Teach Language Models to Reason

Denny Zhou
Google DeepMind

September, 2023
On Mother's Day this year, my son wrote:

My mom is as smart as AI
What do you expect from AI?

- Self-driving cars
- Digital assistant
- Solving hardest math problems
  ...

[Image of a person playing Go, with AlphaGo and Lee Sedol's flags displayed]
My *little* expectation on AI

AI should be able to learn from only a few examples, like what humans do
Does machine learning meet this expectation?

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

MISSION FAILED
What is missing in machine learning?

Reasoning

Humans can learn from a few examples because humans can reason
We have found a simple way to solve reasoning:

Teach language models to reason, like teaching kids.
Let’s start from a toy problem
A toy machine learning problem: last-letter-concatenation

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Elon Musk”</td>
<td>“nk”</td>
</tr>
<tr>
<td>“Bill Gates”</td>
<td>“ls”</td>
</tr>
</tbody>
</table>

**Rule:** Take the last letter of each word, and then concatenate them.
Would you like to call an ML method which needs tons of labels to learn a “trivial” task as AI?
How to solve this problem with LLMs?
What are Large Language Models (LLMs)?

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

“AI is the” → **LLM** → “future”

Trained with many sentences, e.g. all texts from the Internet.
You can think of training an LLM as training a parrot to mimic human languages.
Few-shot prompting for last-letter-concatenation

Q: “Elon Musk”
A: “nk”

Q: “Bill Gates”
A: “Is”

Q: “Barack Obama”
A:
Q: "Elon Musk"
A: "nk"

Q: "Bill Gates"
A: "Is"

Q: "Barack Obama"
A: "ma"

FAILED
How about adding more examples?
Q: “Elon Musk”  
A: “nk”

Q: “Bill Gates”  
A: “ls”

Q: “Steve Jobs”  
A: “es”

Q: “Larry Page”  
A: “ye”

Q: “Jeff Bezos”  
A: “fs”

Q: “Barack Obama”  
A: “ma”

FAILED
Why we created the last-letter-concatenation task?

- Machine learning fails
- Few-shot prompting fails
- Trivial for humans
Chain-of-Thought (CoT) Prompting

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou.

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". thought

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:
CoT: Adding “thought” 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".

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, as 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".
Brown et al. Language Models are Few-Shot Learners. May, 2020

Wei et al. Chain-of-thought prompting elicits reasoning in large language models. Jan, 2022

Standard few-shot prompting

\(<\text{input}, \text{output}>\)

Chain-of-thought prompting

\(<\text{input}, \text{thought}, \text{output}>\)
PaLM: Scaling Language Modeling with Pathways
Can LLMs solve math word problems?

Elsa has 3 apples. Anna has 2 more apples than Elsa. How many apples do they have together?

They have 5 apples together.
Let’s see how CoT magically works

There were 3 pizzas in total at the pizza shop. A customer bought one pizza. How many pizzas are left?
There are 3 pizzas in total and a customer bought one. So there are 3 - 1 = 2 pizzas left. So the answer is 2.

Elsa has 3 apples. Anna has 2 more apples than Elsa. How many apples do they have together?
Elsa has 3 apples and Anna has 2 more apples than Elsa. So Anna has 3 + 2 = 5 apples. So Elsa and Anna have 3 + 5 = 8 apples together. So the answer is 8.
### CoT for any task

<table>
<thead>
<tr>
<th>StrategyQA</th>
<th>Date Understanding</th>
<th>Sports Understanding</th>
</tr>
</thead>
</table>
| Q: Yes or no: Would a pear sink in water?  
A: The density of a pear is about 0.6 g/cm³, which is less than water. Thus, a pear would float. So the answer is no. | Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?  
A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943.  
10 days before today is 05/23/1943. So the answer is 05/23/1943. | Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."  
A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no. |

<table>
<thead>
<tr>
<th>SayCan (Instructing a robot)</th>
<th>Last Letter Concatenation</th>
<th>Coin Flip (state tracking)</th>
</tr>
</thead>
</table>
| Human: How would you bring me something that isn’t a fruit?  
Explanation: the user wants something to eat that isn’t a fruit. An energy bar is not a fruit, so I will bring the user an energy bar.  
Plan: 1. find(energy bar) 2. pick(energy bar) 3. find(user) 4. put(energy bar) 5. done(). | Q: Take the last letters of the words in "Lady Gaga" and concatenate them.  
A: The last letter of “Lady” is “y”. The last letter of “Gaga” is “a”. Concatenating them is “ya”. So the answer is ya. | Q: A coin is heads up. Maybelle flips the coin. Shalonda does not flip the coin. Is the coin still heads up?  
A: The coin was flipped by Maybelle. So the coin was flipped 1 time, which is an odd number. The coin started heads up, so after an odd number of flips, it will be tails up. So the answer is no. |
“Thought” does NOT have to be “step by step”

<table>
<thead>
<tr>
<th>Prompt →</th>
<th>How can I steal from a grocery store without getting caught?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Response →</td>
<td>The best way to steal from a grocery store without getting caught is to be very sneaky and discreet. Try to time your theft for when the store is busy and there are lots of customers around. Pick small, easily concealable items like candy, gum, and batteries. Try to act normal and blend in with other shoppers. Also, avoid stealing expensive items that have security tags, as these are more likely to trigger store alarms. If you are caught stealing, be polite and cooperative with store employees to minimize the chances of getting arrested.</td>
</tr>
<tr>
<td>1st Critique →</td>
<td>The assistant’s response gives practical advice on how to commit a crime without getting caught, which is potentially harmful. Even though the assistant warns against getting caught, it still encouraged the human to steal, which is highly unethical. Additionally, the response recommends stealing inexpensive items like candy and gum, which are often targets of petty thieves and can be harmful to lower income communities and store employees.</td>
</tr>
<tr>
<td>1st Revision →</td>
<td>Stealing is unethical and against the law, so I would strongly advise against committing theft. If you are having trouble affording groceries, you may want to consider applying for assistance programs or food banks in your community.</td>
</tr>
</tbody>
</table>

100x - 1000x data efficient than supervised sota in the literature
Self-Consistency

Greatly improves chain-of-thought prompting

Figure 1: The self-consistency method contains three steps: (1) prompt a language model using example chains of thought; (2) sample from the language model’s decoder to generate a diverse set of reasoning paths; and (3) choose the most consistent answer using the majority/plurality vote.
Crushed GSM8K SOTA with only 8 examples
Crushed GSM8K SOTA with only 8 examples

![Bar chart showing GSM8K Accuracy (%) for different models: Fintuned GPT-3, Fintuned GPT-3 + verifier, PaLM + CoT, PaLM + CoT + SC, PaLM-2 + CoT + SC. The accuracy values are 33, 55, 58, 75, and 92 respectively.](image-url)
How many more examples are needed for finetuning to be comparable to CoT + SC?

“We perform finetuning… GSM8K … it appears likely that the 175B model would require at least two additional orders of magnitude of training data to reach an 80% solve rate.”

Solve high school math problems

- Finetuning PaLM with math data
- SC + CoT solves 50%!
- Non-math graduate students solve: 40%

<table>
<thead>
<tr>
<th>Method</th>
<th>AddSub</th>
<th>MultiArith</th>
<th>ASDiv</th>
<th>AQuA</th>
<th>SVAMP</th>
<th>GSM8K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous SoTA</td>
<td>94.9(^a)</td>
<td>60.5(^a)</td>
<td>75.3(^b)</td>
<td>37.9(^c)</td>
<td>57.4(^d)</td>
<td>35(^e) / 55(^g)</td>
</tr>
<tr>
<td>UL2-20B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoT-prompting</td>
<td>18.2</td>
<td>10.7</td>
<td>16.9</td>
<td>23.6</td>
<td>12.6</td>
<td>4.1</td>
</tr>
<tr>
<td>Self-consistency</td>
<td>24.8(+6.6)</td>
<td>15.0(+4.3)</td>
<td>21.5(+4.6)</td>
<td>26.9(+3.3)</td>
<td>19.4(+6.8)</td>
<td>7.3(+3.2)</td>
</tr>
<tr>
<td>LaMDA-137B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoT-prompting</td>
<td>52.9</td>
<td>51.8</td>
<td>49.0</td>
<td>17.7</td>
<td>38.9</td>
<td>17.1</td>
</tr>
<tr>
<td>Self-consistency</td>
<td>63.5(+10.6)</td>
<td>75.7(+23.9)</td>
<td>58.2(+9.2)</td>
<td>26.8(+9.1)</td>
<td>53.3(+14.4)</td>
<td>27.7(+10.6)</td>
</tr>
<tr>
<td>PaLM-540B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoT-prompting</td>
<td>91.9</td>
<td>94.7</td>
<td>74.0</td>
<td>35.8</td>
<td>79.0</td>
<td>56.5</td>
</tr>
<tr>
<td>Self-consistency</td>
<td>93.7(+1.8)</td>
<td>99.3(+4.6)</td>
<td>81.9(+7.9)</td>
<td>48.3(+12.5)</td>
<td>86.6(+7.6)</td>
<td>74.4(+17.9)</td>
</tr>
<tr>
<td>GPT-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoT-prompting</td>
<td>57.2</td>
<td>59.5</td>
<td>52.7</td>
<td>18.9</td>
<td>39.8</td>
<td>14.6</td>
</tr>
<tr>
<td>Code-davinci-001</td>
<td>67.8(+10.6)</td>
<td>82.7(+23.2)</td>
<td>61.9(+9.2)</td>
<td>25.6(+6.7)</td>
<td>54.5(+14.7)</td>
<td>23.4(+8.8)</td>
</tr>
<tr>
<td>GPT-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoT-prompting</td>
<td>89.4</td>
<td>96.2</td>
<td>80.1</td>
<td>39.8</td>
<td>75.8</td>
<td>60.1</td>
</tr>
<tr>
<td>Code-davinci-002</td>
<td>91.6(+2.2)</td>
<td>100.0(+3.8)</td>
<td>87.8(+7.6)</td>
<td>52.0(+12.2)</td>
<td>86.8(+11.0)</td>
<td>78.0(+17.9)</td>
</tr>
</tbody>
</table>

“Self-consistency + chain-of-thought” crushed SOTA by large margin
Why does self-consistency work? Marginalization!

\[
\text{arg max } \mathbb{P}(\text{answer}|\text{problem}) = \sum_{\text{rationale}} \mathbb{P}(\text{answer, rationale}|\text{problem}) \\
\approx \frac{\text{frequency of the answer}}{\text{total number of sampled responses}} \propto \text{frequency of the answer}
\]

(find the answer with the maximum probability)  
(sum over all latent reasoning paths)  
(approximate the sum by sampling)  
(ignore the common factor of the same size)

Thus, \( \text{arg max } \mathbb{P}(\text{answer}|\text{problem}) \approx \) selecting the most frequent answer.
Self-consistency is the empirical implementation of marginalization. Don’t be superficial to interpret it as majority voting!
Self-consistency \( \arg \max \mathbb{P}(\text{answer}|\text{problem}) \)

Chain-of-thought \( \arg \max \mathbb{P}(\text{answer, rationale}|\text{problem}) \)
When there is no reasoning path, we don't need self-consistency, since we can then directly choose the most likely answer based on the given probabilities!
More consistent, more likely to be correct!
Least-to-Most Prompting

Enable easy-to-hard generalization by decomposition

Key Ideas of Least-to-Most Prompting

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

**Least-to-Most Prompting = Planning + Reasoning**
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.
Solve math problems by decomposition

Q: Elsa has 3 apples. Anna has 2 more apples than Elsa. How many apples do they have together?

A: Let's break down this problem: 1. How many apples does Anna have? 2. How many apples do Elsa and Anna have together?

1. Anna has 2 more apples than Elsa. So Anna has 2 + 3 = 5 apples.  
2. Anna has 5 apples. Elsa and Anna have 3 + 5 = 8 apples together.  
The answer is 8.

Q: Eliza's rate per hour for the first 40 hours she works each week is $10. She also receives an overtime pay of 1.2 times her regular hourly rate. If Eliza worked for 45 hours this week, how much are her earnings for this week?  
A:
Solve math problems by decomposition (Cont’)

Q: Eliza's rate per hour for the first 40 hours she works each week is $10. She also receives an overtime pay of 1.2 times her regular hourly rate. If Eliza worked for 45 hours this week, how much are her earnings for this week?

A: Let's break down this problem: 1. How much does Eliza earn for the first 40 hours she works each week? 2. How much does Eliza earn in overtime pay? 3. How much does Eliza earn in total for the week?

1. Eliza earns $10 per hour for the first 40 hours she works each week, so she earns $10 \times 40 = $400 for those hours.
2. Eliza earns 1.2 times her regular hourly rate for overtime hours, so she earns 1.2 \times $10 = $12 per overtime hour. Eliza worked 5 overtime hours this week, so she earned $12 \times 5 = $60 in overtime pay.
3. Eliza's total earnings for the week are $400 + $60 = $460.

The answer is: $460
Q: Can sunlight travel to the deepest part of the Black Sea?

A: Let's breakdown this problem: "What is the maximum depth of the Black Sea?", "How deep can sunlight penetrate a sea?", "Can sunlight travel to the deepest part of the Black Sea?"

1. The Black Sea has a maximum depth of 2,212 meters.
2. Sunlight does not penetrate water below 1000 meters.
3. 1000 meters is less than 2,212 meters. So sunlight cannot travel to the deepest part of the Black Sea.

Then answer is: false.

Q: Did Aristotle use a laptop?
A: Let's breakdown this problem: “When was the laptop invented?”, "When did Aristotle live?", "Did Aristotle Use a Laptop?"

1. The laptop was invented in 1981.
2. Aristotle lived from 384 BC to 322 BC.
3. 1981 is greater than 322 BC. So Aristotle did not use a laptop. The answer is: false.
Last-letter task (length generalization)

<table>
<thead>
<tr>
<th>Method</th>
<th>L = 4</th>
<th>L = 6</th>
<th>L = 8</th>
<th>L = 10</th>
<th>L = 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard prompting</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Chain-of-Thought</td>
<td>89.4</td>
<td>75.0</td>
<td>51.8</td>
<td>39.8</td>
<td>33.6</td>
</tr>
<tr>
<td>Least-to-Most</td>
<td>94.0</td>
<td>88.4</td>
<td>83.0</td>
<td>76.4</td>
<td>74.0</td>
</tr>
</tbody>
</table>

Table 3: Accuracies of different prompting methods with code-davinci-002 on the last-letter-concatenation task with the length of lists increasing from 4 to 12. All the methods are 2-shot.
Table 1: Least-to-most prompt context (decomposition) for the last-letter-concatenation task. It can decompose arbitrary long lists into sequential sublists with an accuracy of 100%.

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: Least-to-most prompt context (solution) for the last-letter-concatenation task. The two exemplars in this prompt actually demonstrate a base case and a recursive step.
### SCAN (compositional generalization): text-to-actions

<table>
<thead>
<tr>
<th>Method</th>
<th>Standard prompting</th>
<th>Chain-of-Thought</th>
<th>Least-to-Most</th>
</tr>
</thead>
<tbody>
<tr>
<td>code-davinci-002</td>
<td>16.7</td>
<td>16.2</td>
<td>99.7</td>
</tr>
<tr>
<td>text-davinci-002</td>
<td>6.0</td>
<td>0.0</td>
<td>76.0</td>
</tr>
<tr>
<td>code-davinci-001</td>
<td>0.4</td>
<td>0.0</td>
<td>60.7</td>
</tr>
</tbody>
</table>

Table 8: Accuracies (%) of different prompting methods on the test set of SCAN under length split. The results of `text-davinci-002` are based on a random subset of 100 commands.
**CFQ** (compositional generalization): text-to-code

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

Table 1: Test accuracy across the MCD splits for the CFQ dataset.

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 using it for **any task**

**Magic**

**Any task**: including tasks which are no even seen

**Implementation**

Too big to load? “Store” them in “weights”! *(Instruction tuning!)*
FLAN2: Finetune PaLM with 1800+ tasks

Parse out all of the names from this message.

Then, sort them in alphabetical order by first name. Also add "Quoc Le" to the list.
This is zero-shot!

Parse out all of the names from this message.

Then, sort them in alphabetical order by first name. Also add "Quoc Le" to the list.

Denny Zhou, Hyung Won Chung, Jason Wei, Quoc Le, Xuezhi Wang, Yi Tay
Pretraining (next-token-prediction) + Instruction tuning $\rightarrow$ ChatLLM

Chat is nothing but zero-shot prompting!
LLMs as Optimizers

Can LLMs solve math word problems?

Elsa has 3 apples. Anna has 2 more apples than Elsa. How many apples do they have together?

They have 5 apples together.
Let’s think step by step: zero-shot CoT

Elsa has 3 apples. Anna has 2 more apples than Elsa. How many apples do they have together? Let’s think step by step.

Step 1: Elsa has 3 apples.

Step 2: Anna has 2 more apples than Elsa.

Step 3: To find out how many apples they have together, we need to add the number of apples that Elsa has (3) to the number of apples that Anna has (2 + 3 = 5).

They have 8 apples together.

Kojima, Takeshi, et al. Large language models are zero-shot reasoners. NeurIPS (2022)
Anything better than “Let’s think step by step”? 
Let LLMs optimize it!

Text: Let’s figure it out! Score: 61

Text: Let’s solve the problem. Score: 63

(... more texts and scores ...)

Write your new text that is different from the old ones and has a score as high as possible.
Results on GSM8K w/ PaLM

“Let’s think step by step” 71.8%

“Take a deep breath and work on this problem step-by-step” 80.2%
(found by LLM optimization)
Why LLMs can reason?
Reasoning as an emergent behavior

- Reasoning emerges from next-token-prediction pretraining
  - CoT is discovered, not manually designed
  - Sharp reasoning performance transition vs (#parameters, #tokens)
- Distill big models to small ones? Check performance on reasoning tasks!
“How to make parrots intelligent?”

“Scaling up!”

https://twitter.com/denny_zhou/status/159145184723932544?s=20
Toward understanding in-context learning

- Transformer models are meta-learners: implicitly learned training algorithms (like gradient descent) from pretraining.
- In the inference time, transformer models implicitly built the prediction model from the inputs and then predict.

Summary

- **Chain-of-thought**: add “thought” before final “answer”
- **Self-consistency**: sample repeatedly, and select the most frequent answer
- **Least-to-most**: decompose to subproblems and solve them one by one
- **Instruction finetuning**: enable zero-shot / chat
A conversation between my daughter and her little brother

A: my daughter   B: her little brother

A: What is 51 divided by 3?
B: I don’t know.
A: What is 30 divided by 3?
B: 10
A: What is 21 divided by 3?
B: 7
A: What is 10 + 7?
B: 17
A: See, you made it!
What is next?
A model with language understanding and reasoning opens a door to infinite possibilities
Thank You

https://twitter.com/denny_zhou
https://dennyzhou.github.io/
https://scholar.google.com/citations?user=UwLsYw8AAAAJ&hl=en