# LLM Reasoning: Key Ideas and Limitations

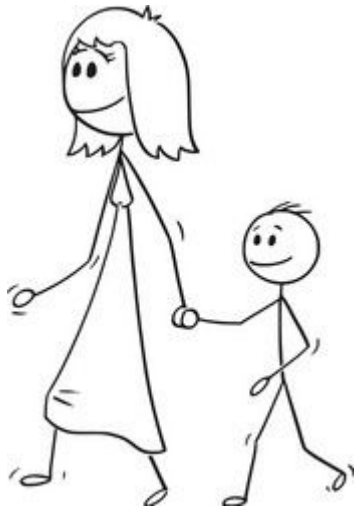
Denny Zhou

Google DeepMind

April, 2024

My mom is as smart as AI

Mother's Day 2023, my little son wrote:



Self-driving cars

Solve hardest math problems

Superhuman intelligence …

# What do you want about AI?

# My little expectation about AI

AI should be able to **learn from only a few examples**, like what humans usually do

# Does machine learning meet this expectation?

Semi-supervised learning
Manifold learning
Sparsity and low rank
Active learning
Reinforcement learning
Bayesian nonparametric
Kernel machines
...

What is missing in machine learning?

# Reasoning

Humans can learn from only a few examples because humans can reason

# Let’s start from a toy problem

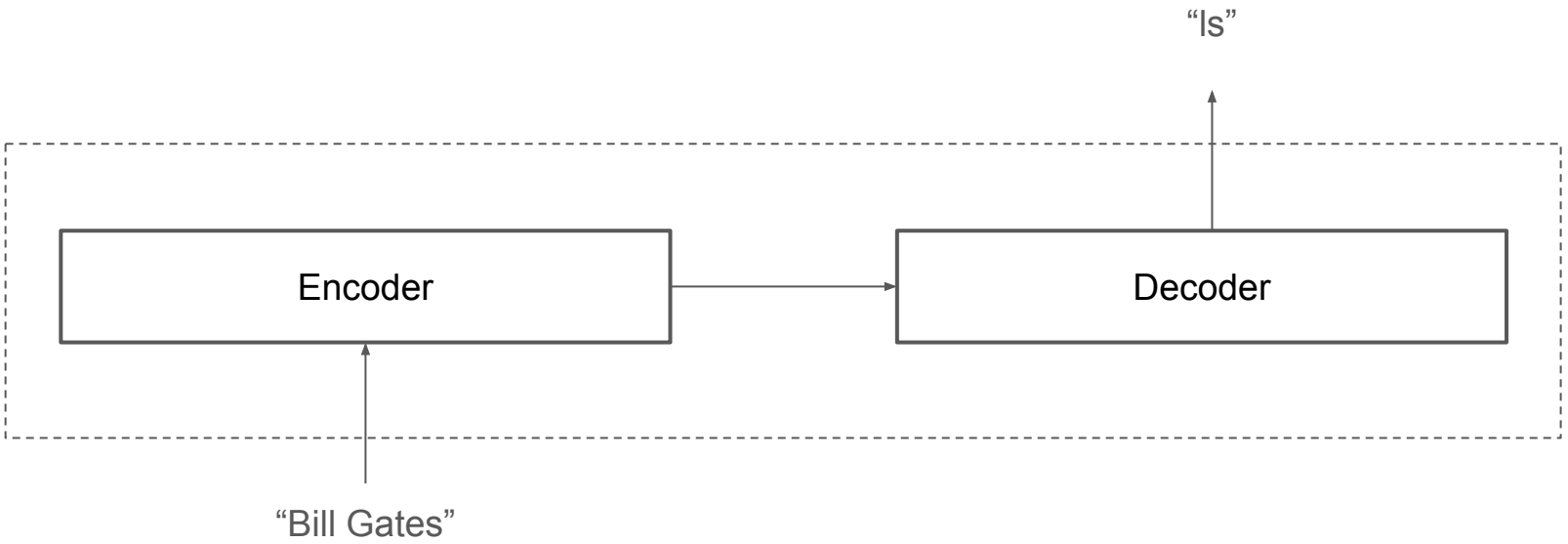
“Make things as simple as possible, but no simpler”

# A silly toy problem: last-letter-concatenation

| Input | Output |
| --- | --- |
| “Elon Musk” | “nk” |
| “Bill Gates” | “ls” |
| “Barack Obama” | ? |

**Rule**: Take the last letter of each word, and then concatenate them

# Solve it by machine learning? Tons of labels needed!

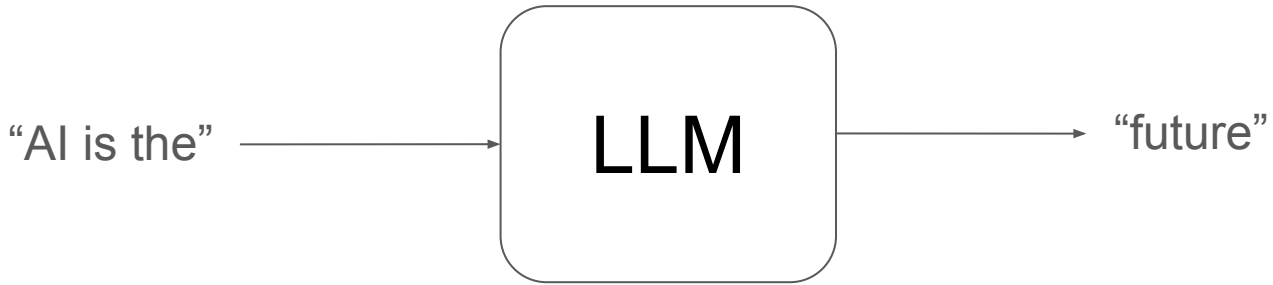



Would you like to call an ML method which needs tons of labels to learn a “trivial” task as AI?

How can this problem be solved by LLMs?

# What are **L**arge **L**anguage **M**odels (LLMs)?

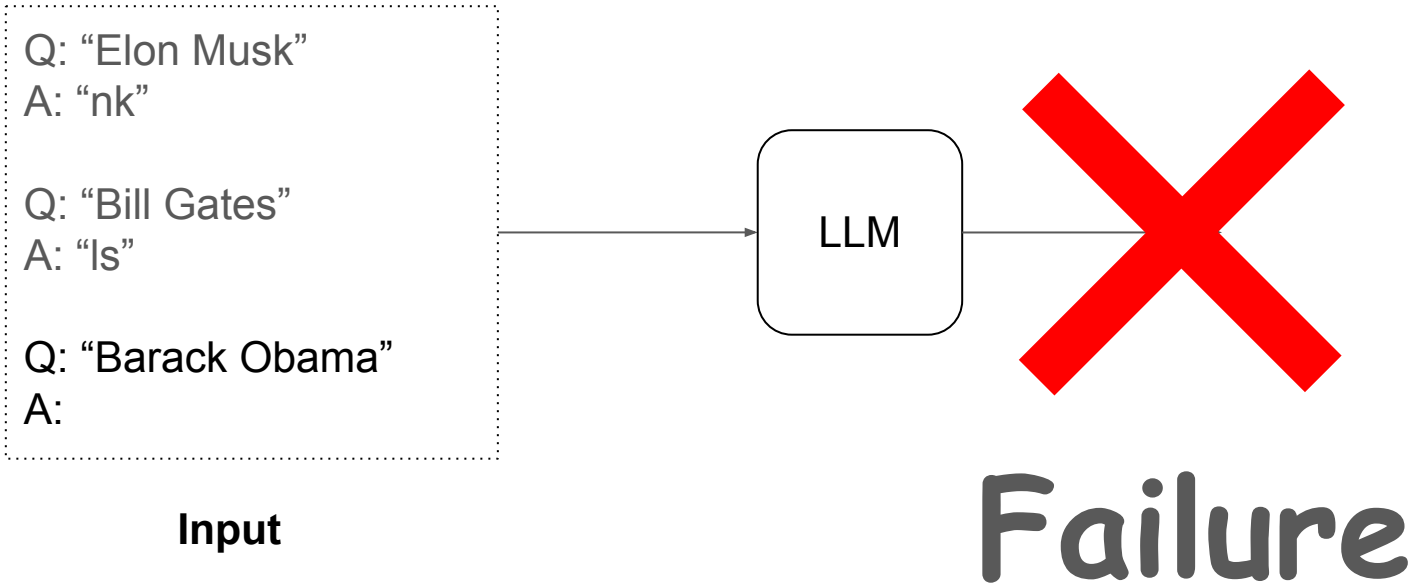
LLM is a “transformer” model trained to predict the next word




Trained with many sentences, e.g. all texts from the Internet

You can think of training LLMs as training parrots to mimic human languages

# Few-shot prompting for last-letter-concatenation




**Input**

# Let’s add “reasoning process” before “answer”

Q: “Elon Musk”
A: the last letter of "Elon" is "n". the last letter of "Musk" is "k". Concatenating "n", "k" leads to "nk". so the output is "nk".

reasoning process

Q: “Bill Gates”
A: the last letter of "Bill" is "l". the last letter of "Gates" is "s". Concatenating "l", "s" leads to "ls". so the output is "ls".

Q: “Barack Obama"
A: the last letter of "Barack" is "k". the last letter of "Obama" is "a". Concatenating "k", "a" leads to "ka". so the output is "ka".

# One demonstration is enough, like humans

Q: “Elon Musk”
A: the last letter of "Elon" is "n". the last letter of "Musk" is "k". Concatenating "n", "k" leads to "nk". so the output is "nk".

Q: “Barack Obama"
A: the last letter of "Barack" is "k". the last letter of "Obama" is "a". Concatenating "k", "a" leads to "ka". so the output is "ka".

# Key Idea: Derive the Final Answer through Intermediate Steps

Ling et al 2017 in DeepMind pioneered using **natural language rationale** to solve math problems by "... **derive the final answer through a series of small steps**". Trained a sequence-to-sequence model from scratch.

**Problem 1**:
**Question**: Two trains running in opposite directions cross a man standing on the platform in 27 seconds and 17 seconds respectively and they cross each other in 23 seconds. The ratio of their speeds is:
**Options**: A) 3/7 B) 3/2 C) 3/88 D) 3/8 E) 2/2
**Rationale**: Let the speeds of the two trains be x m/sec and y m/sec respectively. Then, length of the first train = 27x meters, and length of the second train = 17 y meters. (27x + 17y) / (x + y) = 23 → 27x + 17y = 23x + 23y → 4x = 6y → x/y = 3/2.
**Correct Option**: B

W Ling, D Yogatama, C Dyer, P Blunsom. Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems. ACL 2017.

# GSM8K: <Problem, Solution, Answer>

Following the work by Ling et al 2017, Cobbe et al 2021 in OpenAI built a much larger math word problem dataset (GSM8K) with natural language rationales, and used it to finetune GPT3

**Problem:** Ali is a dean of a private school where he teaches one class. John is also a dean of a public school. John has two classes in his school. Each class has 1/8 the capacity of Ali's class which has the capacity of 120 students. What is the combined capacity of both schools?
**Solution:** Ali's class has a capacity of 120 students. Each of John's classes has a capacity of 120/8 = 15 students. The total capacity of John's two classes is 15 students * 2 classes = 30 students. The combined capacity of the two schools is 120 students + 30 students = 150 students.
**Final answer: 150**

Cobbe et al. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168 [cs.LG]. October 2021.

# Scratchpad for Intermediate Computation

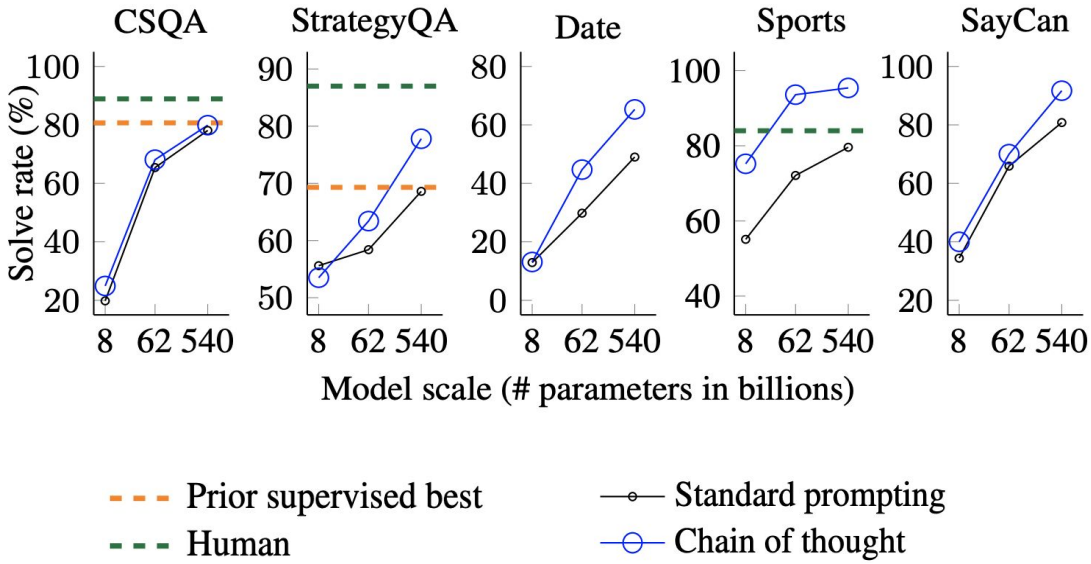
Nye et al 2021 proposed **Scratchpad**: predicting the final output of a program by predicting its intermediate execution result from line to line

```
Input:
2 9 + 5 7

Target:
<scratch>
2 9 + 5 7 ,  C: 0
2 + 5 , 6 C: 1  # added 9 + 7 = 6 carry 1
, 8 6 C: 0  # added 2 + 5 + 1 = 8 carry 0
0 8 6
</scratch>
8 6
```

Nye et al. Show Your Work: Scratchpads for Intermediate Computation with Language Models. arXiv:2112.00114 [cs.LG], November, 2021

# Chain of Thought Prompting




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

# Self-Consistency

Greatly improves step-by-step reasoning

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, Denny Zhou.
Self-Consistency Improves Chain of Thought Reasoning in Language Models. **ICLR 2023**.


Answer in the greedy output

≠

the most likely answer

# Key Idea: Marginalize "Latent" Reasoning Paths

$$\arg\max \mathbb{P}(\text{answer}|\text{problem}) \qquad \text{(find the answer with the maximum probability)}$$

$$= \sum_{\text{rationale}} \mathbb{P}(\text{answer, rationale}|\text{problem}) \qquad \text{(sum over all latent reasoning paths)}$$

$$\approx \frac{\text{frequency of the answer}}{\text{total number of sampled responses}} \qquad \text{(approximate the sum by sampling)}$$

$$\propto \text{frequency of the answer} \qquad \text{(ignore the common denominator)}$$

Thus, $\arg\max \mathbb{P}(\text{answer}|\text{problem}) \approx$ selecting the most frequent answer.

**[Question]** Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder for $2 per egg. How much does she make every day?

## Random sampled responses from LLM

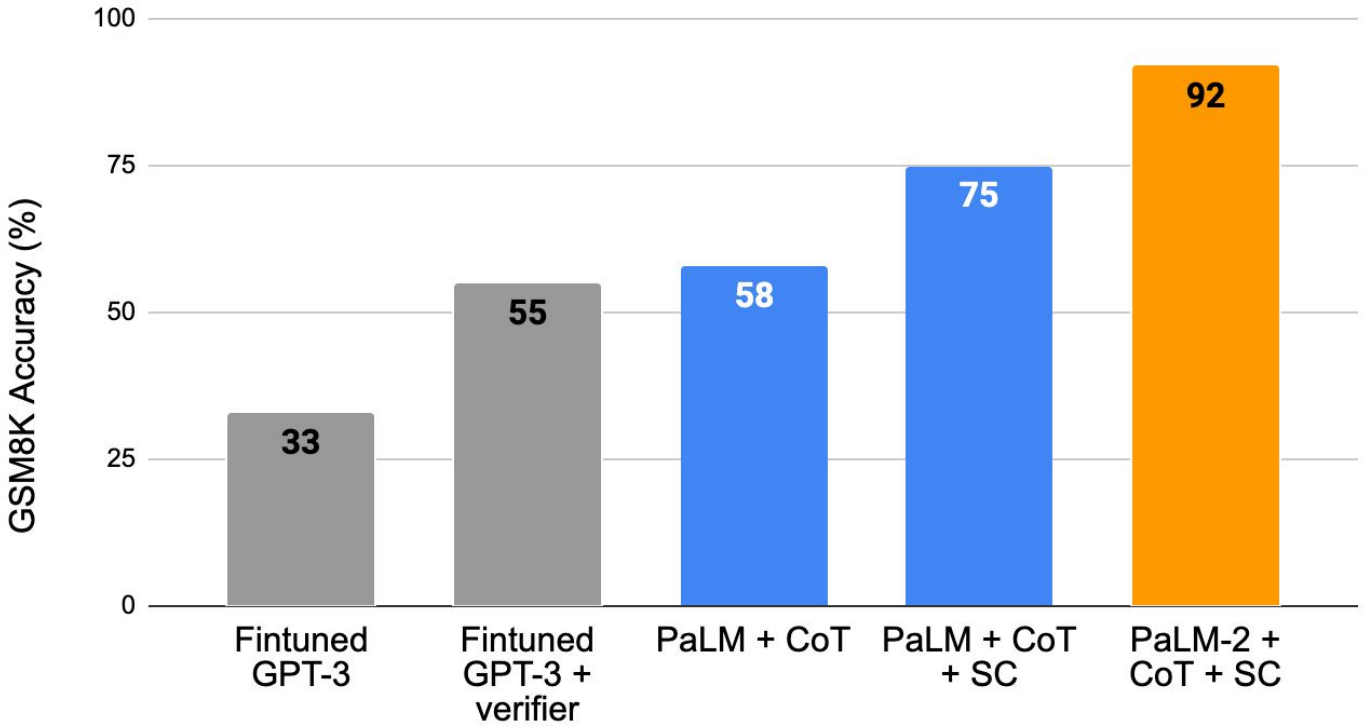
**Response 1:** She has 16 - 3 - 4 = 9 eggs left. So she makes $2 * 9 = $**18** per day.

**Response 2**: This means she she sells the remainder for $2 * (16 - 4 - 3) = $**26** per day.

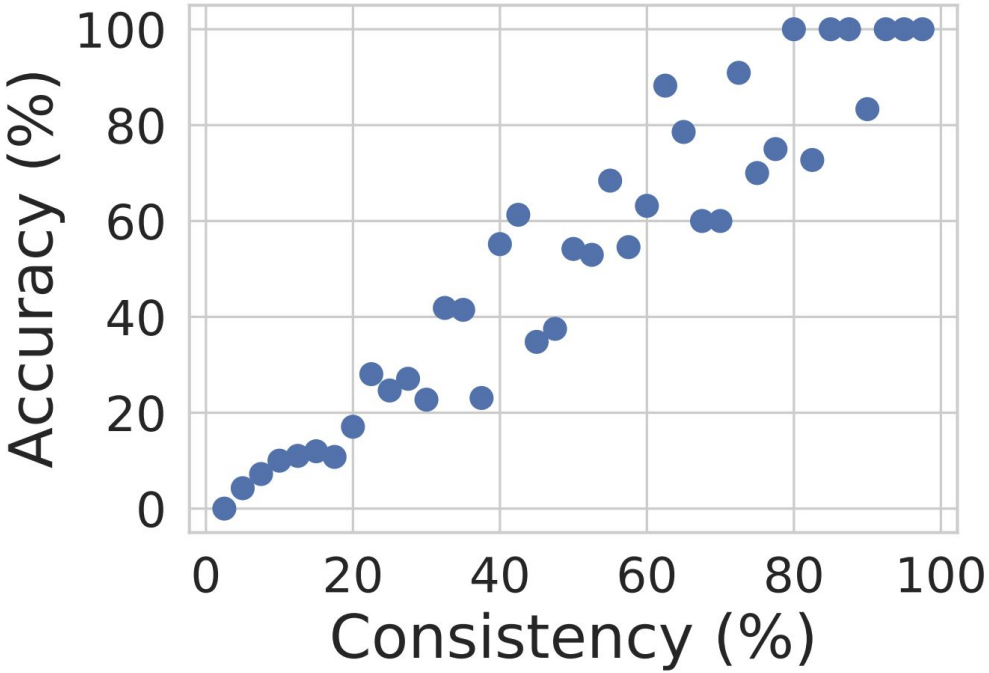
**Response 3**: She eats 3 for breakfast, so she has 16 - 3 = 13 left. Then she bakes muffins, so she has 13 - 4 = 9 eggs left. So she has 9 eggs * $2 = $**18**.

Most frequent answer is: 18
(Not most frequent reasoning path!)

# Crushed GSM8K SOTA with only 8 examples



# More consistent, more likely to be correct!



# How about free-from answers?

## Universal Self-Consistency (USC)

Ask LLMs to self-select the most consistent answer!

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, Denny Zhou. Universal Self-Consistency for Large Language Model Generation. arXiv:2311.17311 [cs.CL], 2023.

**[Question]** Where do people drink less coffee than they do in Mexico?

**Response 1**: ... Some examples include Japan, China and the United Kingdom.

It is important to note that coffee consumption can vary among individuals within these countries, and preferences can change depending on different factors such as…

**Response 2**: People in countries like Japan, China, and India typically drink less coffee than they do in Mexico...

**Response 3**: There are several countries where people generally drink less coffee compared to Mexico. Some of these countries include:

1. Japan:...
2. China...
3. Saudi Arabia...
4. India...

...

The most consistent response: 2

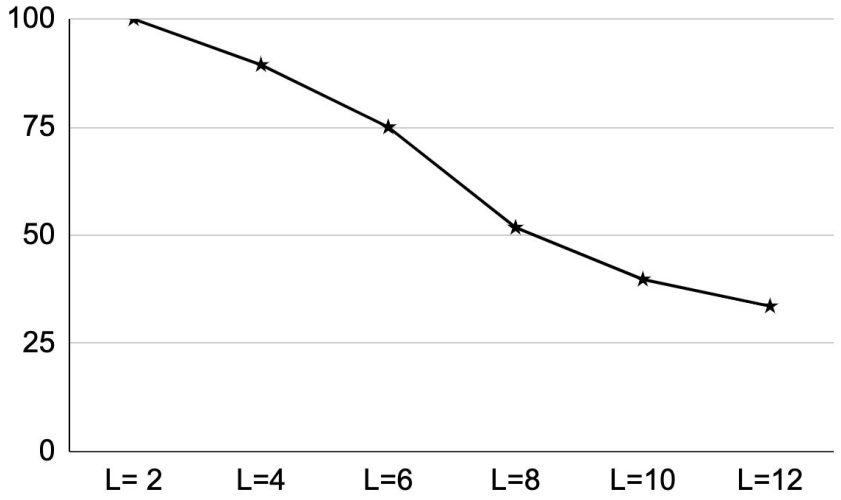
# Least-to-Most Prompting

Enable easy-to-hard generalization by decomposition

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, Ed Chi. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. **ICLR 2023**.

# Revisit Last-Letter-Concatenation

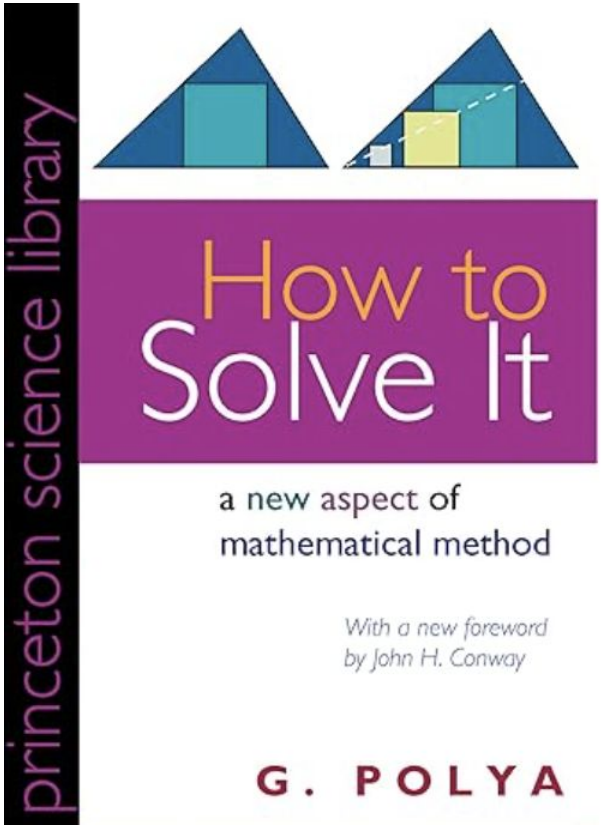
CoT prompting fails when the test lists are longer (length = 4 and above) than the few-shot examples (length = 2 or 3)



# Key Idea: Incorporating Reasoning Strategies

## Decomposing + Sequential Solving

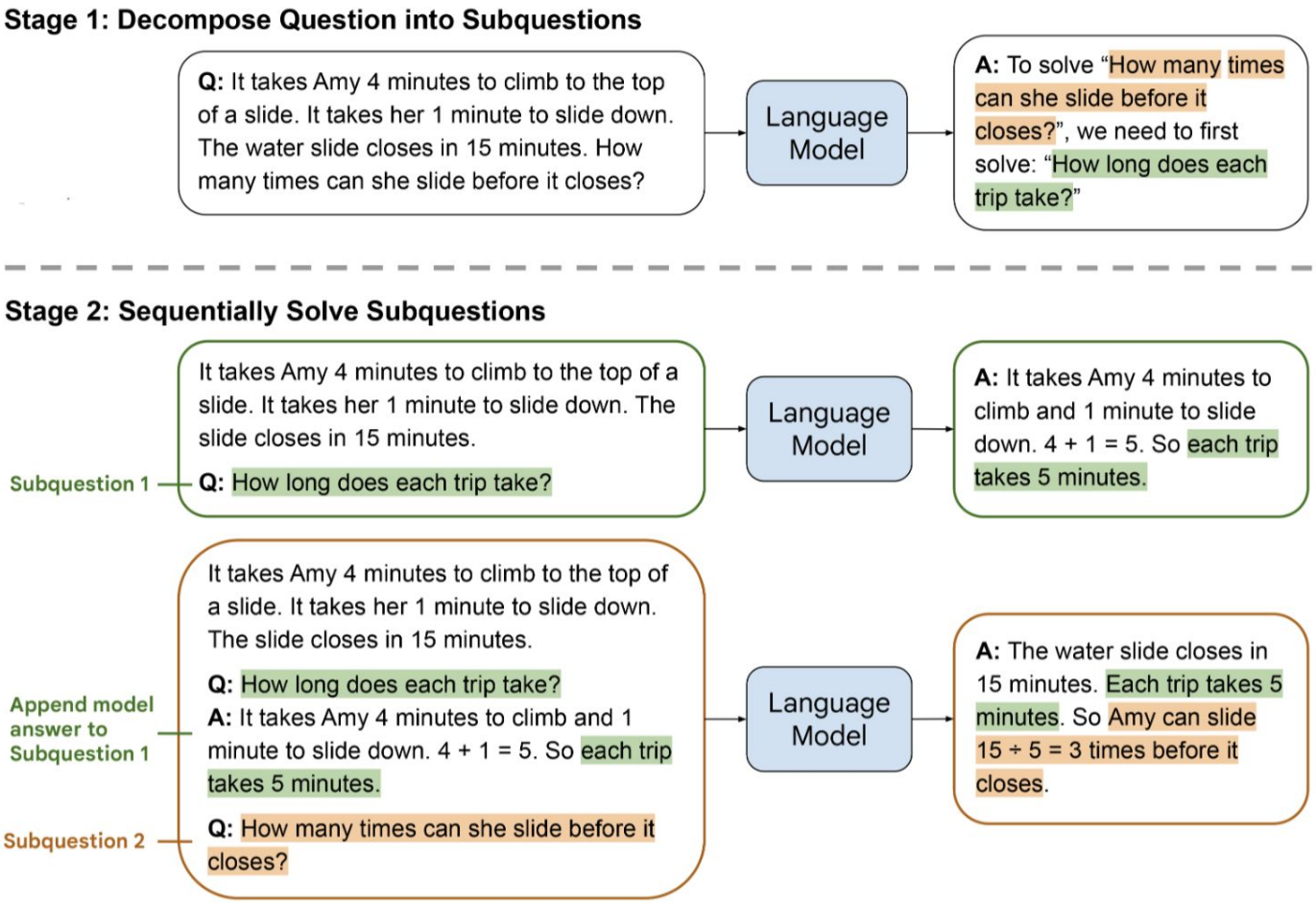
1. **Decompose a complex problem** into a list of easier subproblems
2. **Solve these subproblems** one by one (from least to most complex)




**Decomposing and recombining** are important operations of the mind.

differently. You decompose the whole into its parts, and you recombine the parts into a more or less different whole.

1. If you go into detail you may lose yourself in details. Too many or too minute particulars are a burden on the mind. They may prevent you from giving sufficient attention to the main point, or even from seeing the main point at all. Think of the man who cannot see the forest for the trees.


Stage 1: Decompose Question into Subquestions
Q: It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The water slide closes in 15 minutes. How many times can she slide before it closes?
Language Model
A: To solve "How many times can she slide before it closes?", we need to first solve: "How long does each trip take?"
Stage 2: Sequentially Solve Subquestions
It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.
Subquestion 1
Q: How long does each trip take?
Language Model
A: It takes Amy 4 minutes to climb and 1 minute to slide down. 4 + 1 = 5. So each trip takes 5 minutes.
It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.
Q: How long does each trip take?
Append model answer to Subquestion 1
A: It takes Amy 4 minutes to climb and 1 minute to slide down. 4 + 1 = 5. So each trip takes 5 minutes.
Subquestion 2
Q: How many times can she slide before it closes?
Language Model
A: The water slide closes in 15 minutes. Each trip takes 5 minutes. So Amy can slide 15 ÷ 5 = 3 times before it closes.

# Last-Letter-Concatenation (Length Generalization)

| | $L = 4$ | $L = 6$ | $L = 8$ | $L = 10$ | $L = 12$ |
|---|---|---|---|---|---|
| Standard prompting | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Chain-of-Thought | 84.2 | 69.2 | 50.2 | 39.8 | 31.8 |
| Least-to-Most | **94.0** | **88.4** | **83.0** | **76.4** | **74.0** |

---

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".

---

# SCAN (Compositional Generalization)

| Method | Standard prompting | Chain-of-Thought | Least-to-Most |
|---|---|---|---|
| code-davinci-002 | 16.7 | 16.2 | **99.7** |
| text-davinci-002 | 6.0 | 0.0 | **76.0** |
| code-davinci-001 | 0.4 | 0.0 | **60.7** |

SCAN is a task to translate natural language commands to action sequences

| **Command** | **Action Sequence** |
|---|---|
| "look thrice after jump" | JUMP LOOK LOOK LOOK |
| "run left and walk" | TURN_LEFT RUN WALK |
| "look opposite right" | TURN_RIGHT TURN_RIGHT LOOK |

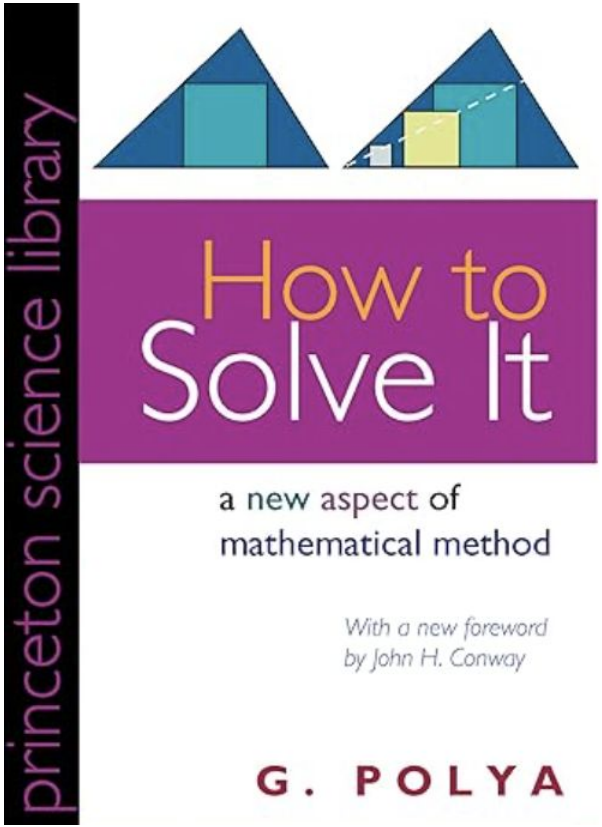
# CFQ (Compositional Generalization): Text-to-Code

| | MCD1 | MCD2 | MCD3 | Ave. |
|---|---|---|---|---|
| **Fully Supervised** | | | | |
| T5-base (Herzig et al., 2021) | 58.5 | 27.0 | 18.4 | 34.6 |
| T5-large (Herzig et al., 2021) | 65.1 | 32.3 | 25.4 | 40.9 |
| T5-3B (Herzig et al., 2021) | 65.0 | 41.0 | 42.6 | 49.5 |
| HPD (Guo et al., 2020) | 79.6 | 59.6 | 67.8 | 69.0 |
| T5-base + IR (Herzig et al., 2021) | 85.8 | 64.0 | 53.6 | 67.8 |
| T5-large + IR (Herzig et al., 2021) | 88.6 | 79.2 | 72.7 | 80.2 |
| T5-3B + IR (Herzig et al., 2021) | 88.4 | 85.3 | 77.9 | 83.9 |
| LeAR (Liu et al., 2021) | 91.7 | 89.2 | 91.7 | 90.9 |
| **Prompting** | | | | |
| (Ours) Dynamic Least-to-Most | **94.3** | **95.3** | **95.5** | **95.0** |

Using only 1% data!

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, Denny Zhou. Compositional Semantic Parsing with Large Language Models. **ICLR 2023**.

Any other reasoning strategies?




**13. Looking back.** Even fairly good students, when they have obtained the solution of the problem and written down neatly the argument, shut their books and look for something else. Doing so, they miss an important and instructive phase of the work. By looking back at the completed solution, by reconsidering and reexamining the result and the path that led to it, they could consoli-

The student has now carried through his plan. He has written down the solution, checking each step. Thus, he should have good reasons to believe that his solution is correct. Nevertheless, errors are always possible, especially if the argument is long and involved. Hence, verifications are desirable. Especially, if there is some rapid and intuitive procedure to test either the result or the argument, it should not be overlooked. *Can you check the result? Can you check the argument?*

# Teaching Large Language Models to Self-Debug

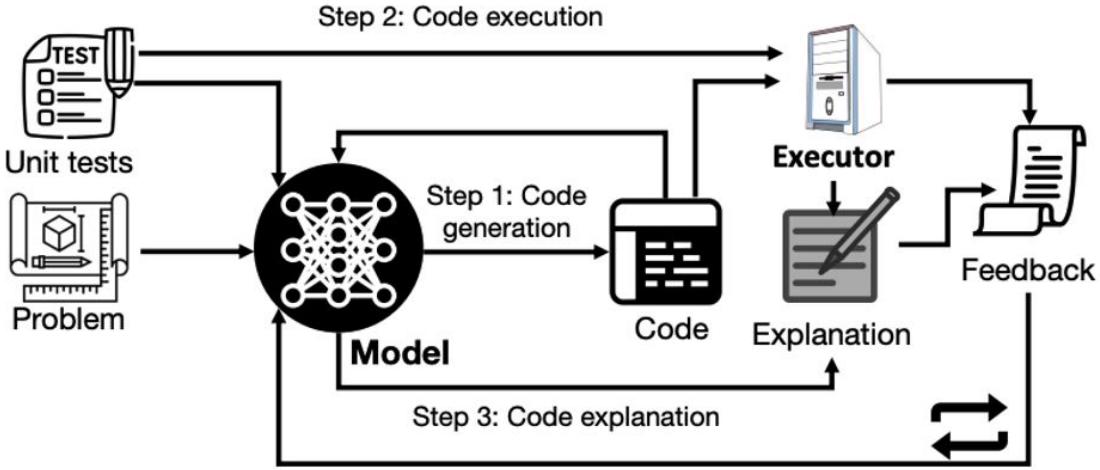



Figure 1: SELF-DEBUGGING for iterative debugging using a large language model. At each debugging step, the model first generates new code, then the code is executed and the model explains the code. The code explanation along with the execution results constitute the feedback message, which is then sent back to the model to perform more debugging steps. When unit tests are not available, the feedback can be purely based on code explanation.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, Denny Zhou. Teaching Large Language Models to Self-Debug. arXiv:2302.00093 [cs.CL], 2023. ICLR 2024.

Is it possible to make one common prompt for all tasks?

**Yes!**

**Key Idea**

Making a big prompt by combining prompts from different tasks, and then applying it to new tasks

**Implementation**

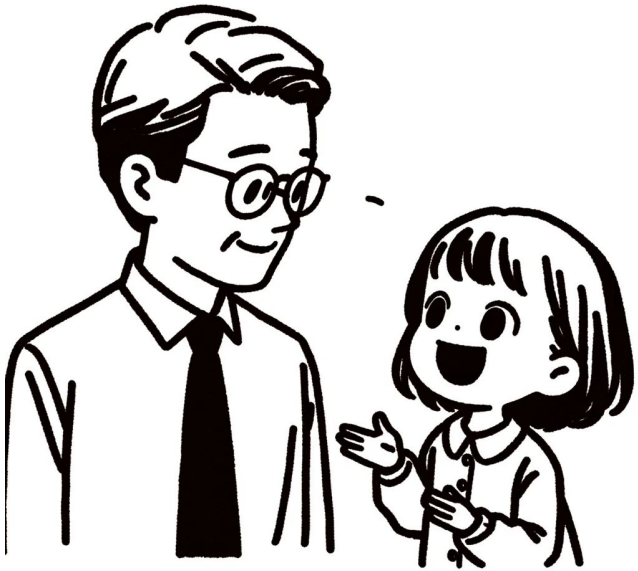
Too big to load? **“Store” them in “weights”** ! (Instruction tuning)

Instruction tuning: using a huge many-shot prompt stored in the model weights to zero-shot new problems

This explains why LLM chat can answer any user query without requiring inputting demonstration examples!

Zero-shot is more convenient than few-shot, but performs worse. Can we make zero-shot match few-shot performance?

“If you can have a superpower what would it be?”

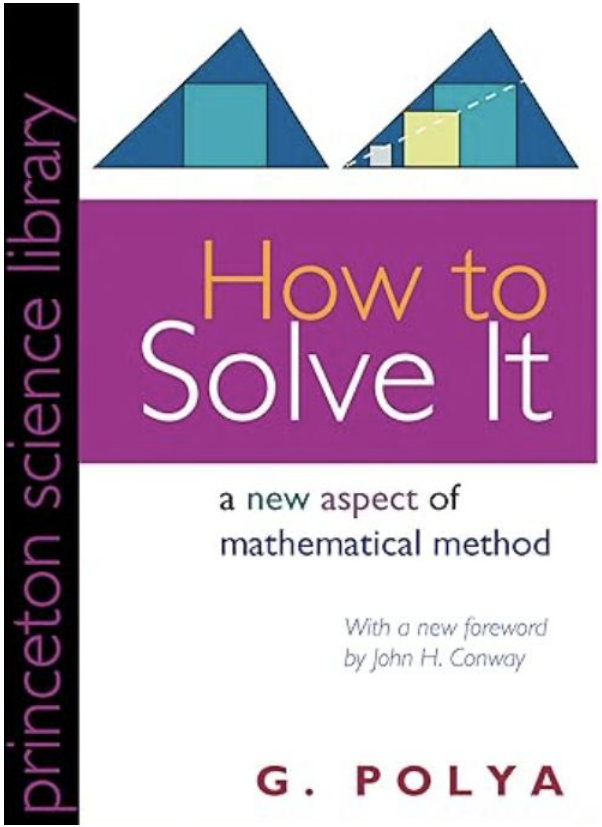


“I want to have a superpower that can let me have whatever power I want any time!”

When instruction-tuned LLMs can “follow instructions”, why not just “instruct” them to self-generate related examples?

# LLMs as Analogical Reasoners

Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H. Chi, Denny Zhou. Large Language Models as Analogical Reasoners. ICLR 2024.



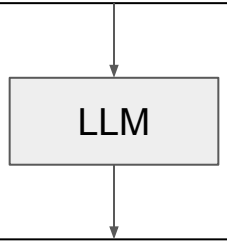

**Do you know a related problem?** We can scarcely imagine a problem absolutely new, unlike and unrelated to any formerly solved problem; but, if such a problem could exist, it would be insoluble. In fact, when solving a problem, we always profit from previously solved problems, using their result, or their method, or the experience we acquired solving them. And, of course, the problems from which we profit must be in some way related to our present problem. Hence the question: *Do you know a related problem?*

There is usually no difficulty at all in recalling formerly solved problems which are more or less related

“A mathematician is a person who can find analogies between theorems; a better mathematician is one who can see analogies between proofs and the best mathematician can notice analogies between theories. One can imagine that the **ultimate mathematician is one who can see analogies between analogies**.”

— Stefan Banach

**What is the area of the square with the four vertices at (-2, 2), (2, -2), (-2, -6), and (-6, -2)?** Recall a related problem, and then solve this one.




**The related problem is finding the distance between two points on a coordinate plane**. To find the distance between two points, we use the distance formula: d = √((x2-x1)^2 + (y2-y1)^2).

In this problem, we have four points: (-2, 2), (2, -2), (-2, -6), and (-6, -2). Let's choose (-2, 2) and (2, -2) to find the distance d between them: d = √((2-(-2))^2 + (-2-2)^2) = √(4^2 + (-4)^2) = √(16 + 16) = √32. We know that the distance between two opposite sides of a square is equal. So, the length of each side of the square is √32. To find the area of the square, we square the length of one side: Area = (√32)^2 = 32.

# Our zero-shot outperforms few-shot!

| Prompting Method | GSM8K Accuracy | | | MATH Accuracy | |
|---|---|---|---|---|---|
| | GPT3.5-turbo | text-davinci-003 | PaLM2 | GPT3.5-turbo | PaLM2 |
| 0-shot | 75.0% | 14.8% | 60.8% | 33.0% | 27.1% |
| 0-shot CoT | 75.8% | 50.3% | 78.2% | 33.9% | 29.8% |
| 5-shot CoT | 76.7% | 54.0% | 80.7% | 34.9% | 34.3% |
| **Ours: Self-generated Exemplars** | **77.8%** | **61.0%**† | **81.7%** | **37.3%** | **34.8%** |

0-shot CoT: “Let’s think step by step” proposed in Kojima et al. “Large language models are zero-shot reasoners." NeurIPS (2022).

# Our zero-shot outperforms few-shot!

| **Prompting Method** | Word sorting | Logical deduction five objects | Temporal sequences | Reasoning about colored objects | Formal fallacies |
|---|---|---|---|---|---|
| 0-shot | 66.8% | 30.0% | 40.4% | 50.4% | 53.6% |
| 0-shot CoT | 67.6% | 35.2% | 44.8% | 61.6% | 55.6% |
| 3-shot CoT | 68.4% | 36.4% | **58.0%** | 62.0% | 55.6% |
| **Ours: Self-generated Exemplars** | **75.2%** | **41.6%** | 57.6% | **68.0%** | **58.8%** |

0-shot CoT: “Let’s think step by step” proposed in Kojima et al. “Large language models are zero-shot reasoners." NeurIPS (2022).

google/**BIG-bench**

# Our zero-shot outperforms few-shot!

| Prompting Method | GPT3.5-turbo-16k | | GPT4 | |
|---|---|---|---|---|
| | Acc@1 | Acc@10 | Acc@1 | Acc@10 |
| 0-shot | 8% | 24% | 16% | 30% |
| 0-shot CoT | 9% | 27% | 16% | 29% |
| 3-shot CoT | 11% | 27% | 17% | 31% |
| **Ours: Self-generated Exemplars** | 13% | 25% | 17% | 32% |
| **Ours: Self-generated Knowledge + Exemplars** | **15%** | **29%** | **19%** | **37%** |

0-shot CoT: “Let’s think step by step” proposed in Kojima et al. “Large language models are zero-shot reasoners." NeurIPS (2022).

Why outperforms? LLMs adaptively generate different related exemples/knowledge for different problems!

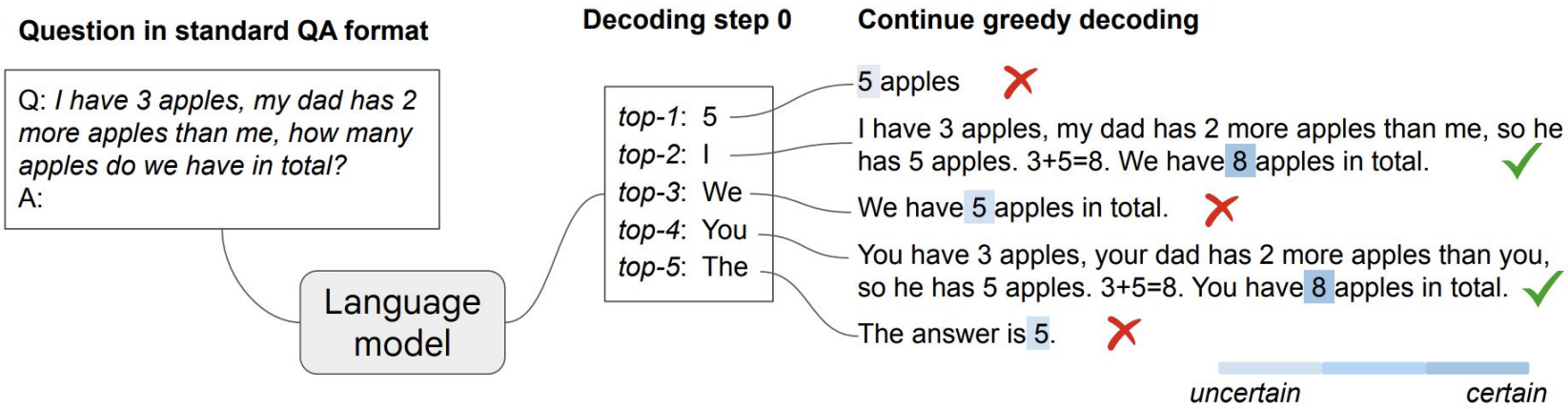
**zero-shot**  **few-shot**

Do we really have to prompt LLMs to generate step by step reasoning?

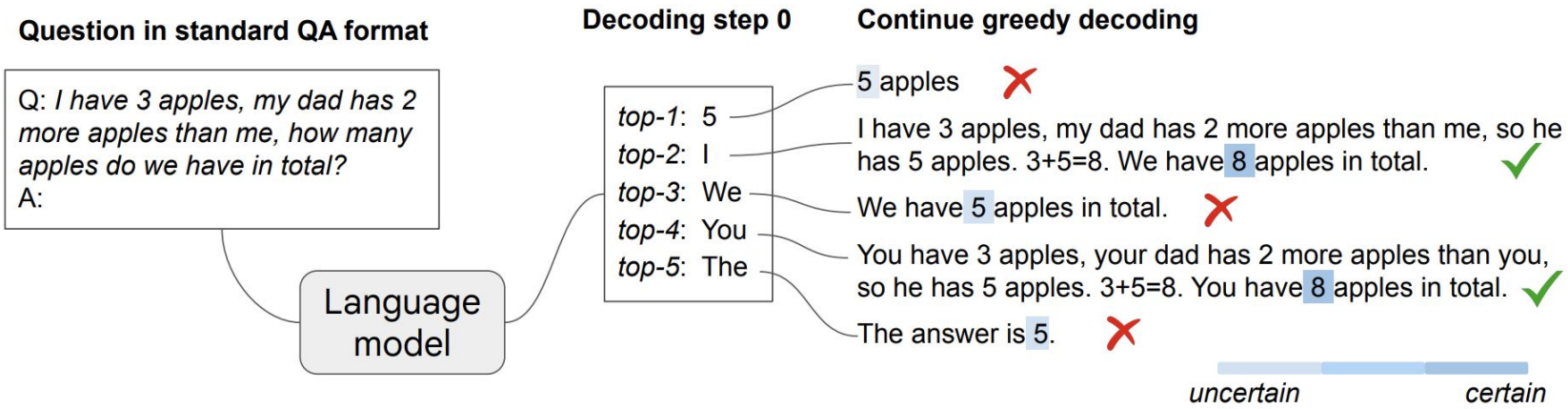
**No**

# Chain-of-Thought Reasoning without Prompting

Xuezhi Wang and Denny Zhou. Chain-of-Thought Reasoning Without Prompting. arXiv preprint arXiv:2402.10200 (2024).




**Key observations:** (1) Pre-trained LLMs have had responses with step-by-step reasoning among the generations started with the top-k tokens; (2) Higher confidence in decoding the final answer when a step-by-step reasoning path is present.




# Chain-of-Thought Decoding

# Chain-of-Thought Decoding

| | | PaLM-2 Pre-trained | | | | PaLM-2 Inst-tuned |
|---|---|---|---|---|---|---|
| | | X-Small | Small | Medium | Large | Large |
| GSM8K | greedy | 9.0 | 14.3 | 21.0 | 34.8 | 67.8 |
| | CoT-decoding | **17.7** (+8.7) | **35.1** (+20.8) | **39.7** (+18.7) | **61.5** (+26.7) | **81.3** (+13.5) |
| MultiArith | greedy | 7.5 | 15.8 | 36.8 | 75.0 | 93.7 |
| | CoT-decoding | **34.8** (+27.3) | **43.5** (+27.7) | **52.5** (+15.7) | **86.7** (+11.7) | **98.7** (+5.0) |

# Chain-of-Thought Decoding

| | PaLM-2 Pre-trained | | |
|---|---|---|---|
| | Small | Medium | Large |
| greedy | 61.0 | 55.0 | 57.0 |
| CoT-decoding | **65.0** (+4.0) | **89.0** (+34.0) | **95.0** (+38.0) |

**[Year Parity]** *Was Nicolas Cage born in an even or odd year?*
**Greedy path:**
$k = 0$: Nicolas Cage was born in an **odd** year. (0.117)
**Alternative top-$k$ paths:**
$k = 1$: **Even** (0.207)
$k = 2$: **Odd** (0.198)
$k = 3$: 1964, an **even** year. (0.949)
$k = 4$: He was born in an **even** year. (0.0)
...
$k = 7$: Cage was born in 1964, an **even** year. (0.978)

# Limitations

# LLMs Can Be Easily Distracted by Irreverent Context

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. Large Language Models Can Be Easily Distracted by Irrelevant Context. **ICML 2023**.

# Humans may be easily distracted by irrelevant context

Psychology studies show that irrelevant information may significantly decrease some children and even adults problem-solving accuracy

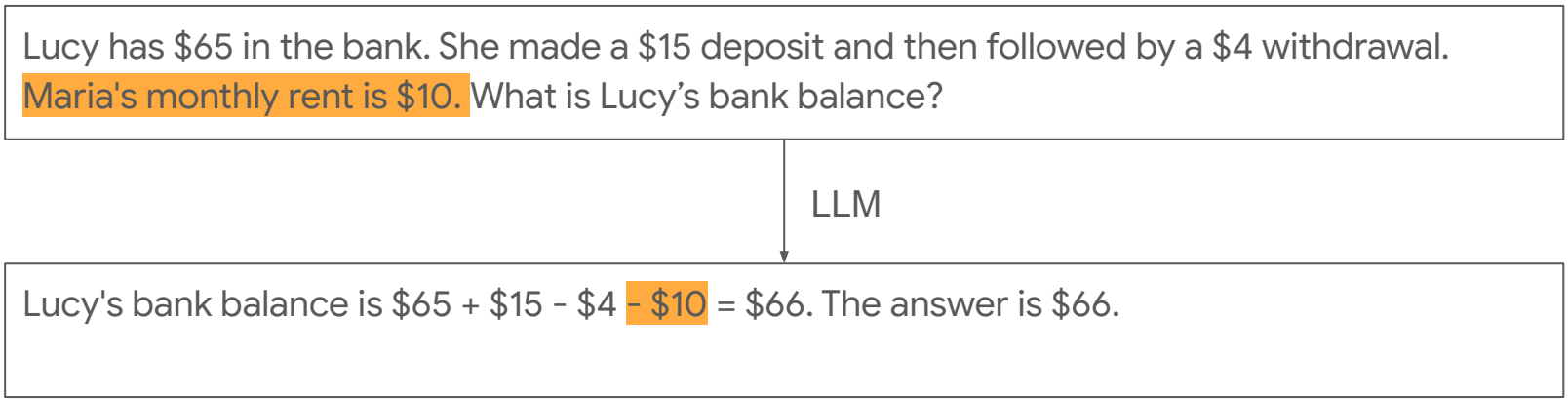
… inattentive children's difficulties in problem solving are partially due to an inability to inhibit irrelevant information …

Marzocchi, G.M., Lucangeli, D., De Meo, T., Fini, F. and Cornoldi, C., 2002. The disturbing effect of irrelevant information on arithmetic problem solving in inattentive children. *Developmental neuropsychology*, *21*(1), pp.73-92.

Does this observation hold for LLMs?

# LLMs can be easily distracted by irreverent context

Adding irrelevant contexts to GSM8K leads to 20+ points performance drop



# LLMs can be easily distracted by irreverent context

**Prompting LLMs to Ignore irrelevant context can take some performance back**

Lucy has $65 in the bank. She made a $15 deposit and then followed by a $4 withdrawal. Maria's monthly rent is $10. What is Lucy's bank balance? Ignore irrelevant context.

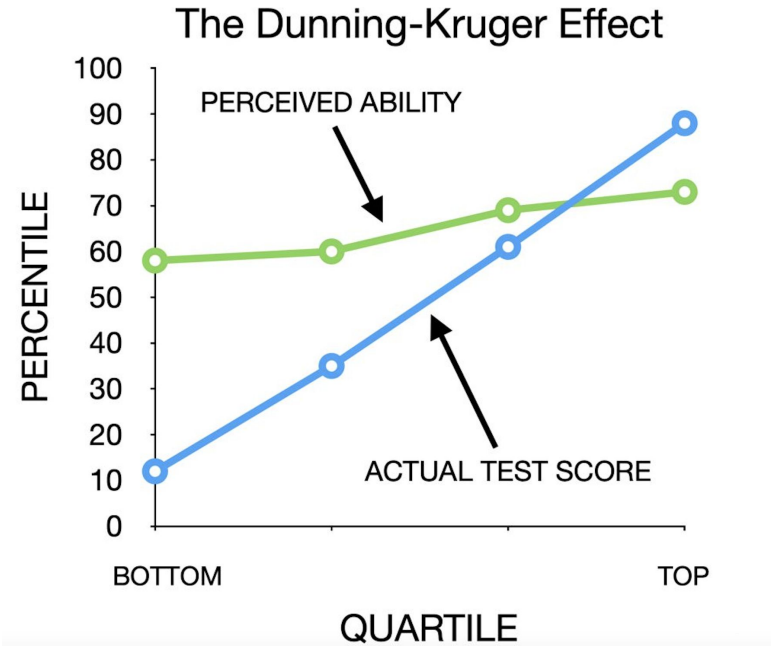

Lucy has $65 in the bank. After making a $15 deposit and then a $4 withdrawal, her bank balance is $76. Maria's monthly rent is not relevant to this question, so it can be ignored. The answer is $76.

# LLMs Cannot Self-Correct Reasoning Yet

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet. ICLR 2024.

“People who are unskilled in these domains suffer a dual burden: Not only do these people reach erroneous conclusions and make unfortunate choices, but their incompetence robs them of the metacognitive ability to realize it.”

J Kruger, D Dunning. Unskilled and unaware of it: how difficulties in recognizing one's own incompetence lead to inflated self-assessments. J Pers Soc Psychol. 1999 Dec;77(6):1121-34.




**WE ARE ALL CONFIDENT IDIOTS** DAVID DUNNING

# Self-correction only leads to worse results

| | | # calls | GSM8K | CommonSenseQA | HotpotQA |
|---|---|---|---|---|---|
| GPT-3.5 | Standard Prompting | 1 | **75.9** | **75.8** | **26.0** |
| | Self-Correct (round 1) | 3 | 75.1 | 38.1 | 25.0 |
| | Self-Correct (round 2) | 5 | 74.7 | 41.8 | 25.0 |
| GPT-4 | Standard Prompting | 1 | **95.5** | **82.0** | **49.0** |
| | Self-Correct (round 1) | 3 | 91.5 | 79.5 | **49.0** |
| | Self-Correct (round 2) | 5 | 89.0 | 80.0 | 43.0 |

Too difficult for LLMs to self realize their mistakes

# Reported Improvements Need Oracle Labels!

| | | GSM8K | CommonSenseQA | HotpotQA |
|---|---|---|---|---|
| GPT-3.5 | Standard Prompting | 75.9 | 75.8 | 26.0 |
| | Self-Correct (Oracle) | 84.3 | 89.7 | 29.0 |
| GPT-4 | Standard Prompting | 95.5 | 82.0 | 49.0 |
| | Self-Correct (Oracle) | 97.5 | 85.5 | 59.0 |

Oracle: Let LLMs self correct only when the answer is wrong

# Multi-Agent Debate? Worse than Self-Consistency!

| | # responses | GSM8K |
|---|---|---|
| Standard Prompting | 1 | 76.7 |
| Self-Consistency | 3 | 82.5 |
| Multi-Agent Debate (round 1) | 6 | 83.2 |
| Self-Consistency | 6 | 85.3 |
| Multi-Agent Debate (round 2) | 9 | 83.0 |
| Self-Consistency | 9 | **88.2** |

Need external oracle feedback to make LLM self-correction work. Self-debug naturally leverages unit tests in code generation tasks.

# Premise Order Matters in LLM Reasoning

Xinyun Chen, Ryan A Chi, Xuezhi Wang, Denny Zhou. Premise Order Matters in Reasoning with Large Language Models. arXiv preprint arXiv:2402.08939. 2024.

# Permuting premises in simple logic inference causes huge performance drops

LLMs achieve the best performance when the premise order aligns with the context required in intermediate reasoning steps

For simple logic inference tasks, **permuting the premise order** causes performance drops which can be more than 30 points:

1. Presenting “If A then B” before “If B then C” generally achieves a higher accuracy compared to the reversed order.
2. The performance gap becomes more significant when the number of premises increases.

# Theoretic Analysis

“There is nothing more practical than a good theory.”

— Kurt Lewin

# What learning algorithm is in-context learning?

**Transformer-based in-context learners implement standard learning algorithms implicitly**, by encoding smaller models in their activations, and updating these implicit models as new examples appear

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? Investigations with linear models. **ICLR 2023**.

# Chain of thought empowers transformers to solve inherently serial problems

For tasks that vanilla transformer either requires a huge depth to solve or cannot solve the tasks at all, **transformer generating intermediate steps** can solve these tasks as long as the depth exceeds a small threshold

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of Thought Empowers Transformers to Solve Inherently Serial Problems. ICLR 2024.

# Summary

**Key Ideas**

- Step by step/chain-of-thought
- Self-consistency/marginalization
- Incorporating reasoning strategies
  - Decompose(least-to-most)
  - Self-examine (self-debug)
  - Analogical reasoning

**Limitations**

- Easily distracted by irrelevant context
- Cannot self-correct reasoning yet
- Premise order matters

Few-shot prompting ⟶ Zero-shot prompting ⟶ No prompting

# What is next?

“If I were given one hour to save the planet, I would spend 59 minutes defining the problem and one minute resolving it.”

— Albert Einstein

https://colmweb.org

# Conference on Language Modeling (COLM)

# Thank You

https://twitter.com/denny_zhou

https://dennyzhou.github.io/

https://scholar.google.com/citations?user=UwLsYw8AAAAJ&hl=en