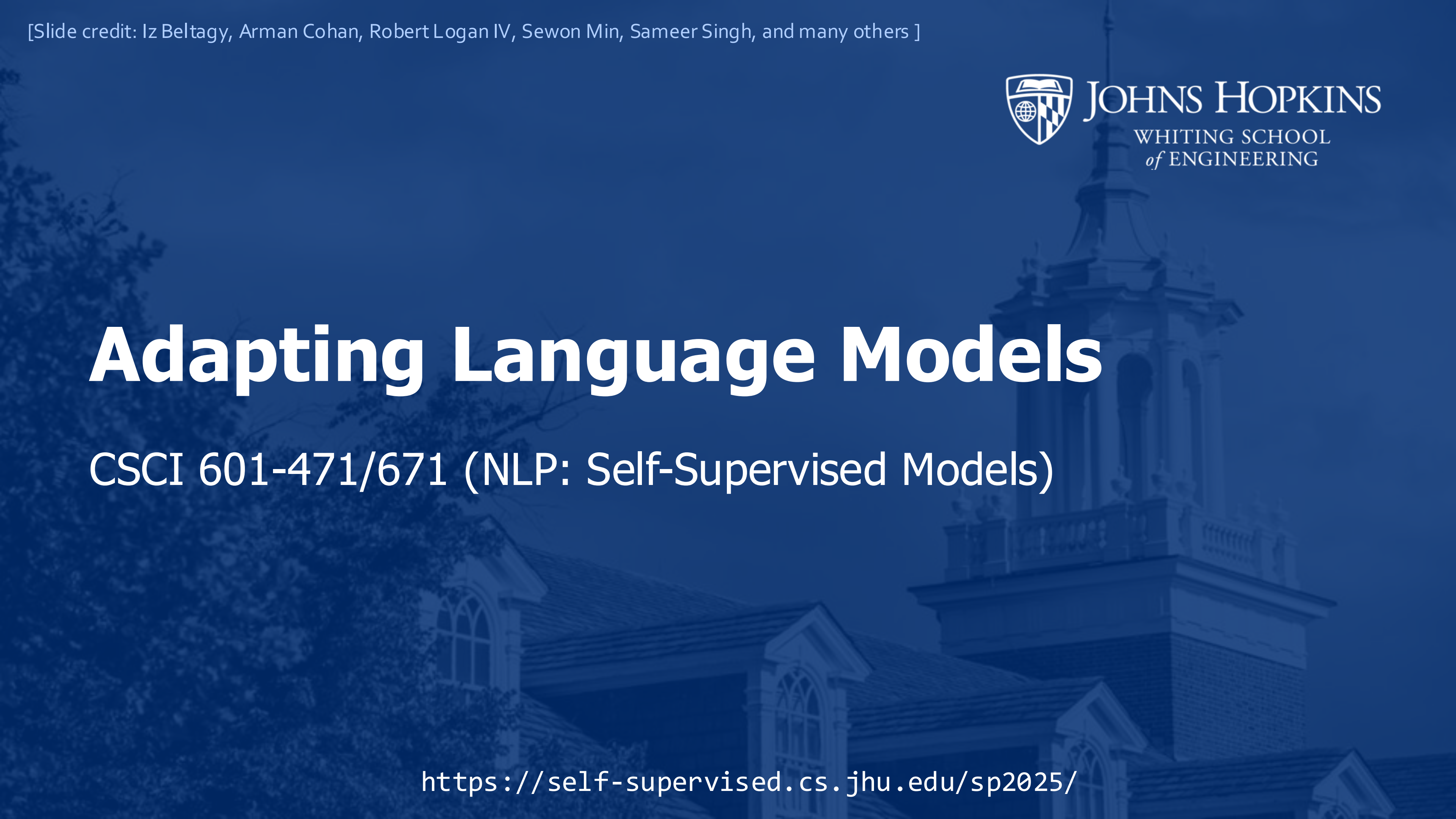
[Slide credit: Iz Beltagy, Arman Cohan, Robert Logan IV, Sewon Min, Sameer Singh, and many others ]

JOHNS HOPKINS
WHITING SCHOOL
*of* ENGINEERING



# Adapting Language Models

CSCI 601-471/671 (NLP: Self-Supervised Models)

https://self-supervised.cs.jhu.edu/sp2025/

# Logistics update

- Quiz 2 is this Thursday.
  - Will cover everything until the beginning of “LM adaptation”.

- Grade distributions so far are online on Piazza.

# Adapting Language Models: Chapter Plan

You have a pre-trained language model that is pre-trained on massive amounts of data. They do not necessarily do useful things—they only complete sentences.

**Now how to you ”adapt” them for your use-case?**

- **Prompting:** adapting model inputs (language statements)
- **Tuning:** adapting (modifying) model parameters

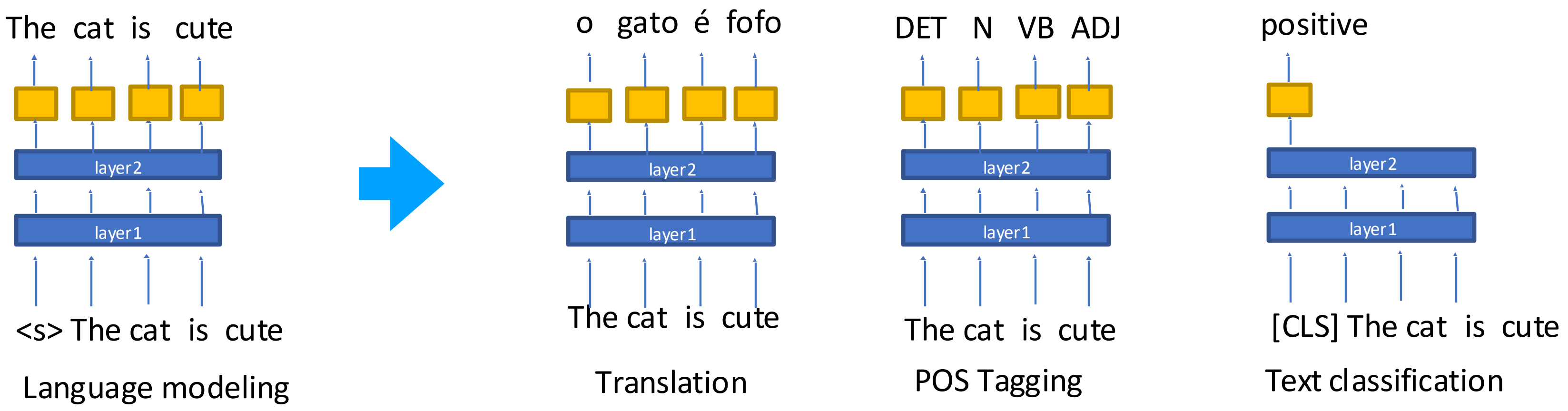
# Adapting Language Models: Chapter Plan

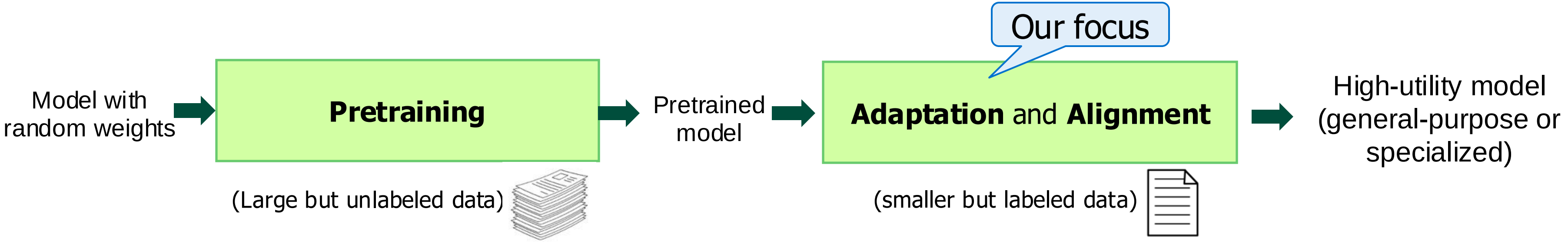
1. Adaptation via Fine-tuning
2. Parameter-efficient fine-tuning
3. Adaptation via In-Context Learning
4. In-context learning of multi-step problems.

**Chapter goal:** Get comfortable with various forms of adapting LMs for solving your tasks!

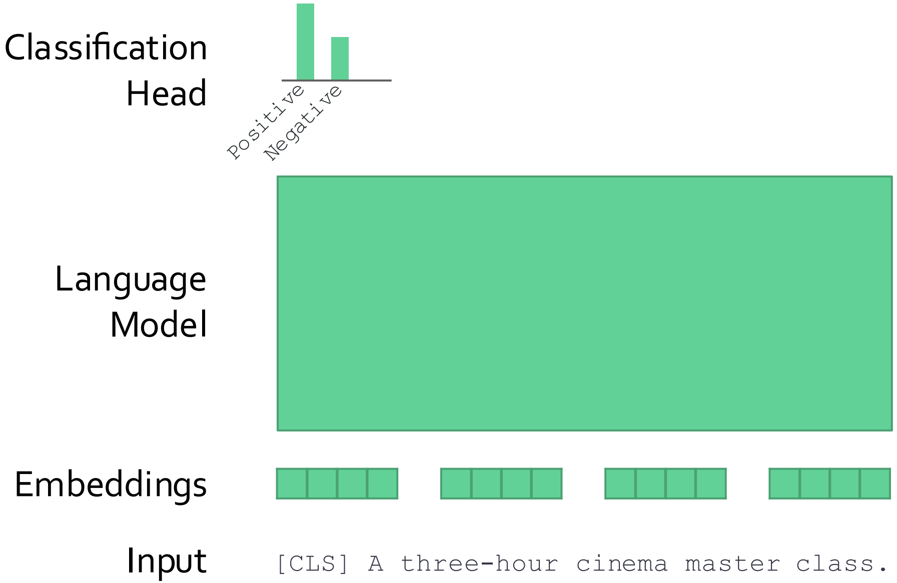
# Adaptation via Fine-Tuning

# Fine-Tuning for Tasks




Model with random weights
Pretraining
(Large but unlabeled data)
Pretrained model
Our focus
Adaptation and Alignment
(smaller but labeled data)
High-utility model (general-purpose or specialized)

# Fine-tuning Pre-trained Models



- Whole model tuning:
  - Run an optimization defined on your task data that updates **all** model parameters
- Head-tuning:
  - Run an optimization defined on your task data that updates the parameters of the model "head"

[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# Fine-tuning Pre-trained Models

- In fine-tuning we need to updating and storing all the parameters of  the LM
  - We would need to store a copy of the LM for each task
- With large models, storage management becomes difficult
  - E.g., A model of size 170B parameters requires ~340Gb of storage
  - If you fine-tune a separate model for 100 tasks:
    - 340 * 100 = 34 TB of storage!

# Parameter-efficient Fine-tuning

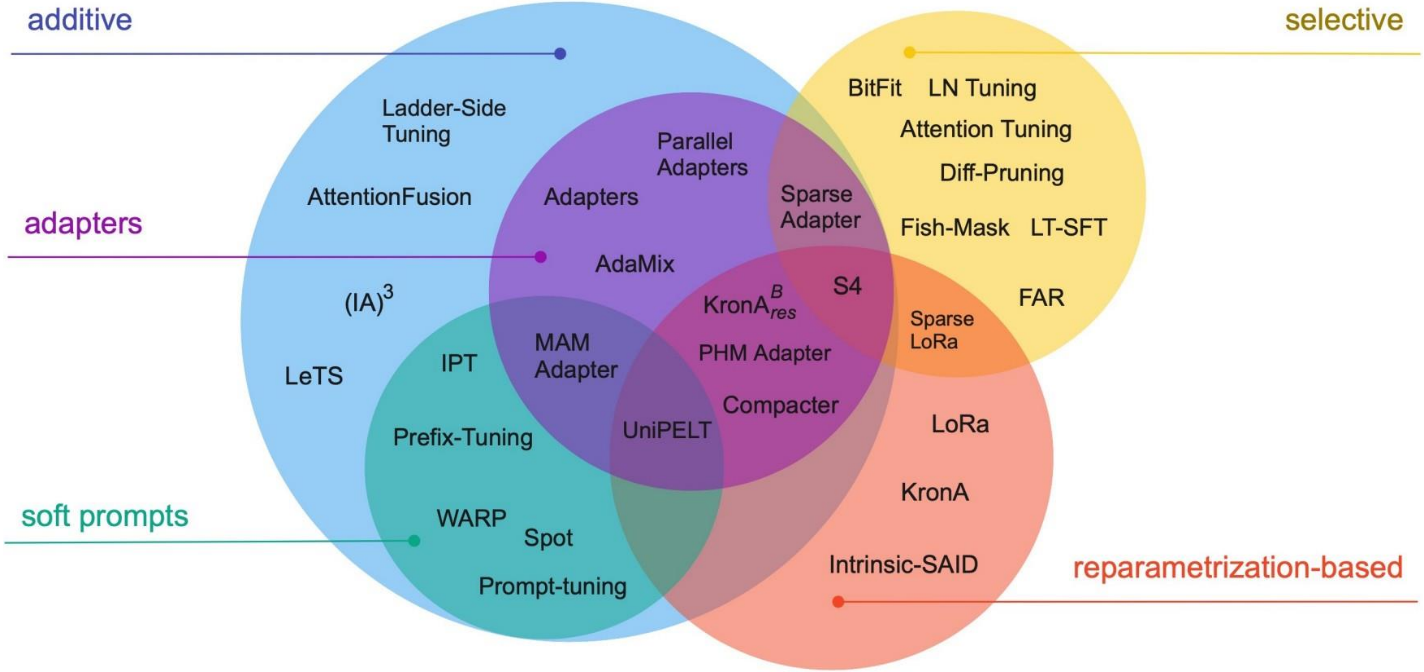
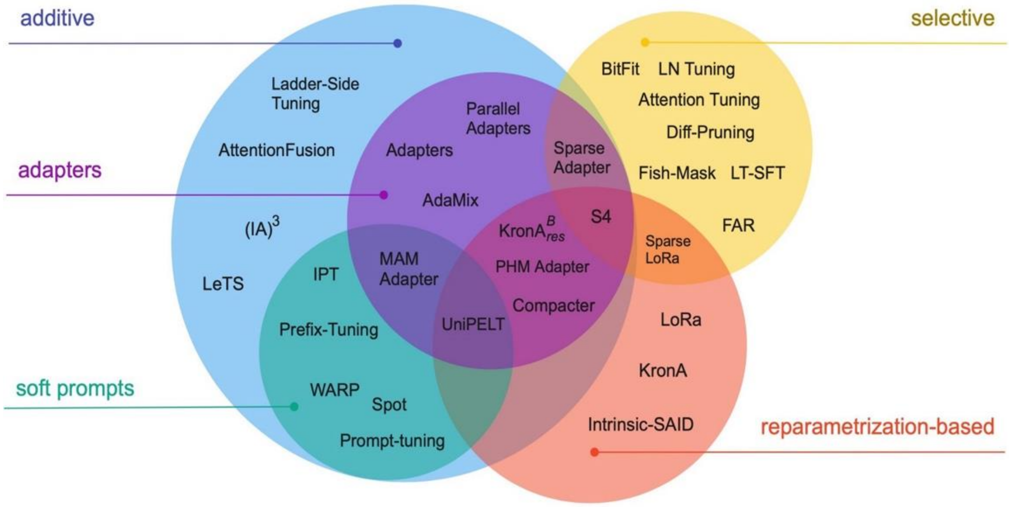


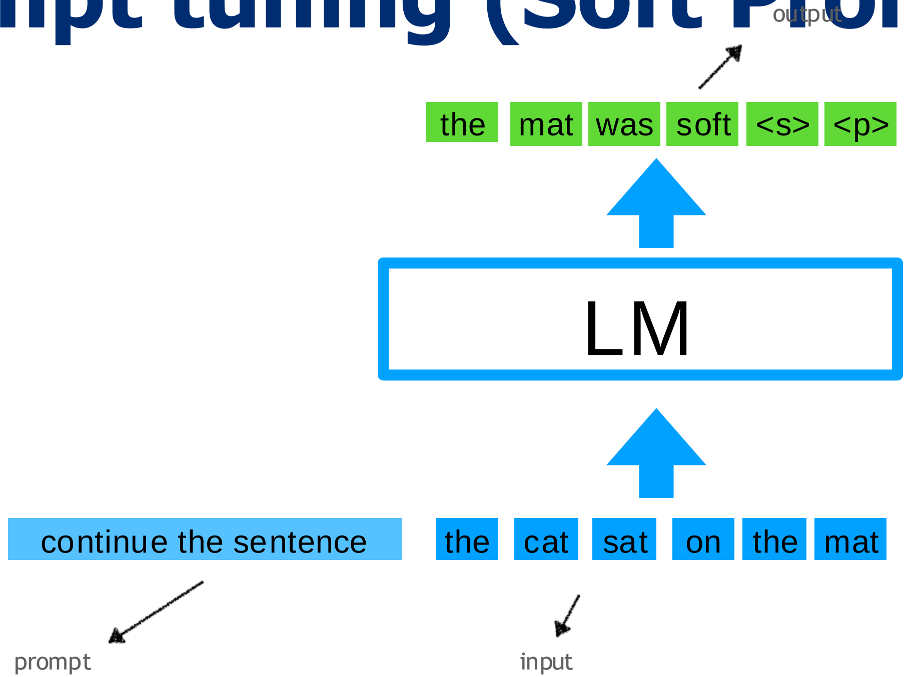
fig source https://arxiv.org/pdf/2303.15647.pdf

# Param-efficient Tuning: Additive Models

- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- Two commonly used methods:
  - **Soft prompts**
  - Adapters

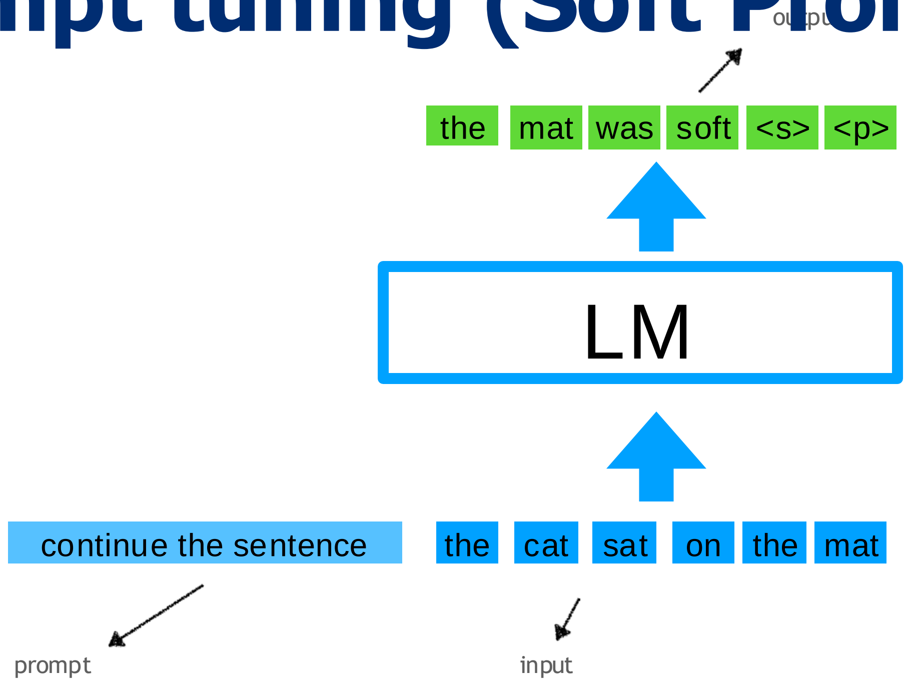


# Prompt tuning (Soft Prompts)



[Slide: Arman Cohan]

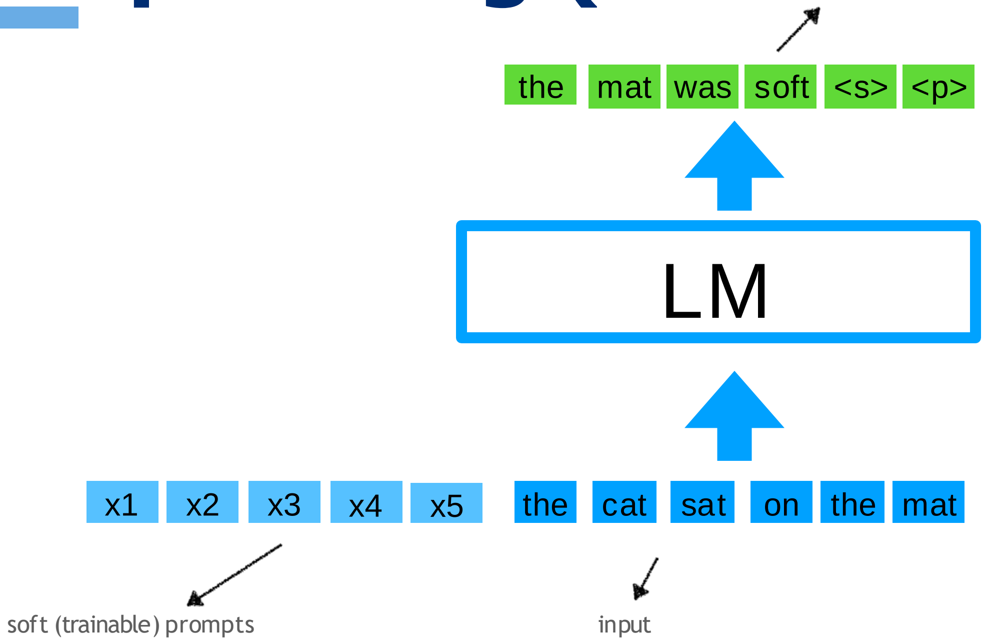
# Prompt tuning (Soft Prompts)



- Designing good prompts might be difficult for each task
- Maybe we can “learn” the prompts?

[Slide: Arman Cohan]

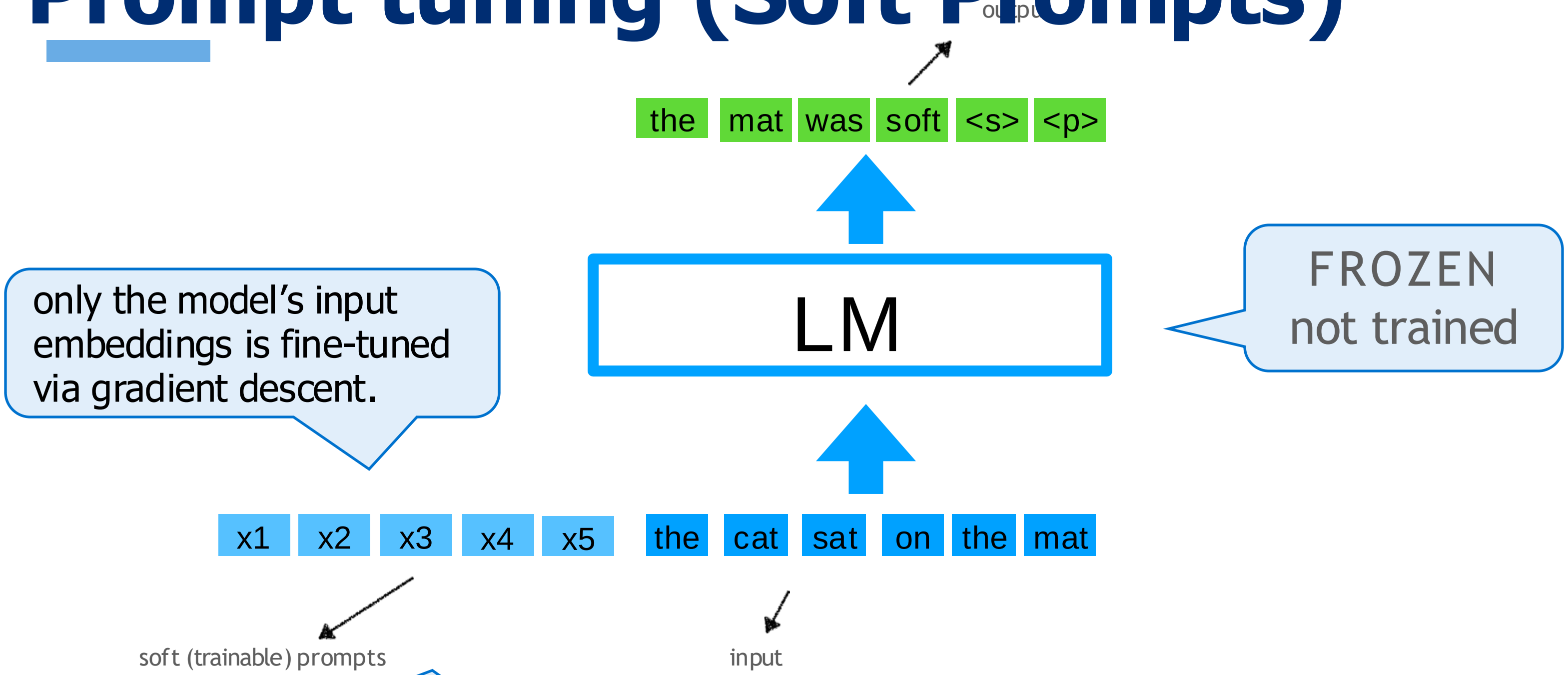
# Prompt tuning (Soft Prompts)



- Designing good prompts might be difficult for each task
- Maybe we can "learn" the prompts?

[Slide: Arman Cohan]

# Prompt tuning (Soft Prompts)

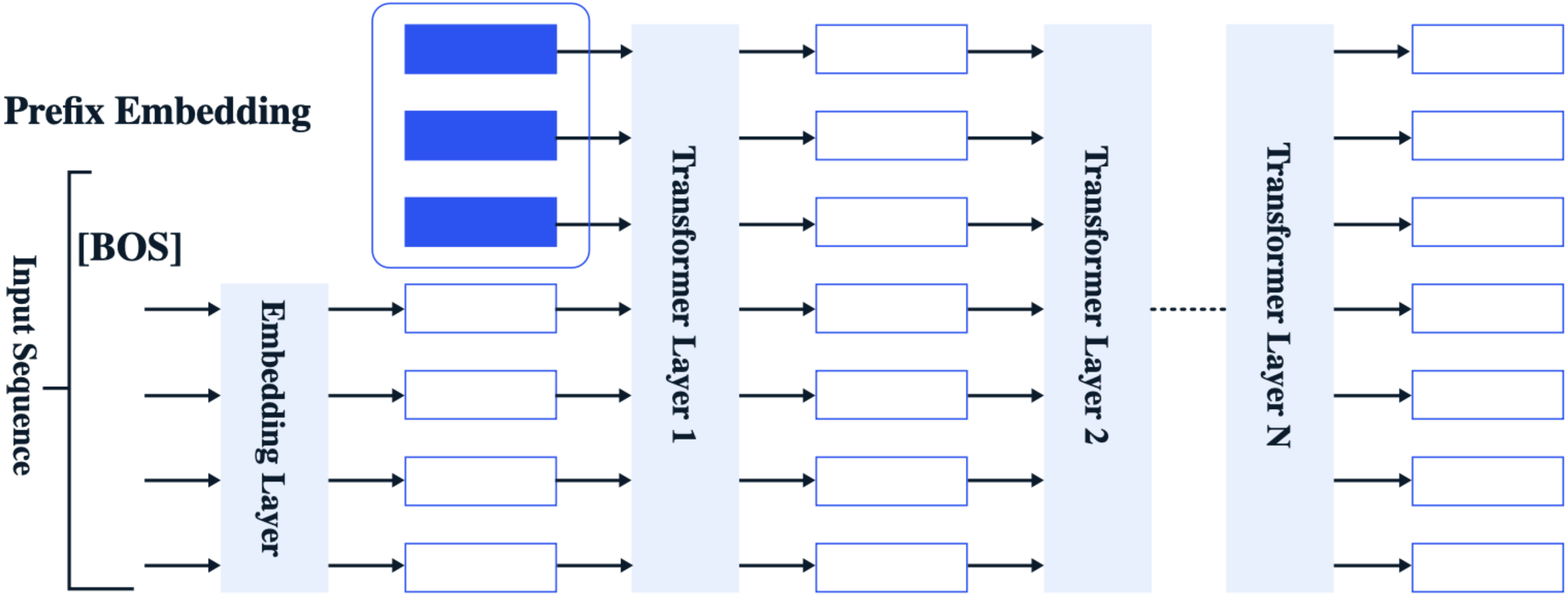


only the model's input embeddings is fine-tuned via gradient descent.

FROZEN
not trained

It is better to initialize soft prompts from existing vocab than randomly. [Lester et al., 2021]

# Prompt tuning (Soft Prompts)



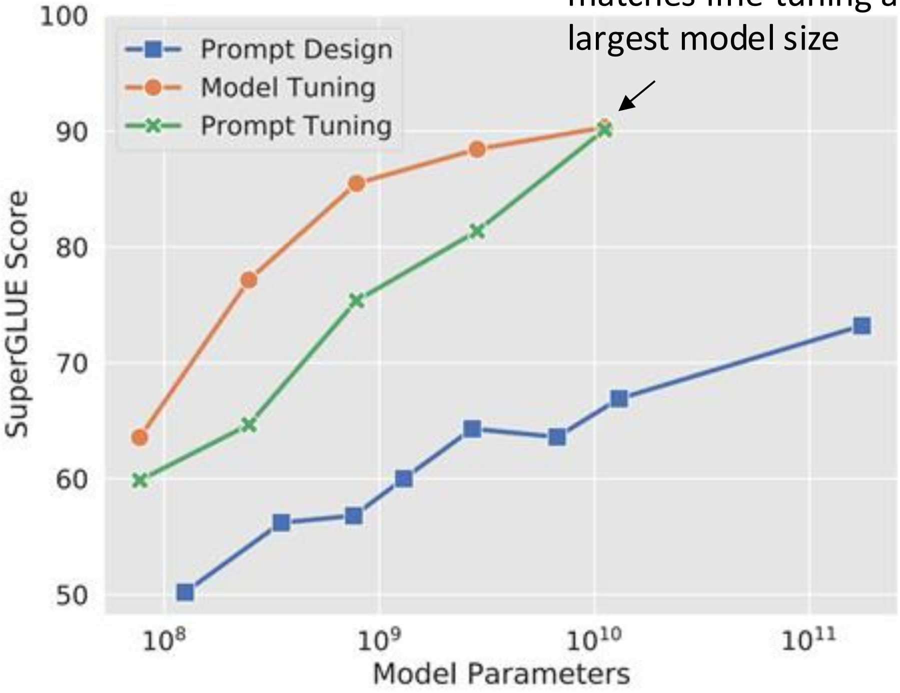
[Slide credit: Samet Oymak]

# Prompt Tuning: Effect of Model Size

- Model tuning: fine-tuning the model for your task.

- Prompt design: manually designing a problem.

- Prompt tuning performs poorly at smaller model sizes and on harder tasks.



Prompt tuning only matches fine-tuning at th
largest model size

[The Power of Scale for Parameter-Efficient Prompt Tuning, Lester et al. 2021]

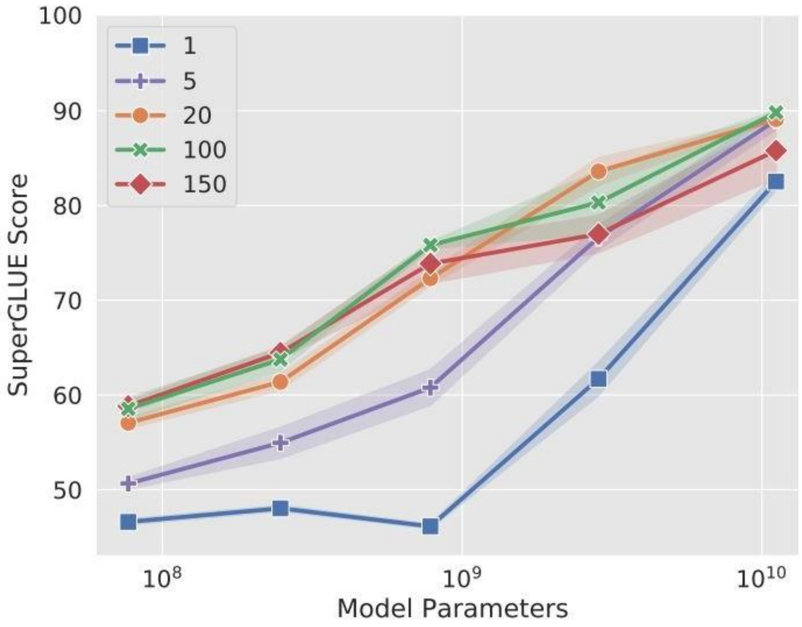
# Prompt Tuning: Effect of Prompt Length

- The shorter the prompt, the fewer new parameters must be tuned

- Increasing prompt length is critical to achieving good performance

- The largest model still gives strong results with a single-token prompt

- Increasing beyond 20 tokens only yields marginal gains



[The Power of Scale for Parameter-Efficient Prompt Tuning, Lester et al. 2021]

# Prefix-tuning vs Prompt-tuning

- **Prompt-tuning**, extra trainable embeddings are simply prepended to the model's input.

- **Prefix-tuning**, similar to prompt tuning but applied at every attention layer of the model.

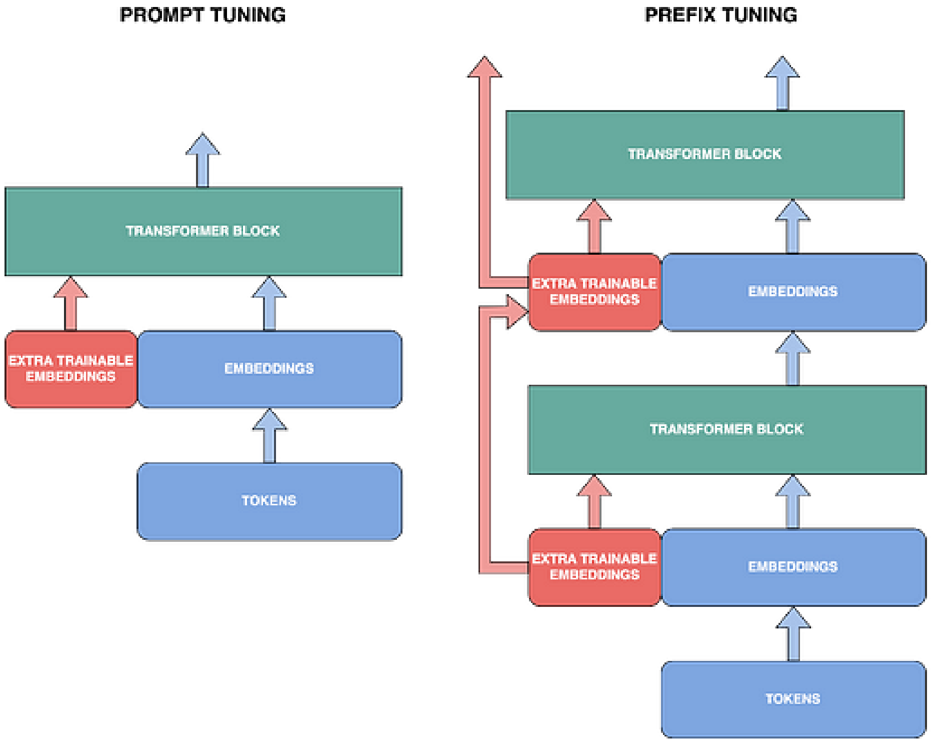
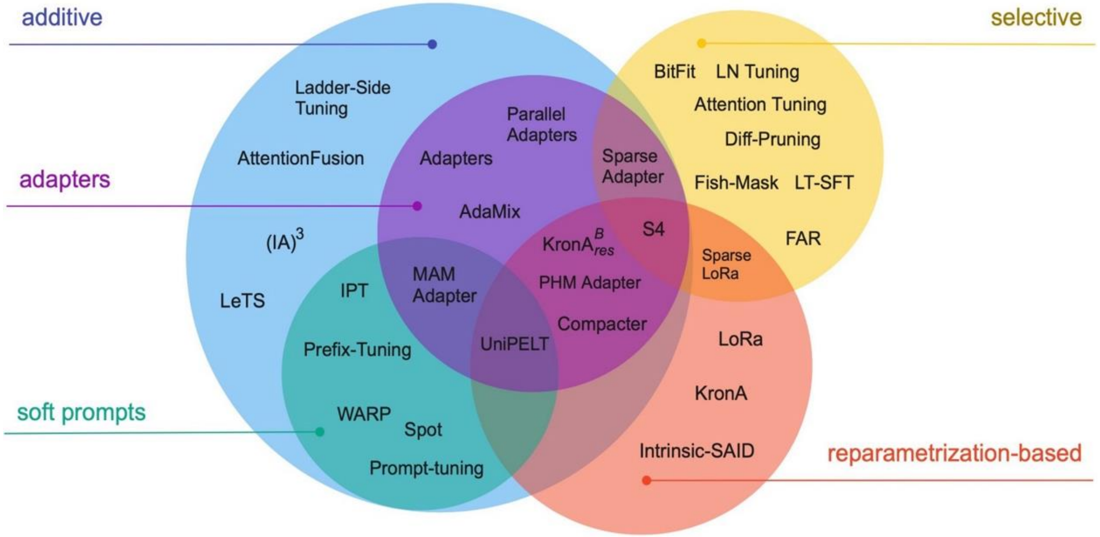


Figure source: https://blog.dataiku.com/parameter-efficient-llm-fine-tuning

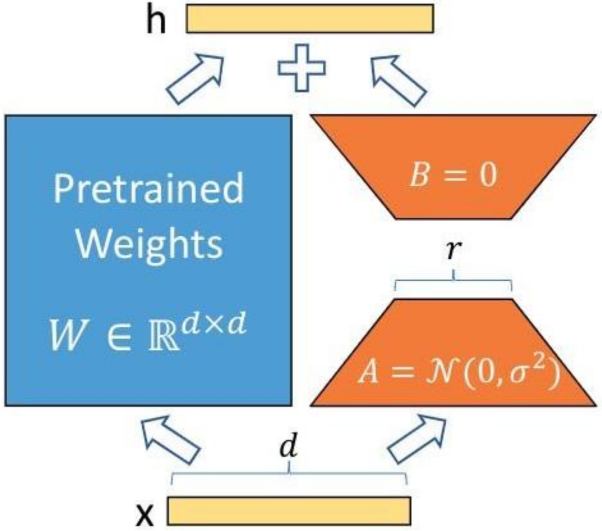
# Reparametrization based methods

- Reparametrize the weights of the network using a low-rank transformation. This decreases the trainable parameter count while still allowing the method to work with high-dimensional matrices



# LoRA: Low-Rank Adaptation

- Hypothesis: the intrinsic rank of the weight matrices in a large language model is low.
- Parameter update for a weight matrix is decomposed into a product of two low-rank matrices.


h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
$r$
$A = \mathcal{N}(0, \sigma^2)$
$d$
x

$$W \leftarrow W + \Delta W$$
$$\Delta W = BA$$
$$B \in \mathbb{R}^{d \times r}, A \in \mathbb{R}^{r \times d}, \qquad r \ll d$$

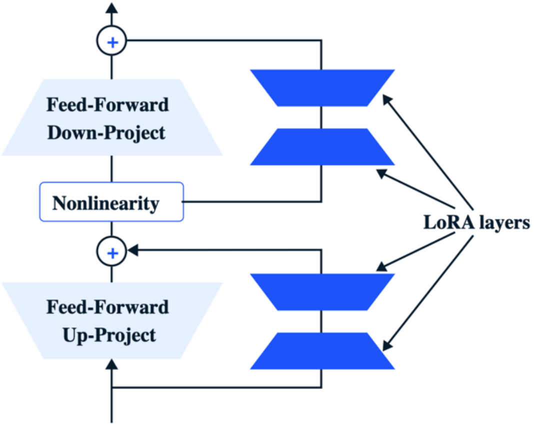
- A is initialized with random Gaussian Initialization, B is initialized to zero.

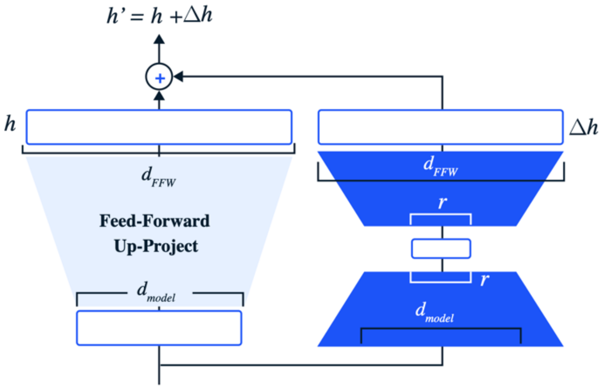
["LoRA: Low-Rank Adaptation of Large Language Models" Hu et al., 2021.]

# LoRA: Where you place them?

- LoRA used to be applied to $W_q$ and $W_v$ matrices in SA.
  - But since then, it's been common to target all matrices in SA and FFN of all layers.

$$\{W_q^{(l)}, W_k^{(l)}, W_v^{(l)}, W_o^{(l)}\}_{l=1}^L \quad \{W_{\text{gate}}^{(l)}, W_{\text{up}}^{(l)}, W_{\text{down}}^{(l)}\}_{l=1}^L$$

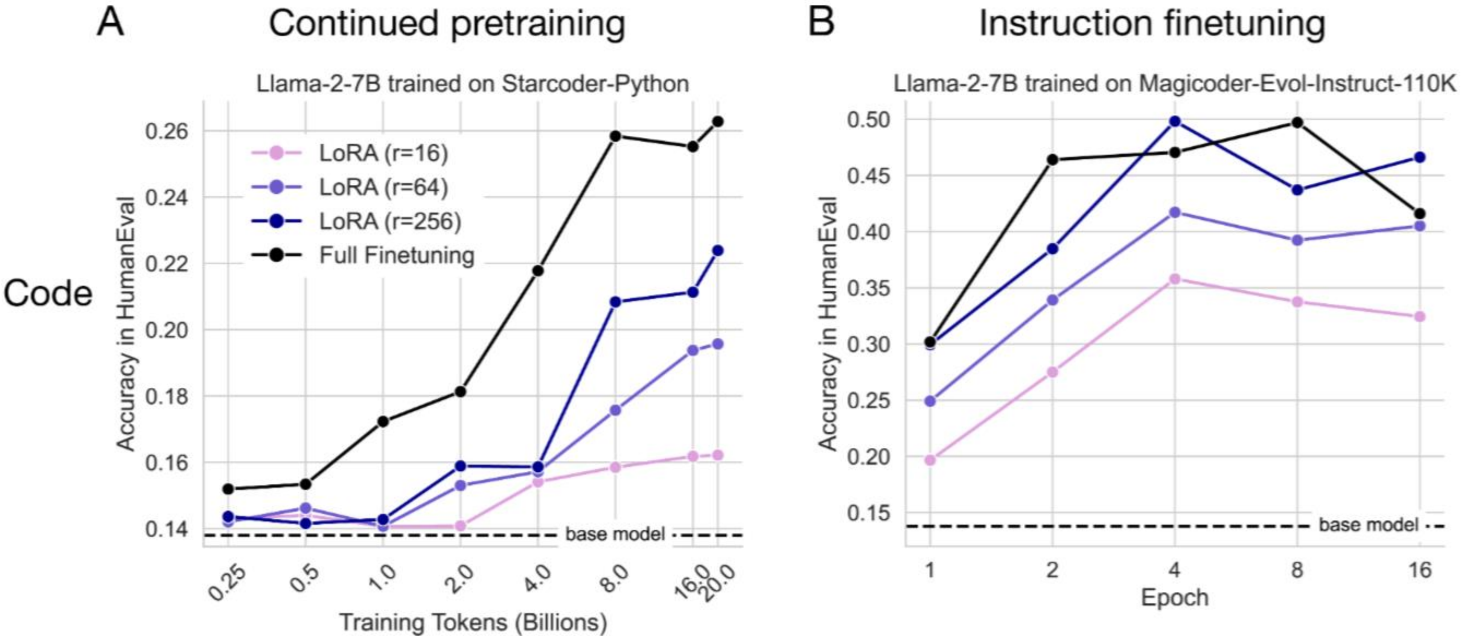




[Figure credit: Samet Oymak]

# LoRA, empirically

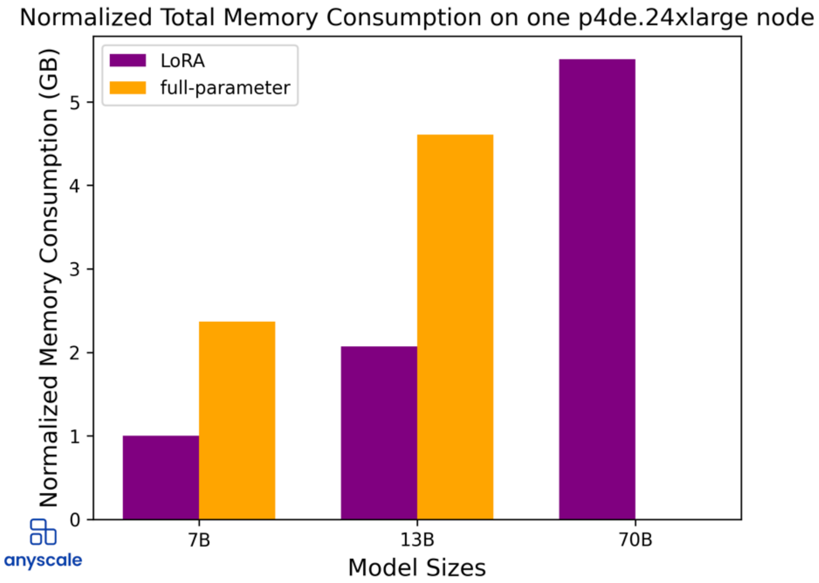
- LoRA [at low ranks] underperforms full finetuning.



[LoRA Learns Less and Forgets Less, 2024]

# LoRA memory footprint

- LoRA reduces memory requirements during fine-tuning (up to x3 for LoRA).
- Why?



https://www.anyscale.com/blog/fine-tuning-llms-lora-or-full-parameter-an-in-depth-analysis-with-llama-2

# LoRA: Compute Cost

- Does LoRA lead to computational speedup?

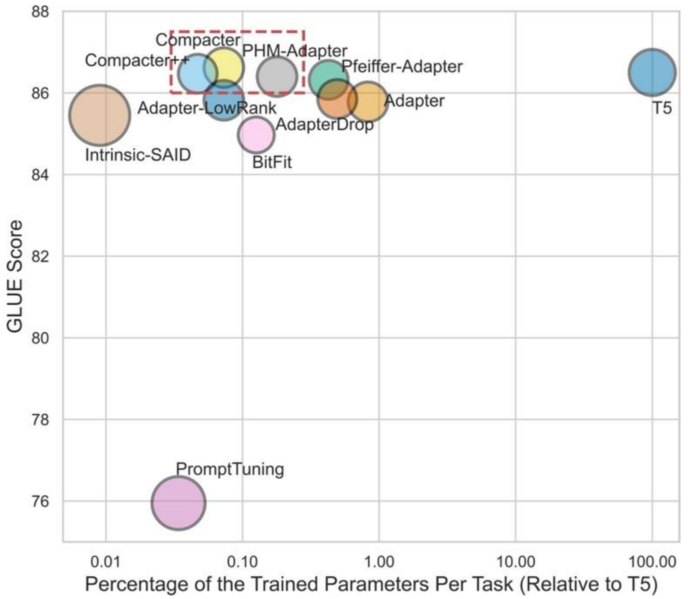
- No, in the sense that you still need to run the same operations during forward and backward.

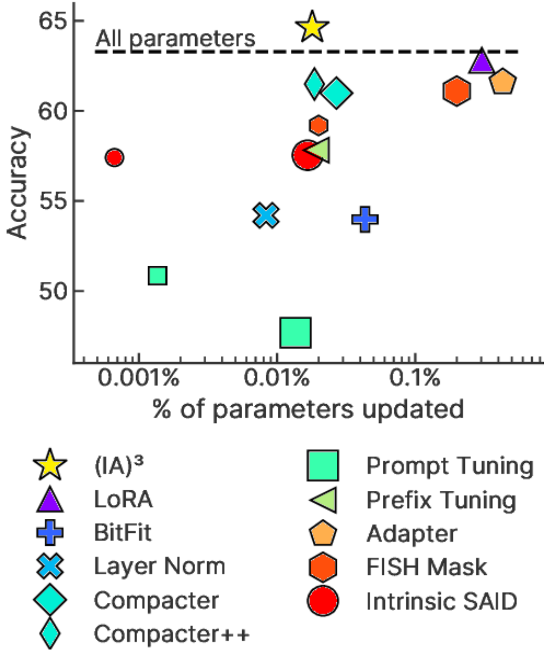
- Yes, in the sense that you can fit larger batches in your memory.

# Recap: LoRA

- LoRA: Low-rank adaptation to Transformer weight matrices.
- Memory footprint: LoRa reduces memory requirements during fine-tuning (up to x3 for LoRA).
- Impact on floating-point operations: LoRA's impact on on FLOPs, during training or inference, is minimal. You still need same FLOPs for forward/backward passes even if you update few parameters. You just apply less number of gradient updates.
- The main speedup is coming from the fast that you can pack more things in a batch.

# Performance/compactness comparison





# Summary: Adaptation via Fine-Tuning

- Parameter efficient optimization — optimize fewer parameters than the whole model.
  - Space efficiency — fewer parameters to store
  - Computation efficiency? A bit unclear

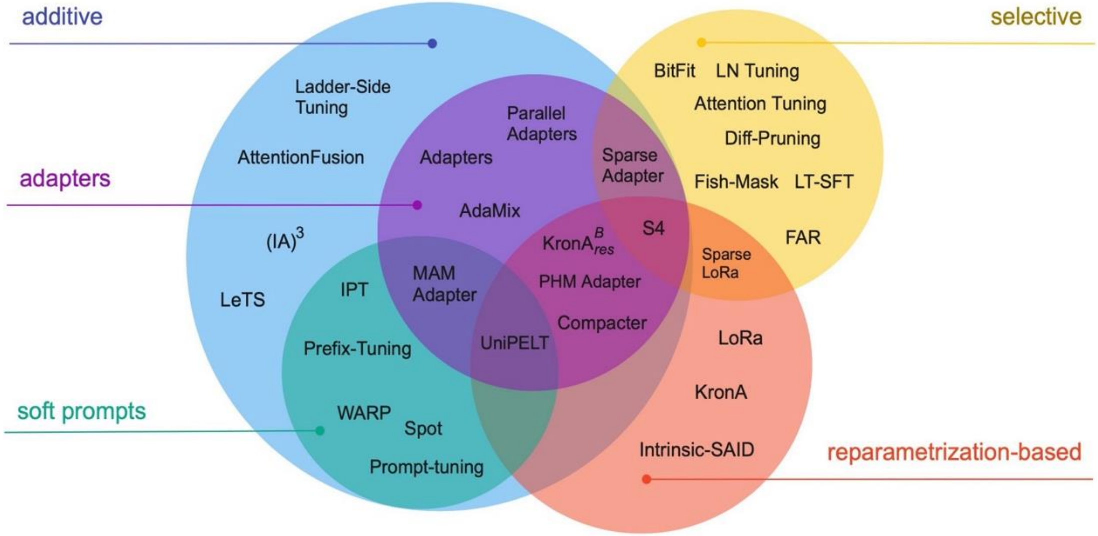
- Their interpretability is not quite clear.

- See: https://huggingface.co/docs/trl/index

Bonus

# Selective methods

- Selective methods fine-tune a subset of the existing parameters of the model.
- It could be a layer depth-based selection, layer type-based selection, or even individual parameter selection.

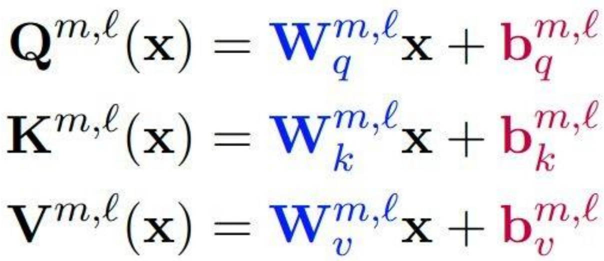


Ben Zaken et al., 2021. "BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models"

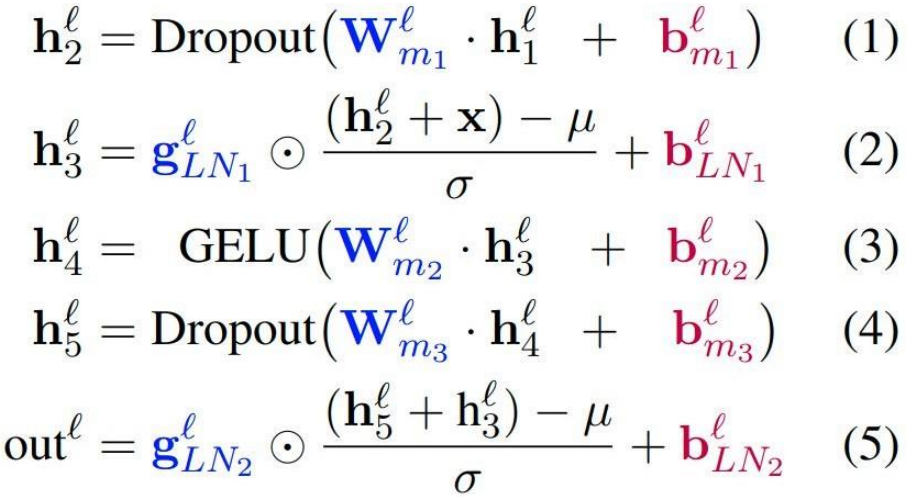
Bonus

# BitFit

- BitFit adds bias terms in self-attention and MLP layers and tunes those.



$$\mathbf{Q}^{m,\ell}(\mathbf{x}) = \mathbf{W}_q^{m,\ell}\mathbf{x} + \mathbf{b}_q^{m,\ell}$$
$$\mathbf{K}^{m,\ell}(\mathbf{x}) = \mathbf{W}_k^{m,\ell}\mathbf{x} + \mathbf{b}_k^{m,\ell}$$
$$\mathbf{V}^{m,\ell}(\mathbf{x}) = \mathbf{W}_v^{m,\ell}\mathbf{x} + \mathbf{b}_v^{m,\ell}$$



$$\mathbf{h}_2^\ell = \text{Dropout}\big(\mathbf{W}_{m_1}^\ell \cdot \mathbf{h}_1^\ell \;+\; \mathbf{b}_{m_1}^\ell\big) \qquad (1)$$
$$\mathbf{h}_3^\ell = \mathbf{g}_{LN_1}^\ell \odot \frac{(\mathbf{h}_2^\ell + \mathbf{x}) - \mu}{\sigma} + \mathbf{b}_{LN_1}^\ell \qquad (2)$$
$$\mathbf{h}_4^\ell = \text{GELU}\big(\mathbf{W}_{m_2}^\ell \cdot \mathbf{h}_3^\ell \;+\; \mathbf{b}_{m_2}^\ell\big) \qquad (3)$$
$$\mathbf{h}_5^\ell = \text{Dropout}\big(\mathbf{W}_{m_3}^\ell \cdot \mathbf{h}_4^\ell \;+\; \mathbf{b}_{m_3}^\ell\big) \qquad (4)$$
$$\text{out}^\ell = \mathbf{g}_{LN_2}^\ell \odot \frac{(\mathbf{h}_5^\ell + \mathbf{h}_3^\ell) - \mu}{\sigma} + \mathbf{b}_{LN_2}^\ell \qquad (5)$$

- BitFit only updates about 0.05% of the model parameters.

Ben Zaken et al., 2021. “BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models”

Bonus

# BitFit

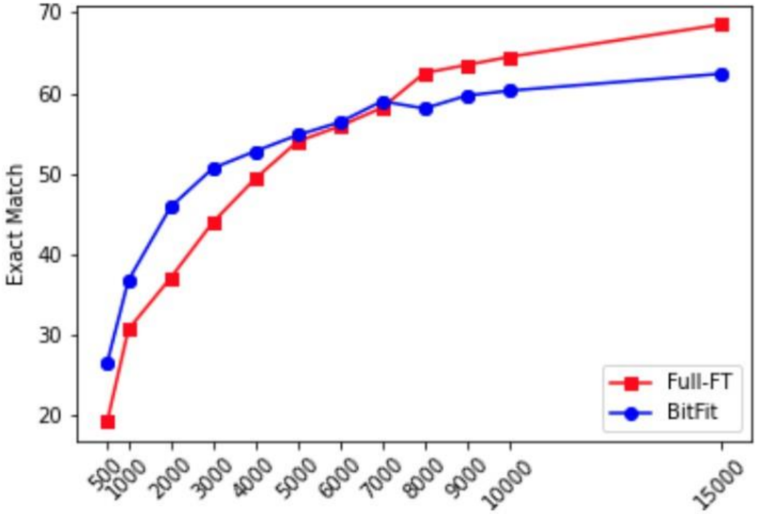


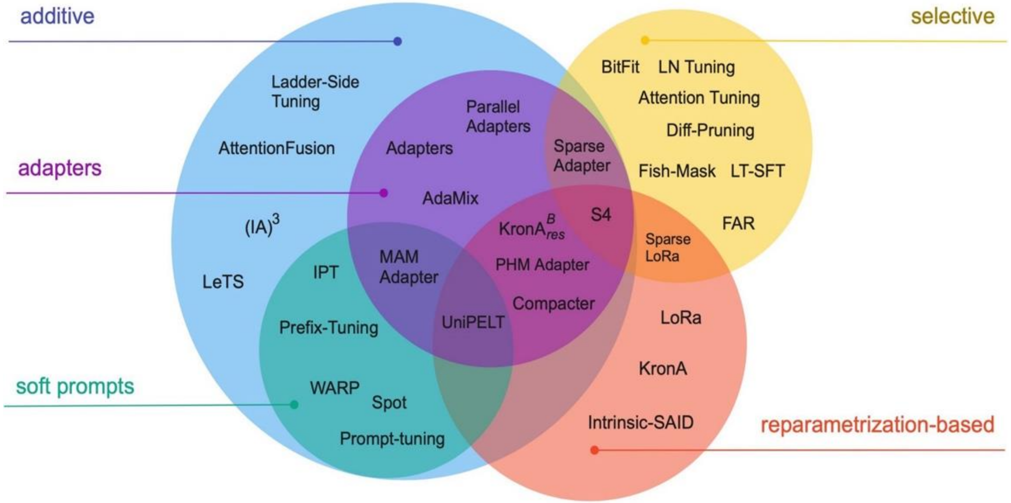
Figure 2: Comparison of BitFit and Full-FT with $\text{BERT}_{\text{BASE}}$ exact match score on SQuAD validation set.

Ben Zaken et al., 2021. "BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models"

Bonus

# Parameter-efficient Fine-tuning: Adding Models

- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- Two commonly used methods:
  - Soft prompts
  - **Adapters**

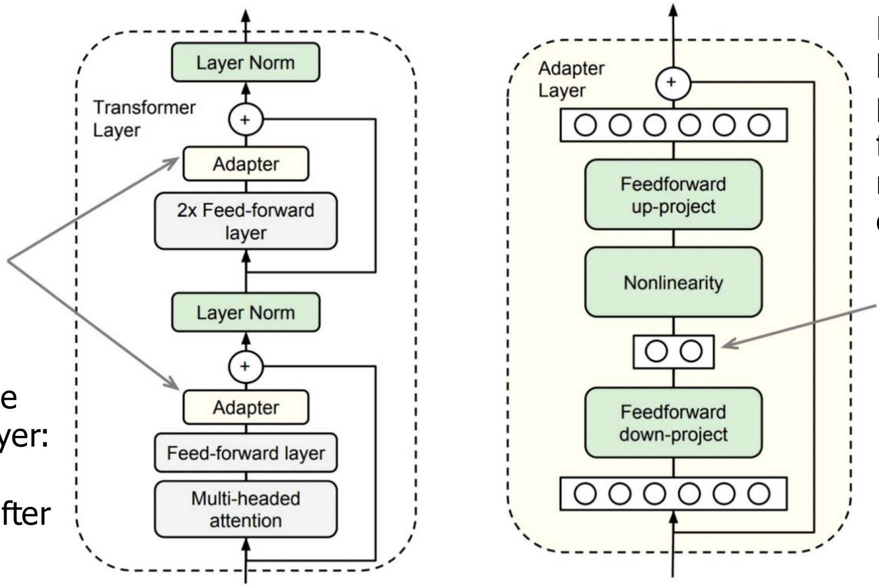


Bonus

# Adapters

- **Idea:** train small sub-networks and only tune those.
  - FF projects to a low dimensional space to reduce parameters.
- No need to store a full model for each task, **only the adapter params**.

Only these are trained, everything else is fixed and is the same for all tasks



**Right**: The adapter consists of a bottleneck which contains few parameters relative to the attention and feedforward layers in the original model. The adapter also contains a skip connection.

Small hidden size, i.e. an adaptor has only a few parameters (which is good!)

**Left**: Add the adapter module twice to each Transformer layer: after the projection following multi-headed attention and after the two feed-forward layers.

[“Parameter-Efficient Transfer Learning for NLP”, Houlsby et al., 2019.]

Bonus

# Question

- Is parameter-efficient tuning more (1) computationally efficient; (2) memory-efficient than whole-model tuning?

- It is not faster! You still need to do the entire forward and backward pass.
- It is more memory efficient.
  - You only need to keep the optimizer state for parameters that you are fine-tuning and not all the parameters.

Bonus

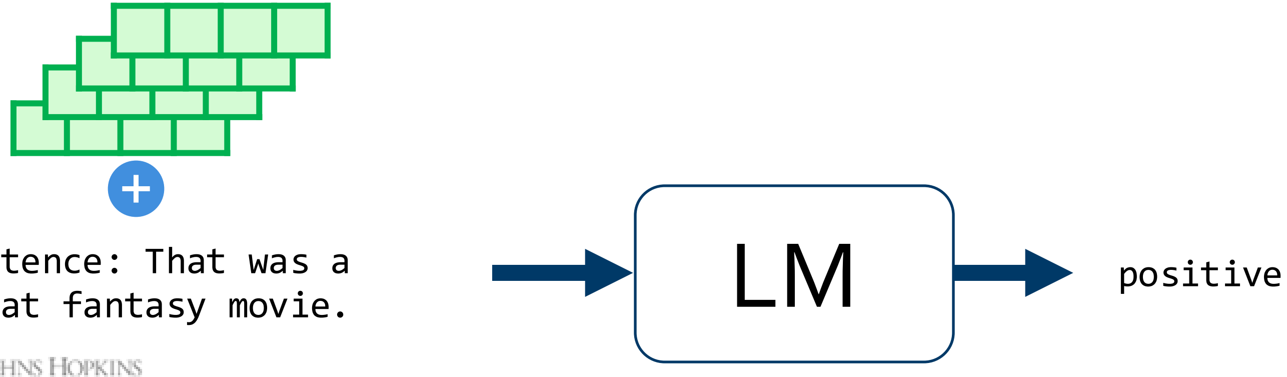
Are continuous prompts as interpretable as Discrete (Language) prompts?

Bonus

# [Continuous] Prompt Waywardness

Prompt waywardness hypothesis, states that:

- There are infinite-many continuous prompts that can solve a given task.



Sentence: That was a great fantasy movie.

["Prompt Waywardness: The Curious Case of Discretized Interpretation of Continuous Prompts" Khashabi et al., 2022.]

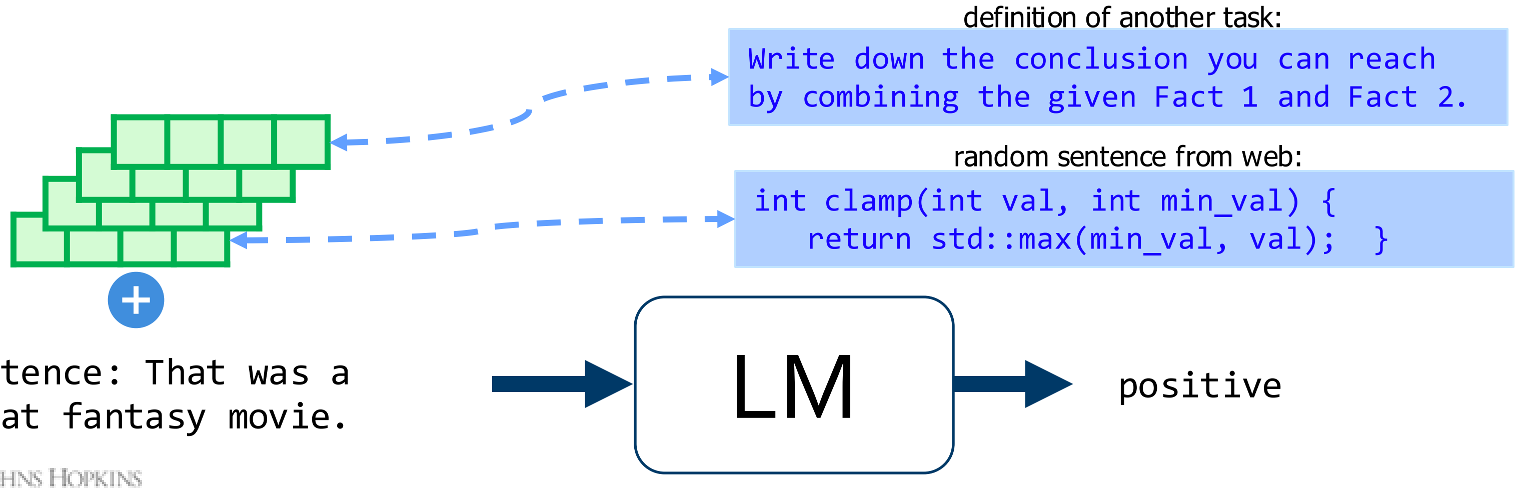
Bonus

# [Continuous] Prompt Waywardness

Prompt waywardness hypothesis, states that:

- There are infinite-many continuous prompts that can solve a given task.
- There exist continuous prompts that ”project” to the description of irrelevant or contradictory tasks.

Project := nearest-neighbor mapping of continuous prompt onto the word embeddings



definition of another task:

```
Write down the conclusion you can reach
by combining the given Fact 1 and Fact 2.
```

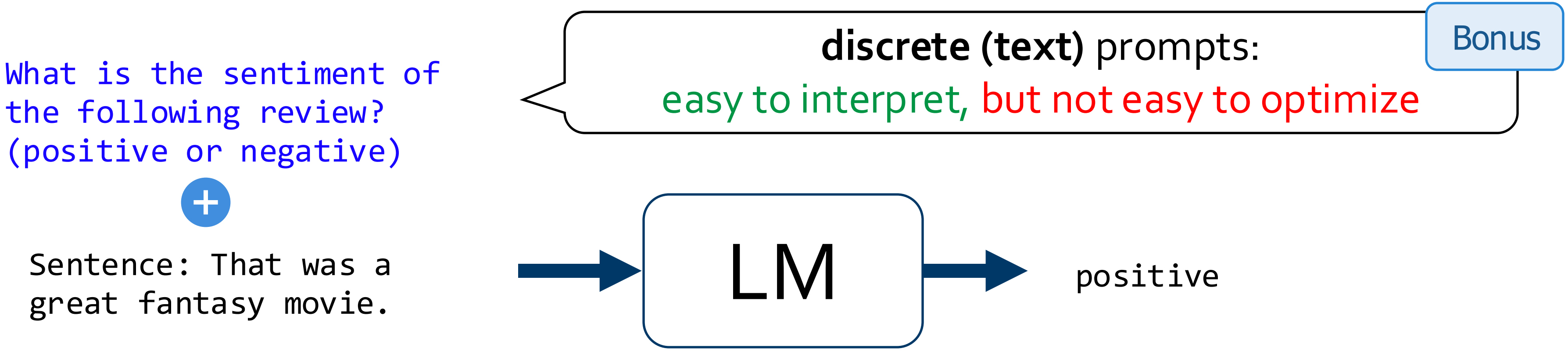
random sentence from web:

```
int clamp(int val, int min_val) {
   return std::max(min_val, val);  }
```

```
Sentence: That was a
great fantasy movie.
```

["Prompt Waywardness: The Curious Case of Discretized Interpretation of Continuous Prompts" Khashabi et al., 2022.]

Bonus



What is the sentiment of the following review? (positive or negative)

+

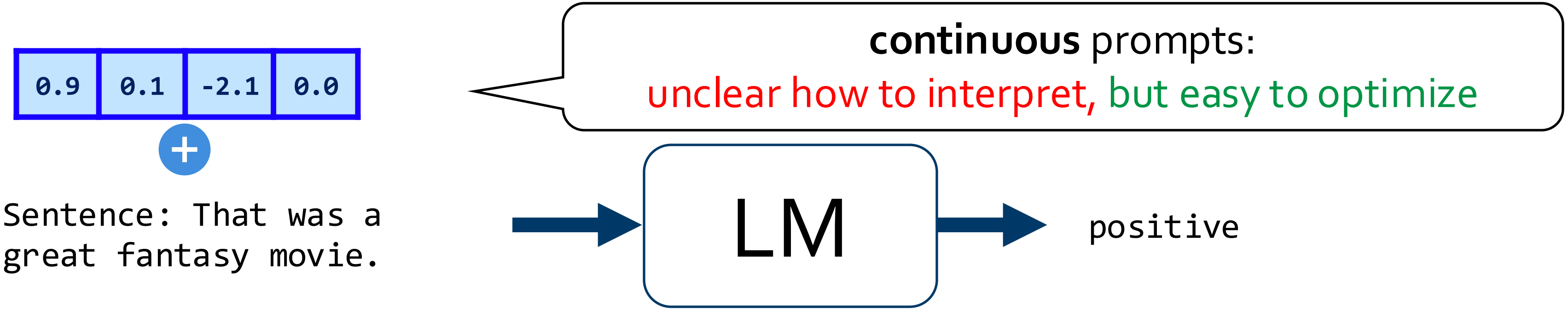
Sentence: That was a great fantasy movie.

LM

positive

**discrete (text)** prompts:

easy to interpret, but not easy to optimize



|     |     |      |     |
|-----|-----|------|-----|
| 0.9 | 0.1 | -2.1 | 0.0 |

+

Sentence: That was a great fantasy movie.

LM

positive

**continuous** prompts:

unclear how to interpret, but easy to optimize

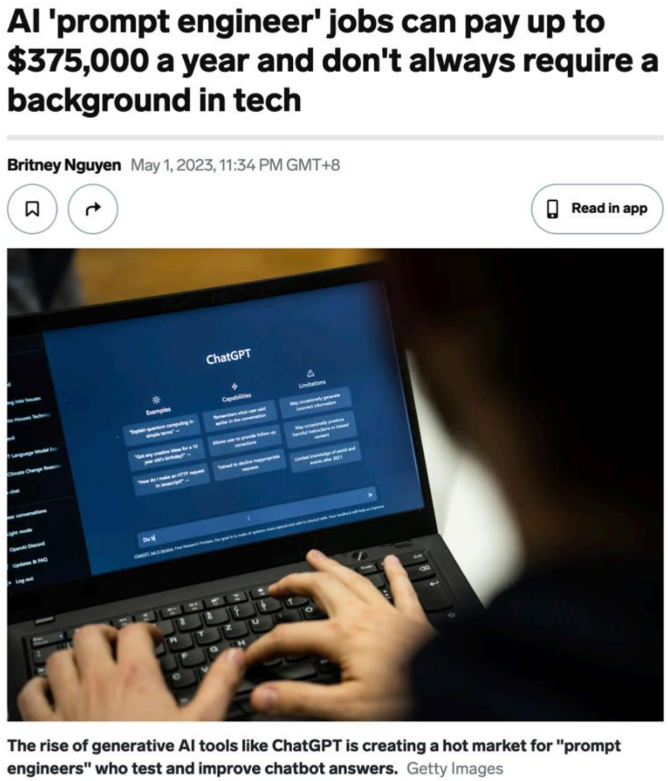
["Prompt Waywardness: The Curious Case of Discretized Interpretation of Continuous Prompts" Khashabi et al., 2022.]

# Prompting and In-Context Learning

# Prompt Engineering

- Reformulating tasks to a language that is easier to for the models.
  - Show demonstrations
  - Decompose your problem
  - Ask for rationales (a la CoT)
  - Check for consistency
  - …

- **Question for you:** will ”prompt engineering” be relevant topic in the coming years?



AI 'prompt engineer' jobs can pay up to $375,000 a year and don't always require a background in tech

Britney Nguyen May 1, 2023, 11:34 PM GMT+8

The rise of generative AI tools like ChatGPT is creating a hot market for "prompt engineers" who test and improve chatbot answers. Getty Images

# Limitations of Pre-training, then Fine-tuning

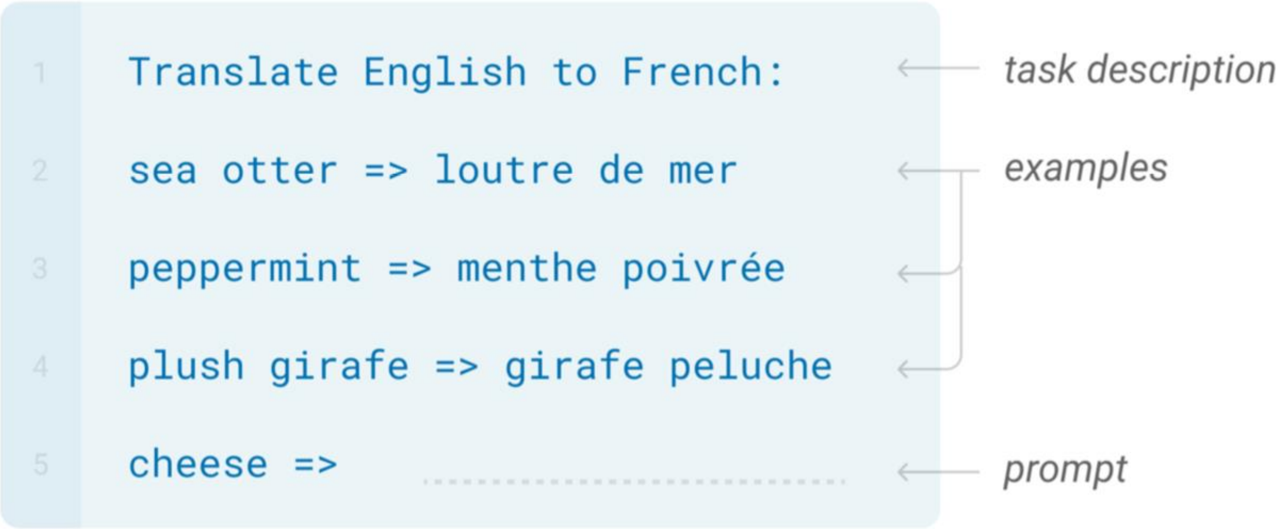
- Often you need a large labeled data
  - Though more pre-training can reduce the need for labeled data

# Limitations of Pre-training, then Fine-tuning

- Often you need a large labeled data
  - Though more pre-training can reduce the need for labeled data

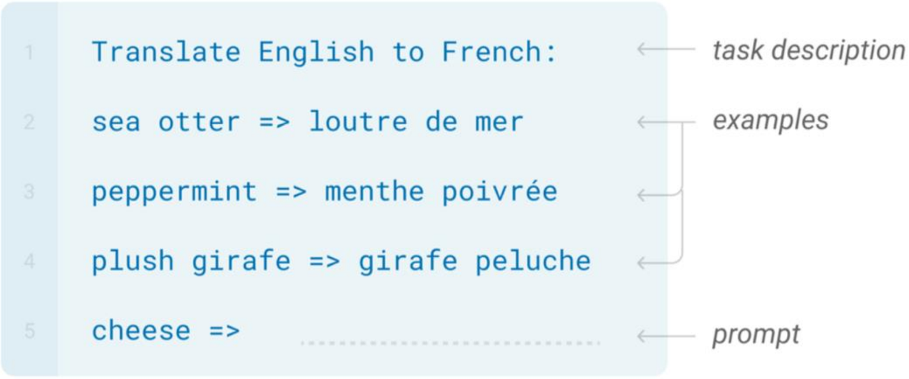
- End up with many copies (or sub-copies) of the same [sub]model

# In-Context Learning



```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese => ..............................
```

# In-Context Learning



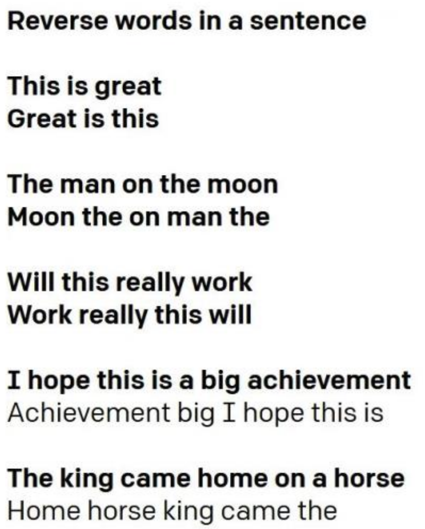
```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese => ............................
```

- Learns to do a downstream task by conditioning on input-output examples!

- **No weight update** — our model is not **explicitly pre-trained** to learn from examples
  - The underlying models are quite general

- Today’s focus:
  - How to use effectively in practice?
  - Fundamentally, why does it work?

# In-Context Learning



Reverse words in a sentence

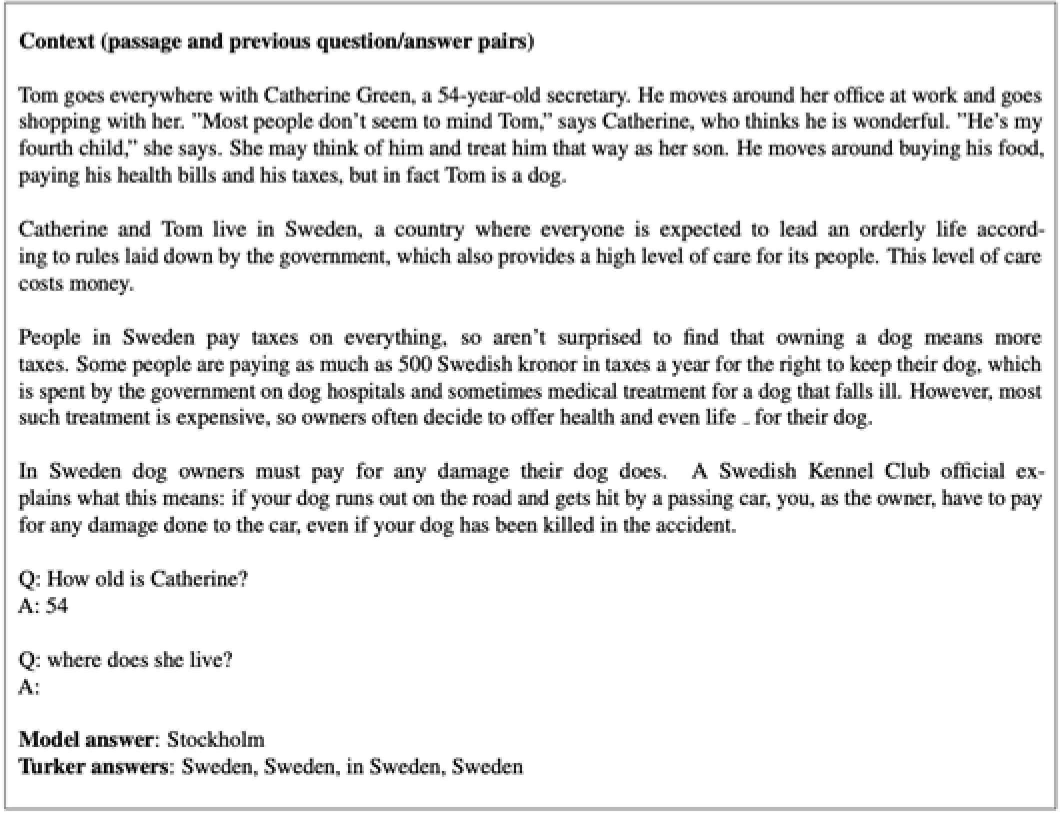
This is great
Great is this

The man on the moon
Moon the on man the

Will this really work
Work really this will

I hope this is a big achievement
Achievement big I hope this is

The king came home on a horse
Home horse king came the



**Context (passage and previous question/answer pairs)**

Tom goes everywhere with Catherine Green, a 54-year-old secretary. He moves around her office at work and goes shopping with her. "Most people don't seem to mind Tom," says Catherine, who thinks he is wonderful. "He's my fourth child," she says. She may think of him and treat him that way as her son. He moves around buying his food, paying his health bills and his taxes, but in fact Tom is a dog.

Catherine and Tom live in Sweden, a country where everyone is expected to lead an orderly life according to rules laid down by the government, which also provides a high level of care for its people. This level of care costs money.

People in Sweden pay taxes on everything, so aren't surprised to find that owning a dog means more taxes. Some people are paying as much as 500 Swedish kronor in taxes a year for the right to keep their dog, which is spent by the government on dog hospitals and sometimes medical treatment for a dog that falls ill. However, most such treatment is expensive, so owners often decide to offer health and even life _ for their dog.

In Sweden dog owners must pay for any damage their dog does. A Swedish Kennel Club official explains what this means: if your dog runs out on the road and gets hit by a passing car, you, as the owner, have to pay for any damage done to the car, even if your dog has been killed in the accident.

Q: How old is Catherine?
A: 54

Q: where does she live?
A:

**Model answer**: Stockholm
**Turker answers**: Sweden, Sweden, in Sweden, Sweden

https://gpt3experiments.substack.com/p/is-gpt-3-really-doing-few-shot-learning

# In-Context Prompting: Implementation

**Movie review dataset**

Input: An effortlessly accomplished and richly resonant work.
Label: positive

Input: A mostly tired retread of several other mob tales.
Label: negative

An effortlessly accomplished and richly resonant work. It was great! A mostly tired retread of several other mob tales. It was terrible!

A three-hour cinema master class. It was ______________

**Language Model**

```
p1 = P(It was great!    | 1st train input+output \n 2nd train input+output \n A three-hour cinema master class.)
p2 = P(It was terrible! | 1st train input+output \n 2nd train input+output \n A three-hour cinema master class.)
```

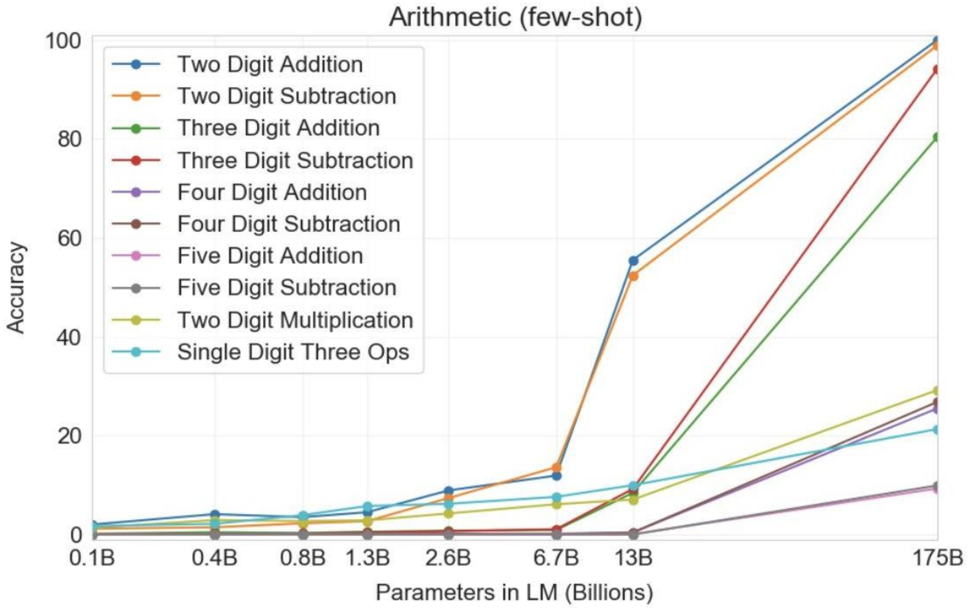
```
p1 > p2  → "positive"
p1 < p2  → "negative"
```

[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# In-Context learning Results

- Example:
  - Q: What is 48 plus 76?
  - A: 124

- Observations:
  - Scale is important
  - Number of digits correlate with their difficulty.
  - Multiplication is harder than summation!



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

# Why Do We Care About In-Context Learning?

Practically Useful

Intellectually Intriguing

[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# In-Context Learning: Practically Useful

- Labeling data is costly
  - May require domain expertise
    - Medical, legal, financial
  - You don’t want to get more data
  - Emergent, time-sensitive scenarios
    - Something new happened—need to react quickly!
- Finetuning can tricky
  - Training is sensitive to hyperparams
  - Not enough validation data
  - We don’t quite understand how finetuning works
  - Expensive to train, time and memory

[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh; quote credit: Colin Raffel]

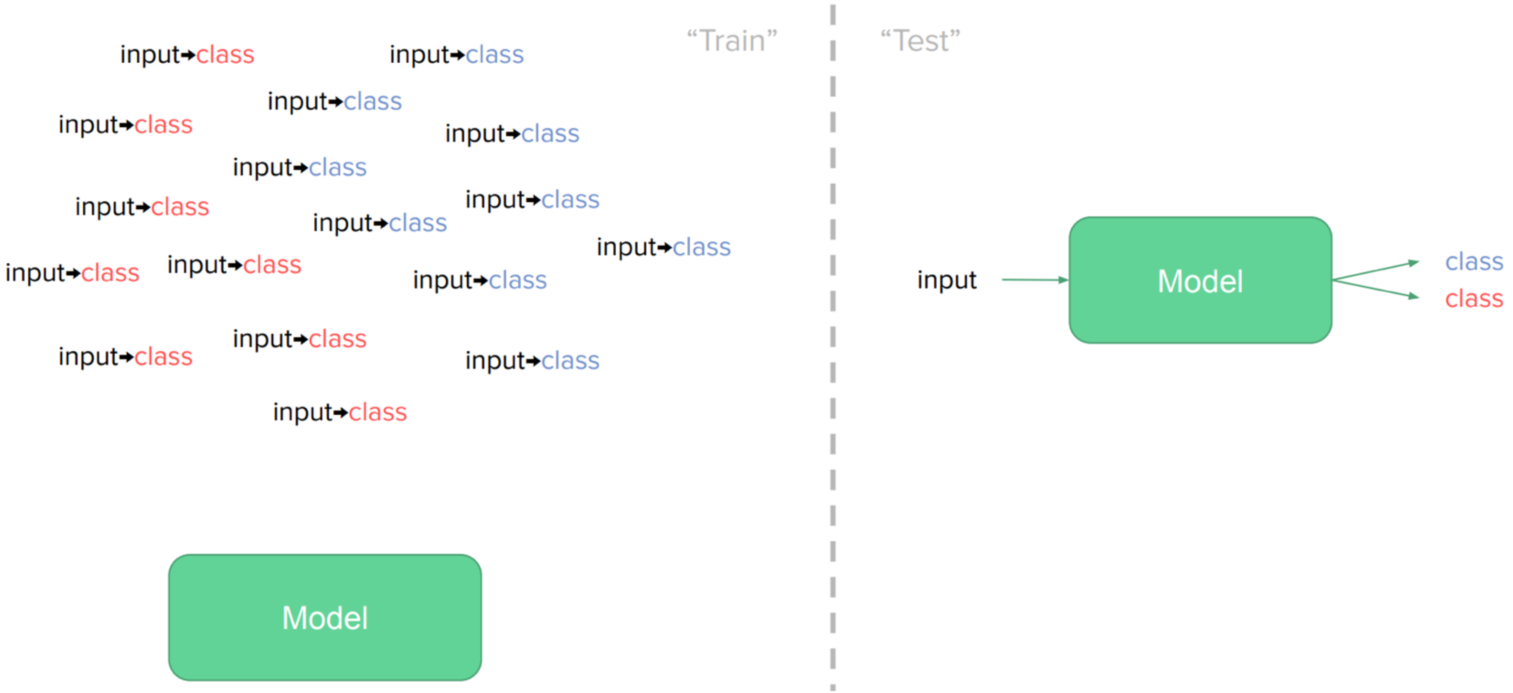
# In-Context Learning: Intellectually Intriguing

- Potential test for "Intelligent Behavior"
  - Generalization from few examples
    - Fundamental piece of intelligence
    - Often used in psychology
    - Quickly adjust to environment

- Insights into Language Modeling
  - What does an LLM "know"?
  - What are the biases/limitations of LLMs?
  - …

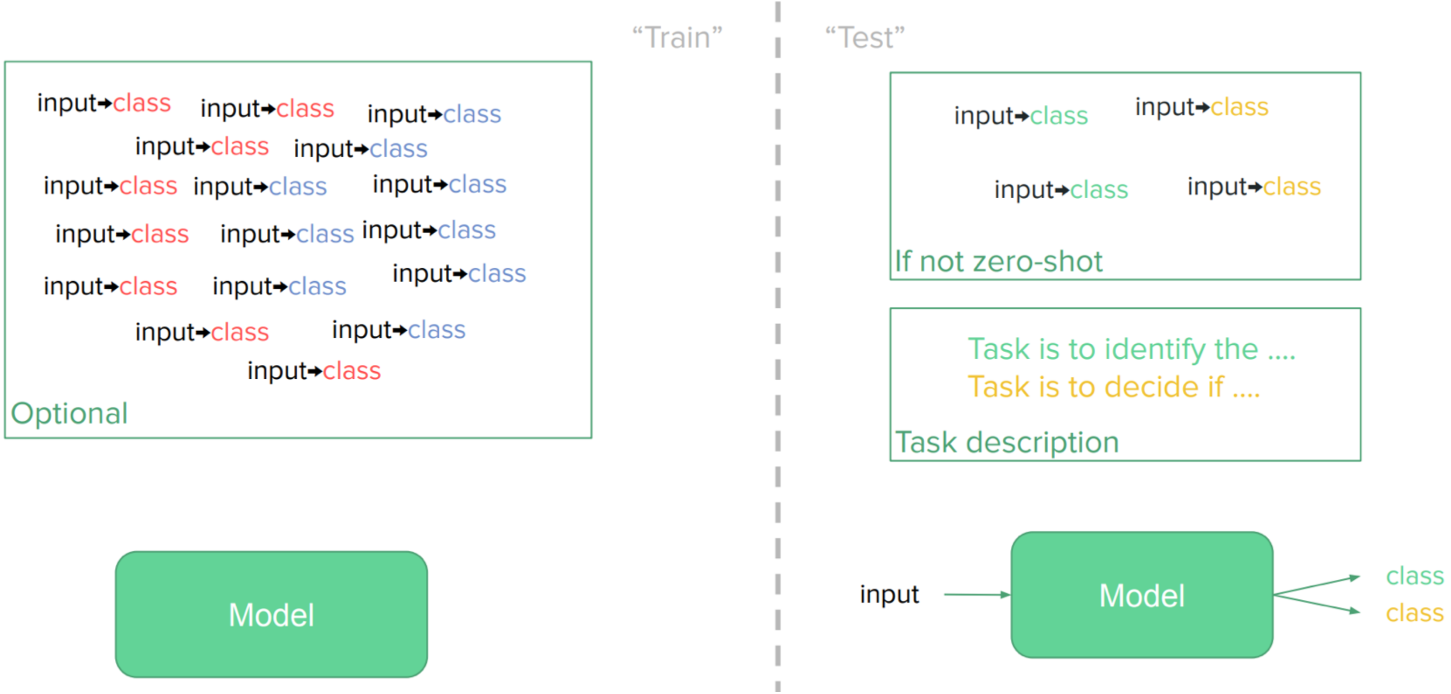
[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# The Broader Context: Supervised Learning



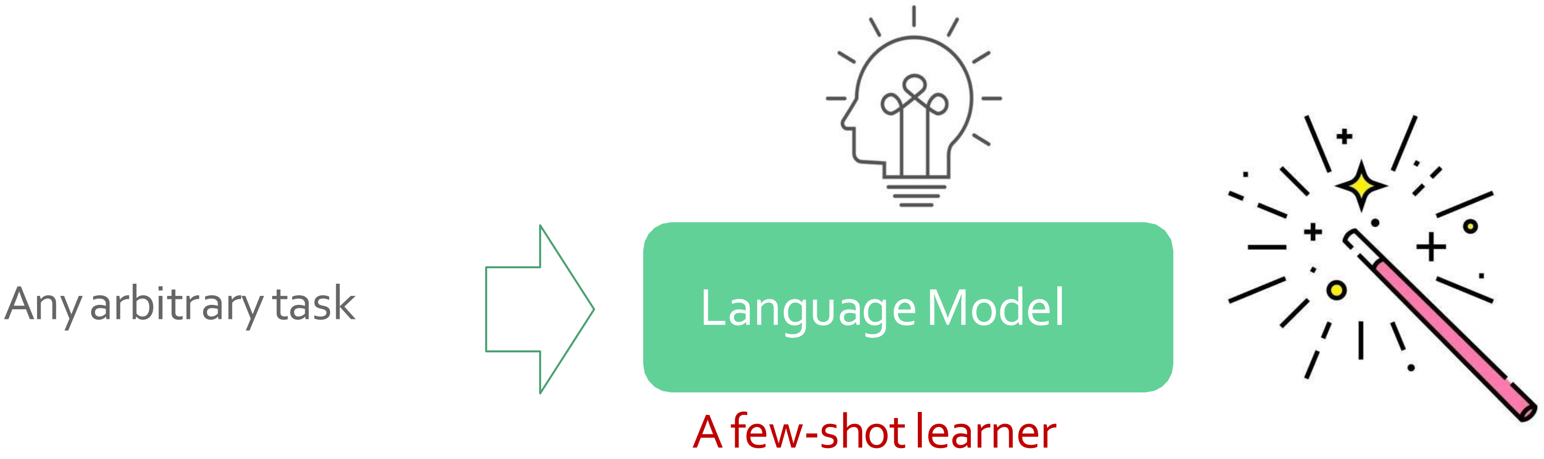
[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# The Broader Context: [Modern] Few-shot Learning



[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# ICL as a General-Purpose Few-Show Learning Mechanism



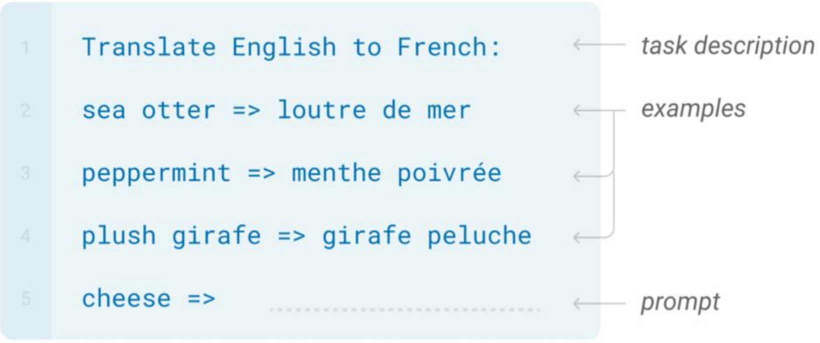
Any arbitrary task

Language Model

A few-shot learner

[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

# Summary



```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese => ..............................
```

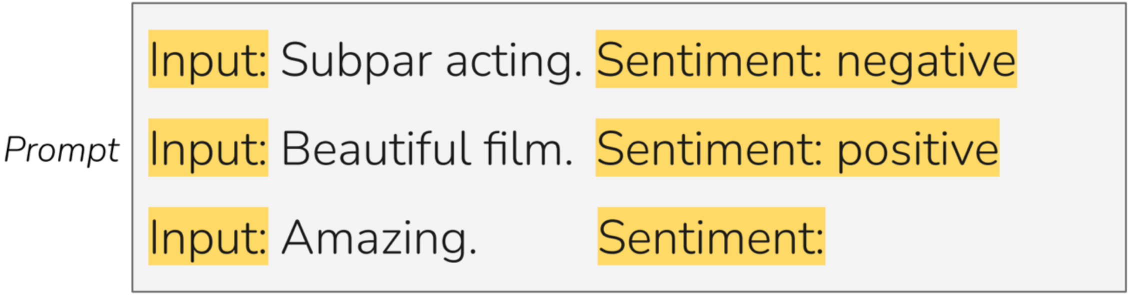
- ICL is a surprising phenomenon
  that came out of large-scale pre-training LMs.

- It allows test-time adaptation via examples.
  - Few examples are often enough—data efficient!
  - No parameter updates—no model replicas!

- When does it work? What are its strength or weaknesses? 🤔

# ICL’s Sensitivity

# LM Prompting: Choices of Encoding



*Prompt*

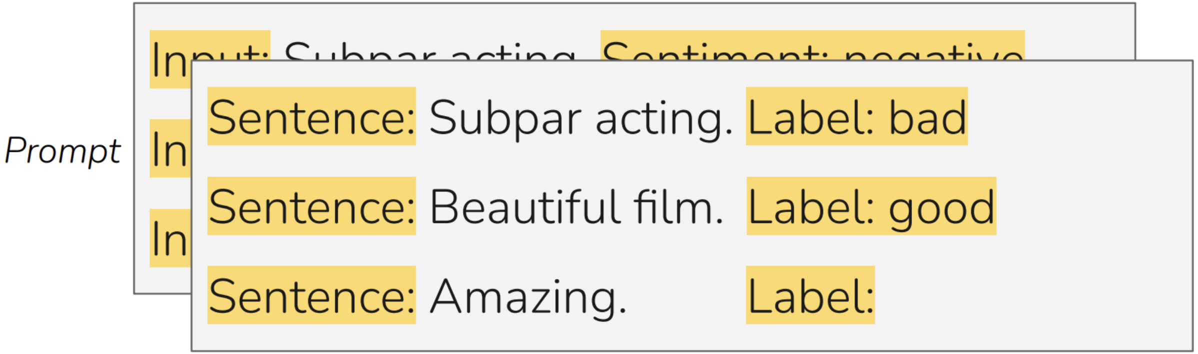
Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

[Slide credit: Eric Wallace]

# LM Prompting: Choices of Encoding



Prompt

Input: Subpar acting. Sentiment: negative

In

In

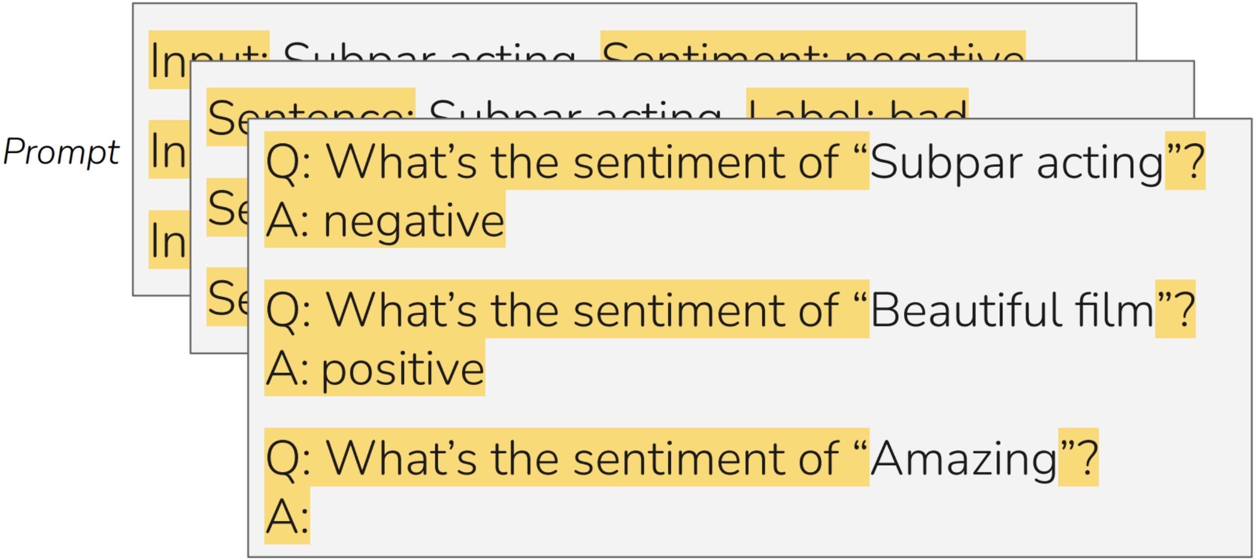
Sentence: Subpar acting. Label: bad

Sentence: Beautiful film. Label: good

Sentence: Amazing. Label:

[Slide credit: Eric Wallace]

# LM Prompting: Choices of Encoding



*Prompt*

Input: Subpar acting. Sentiment: negative

In

In

Sentence: Subpar acting. Label: bad

Se

Se

Q: What's the sentiment of "Subpar acting"?
A: negative

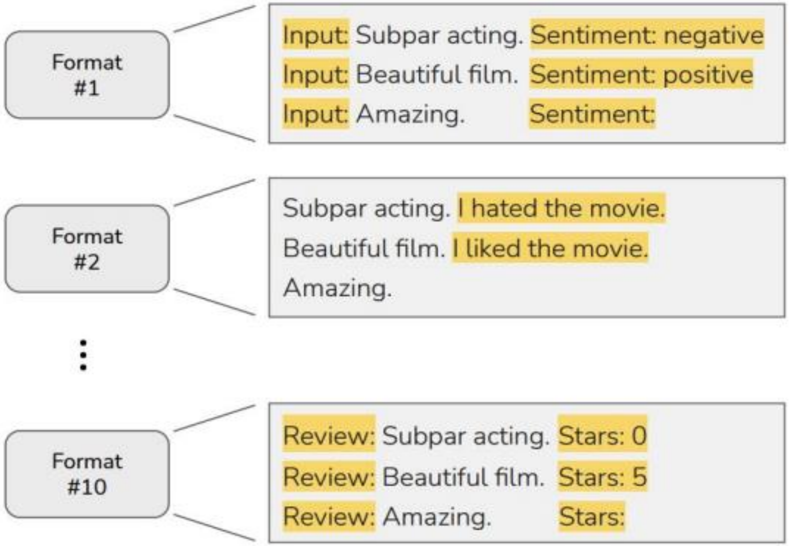
Q: What's the sentiment of "Beautiful film"?
A: positive

Q: What's the sentiment of "Amazing"?
A:

[Slide credit: Eric Wallace]

[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Encoding



Format #1

Input: Subpar acting. Sentiment: negative
Input: Beautiful film. Sentiment: positive
Input: Amazing. Sentiment:

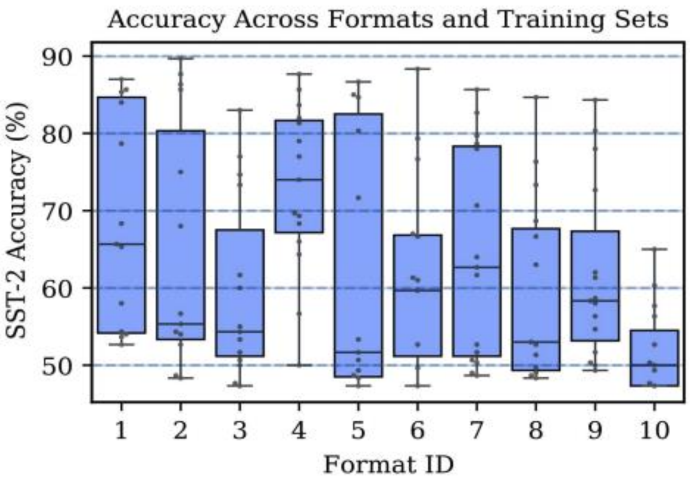
Format #2

Subpar acting. I hated the movie.
Beautiful film. I liked the movie.
Amazing.

⋮

Format #10

Review: Subpar acting. Stars: 0
Review: Beautiful film. Stars: 5
Review: Amazing. Stars:


Accuracy Across Formats and Training Sets

In-context learning is highly sensitive to prompt format (training sets and patterns/verbalizers)

[“Calibrate Before Use: Improving Few-Shot Performance of Language Models.” Zhao et al. 2021]

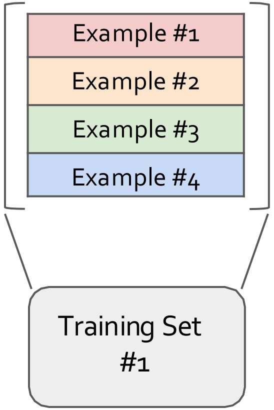
[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Demo. Permutations

Training Set #1

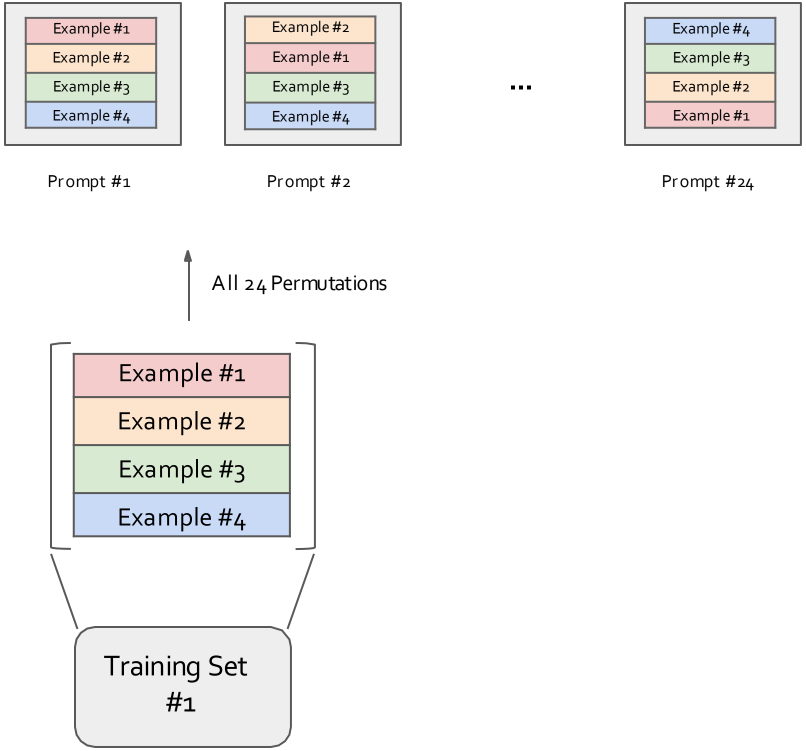
[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Demo. Permutations



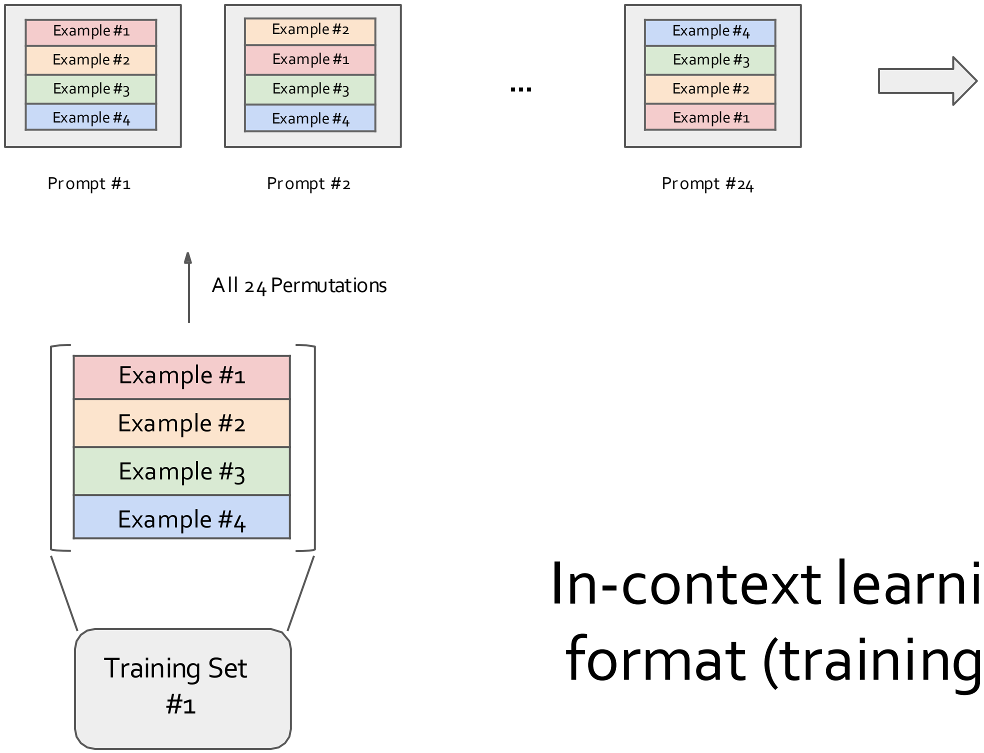
[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Demo. Permutations



[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Demo. Permutations



Example #1
Example #2
Example #3
Example #4

Prompt #1

Example #2
Example #1
Example #3
Example #4

Prompt #2

...

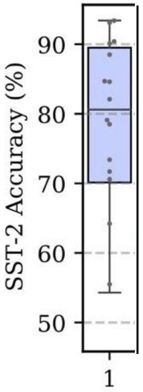
Example #4
Example #3
Example #2
Example #1

Prompt #24

All 24 Permutations

Example #1
Example #2
Example #3
Example #4

Training Set #1

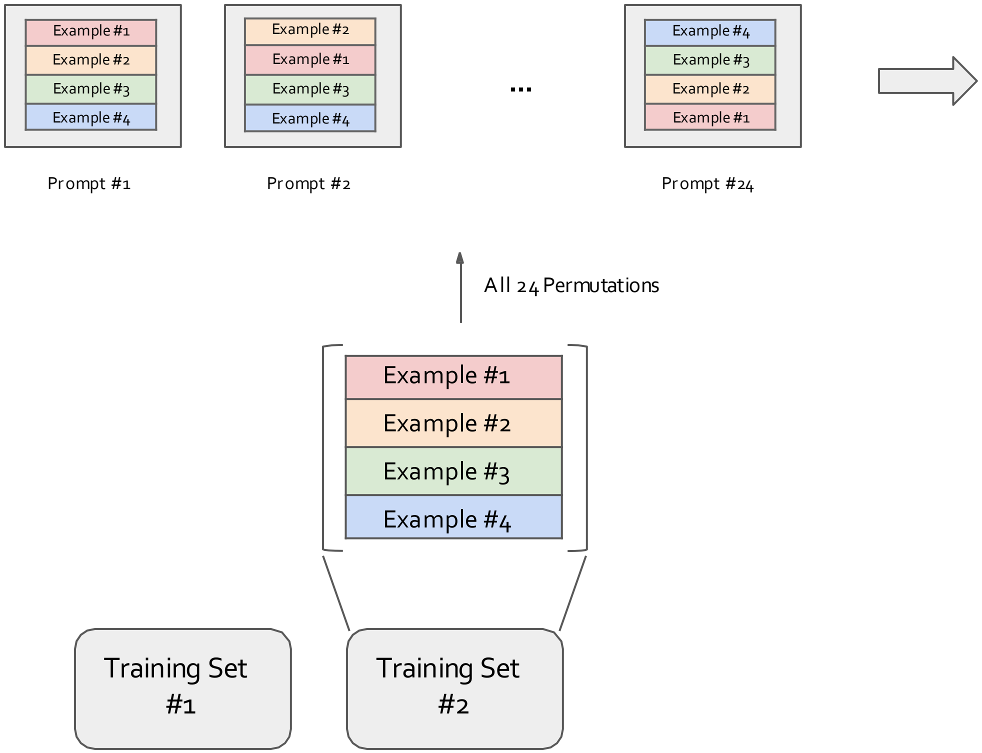


In-context learning is highly sensitive to prompt format (training sets and patterns/verbalizers)

[“Calibrate Before Use: Improving Few-Shot Performance of Language Models.” Zhao et al. 2021]

[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Demo. Permutations



Example #1
Example #2
Example #3
Example #4

Prompt #1

Example #2
Example #1
Example #3
Example #4

Prompt #2

...

Example #4
Example #3
Example #2
Example #1

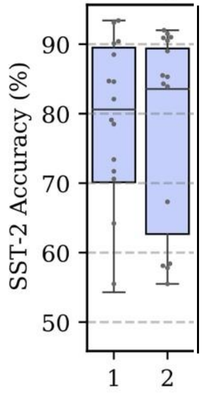
Prompt #24

All 24 Permutations

Example #1
Example #2
Example #3
Example #4

Training Set #1

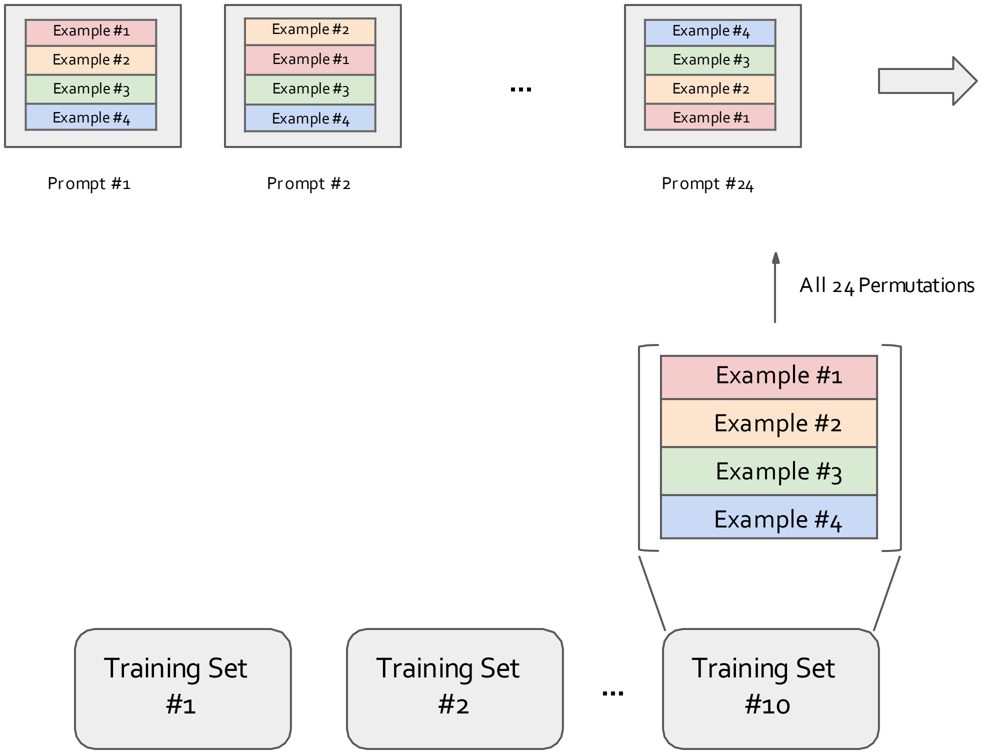
Training Set #2



[“Calibrate Before Use: Improving Few-Shot Performance of Language Models.” Zhao et al. 2021]

[Slide credit: Eric Wallace]

# In-Context Learning: Sensitivity to Demo. Permutations



Example #1
Example #2
Example #3
Example #4

Prompt #1

Example #2
Example #1
Example #3
Example #4

Prompt #2

...

Example #4
Example #3
Example #2
Example #1

Prompt #24

All 24 Permutations

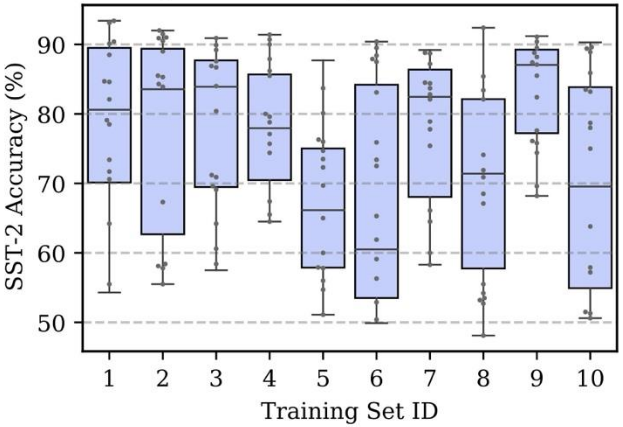
Example #1
Example #2
Example #3
Example #4

Training Set #1

Training Set #2

...

Training Set #10



The choice of demonstrations and their order is quite important.

[“Calibrate Before Use: Improving Few-Shot Performance of Language Models.” Zhao et al. 2021]

# Sensitivity to Wording (Framing) of Prompts

- Framing of prompts matters a lot.

```
Craft a question that 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?  ...
```

```
Generate questions such that you 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.
```

[“Reframing Instructional Prompts to GPTk's Language.” Mishra et al. 2021.]

# Sensitivity to Wording (Framing) of Prompts

- Prompts can often be phrased in a language that are easier to be understood by language models.

- Generally, it is easier for LMs to follow shorter, crisp, itemized prompts.

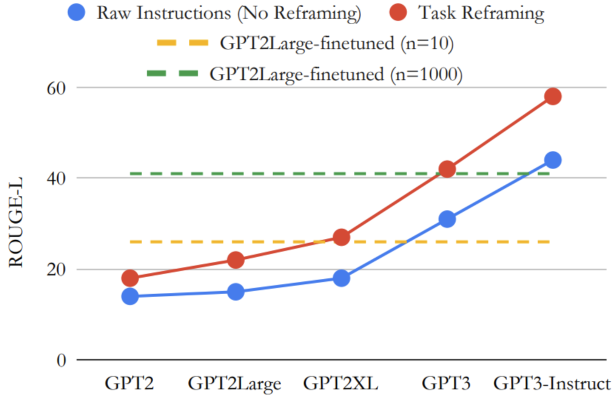


Figure 2: Across a variety of model sizes, reframed prompts consistently show considerable performance gain over raw task instructions (no reframing) in a few-shot learning setup. Since fine-tuning GPT3 is prohibitively expensive, we show the performance of fine-tuning smaller models (horizontal lines). This results indicates that *evaluating* reframed prompts on a large model like GPT3-instruct (red line) might be more effective that *fine-tuning* a smaller model like GPT2Large (green line) with 200× more data. Details of the experiments in §4.

[“Reframing Instructional Prompts to GPTk's Language.” Mishra et al. 2021.]

# Summary Thus Far

- In-context learning:
  - Pre-trained LMs imitated examples provided in their context. (why??)

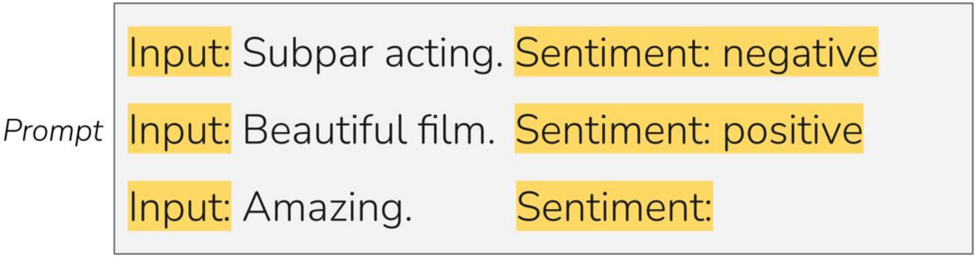
- It turns out there is a huge variance in performance depending on the encoding.
  - The choice of demonstrations, their order, wording, etc.
  - You can treat them as hyper-parameters
  - You should **not** choose these encodings based on the test data.

- Generally, you want to an encoding that makes your task similar to language modeling — closer to what is observed during pretraining.

# What Causes These Variances?

- Here we will provide several justifications …

# Majority Label Bias



*Prompt*

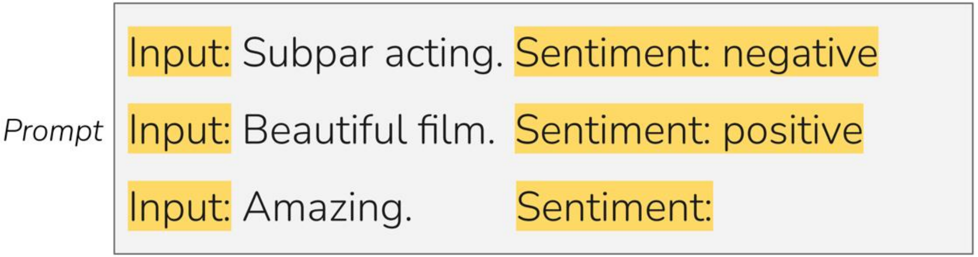
Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

- Among 4 demonstrations, count how many are “positive”.
- Then check if the model output correlates with the number of “positive” demos.

# Majority Label Bias

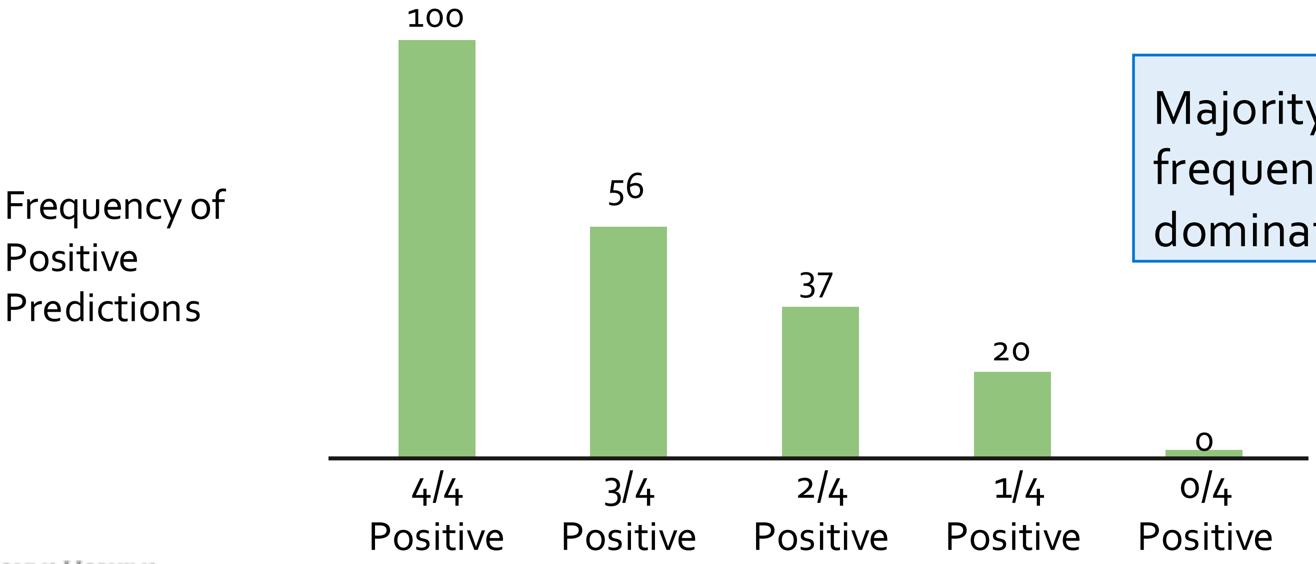


Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

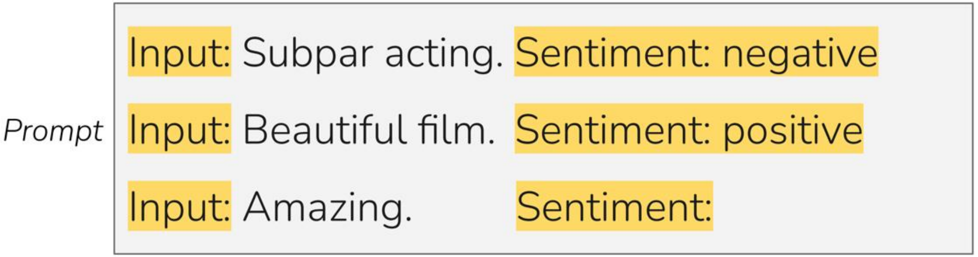
- Among 4 demonstrations, count how many are "positive".
- Then check if the model output correlates with the number of "positive" demos.



Majority label bias: frequent training answers dominate predictions.

["Calibrate Before Use: Improving Few-Shot Performance of Language Models." Zhao et al. 2021]

# Recency Bias



Prompt

Input: Subpar acting. Sentiment: negative

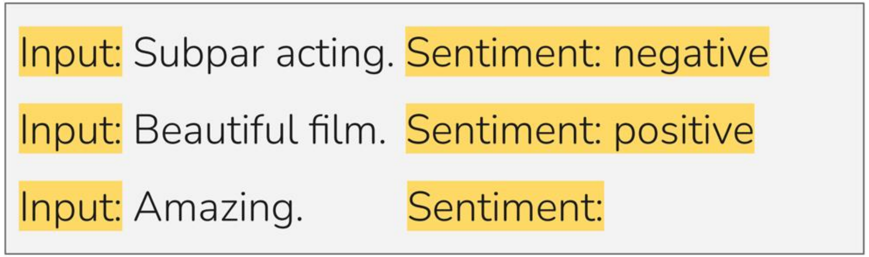
Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

- Check if the label of the most-recent demo biases the model output.

# Recency Bias

*Prompt*



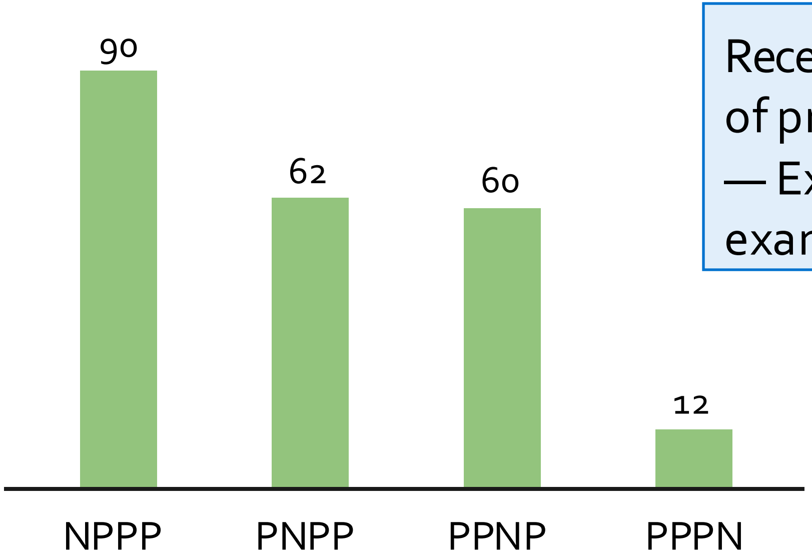
Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

- Check if the label of the most-recent demo biases the model output.

Frequency of Positive Predictions

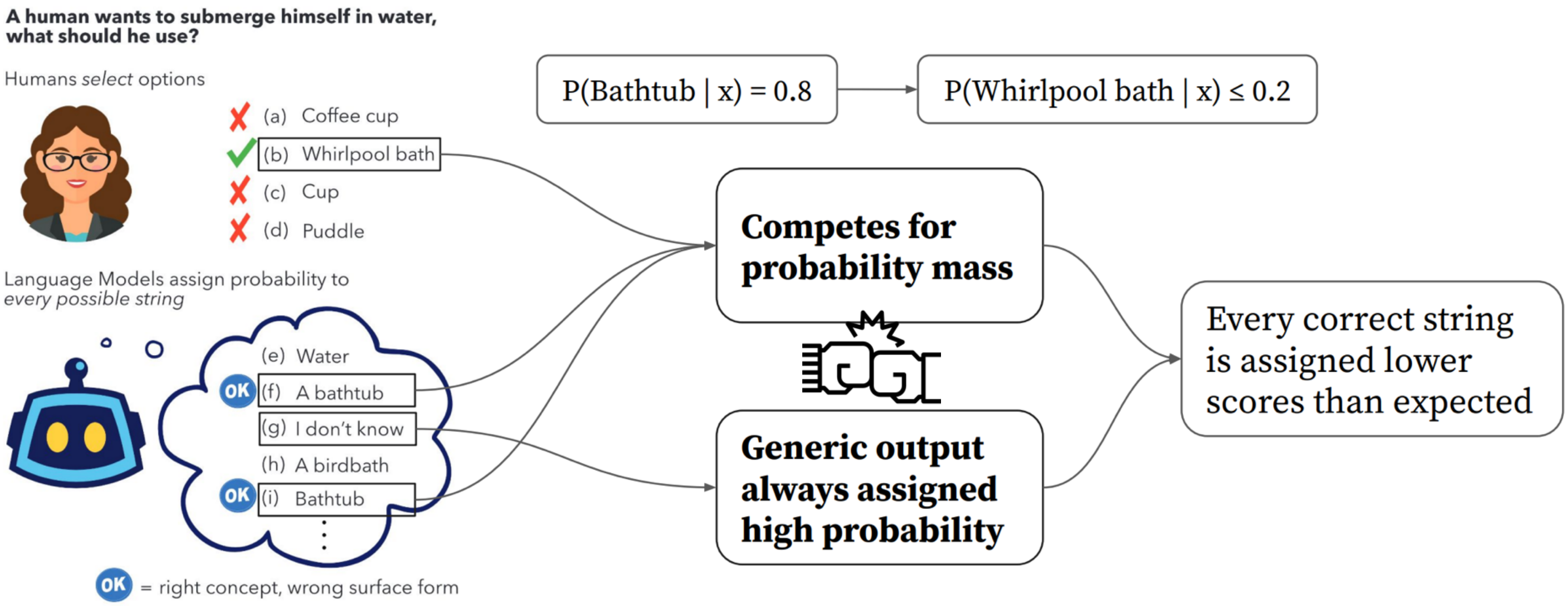


Recency bias: examples near end of prompt dominate predictions — Explains variance across example permutations!

[“Calibrate Before Use: Improving Few-Shot Performance of Language Models.” Zhao et al. 2021]

Bonus

# Surface Form Competition



**A human wants to submerge himself in water, what should he use?**

Humans *select* options

✗ (a) Coffee cup

✓ (b) Whirlpool bath

✗ (c) Cup

✗ (d) Puddle

Language Models assign probability to *every possible string*

(e) Water

OK (f) A bathtub

(g) I don't know

(h) A birdbath

OK (i) Bathtub

⋮

OK = right concept, wrong surface form

P(Bathtub | x) = 0.8 → P(Whirlpool bath | x) ≤ 0.2

**Competes for probability mass**

**Generic output always assigned high probability**

Every correct string is assigned lower scores than expected

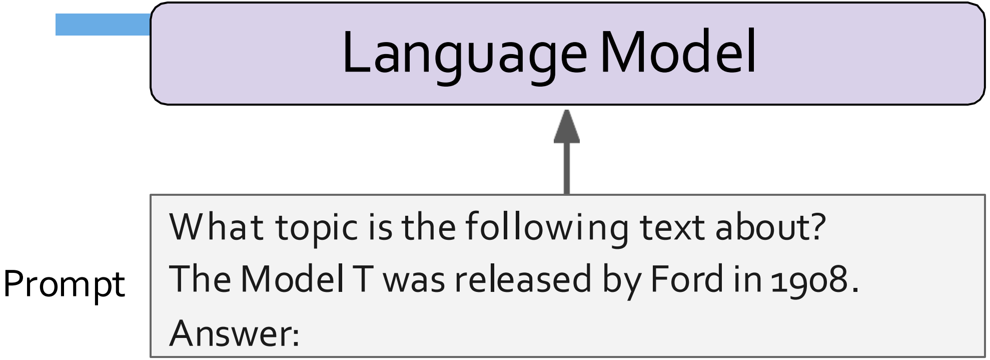
Surface forms are competing for probability mass (Holtzman et al 2021).

[Slide credit: Howard Yen, Vishvak Murahari]

# Common Token Bias

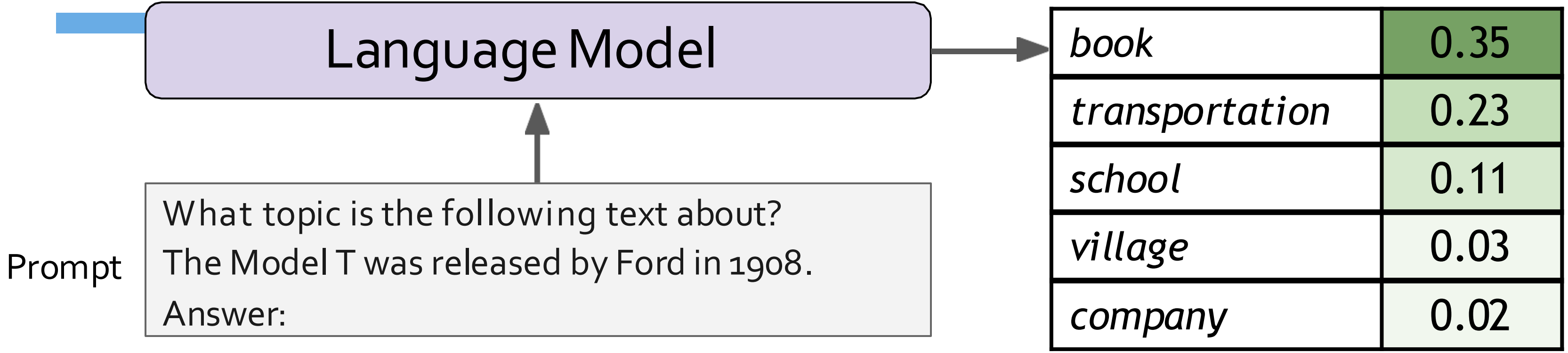
- Is a bias toward words that are more frequently observed in pre-training data?

# Common Token Bias



What topic is the following text about?
The Model T was released by Ford in 1908.
Answer:

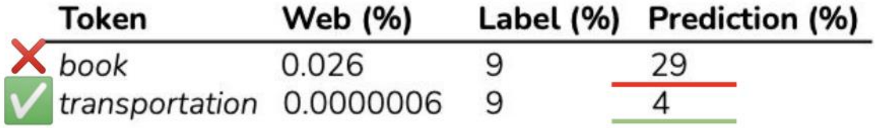
# Common Token Bias



What topic is the following text about?
The Model T was released by Ford in 1908.
Answer:

| Token | Prob |
|---|---|
| *book* | 0.35 |
| *transportation* | 0.23 |
| *school* | 0.11 |
| *village* | 0.03 |
| *company* | 0.02 |

Model is biased towards predicting the incorrect frequent token "book" even when both "book" and "transportation" are equally likely labels in the dataset

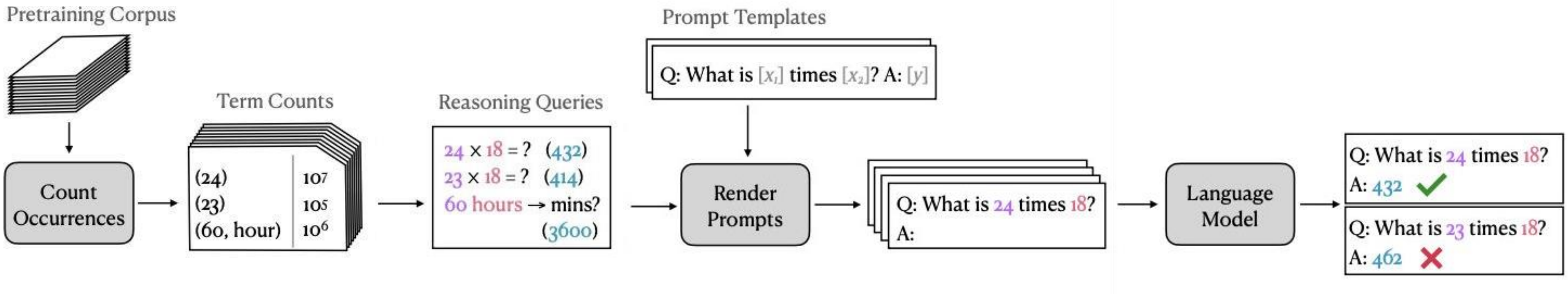


|  | Token | Web (%) | Label (%) | Prediction (%) |
|---|---|---|---|---|
| ❌ | *book* | 0.026 | 9 | 29 |
| ✅ | *transportation* | 0.0000006 | 9 | 4 |

- Common token bias: common n-grams dominate predictions
  - helps explain variance across prompt formats

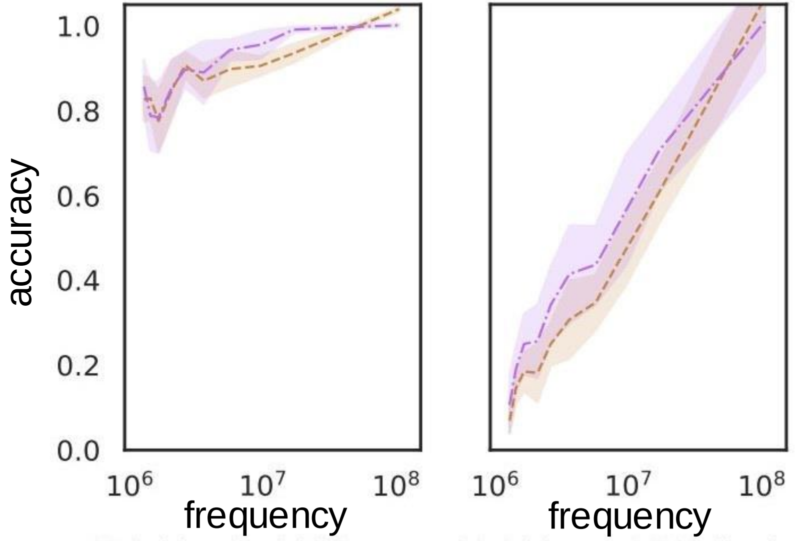
# Impact of Pretraining Term Frequencies

- For each task, identify relevant terms from each instance—numbers and units
- Count co-occurrences of these terms in the pretraining data (term pairs or triples within a fixed window)



["Impact of Pretraining Term Frequencies on Few-Shot Reasoning" Razeghi et al. 2022.]

# Impact of Pretraining Term Frequencies



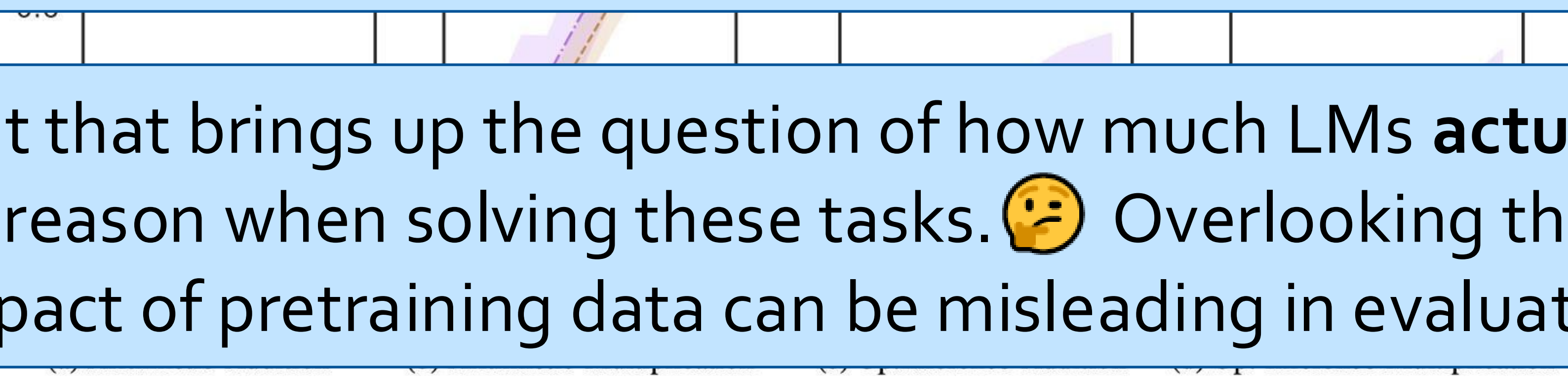
(a) Arithmetic-Addition (b) Arithmetic-Multiplication

In-context learning performance is highly correlated with term frequencies during pretraining

["Impact of Pretraining Term Frequencies on Few-Shot Reasoning" Razeghi et al. 2022.]

# Impact of Pretraining Term Frequencies

This may also indicate that, demonstrations do not teach a new task; instead, it is about locating an already-learned task during pretraining (Reynolds & McDonell, 2021)



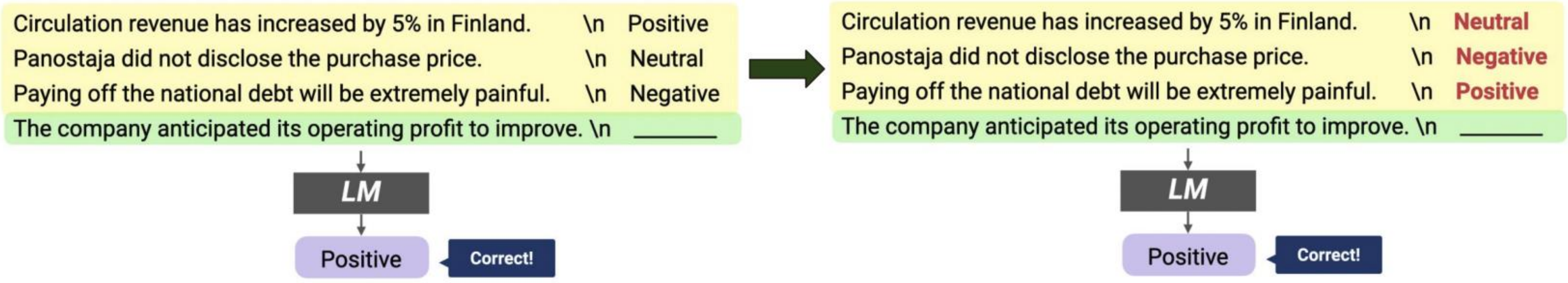
But that brings up the question of how much LMs **actually** reason when solving these tasks. 🤔 Overlooking the impact of pretraining data can be misleading in evaluation!

In-context learning performance is highly correlated with term frequencies during pretraining

["Impact of Pretraining Term Frequencies on Few-Shot Reasoning" Razeghi et al. 2022.]

# Impact of Input-Output Mapping

- Study the effect of randomizing labels in demonstrations.
  - Randomly sample a label from the correct label space



Circulation revenue has increased by 5% in Finland. \n Positive
Panostaja did not disclose the purchase price. \n Neutral
Paying off the national debt will be extremely painful. \n Negative
The company anticipated its operating profit to improve. \n ______

Prompt with true labels

Circulation revenue has increased by 5% in Finland. \n Neutral
Panostaja did not disclose the purchase price. \n Negative
Paying off the national debt will be extremely painful. \n Positive
The company anticipated its operating profit to improve. \n ______

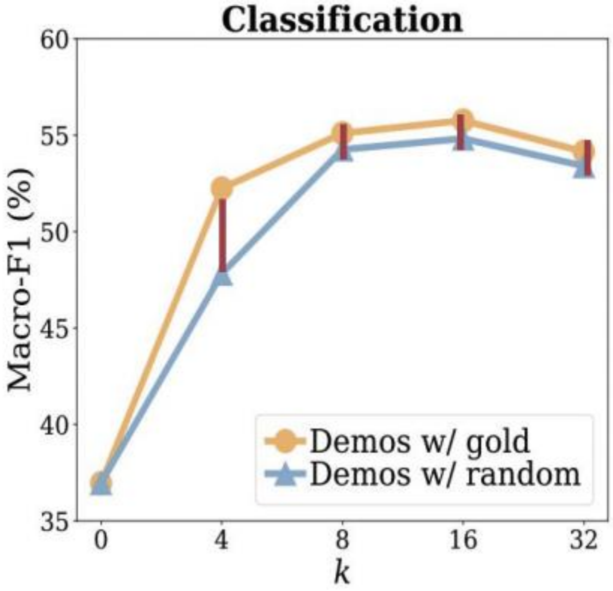
Prompt with random labels

["Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?" Min et al. 2022.]

# Impact of Input-Output Mapping

- Vary number of demonstrations.
- Models see a small performance drop (0–5%) with random labels

- **Takeaway:**
  - Ground truth input-label mapping in the prompt is not as important as we thought!



["Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?" Min et al. 2022.]

# Summary Thus Far

- In-context learning, while being fascinating, is highly unstable/brittle.

- Next: why does ICL emerge?

[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

Bonus

# Why Does ICL Emerge?

Bonus

# Why Does ICL Emerge?

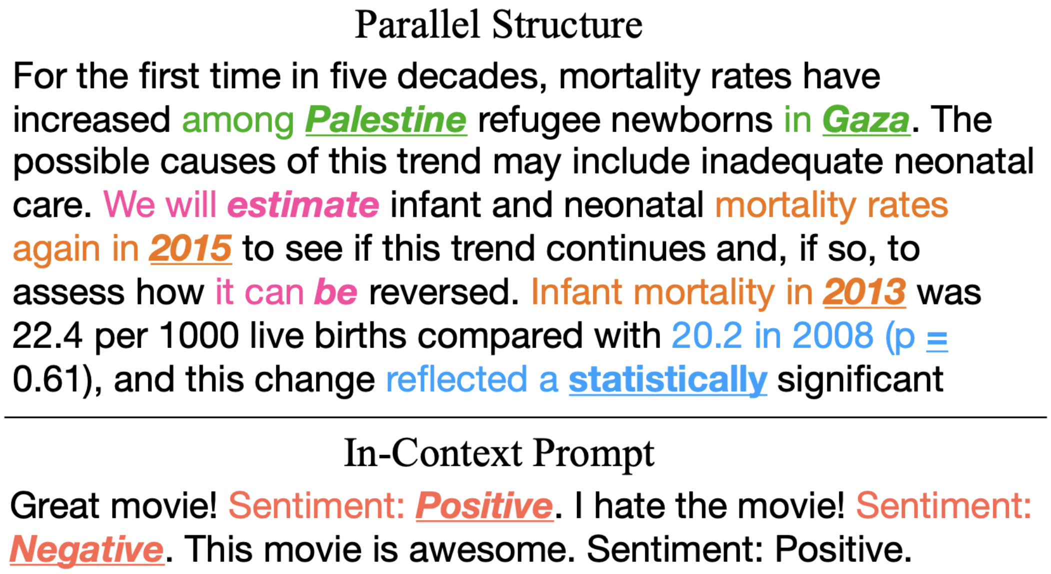
- We don’t know!
- We have partial empirical explanations.
- And some theoretical analogies.
- But none of them fully explain ICL.

# ICL Emerges Due to Parallel Structures in Pre-training

Bonus

- Natural text is abundant with “parallel structure”.
  - Various forms of coreference
  - Tables
  - Lists
  - …

- **Hypothesis:** Such parallel structure is essential for the emergence of ICL.



Parallel Structure

For the first time in five decades, mortality rates have increased among ***Palestine*** refugee newborns in ***Gaza***. The possible causes of this trend may include inadequate neonatal care. We will ***estimate*** infant and neonatal mortality rates again in ***2015*** to see if this trend continues and, if so, to assess how it can ***be*** reversed. Infant mortality in ***2013*** was 22.4 per 1000 live births compared with 20.2 in 2008 (p ***=*** 0.61), and this change reflected a ***statistically*** significant

In-Context Prompt

Great movie! Sentiment: ***Positive***. I hate the movie! Sentiment: ***Negative***. This movie is awesome. Sentiment: Positive.

Parallel Structures in Pre-training Data Yield In-Context Learning, 2024

Bonus

# In-context Learning as Bayesian Inference

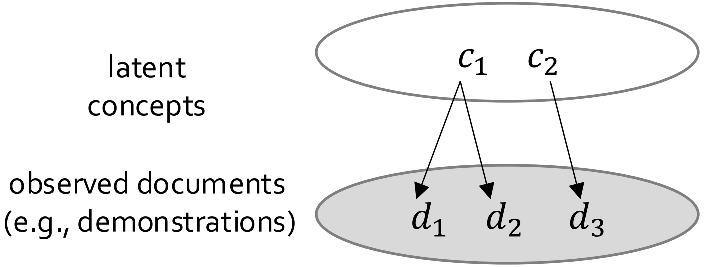
- (Xie et al., 2022) try to explain ICL as an implicit Bayesian inference.

**Idea:**

- (Pre-trained LM learn to represent "concepts", i.e. the ideas described by words.
- ICL enables LMs to "locate" the learned concepts.

- Can formulate this intuition as a Bayesian inference
  - Prior over latent "concepts"
  - Likelihood describes connection between text and concepts
  - Given an incomplete doc, use Bayes formula to infer what concept is likely it is generated from and then complete the document.

- Does not explain everything.
  - GPT-3 can handle "unseen" concepts

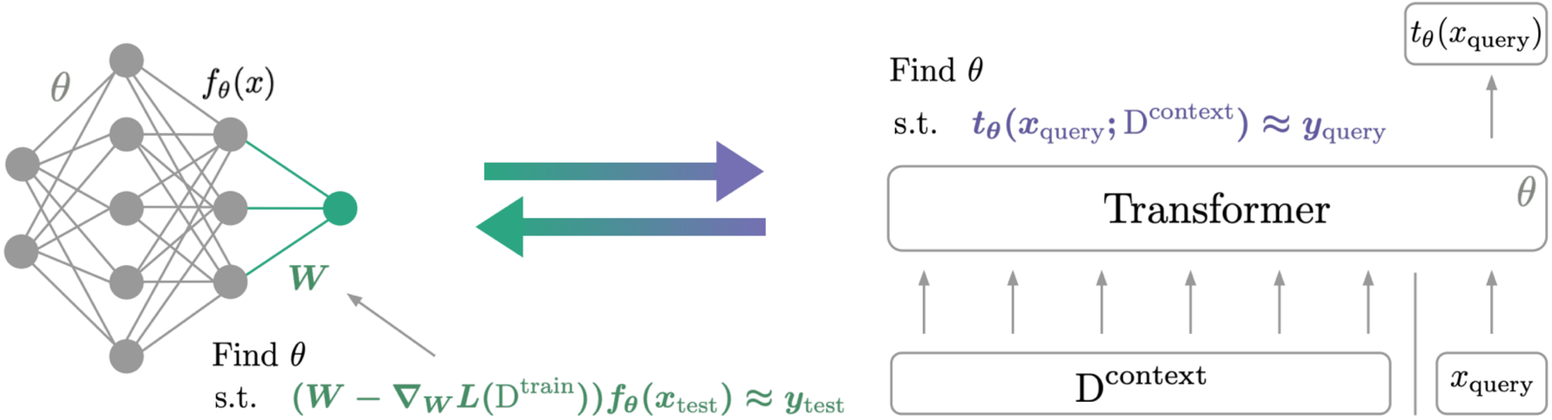


Bonus

# In-context Learning as Gradient Descent?

[von Oswald et al. 2022; Akyurek et al. 2022; Dai et al. 2022, …]

- Hypothesis: ICL is implicitly equivalent to SGD on in-context demonstrations



$\theta$
$f_\theta(x)$
$W$
Find $\theta$
s.t. $(W - \nabla_W L(\mathrm{D}^{\text{train}}))f_\theta(x_{\text{test}}) \approx y_{\text{test}}$
$t_\theta(x_{\text{query}})$
Find $\theta$
s.t. $t_\theta(x_{\text{query}}; \mathrm{D}^{\text{context}}) \approx y_{\text{query}}$
Transformer
$\theta$
$\mathrm{D}^{\text{context}}$
$x_{\text{query}}$

- However, unclear whether these formalisms hold in reality.

[Do Pre-trained Transformers Really Learn In-context by Gradient Descent?, 2024]

Bonus

# ICL Operation: Task learning vs retrieval

- TODO

Bonus

# Summary

- In-context learning has been a promising few-shot learning approach
  - No gradient updates → Much easier to use large models!

- Many open questions …
  - Understanding how/why it works,
  - Can we predict whether in-context learning would work on a given task or not?

# Prompting to Solve Multi-step Problems

# Some Problems Involve Reasoning

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**

Arithmetic Reasoning (AR) (+ −×÷…)

Q: Take the last letters of the words in "Elon Musk" and concatenate them

A: The answer is **nk.**

Symbolic Reasoning (SR)

Q: What home entertainment equipment requires cable? Answer Choices: (a) radio shack (b) substation (c) television (d) cabinet

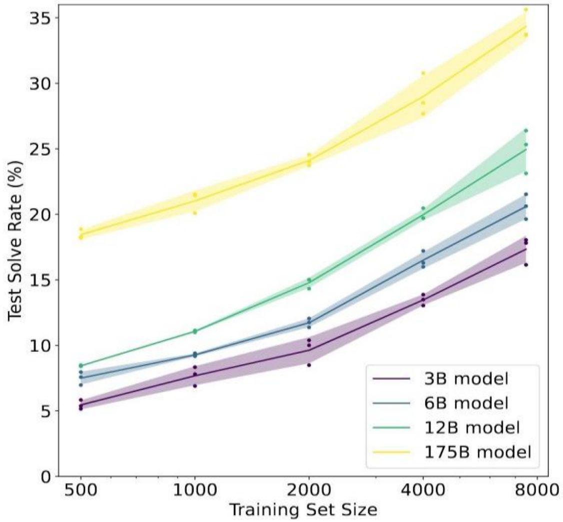
A: The answer is **(c).**

Commonsense Reasoning (CR)

# Fine-tuning on Reasoning Problems

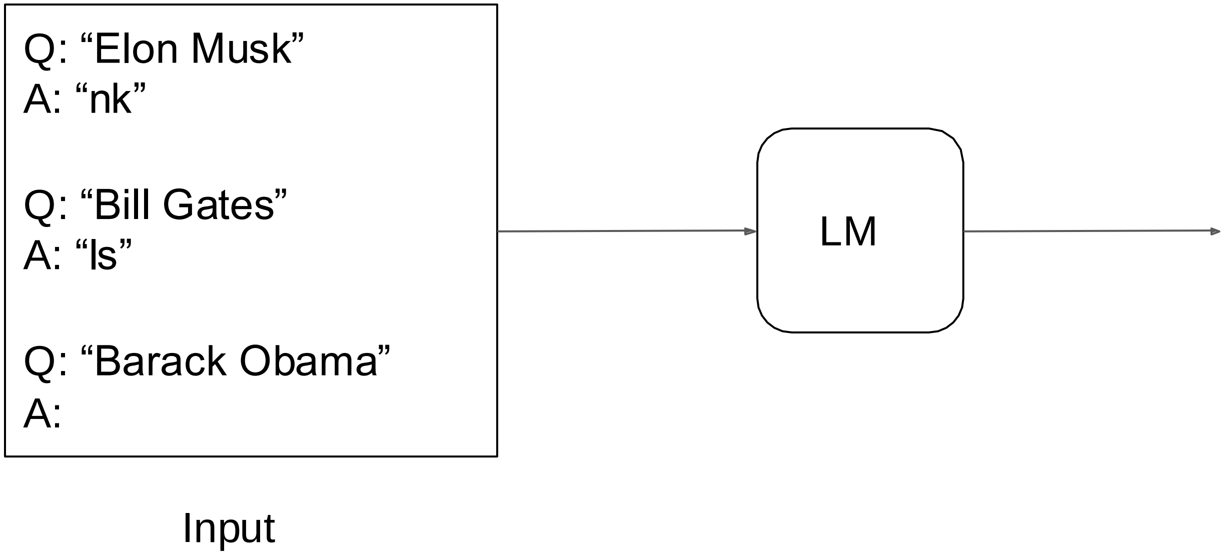
(Cobbe et al. 2021)

- Fine-tune LMs on GSM8K (arithmetic reasoning)
- One may conjecture that, to achieve >80%, one needs 100x more training data for 175B model
- Another option is to increase model sizes, which is expensive.



- Other than these, how else can we improve the model performance on tasks that require multi-step reasoning?

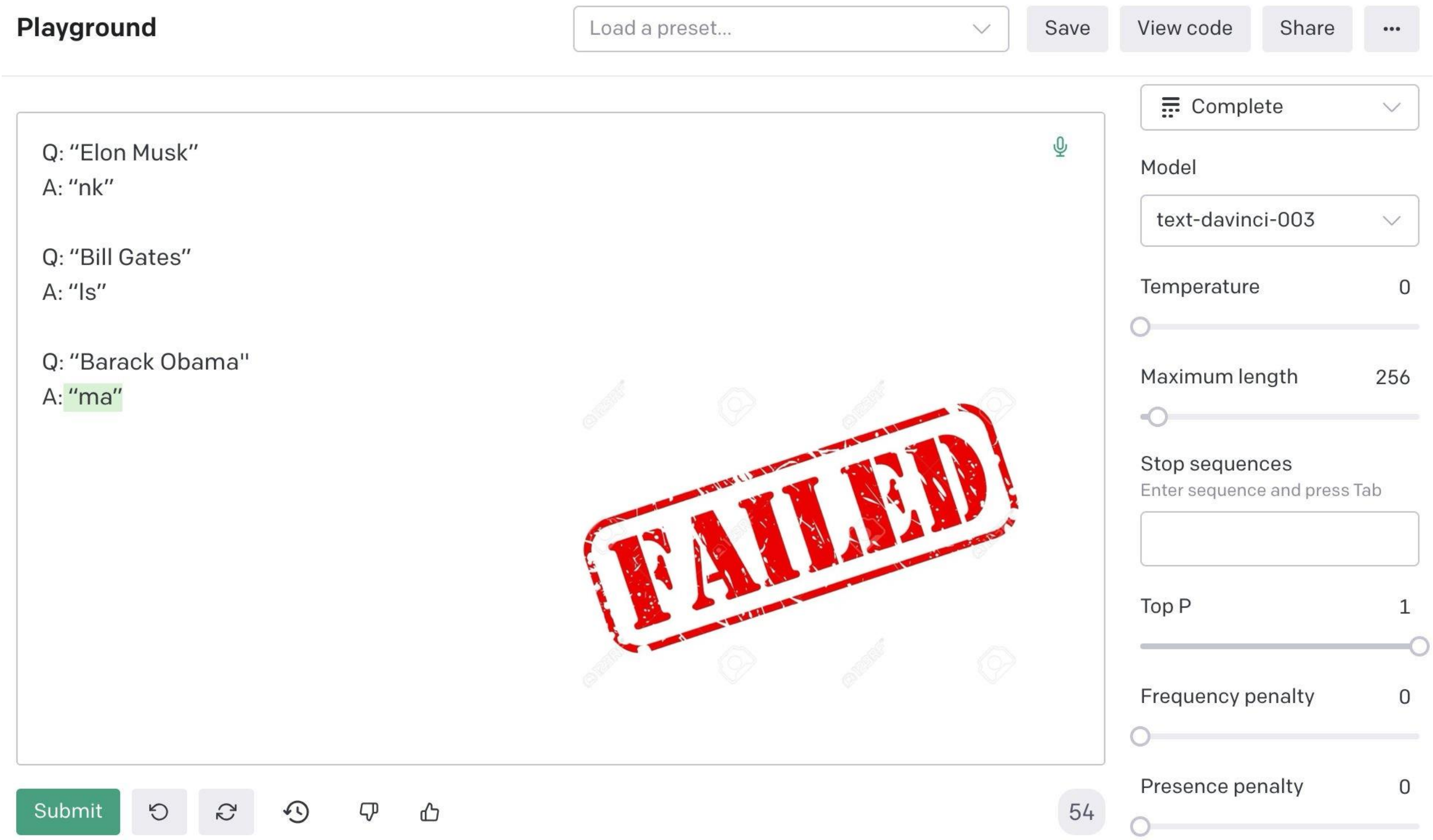
# Vanilla ICL on Reasoning Problems



Q: “Elon Musk”
A: “nk”

Q: “Bill Gates”
A: “ls”

Q: “Barack Obama”
A:



Q: “Elon Musk”
A: “nk”

Q: “Bill Gates”
A: “ls”

Q: “Barack Obama"
A: “ma”

Complete

Model
text-davinci-003

Temperature 0

Maximum length 256

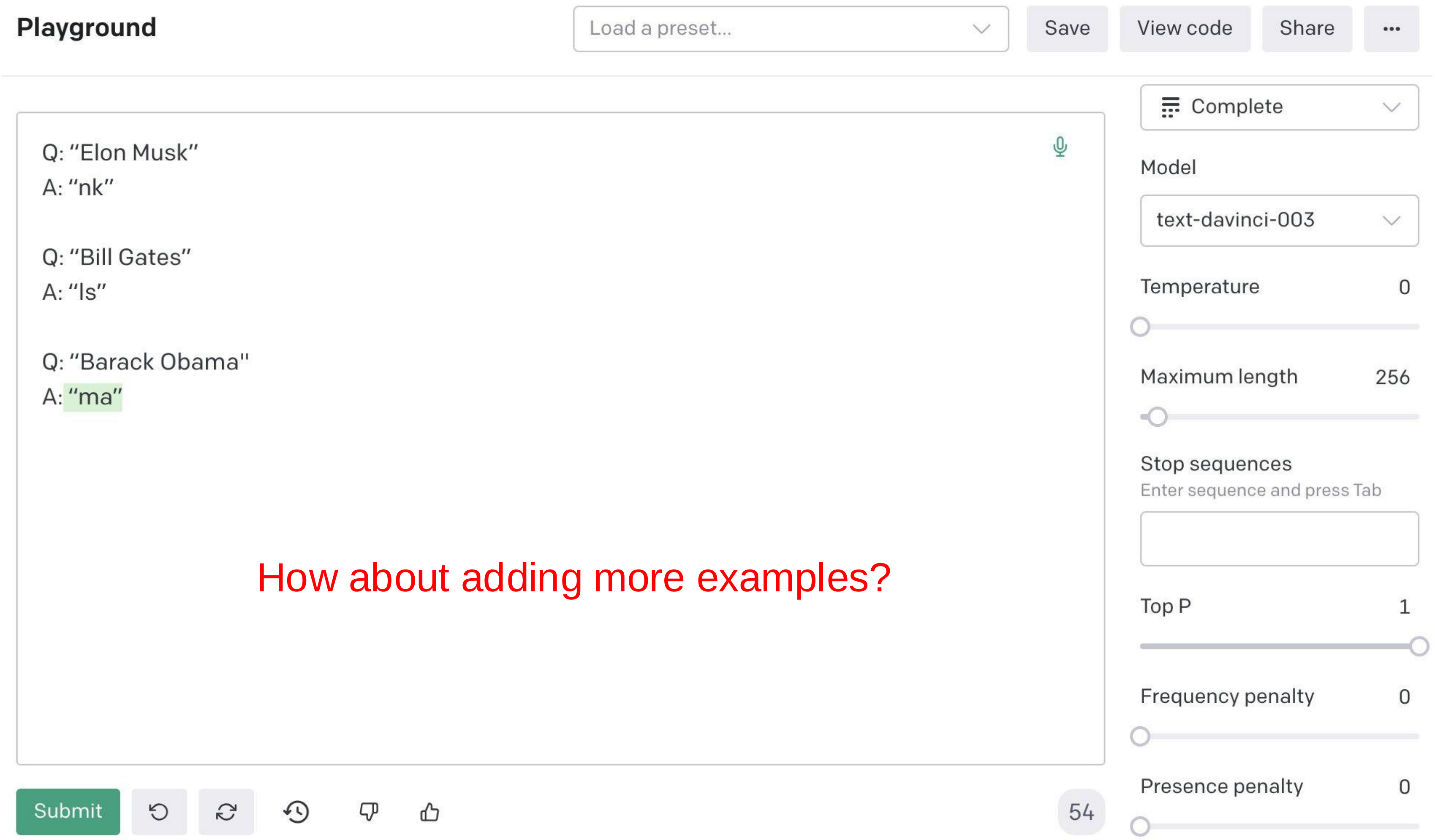
Stop sequences
Enter sequence and press Tab

Top P 1

Frequency penalty 0

Presence penalty 0

[Denny Zhou]



Q: “Elon Musk”
A: “nk”

Q: “Bill Gates”
A: “ls”

Q: “Barack Obama"
A: “ma”

Model: text-davinci-003

Temperature 0

Maximum length 256

Stop sequences
Enter sequence and press Tab

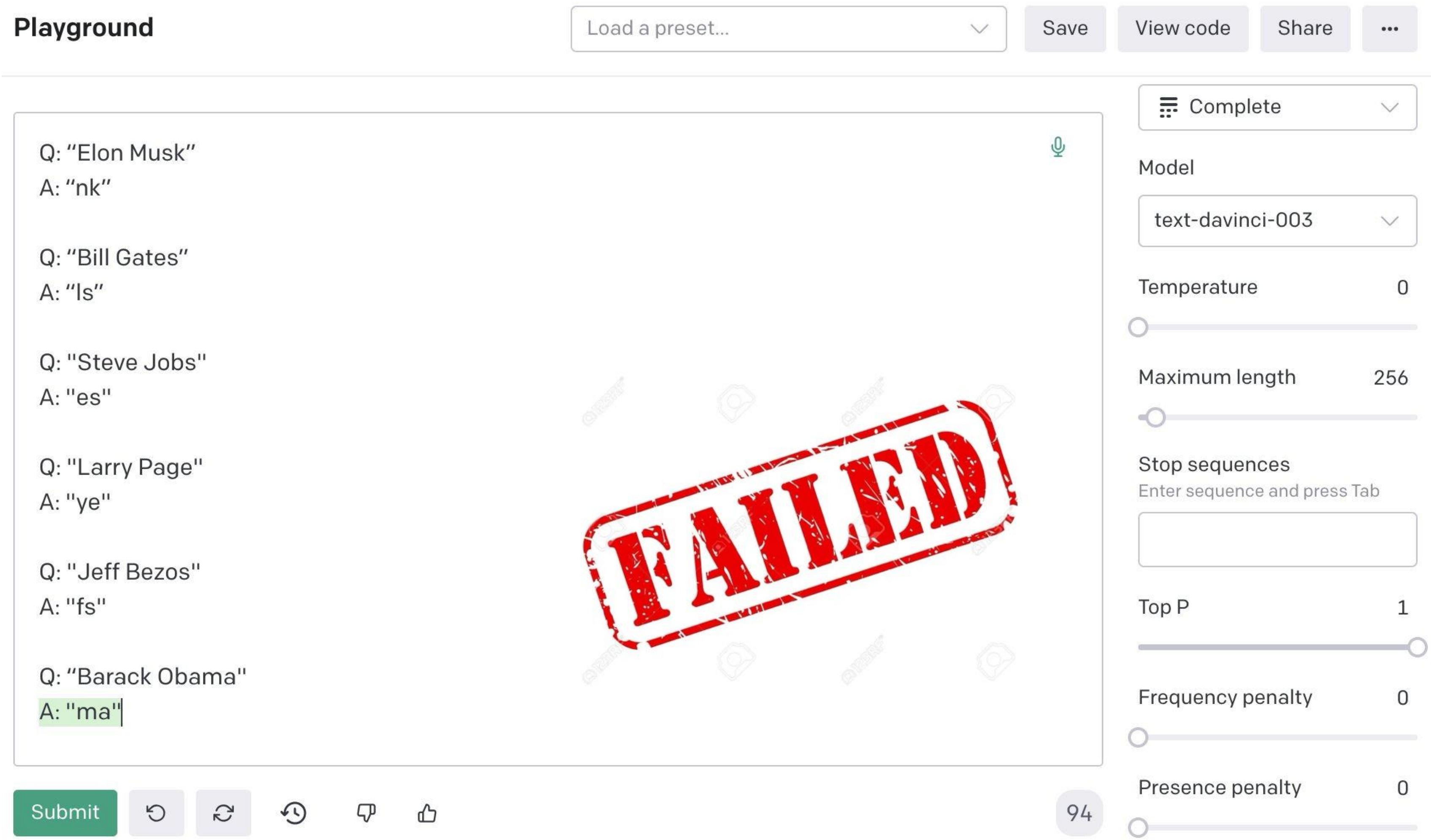
Top P 1

Frequency penalty 0

Presence penalty 0

How about adding more examples?

[Denny Zhou]



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

Complete

Model: text-davinci-003

Temperature 0

Maximum length 256

Stop sequences
Enter sequence and press Tab

Top P 1

Frequency penalty 0

Presence penalty 0

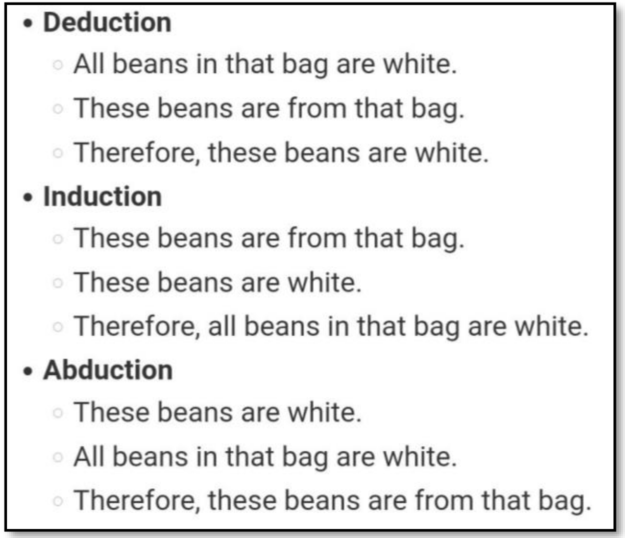
[Denny Zhou]

# Reasoning Problems via Multi-Step Prompting

- **Basic idea:** Rather than showing input-output pairs, prompting the model such that it shows its proof steps.

- **Note:** ideas around models that are capable of multi-step reasoning go way back.
  - Aristotle (deduction),
  - Hume (induction),
  - Peirce (abduction)
  - Lots of other works in pre-LM era
  - Namely, my Ph.D. thesis ☺ on multi-step reasoning in semantic representations of language!

    [Reasoning-Driven Question-Answering for Natural Language Understanding]



- **Deduction**
  - All beans in that bag are white.
  - These beans are from that bag.
  - Therefore, these beans are white.
- **Induction**
  - These beans are from that bag.
  - These beans are white.
  - Therefore, all beans in that bag are white.
- **Abduction**
  - These beans are white.
  - All beans in that bag are white.
  - Therefore, these beans are from that bag.

# 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".

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:

[Denny Zhou]

# 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".

thought

Q: “Bill Gates”

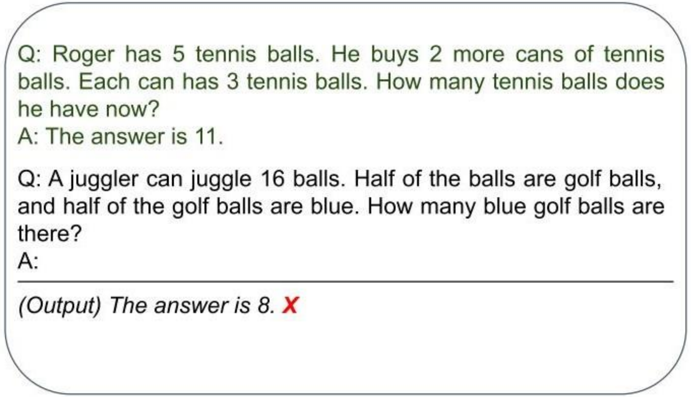
A: the last letter of "Bill" is "l". the last letter of "Gates" is "s". Concatenating "l", "s" leads to "ls". so the output is "ls".

Q: “Barack Obama"

A: 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".

[Denny Zhou]

# CoT: Adding “thought” before “answer”



(a) Few-shot

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.

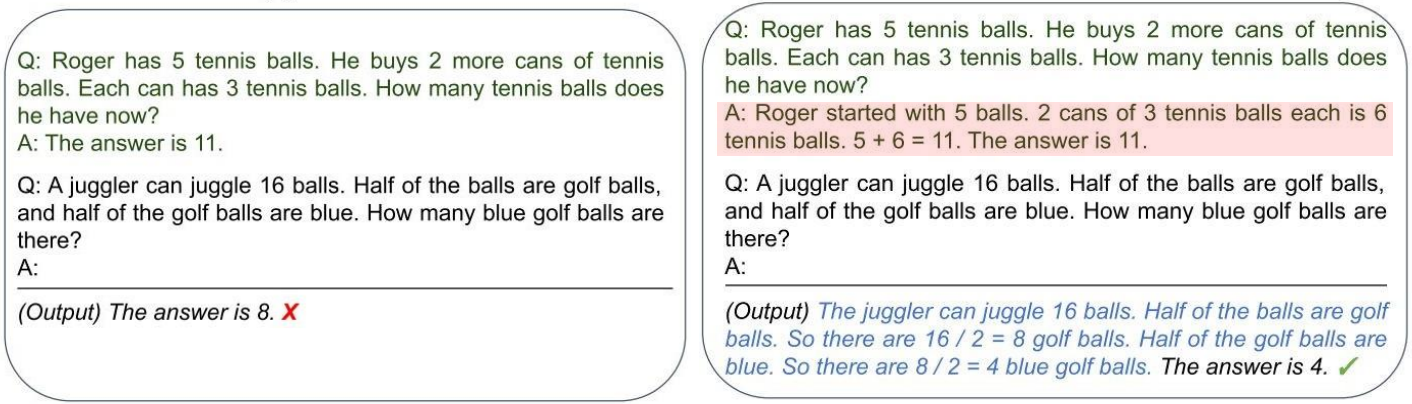
Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The answer is 8.* ✗

# CoT: Adding “thought” before “answer”

(a) Few-shot

(b) Few-shot-CoT （Wei et al., 2022）



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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The answer is 8.* ✗

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4.* ✓

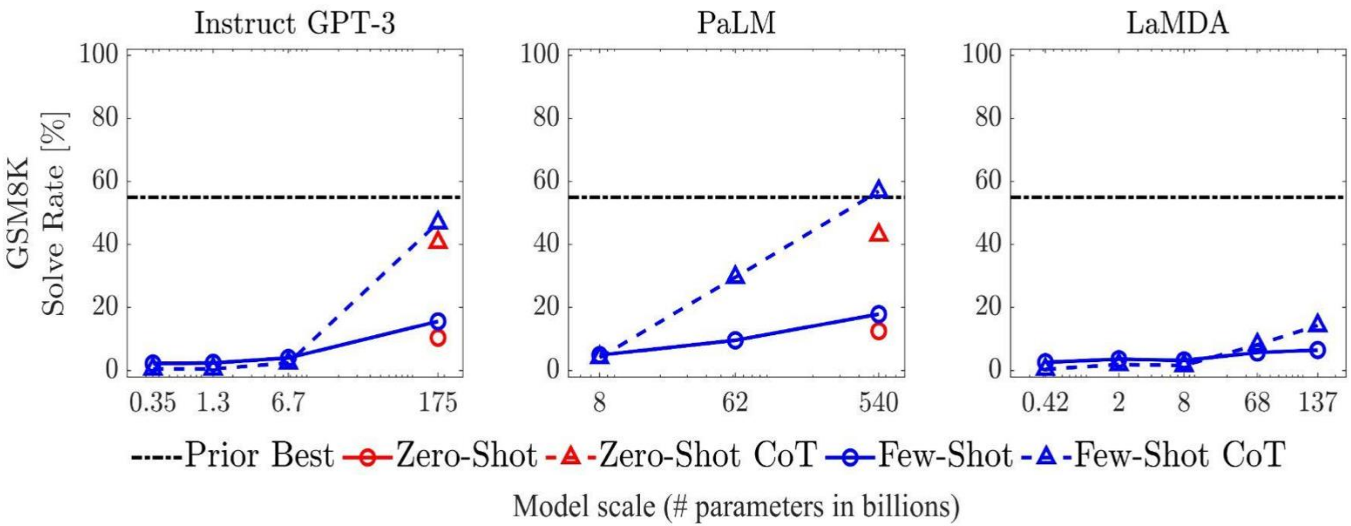
Step-by-step demonstration

Step-by-step Answer

The use of natural language to describe rationales is critical for the success of CoT.

# CoT Prompting: Empirical Results

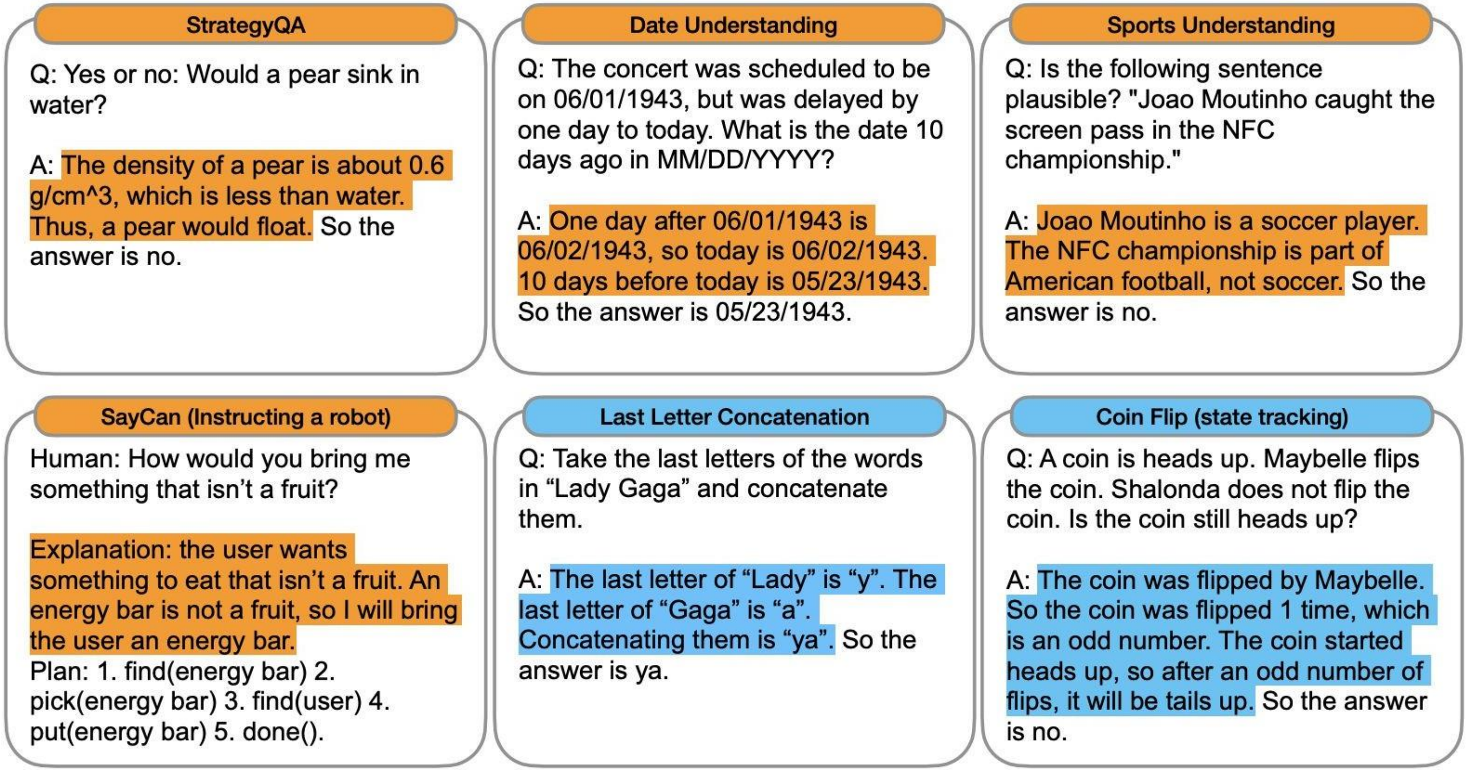
- **Setup:** show demonstrations that contain the decompositions
- The gains of multi-step prompting increases with scale.
- Prompting achieves better perf than [smaller] models that are fine-tuned on a lot more data.



["Chain of thought prompting elicits reasoning in large language models", Wei et al. 2022]

# Apply CoT to Any Task

Though each task's demonstrations need to be "engineered" manually! ☹



**StrategyQA**

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm^3, which is less than water. Thus, a pear would float. So the answer is no.

**Date Understanding**

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.

**Sports Understanding**

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.

**SayCan (Instructing a robot)**

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().

**Last Letter Concatenation**

Q: Take the last letters of the words in "Lady Gaga" and concatenate them.

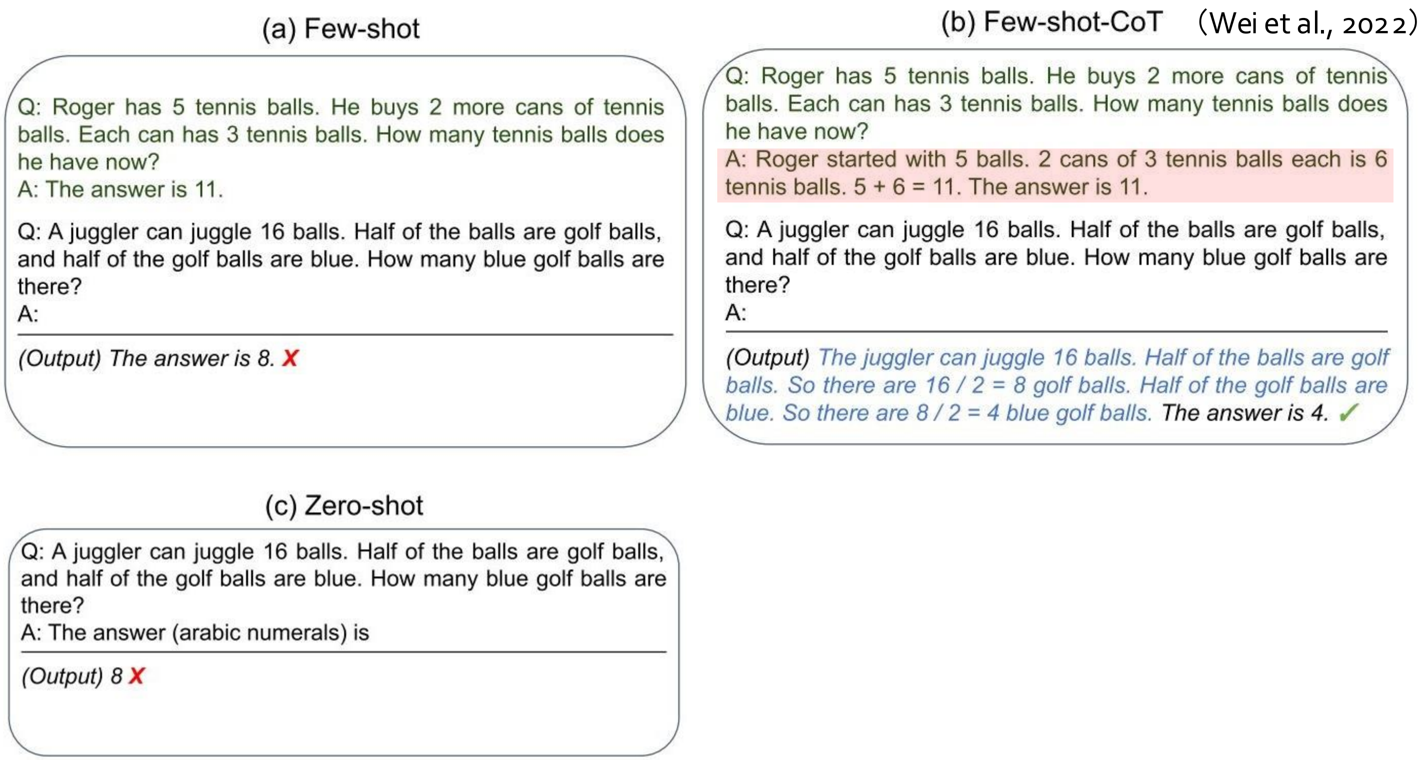
A: The last letter of "Lady" is "y". The last letter of "Gaga" is "a". Concatenating them is "ya". So the answer is ya.

**Coin Flip (state tracking)**

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.

# Zero-Shot CoT



(a) Few-shot

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The answer is 8.* X

(b) Few-shot-CoT （Wei et al., 2022）

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4.* ✓

(c) Zero-shot

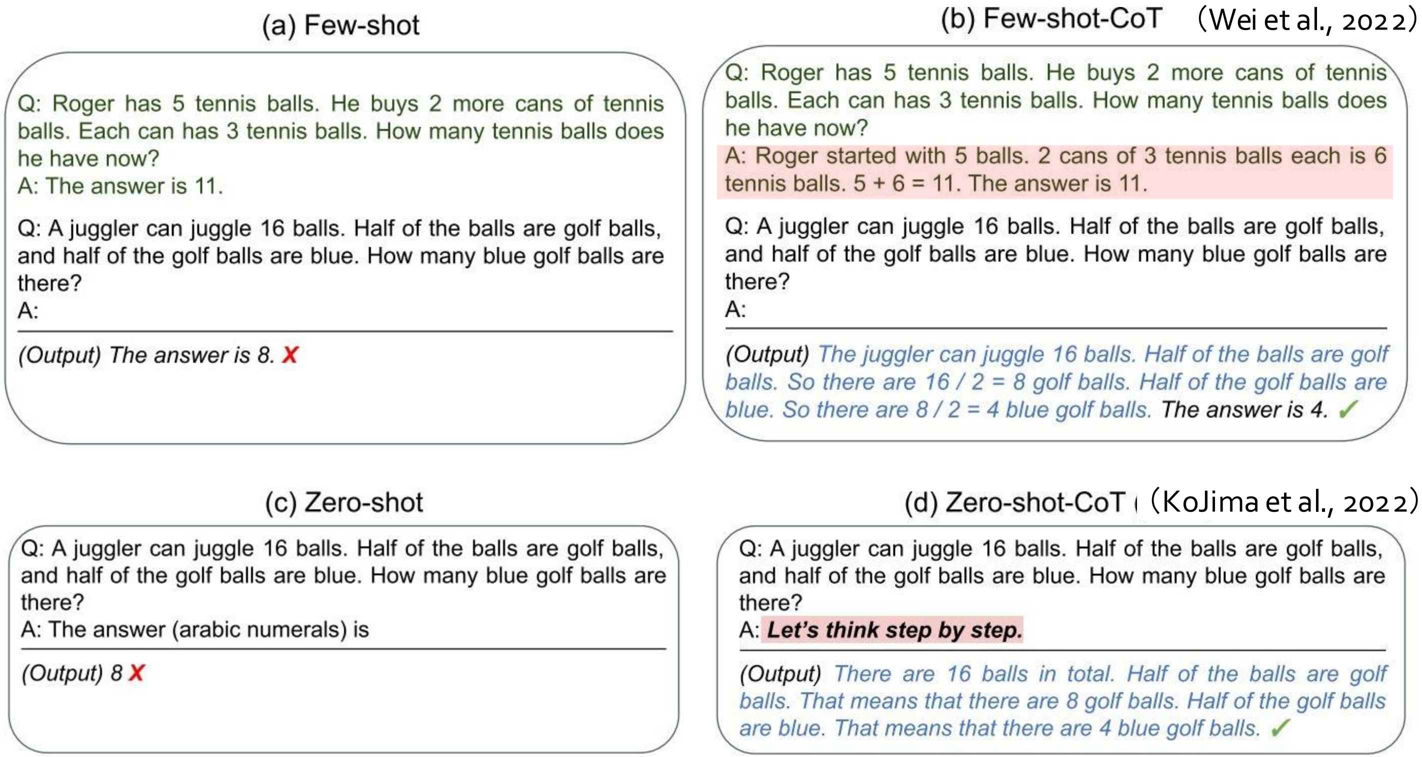
Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

*(Output) 8* X

Step-by-step demonstration

Step-by-step Answer

# Zero-Shot CoT



(a) Few-shot

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The answer is 8.* ✗

(b) Few-shot-CoT （Wei et al., 2022）

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4.* ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

*(Output) 8* ✗

(d) Zero-shot-CoT （KoJima et al., 2022）

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

*(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls.* ✓

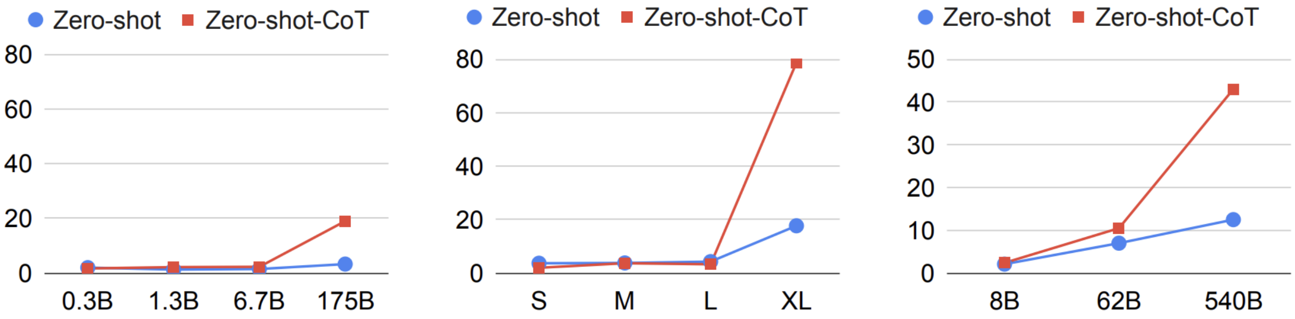
Step-by-step demonstration

Step-by-step Answer

Two-stage Prompting
Step-by-step Answer

# Multi-Step Prompting: Empirical Results

- **Setup:** show demonstrations that contain the decompositions
- The gains of multi-step prompting increases with scale.
- Prompting achieves better perf than [smaller] models that are fine-tuned on a lot more data.



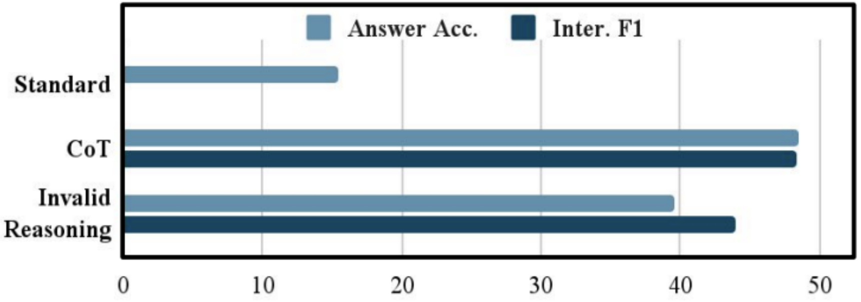
(a) MultiArith on Original GPT-3 (b) MultiArith on Instruct GPT-3 (c) GMS8K on PaLM

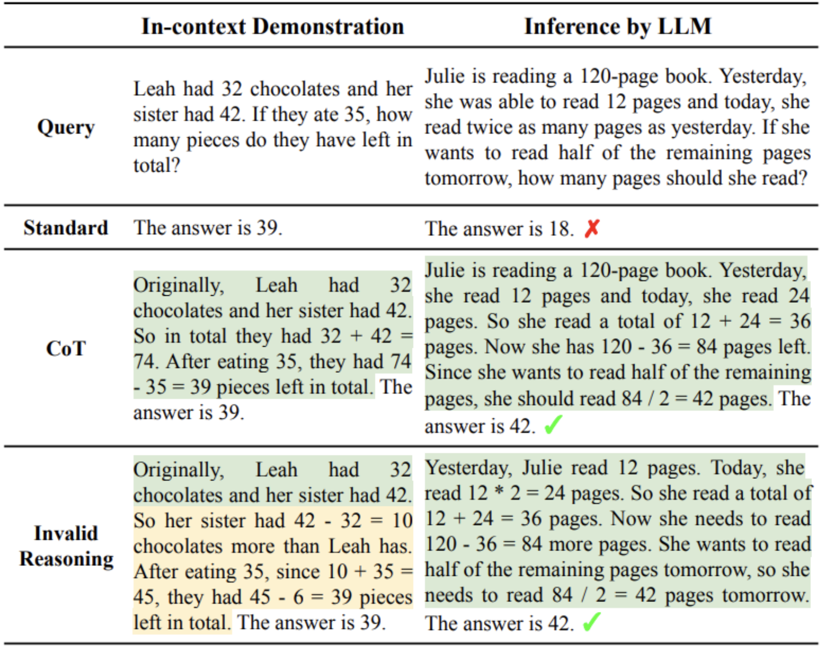
Figure 3: Model scale study with various types of models. S: text-ada-001, M: text-babbage-001, L: text-curie-001, XL: text-davinci-002. See Appendix A.3 and E for the detail.

[“Large Language Models are Zero-Shot Reasoners”, Kojima et al. 2022]

# Multi-Step Prompting: Steps Don’t Have to Be Correct!

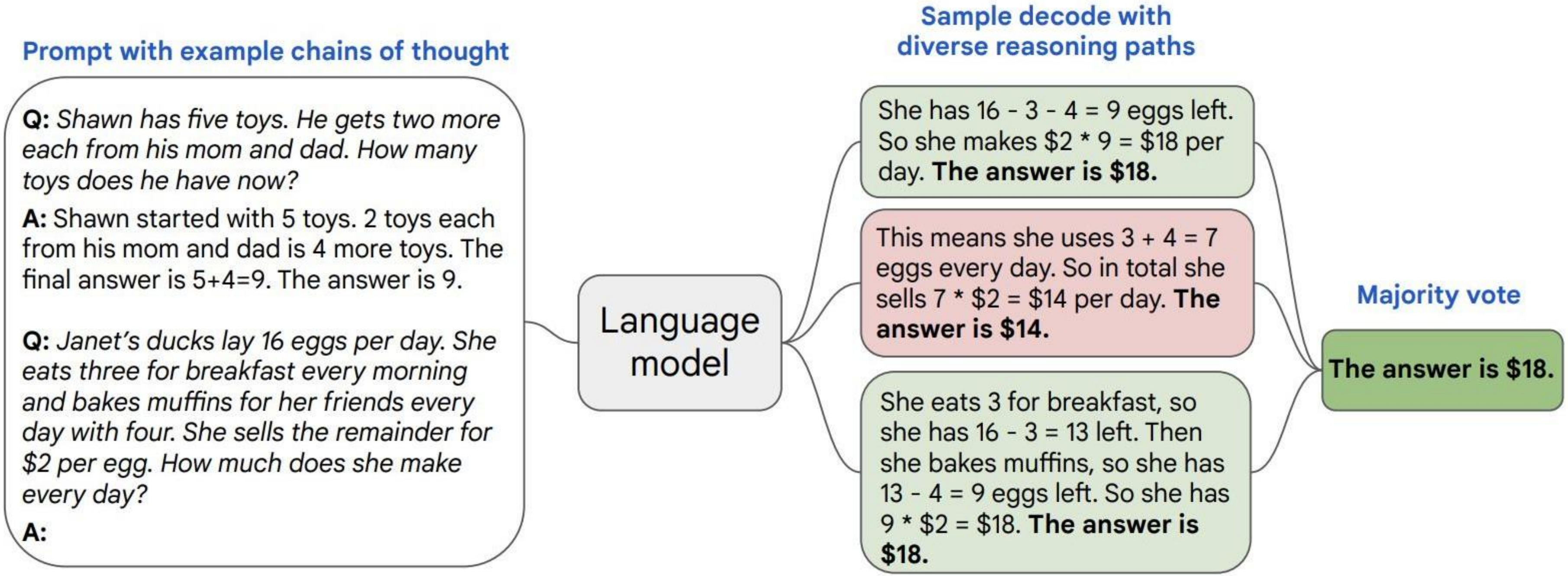
- It is possible even with invalid demonstration
- Prompting with invalid reasoning steps can ac with correct reasoning steps.





| | In-context Demonstration | Inference by LLM |
|---|---|---|
| **Query** | Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total? | Julie is reading a 120-page book. Yesterday, she was able to read 12 pages and today, she read twice as many pages as yesterday. If she wants to read half of the remaining pages tomorrow, how many pages should she read? |
| **Standard** | The answer is 39. | The answer is 18. ✗ |
| **CoT** | Originally, Leah had 32 chocolates and her sister had 42. So in total they had 32 + 42 = 74. After eating 35, they had 74 - 35 = 39 pieces left in total. The answer is 39. | Julie is reading a 120-page book. Yesterday, she read 12 pages and today, she read 24 pages. So she read a total of 12 + 24 = 36 pages. Now she has 120 - 36 = 84 pages left. Since she wants to read half of the remaining pages, she should read 84 / 2 = 42 pages. The answer is 42. ✓ |
| **Invalid Reasoning** | Originally, Leah had 32 chocolates and her sister had 42. So her sister had 42 - 32 = 10 chocolates more than Leah has. After eating 35, since 10 + 35 = 45, they had 45 - 6 = 39 pieces left in total. The answer is 39. | Yesterday, Julie read 12 pages. Today, she read 12 * 2 = 24 pages. So she read a total of 12 + 24 = 36 pages. Now she needs to read 120 - 36 = 84 more pages. She wants to read half of the remaining pages tomorrow, so she needs to read 84 / 2 = 42 pages tomorrow. The answer is 42. ✓ |

[“Towards Understanding Chain-of-Thought Prompting”, Wang et al. 2022]


Prompt with example chains of thought
Sample decode with diverse reasoning paths
Language model
Majority vote

**Q:** *Shawn has five toys. He gets two more each from his mom and dad. How many toys does he have now?*

**A:** Shawn started with 5 toys. 2 toys each from his mom and dad is 4 more toys. The final answer is 5+4=9. The answer is 9.

**Q:** *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?*

**A:**

She has 16 - 3 - 4 = 9 eggs left. So she makes $2 * 9 = $18 per day. **The answer is $18.**

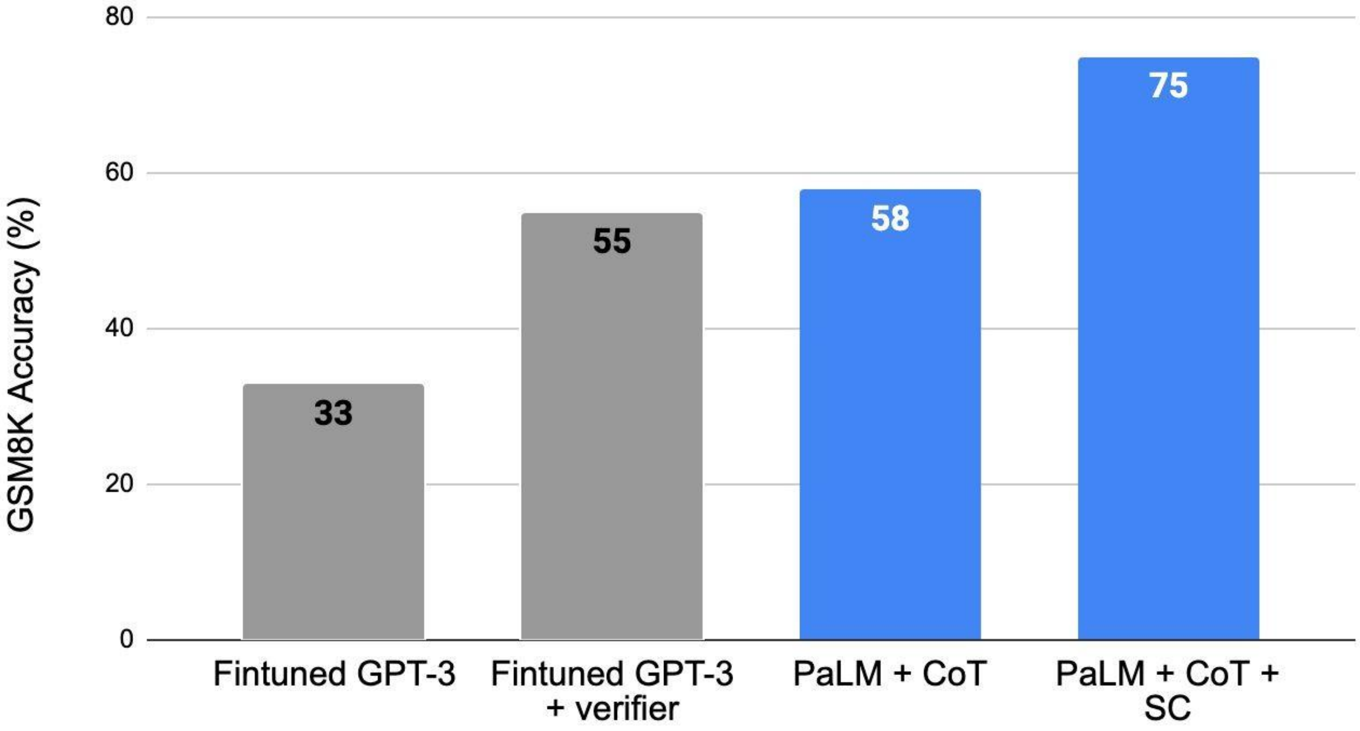
This means she uses 3 + 4 = 7 eggs every day. So in total she sells 7 * $2 = $14 per day. **The answer is $14.**

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 * $2 = $18. **The answer is $18.**

**The answer is $18.**

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.

# Self-Consistency Leads to Improved Results



# Multi-Step Prompting: Parting Comments

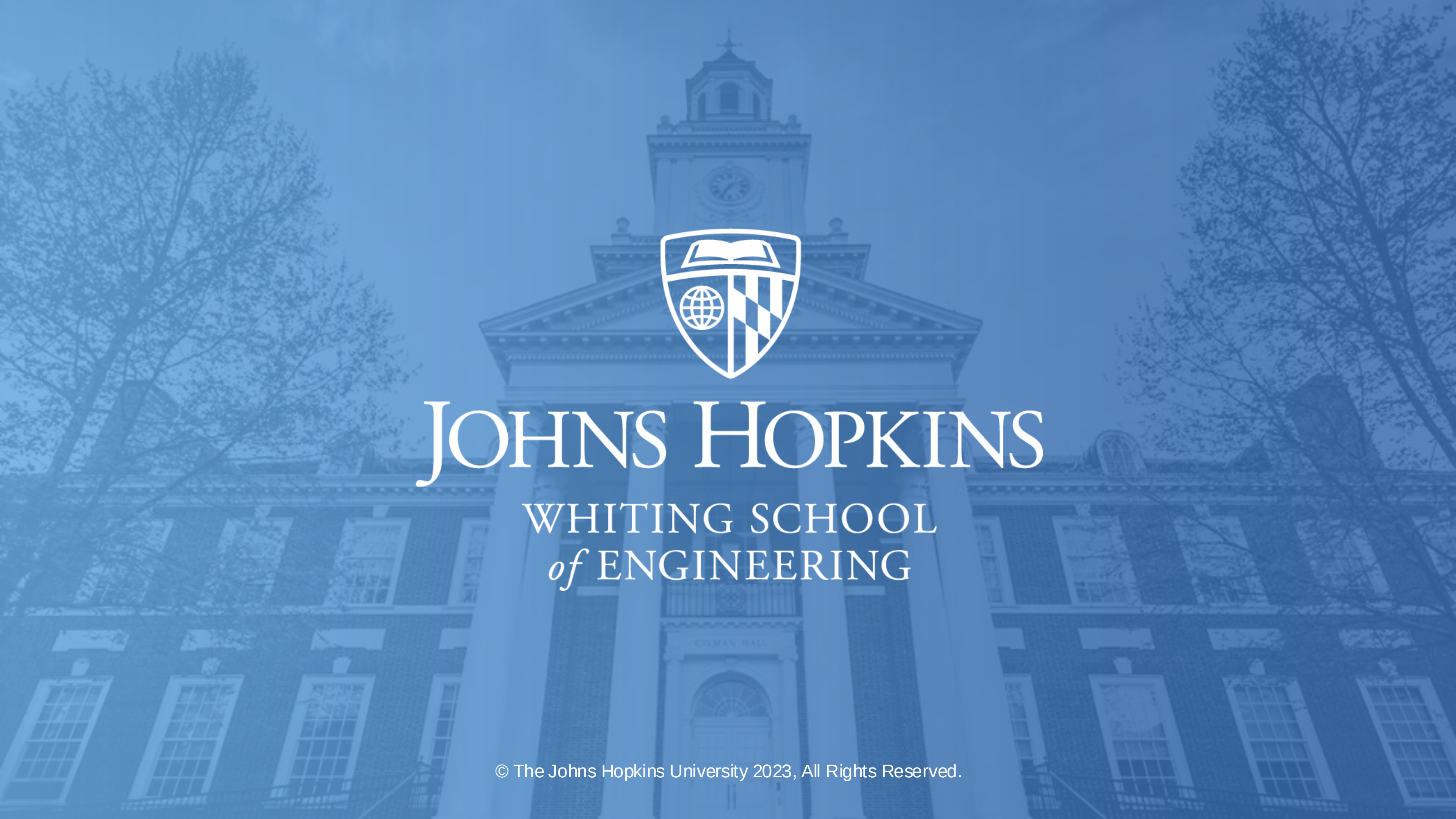
- Prompting LMs to explain their reasoning improves their performance.
- However, their steps aren't always correct.
  - A useful repository of annotation: https://github.com/OpenBioLink/ThoughtSource
- There is much to research on here:
  - When do LMs over-reason or under-reason?
  - How do adjust the granularity of step?
  - How to use use given references in the proofs?
  - How do use external "tools" (e.g., logic, calculator, Python) in forming proofs?

# Summary

- Prompting language models is a powerful way to adapt them to our desired tasks.
  - We saw prompting via in-context demonstrations
  - We also saw various variants and extensions

- They also serve as a gateway to understand the underlying dynamics inside models.

- Lots of activity in this area and room for a lot of research progress.



JOHNS HOPKINS

WHITING SCHOOL *of* ENGINEERING