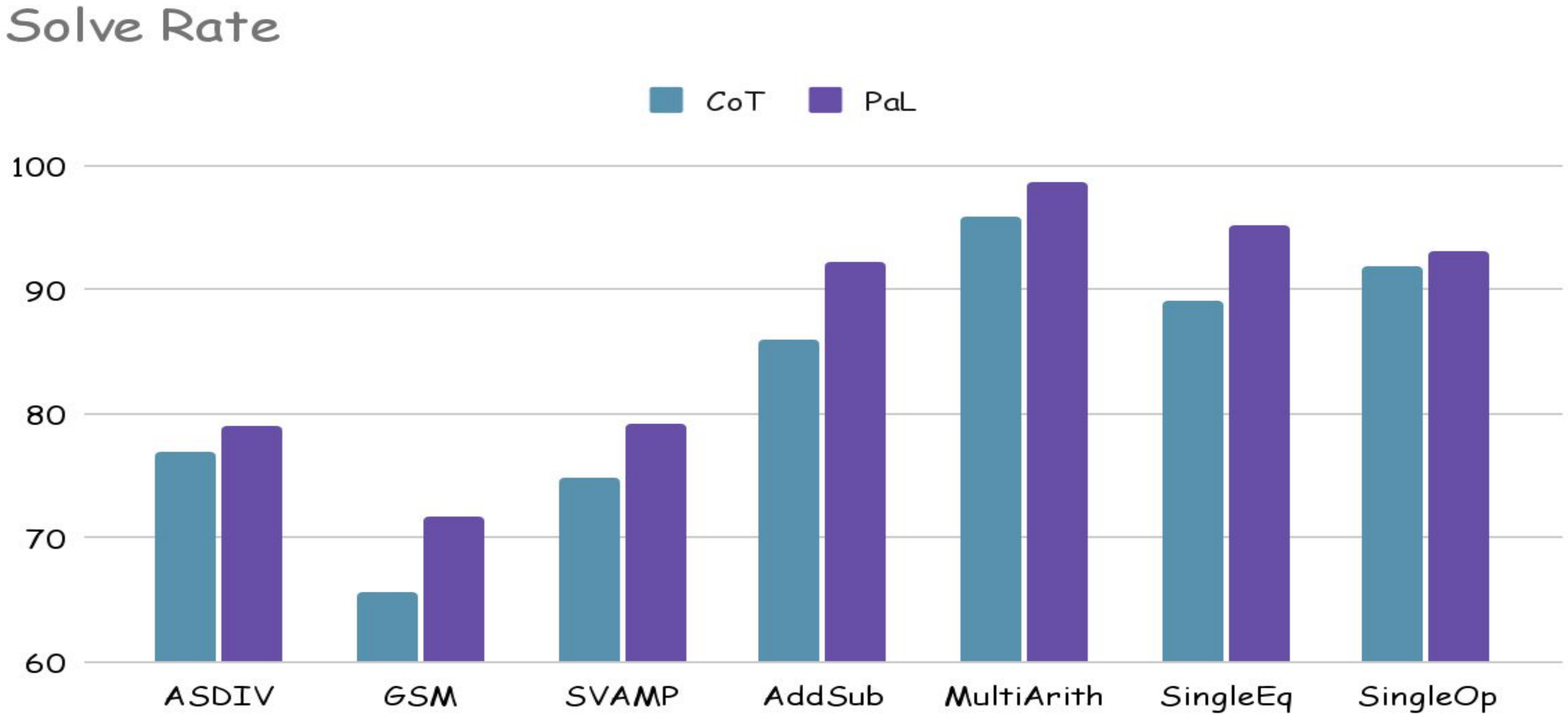
```
def solution():
    money_initial = 23
    bagels = 5
    bagel_cost = 3
    money_spent = bagels * bagel_cost
    money_left = money_initial - money_spent
    result = money_left
    return result
```

PaL

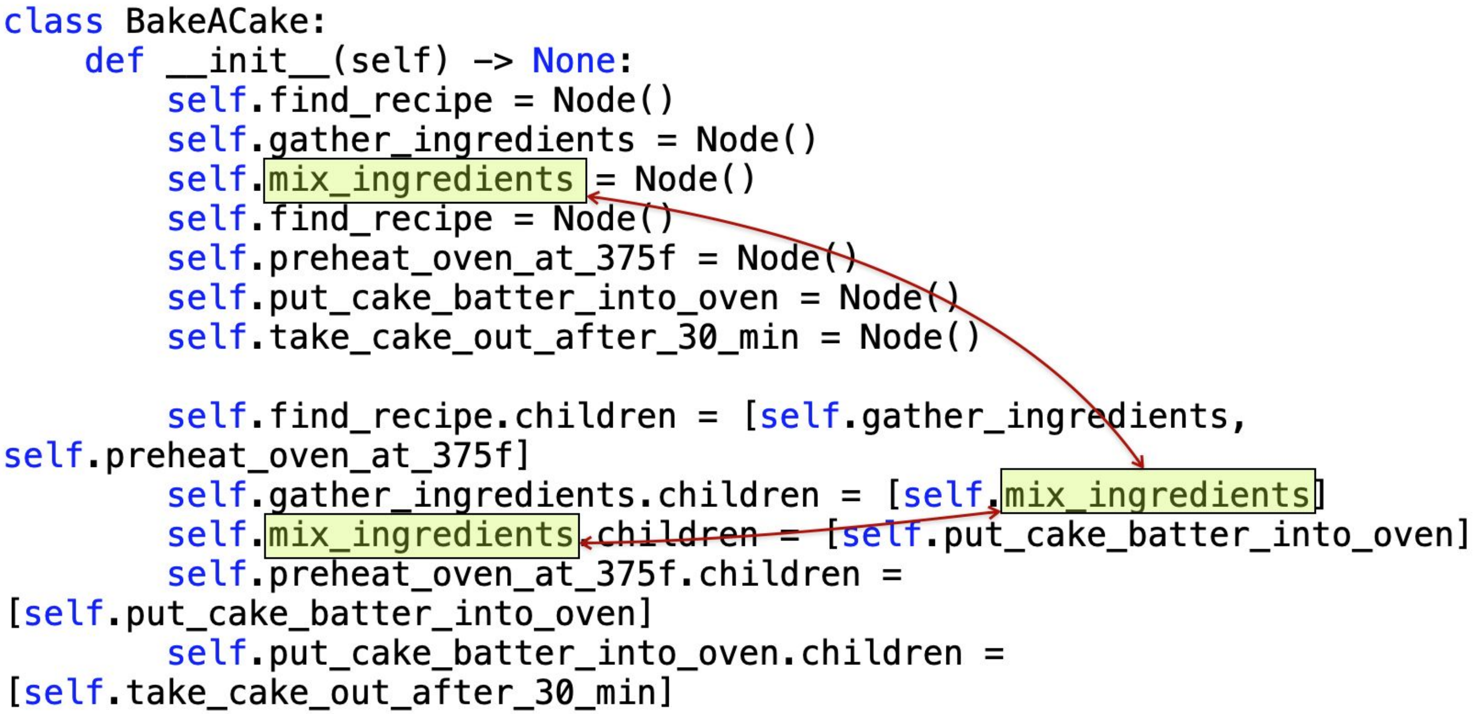
Comparison with CoT:

- The language model is responsible for generating a high-level plan that is **executed** to derive the answer
- The results are obtained after running the program

Improves Solve Rate for Multiple Maths Reasoning Tasks

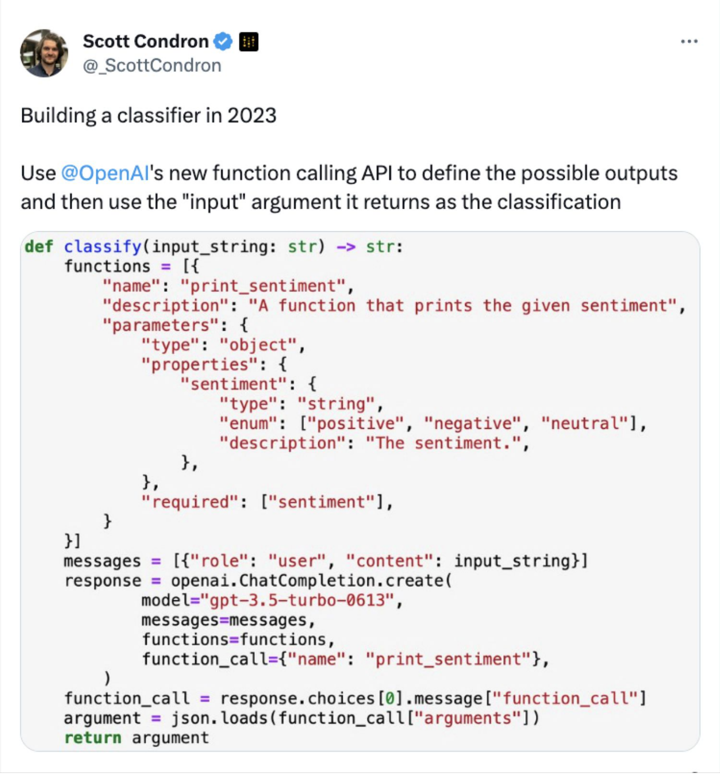


Why should code help?



```
class BakeACake:
    def __init__(self) -> None:
        self.find_recipe = Node()
        self.gather_ingredients = Node()
        self.mix_ingredients = Node()
        self.find_recipe = Node()
        self.preheat_oven_at_375f = Node()
        self.put_cake_batter_into_oven = Node()
        self.take_cake_out_after_30_min = Node()

        self.find_recipe.children = [self.gather_ingredients,
self.preheat_oven_at_375f]
        self.gather_ingredients.children = [self.mix_ingredients]
        self.mix_ingredients.children = [self.put_cake_batter_into_oven]
        self.preheat_oven_at_375f.children =
[self.put_cake_batter_into_oven]
        self.put_cake_batter_into_oven.children =
[self.take_cake_out_after_30_min]
```

Structured Generation



Scott Condron @_ScottCondron

Building a classifier in 2023

Use @OpenAI's new function calling API to define the possible outputs and then use the "input" argument it returns as the classification

```python
def classify(input_string: str) -> str:
    functions = [{
        "name": "print_sentiment",
        "description": "A function that prints the given sentiment",
        "parameters": {
            "type": "object",
            "properties": {
                "sentiment": {
                    "type": "string",
                    "enum": ["positive", "negative", "neutral"],
                    "description": "The sentiment.",
                },
            },
            "required": ["sentiment"],
        }
    }]
    messages = [{"role": "user", "content": input_string}]
    response = openai.ChatCompletion.create(
            model="gpt-3.5-turbo-0613",
            messages=messages,
            functions=functions,
            function_call={"name": "print_sentiment"},
        )
    function_call = response.choices[0].message["function_call"]
    argument = json.loads(function_call["arguments"])
    return argument
```

https://twitter.com/_ScottCondron/status/1670827747684364288

More on future directions

Complex Few-shot Reasoning with LLMs: Key Techniques

✓ **Reasoning Elaboration: Spell out the reasoning process before generating the answer**

✓ **Instructions Provide explicit instructions to the LLM, capture all expectations**

✓ **Tool Augmentation: Enhance LLMs for specialized tasks**

✓ **Structured Generation: Use structure to guide model**

Feedback

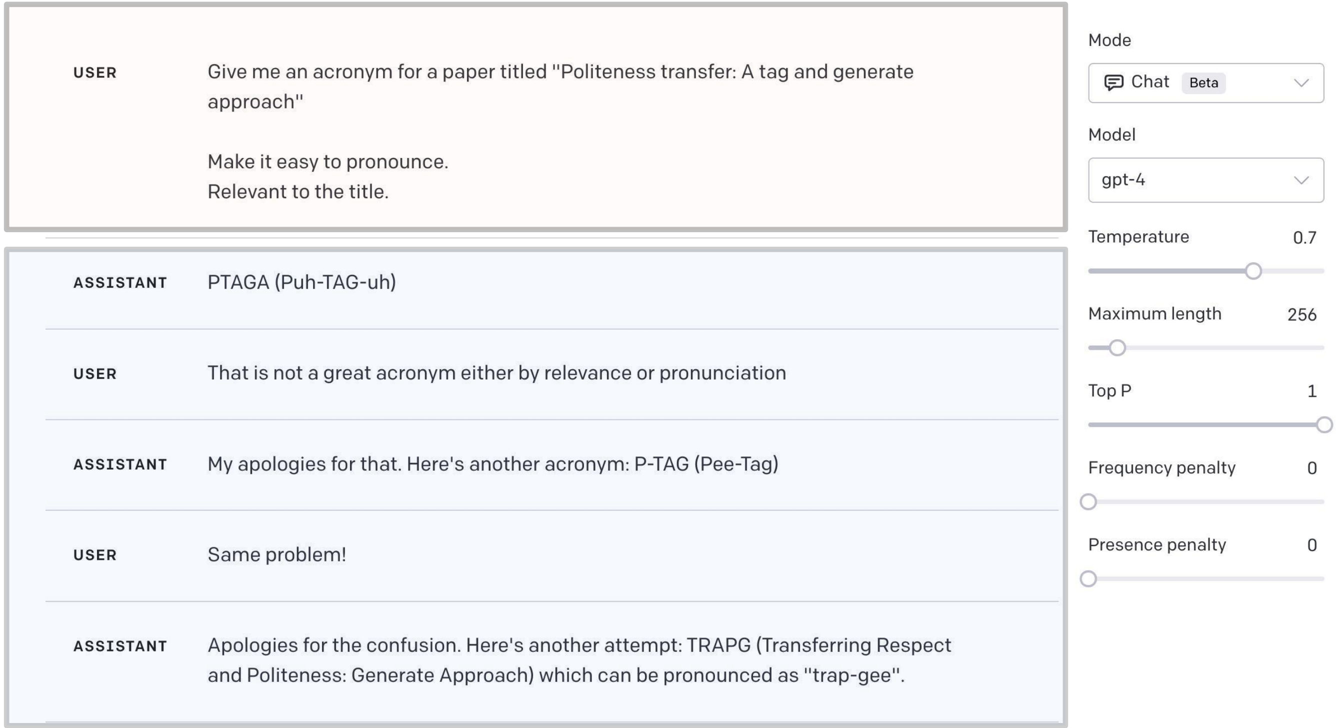
Memory

Feedback

- Humans typically generate content iteratively:
    - Generate → Feedback → Refine

- LLMs, with the right prompt, have recently shown promise at mimicking this process
    - Generate an output
    - Gather feedback
    - Improve output with feedback, repeat

Self-Refine (Madaan et al. 2023)



USER Give me an acronym for a paper titled "Politeness transfer: A tag and generate approach"

Make it easy to pronounce.
Relevant to the title.

ASSISTANT PTAGA (Puh-TAG-uh)

USER That is not a great acronym either by relevance or pronunciation

ASSISTANT My apologies for that. Here's another acronym: P-TAG (Pee-Tag)

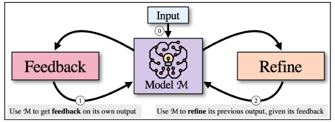
USER Same problem!

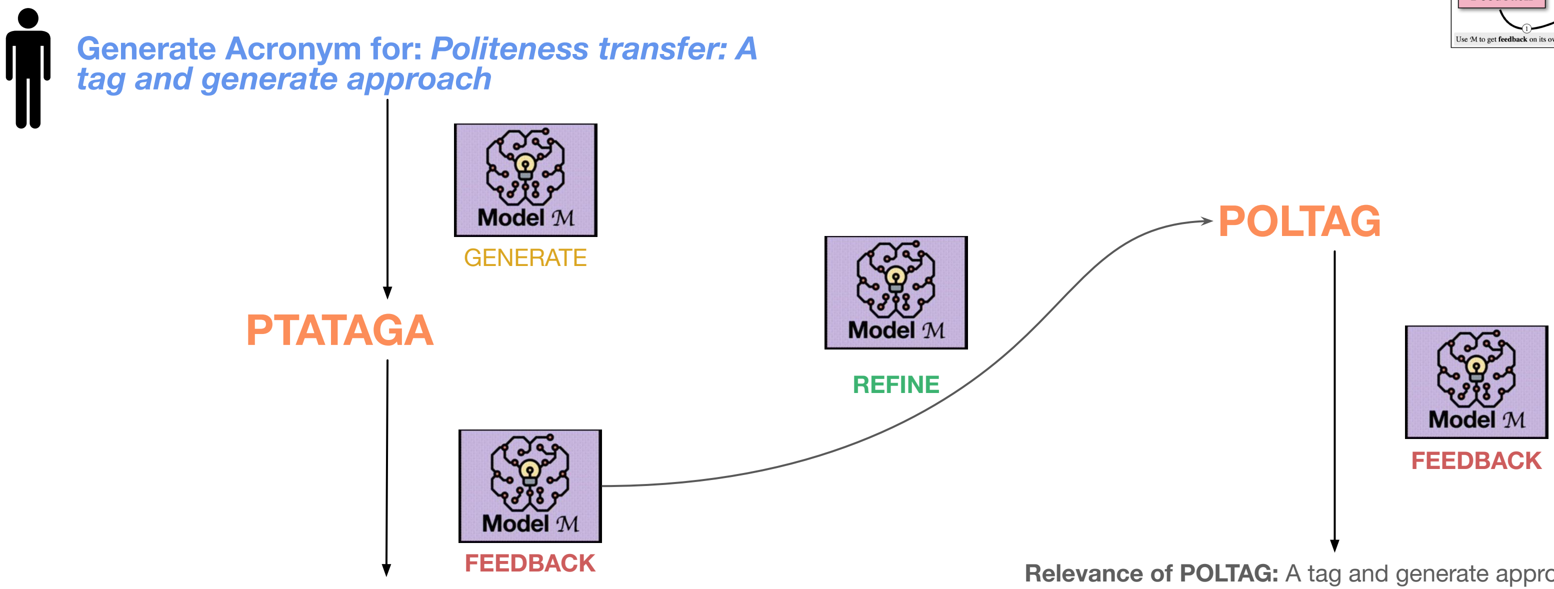
ASSISTANT Apologies for the confusion. Here's another attempt: TRAPG (Transferring Respect and Politeness: Generate Approach) which can be pronounced as "trap-gee".

Mode: Chat Beta
Model: gpt-4
Temperature 0.7
Maximum length 256
Top P 1
Frequency penalty 0
Presence penalty 0

GPT-3.5/4 is good!
But often requires a few rounds of talking through the problem

Let LLM Self-Refine


Input
0
Feedback
Model M
Refine
1
2
Use M to get feedback on its own output
Use M to refine its previous output, given its feedback



Generate Acronym for: *Politeness transfer: A tag and generate approach*

PTATAGA

POLTAG

Relevance of PTATAGA: The acronym includes the first letters of each major word in the title but doesn't fully capture the meaning of the title. <3/5>

Memorability of PTATAGA: The acronym is not the most memorable. <2/5>

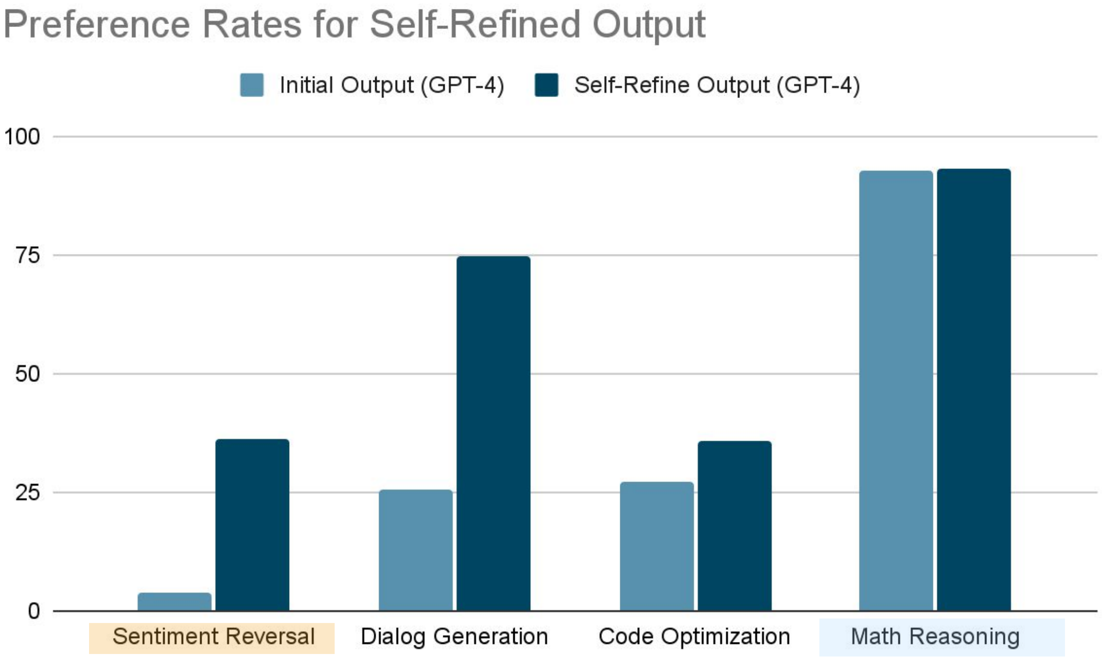
Pronunciation of PTATAGA: The acronym is pronounceable but not very fluid. <3/5>

Relevance of POLTAG: A tag and generate approach: The acronym POLTAG is a good representation of the title as it combines the first few letters of Politeness and Tag. <4/5>

Memorability of POLTAG: The acronym POLTAG is easy to remember as it is short, simple, and has a clear structure. <4/5> → STOP

Pronunciation of POLTAG: The acronym is easily pronounced as it follows the common structure of acronyms. <5/5>

Self-Refine Results

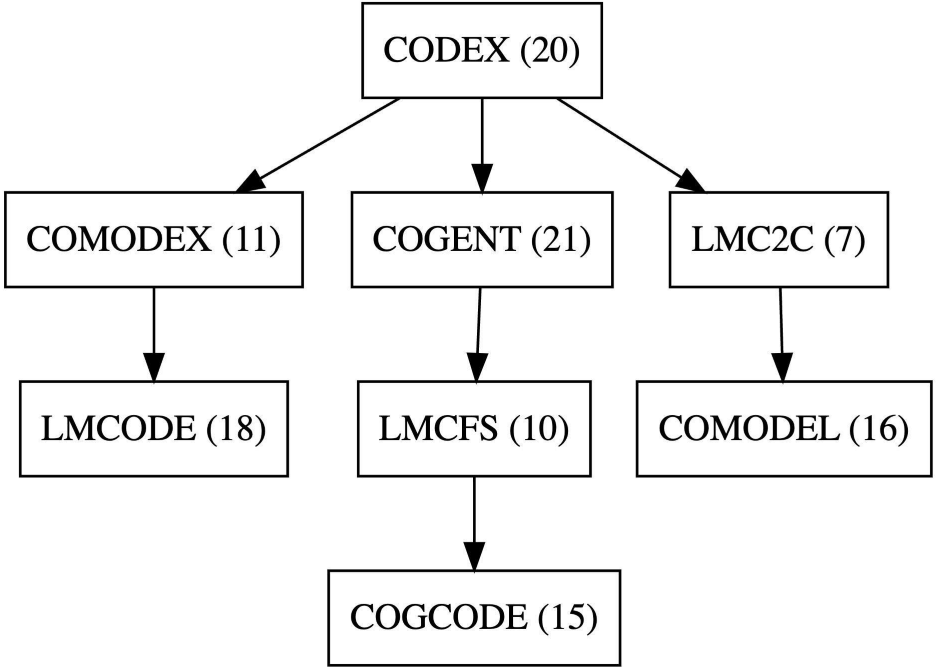


- Larger gains where feedback is obvious + easy
- Lots of room for improvement!

Future Directions: Feedback + Planning

- Monte-Carlo Tree Search + Self-Refine
  - Extend Self-Refine Beyond Linear Search

- Other exciting work:
  - Language Models can Solve Computer Tasks (Kim et al. 2023)
  - Tree-of-thought (Yao et al. 2023)
  - Reflexion (Shinn et al. 2023)
  - Refiner (Paul et al. 2023)



CODEX (20)

COMODEX (11)

COGENT (21)

LMC2C (7)

LMCODE (18)

LMCFS (10)

COMODEL (16)

COGCODE (15)

Complex Few-shot Reasoning with LLMs: Key Techniques

✓ **Reasoning Elaboration: Spell out the reasoning process before generating the answer**

✓ **Instructions Provide explicit instructions to the LLM, capture all expectations**

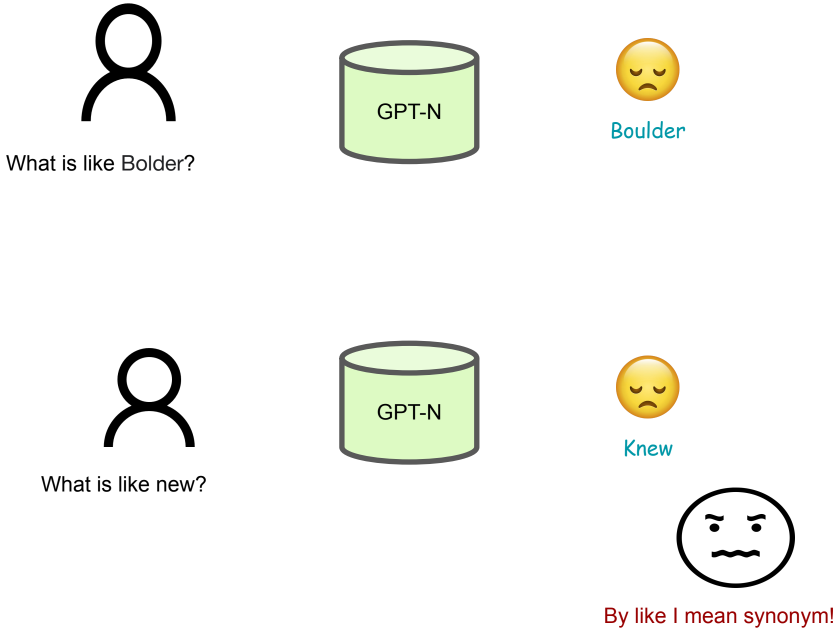
✓ **Tool Augmentation: Enhance LLMs for specialized tasks**

✓ **Structured Generation: Use structure to guide model**

✓ **Feedback: Refine model outputs during inference**

Memory

Models Repeat Mistakes



What is like Bolder?

GPT-N

Boulder

What is like new?

GPT-N

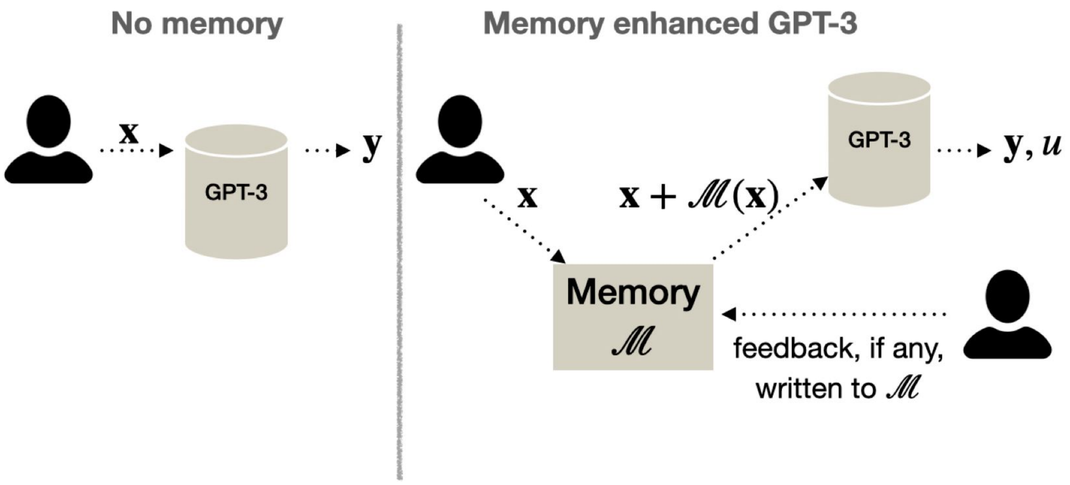
Knew

By like I mean synonym!

Memory

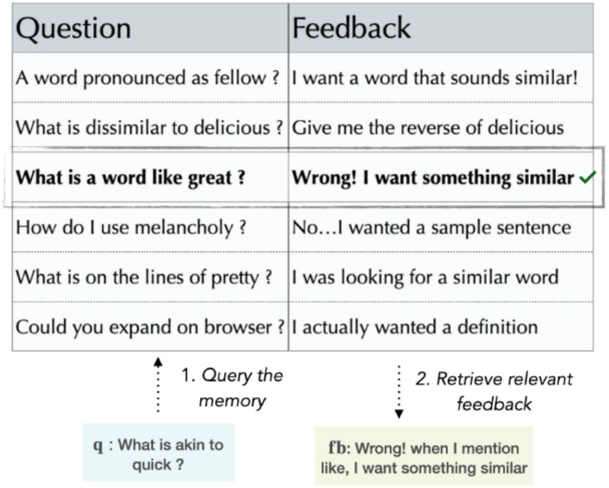
- Standard few-shot prompting setup
    - Prompt **P**: [X_i, Y_i]
    - Test examples X_1, X_2
        - **P** + $X_1 \rightarrow Y'_1$
        - **P** + $X_2 \rightarrow Y'_2$
    - What if user provides a feedback on $X_1 \rightarrow Y'_1$?
    - Can we improve the output generated for X_2 <u>without re-training</u>?
- Solution: maintain a memory of examples seen so far, and any feedback
    - Update the prompt **P** dynamically
    - Stateful inferences

MemPrompt


No memory
x
GPT-3
y
Memory enhanced GPT-3
x
$\mathbf{x} + \mathcal{M}(\mathbf{x})$
GPT-3
$\mathbf{y}, u$
Memory $\mathcal{M}$
feedback, if any, written to $\mathcal{M}$

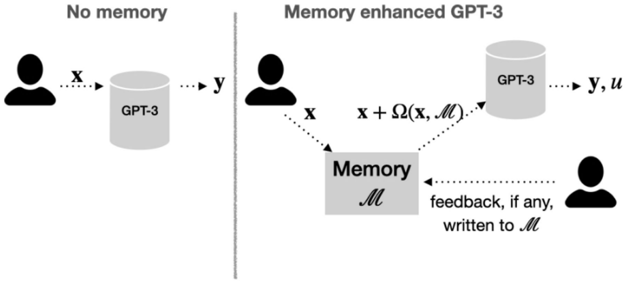
MemPrompt: Workflow

- **Step 1:** User asks a question
- **Step 2:** Check if the same question has been asked before, and a clarification is present in memory
  - Step 2.1: If a clarification is present, add question + clarification to the **prompt**
  - Step 2.2: If not, just ask a question
- **Step 3:** Model generates an answer
- **Step 4:** Take clarification on answer if needed, add clarification to memory



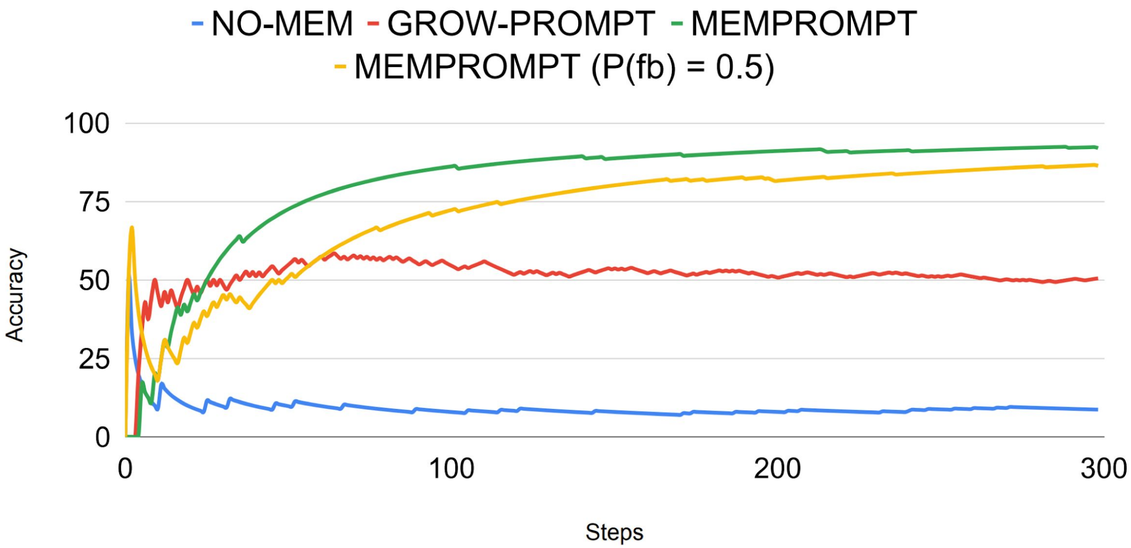
Question	Feedback
A word pronounced as fellow ?	I want a word that sounds similar!
What is dissimilar to delicious ?	Give me the reverse of delicious
What is a word like great ?	**Wrong! I want something similar** ✓
How do I use melancholy ?	No…I wanted a sample sentence
What is on the lines of pretty ?	I was looking for a similar word
Could you expand on browser ?	I actually wanted a definition

1. Query the memory
2. Retrieve relevant feedback
q : What is akin to quick ?
fb: Wrong! when I mention like, I want something similar


No memory
Memory enhanced GPT-3
x
GPT-3
y
x
x + Ω(x, ℳ)
GPT-3
y, u
Memory ℳ
feedback, if any, written to ℳ

MemPrompt: Personalization

Queries in Punjabi



Memory for Few-shot Prompting

- Simpler but effective variants
  - What Makes Good In-Context Examples for GPT-3? (Liu et al. 2021): Store the training set in a database, retrieve most relevant examples on the fly

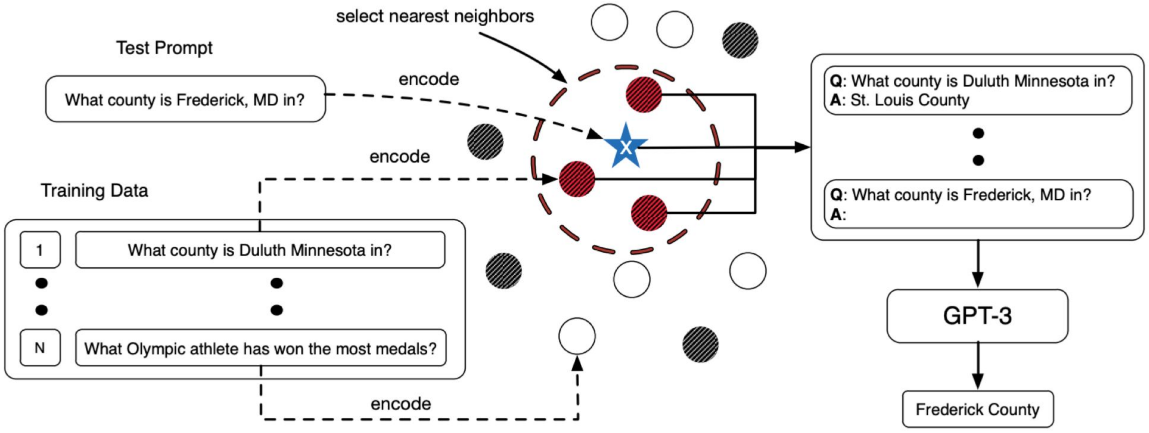

select nearest neighbors
Test Prompt
What county is Frederick, MD in?
encode
encode
Training Data
1
What county is Duluth Minnesota in?
N
What Olympic athlete has won the most medals?
encode
x
Q: What county is Duluth Minnesota in?
A: St. Louis County
Q: What county is Frederick, MD in?
A:
GPT-3
Frederick County

Figure 2: In-context example selection for GPT-3. White dots: unused training samples; grey dots: randomly sampled training samples; red dots: training samples selected by the k-nearest neighbors algorithm in the embedding space of a sentence encoder.

Complex Few-shot Reasoning with LLMs: Key Techniques

- ✓ **Reasoning Elaboration: Spell out the reasoning process before generating the answer**
- ✓ **Instructions: Provide explicit instructions to the LLM, don’t expect mind reading**
- ✓ **Tool Augmentation: Blend tools to leverage LLMs for specialized tasks**
- ✓ **Structured Generation: Use structure to guide model**
- ✓ **Feedback: Refine model outputs during inference**
- ✓ **Memory: Maintain a history of interactions with LLM**

Future Directions

- LLMs are getting better at following instructions, exciting new possibilities
    - Planning, Search – resurgence of classical AI techniques
    - Generating actionable feedback from tools
- Appendix: Why does few-shot prompting work?

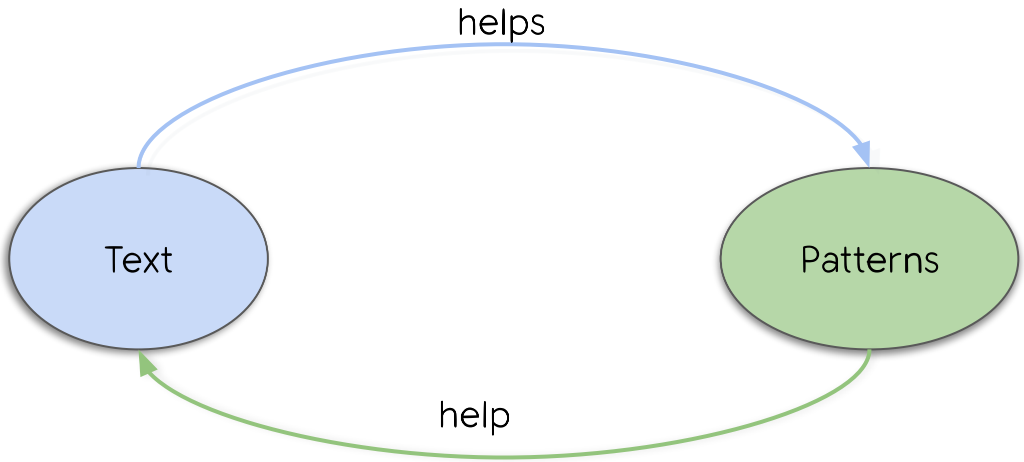
Complex Few-shot Reasoning with LLMs: Key Techniques

Questions @ Rocket Chat

Appendix

But Why Does it Work?

Text and Patterns: For Effective Chain of Thought, It Takes Two to Tango

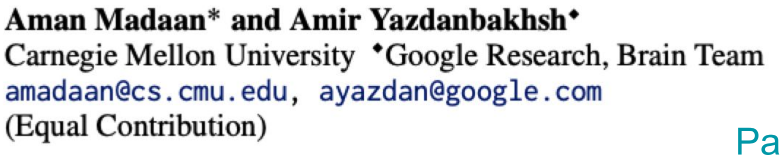


helps

Text

Patterns

help

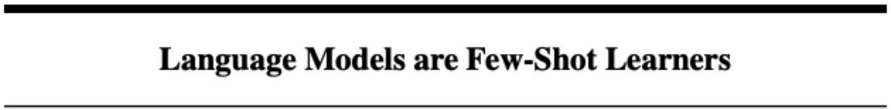


Aman Madaan* and Amir Yazdanbakhsh[♦]
Carnegie Mellon University [♦]Google Research, Brain Team
amadaan@cs.cmu.edu, ayazdan@google.com
(Equal Contribution)

Paper

What makes the chain of thought prompting so effective?

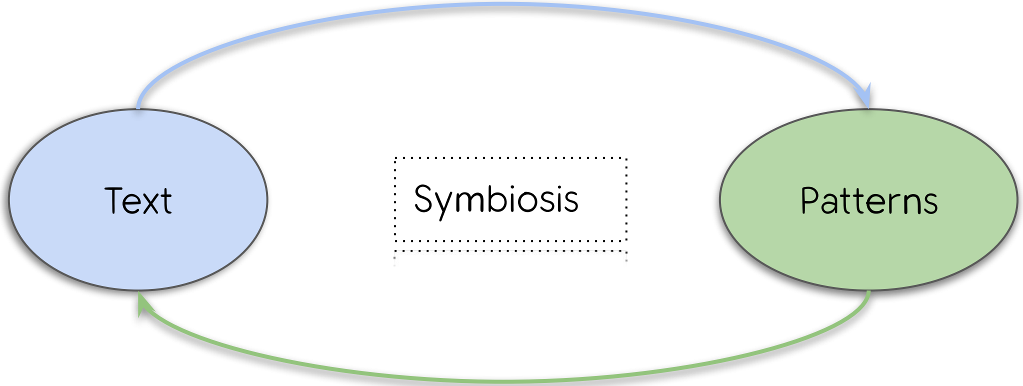
- *The thought makes the model think about the problem?*
- *The thought helps the model learn better*
- *The thought serves as an additional example of the task*
- *The thought helps the model remind of the task*
- *The thought helps extract relevant information for solving the task*



Language Models are Few-Shot Learners

What makes chain of thought prompting so effective

Text: Extracting common sense, copy mechanism

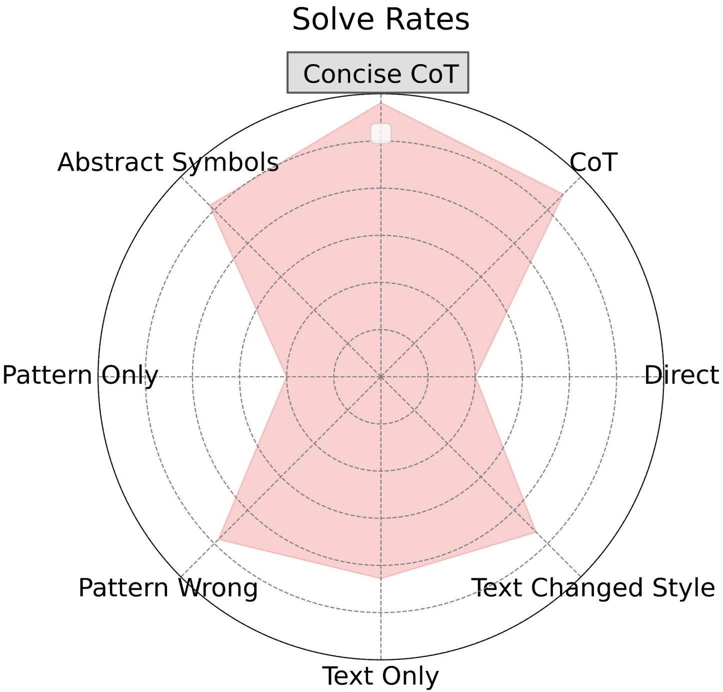


Text

Symbiosis

Patterns

Patterns: Task understanding, final answer generation



Approach

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

T: There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5.

A: The answer is 5 cars.

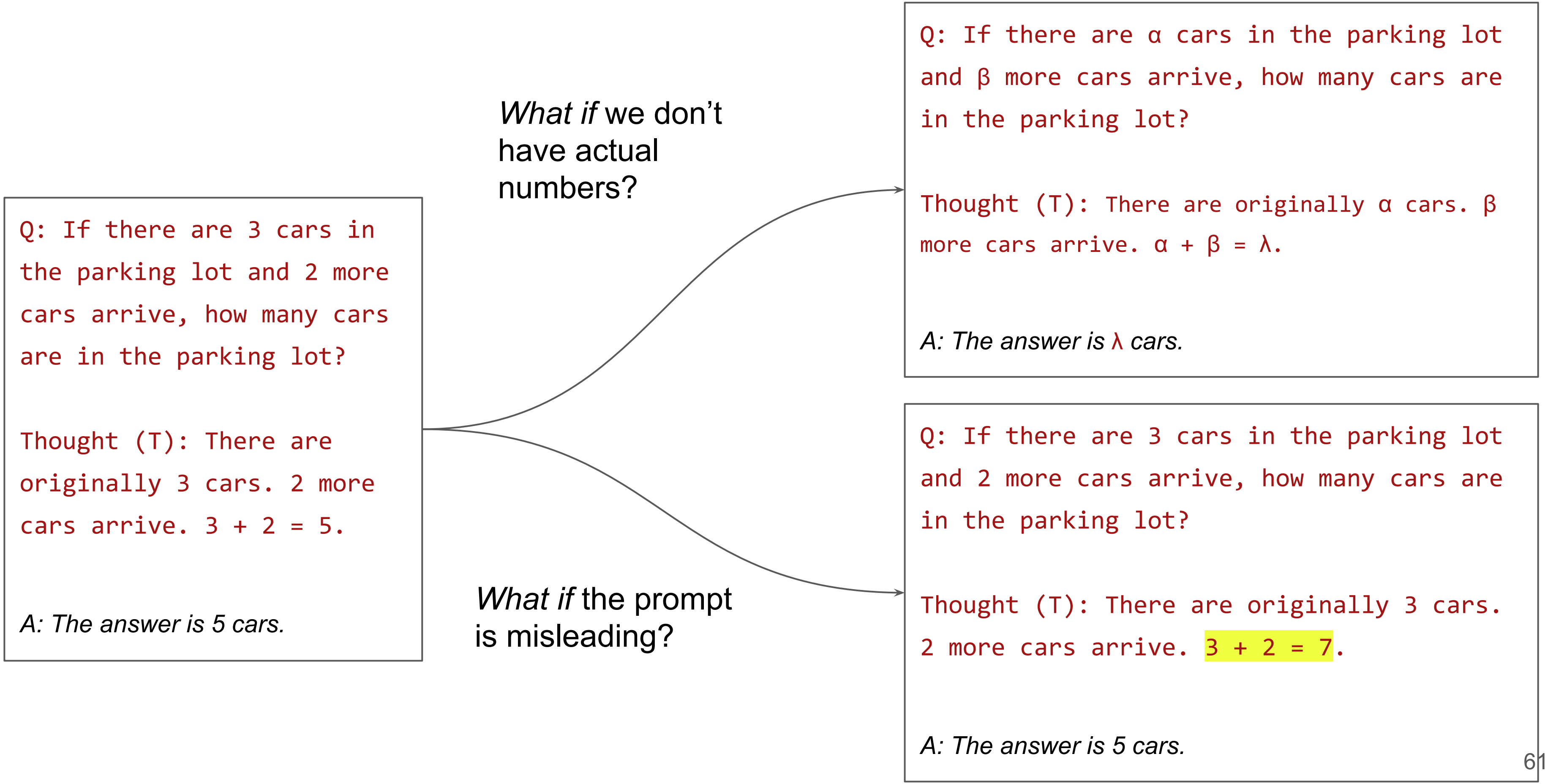
Symbols

Patterns

Text

- ***Counterfactual prompting:***
  - Change one *knob* at a time (symbol, patterns, text)

What if? prompting (counterfactual prompting)



Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

Thought (T): There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5.

A: The answer is 5 cars.

What if we don't have actual numbers?

Q: If there are α cars in the parking lot and β more cars arrive, how many cars are in the parking lot?

Thought (T): There are originally α cars. β more cars arrive. $\alpha + \beta = \lambda$.

A: The answer is λ cars.

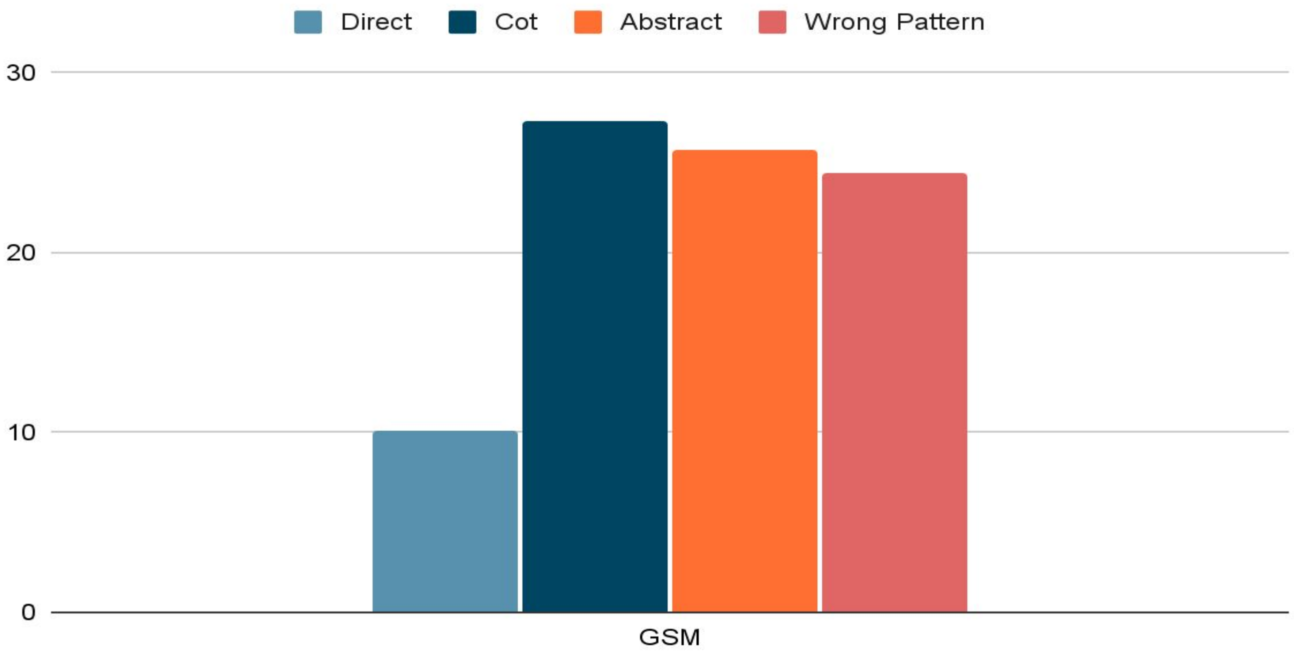
What if the prompt is misleading?

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

Thought (T): There are originally 3 cars. 2 more cars arrive. 3 + 2 = 7.

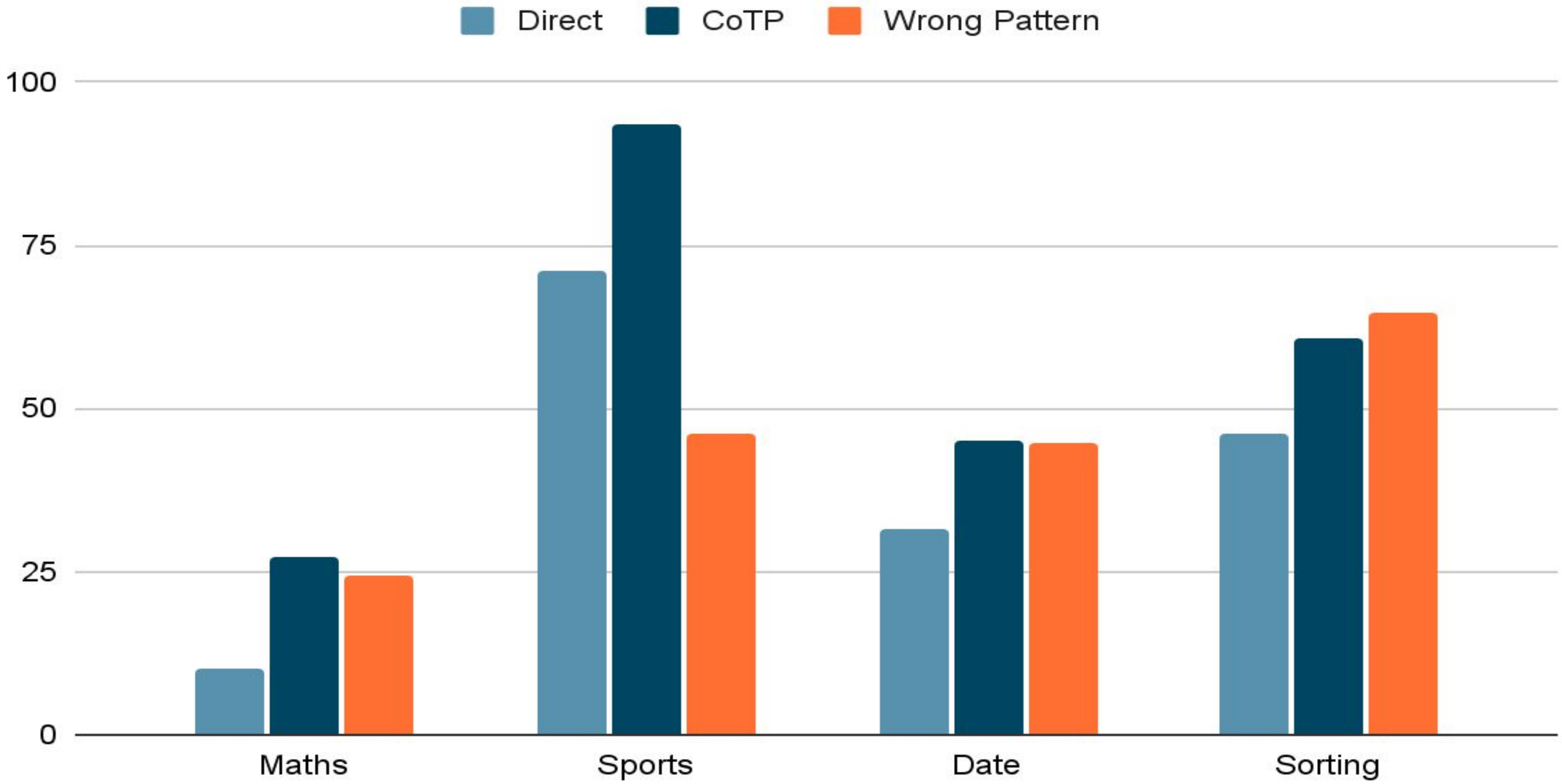
A: The answer is 5 cars.

Performance with Counterfactual Prompts

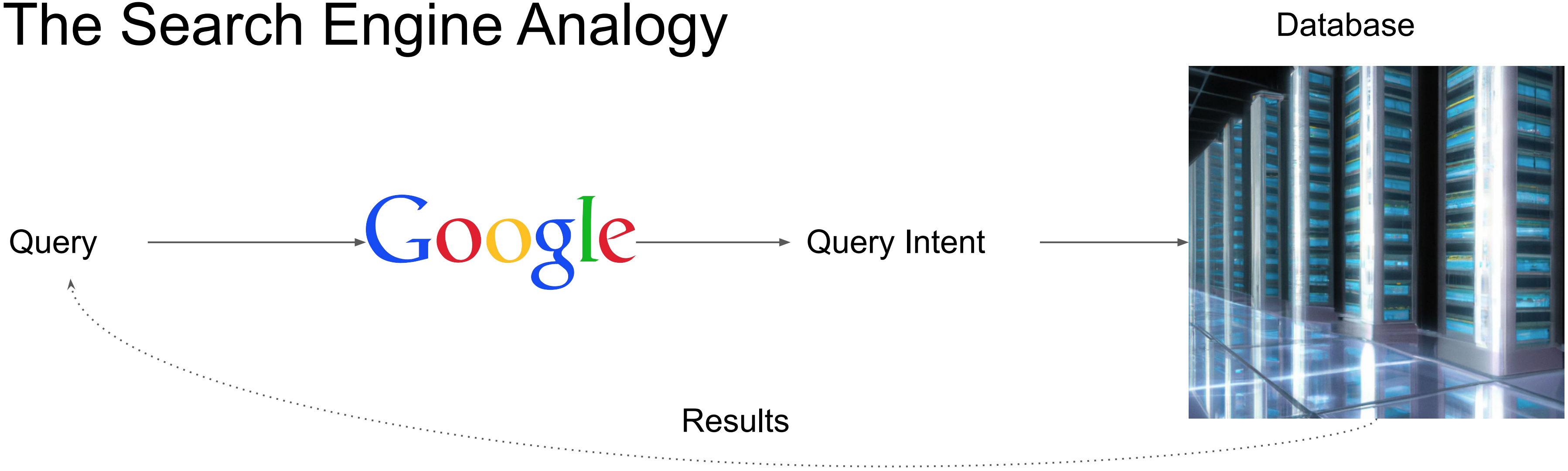


Performance does not change!

Other Tasks


Direct
CoTP
Wrong Pattern
100
75
50
25
0
Maths
Sports
Date
Sorting

The Search Engine Analogy



Query ≅ Prompt

Query Intent ≅ Task Understanding

Database ≅ Weights

Also see: https://ai.stanford.edu/blog/understanding-incontext/