

[Slide credit: Iz Beltagy, Arman Cohan, Eric Wallace, Sewon Min, Sameer Singh, and many others ]



# Adapting Language Models

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

<https://self-supervised.cs.jhu.edu/sp2024/>

# Language Models are not trained to do what you want

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION GPT-3

Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.

There is a mismatch between LLM pre-training and user intents.

# 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

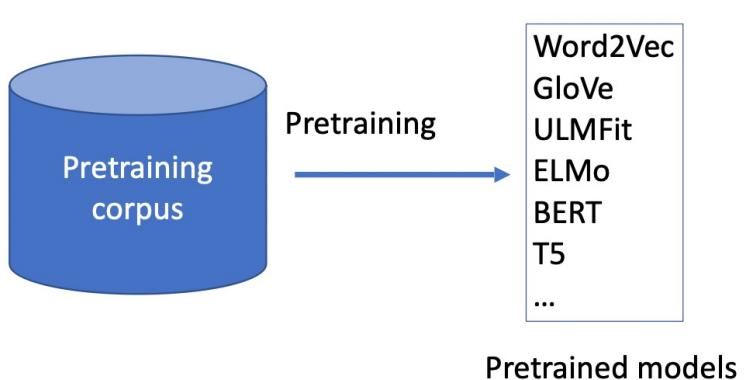
# Adapting Language Models: Chapter Plan

---

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

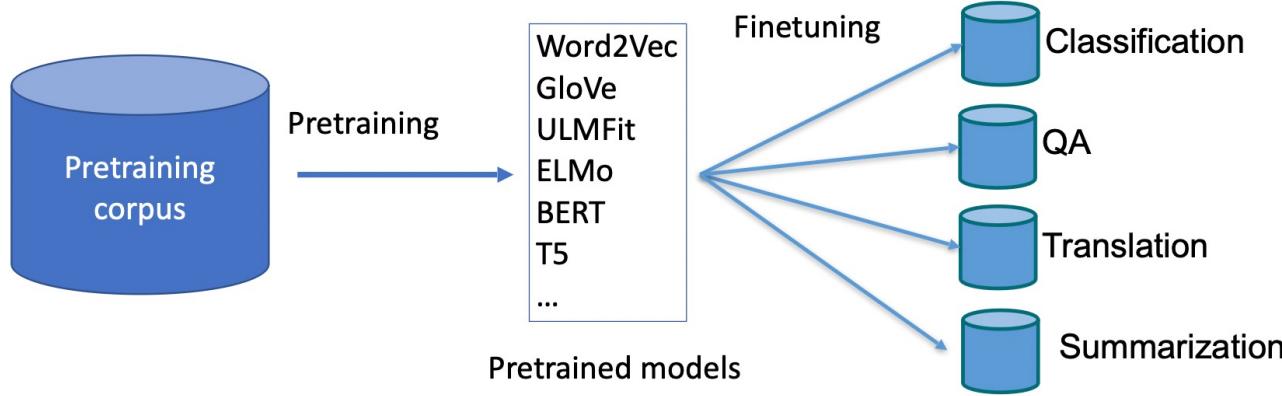
**Chapter goal:** Get comfortable with various forms of adapting LMs for solving your tasks!

# Adaptation via Fine-Tuning

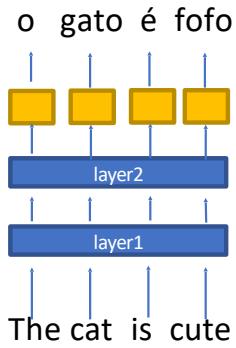


# Fine-Tuning for Tasks

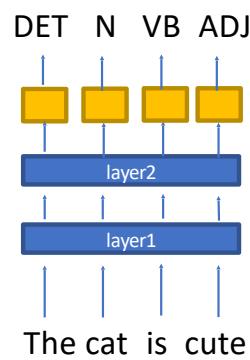
---



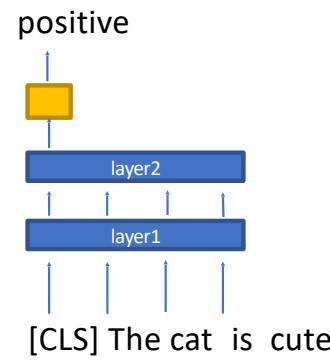
# Fine-Tuning for Tasks



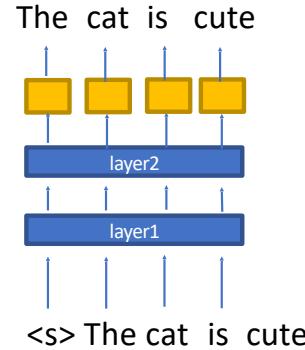
Translation



POS Tagging



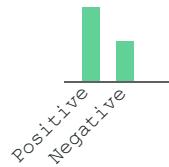
Text classification



Language modeling

# Fine-tuning Pre-trained Models

Classification Head



Language Model



Embeddings



Input

[CLS] A three-hour cinema master class.

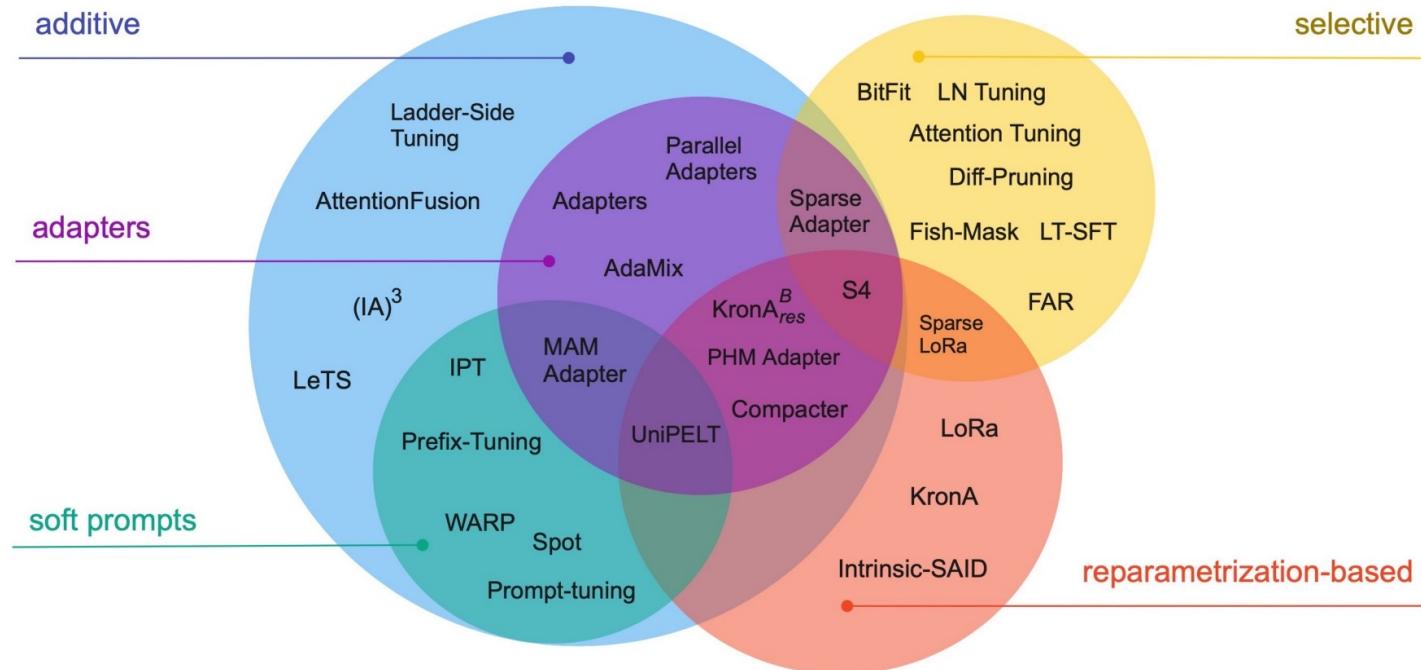
- Whole model tuning:
  - Run an optimization defined on your task data that updates **all** model parameters
- Head-tuning:
  - Run an optimization defined on your task data that updates the parameters of the model “head”

# Parameter-efficient Fine-tuning

---

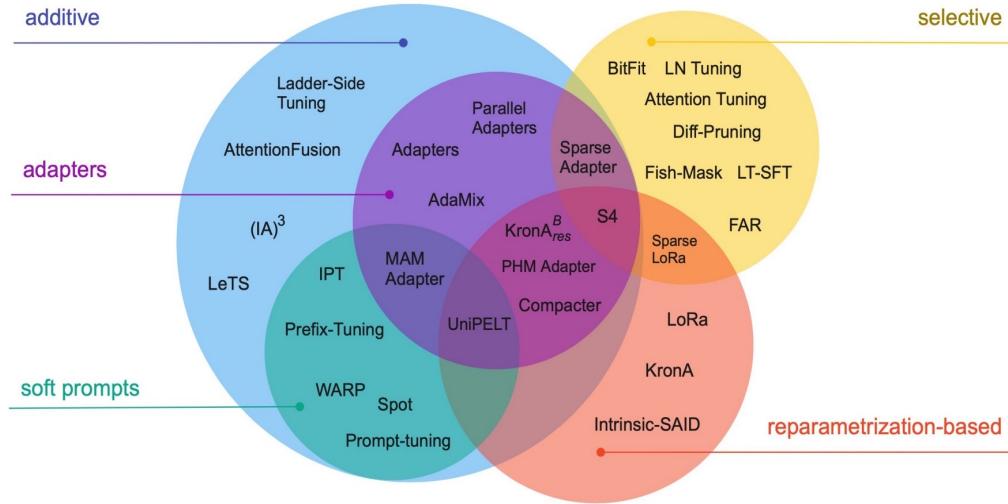
- In fine-tuning we need to updating and storing all the parameters of the LM
  - We would need to store a copy of the LM for each task
- With large models, storage management becomes difficult
  - E.g., A model of size 170B parameters requires ~340Gb of storage
  - If you fine-tune a separate model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

# Parameter-efficient Fine-tuning

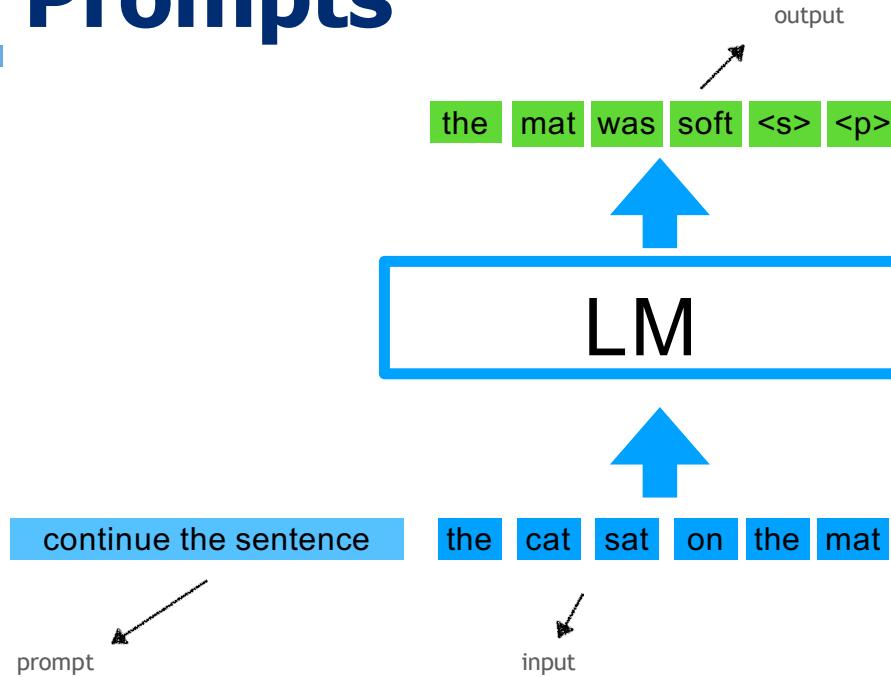


# Parameter-efficient Fine-tuning: Adding Models

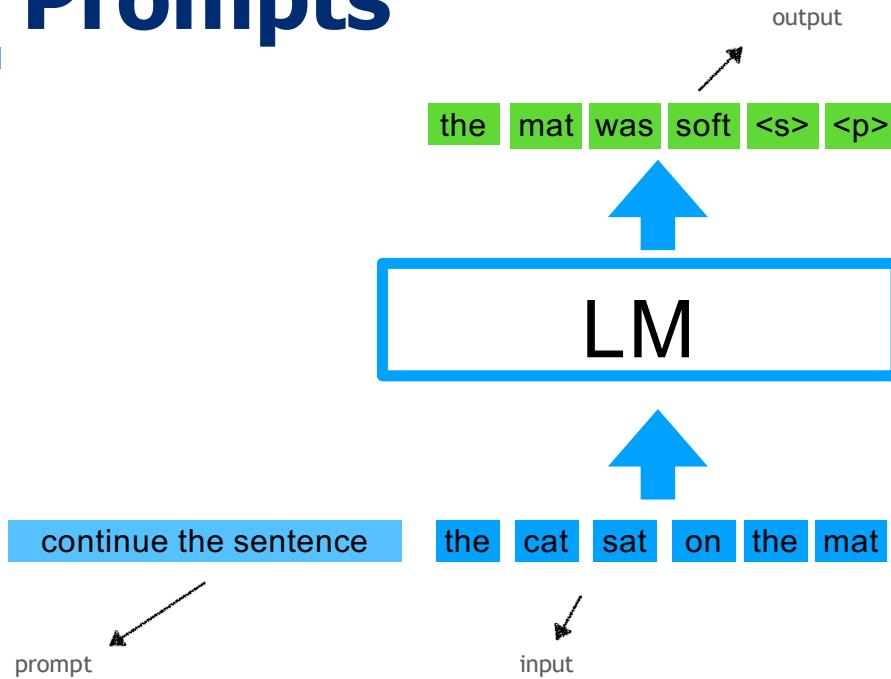
- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- Two commonly used methods:
  - **Soft prompts**
  - Adapters



# Soft Prompts

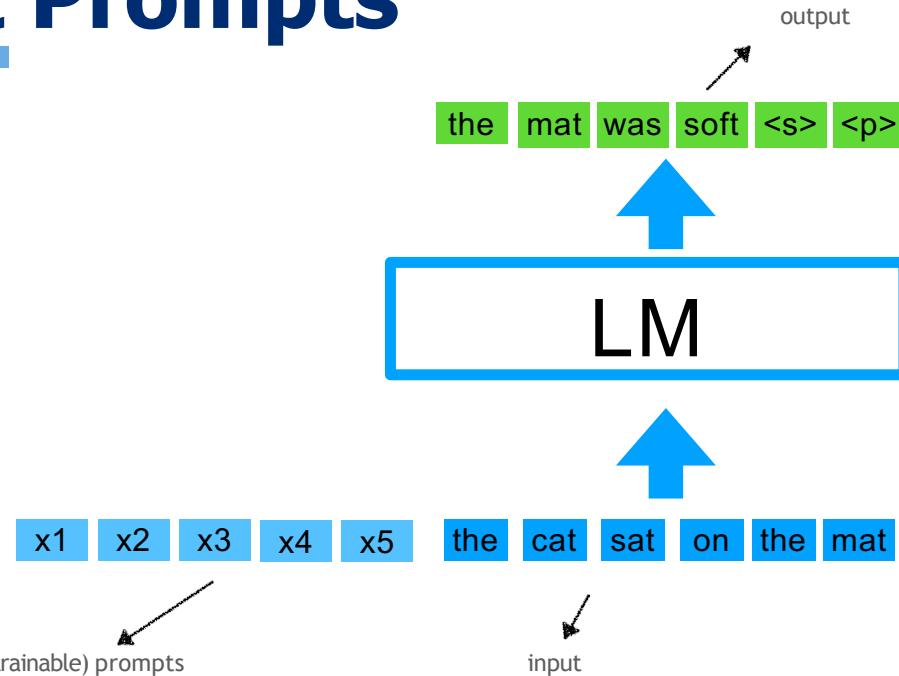


# Soft Prompts



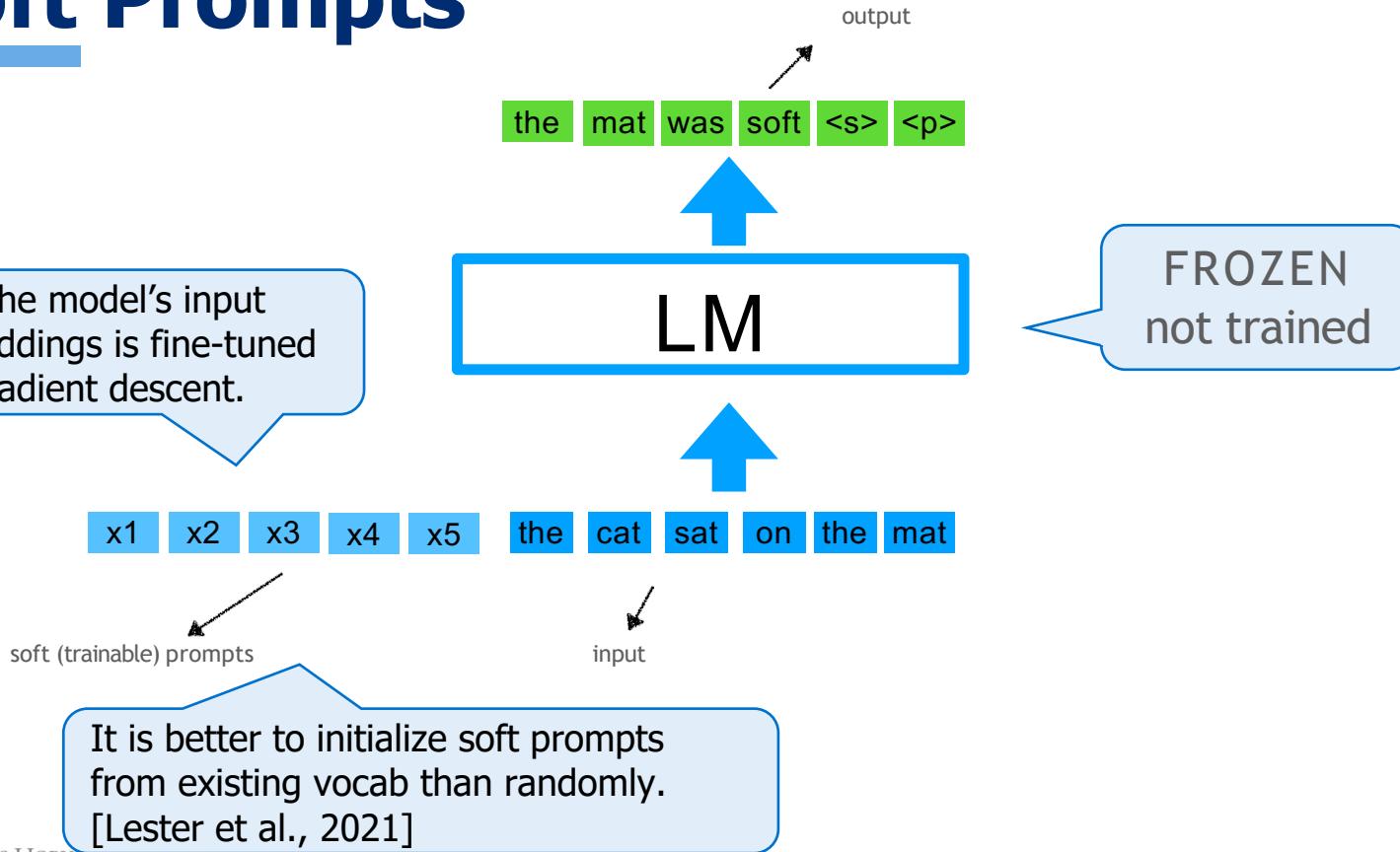
- Designing good prompts might be difficult for each task
- Maybe we can “learn” the prompts?

# Soft Prompts



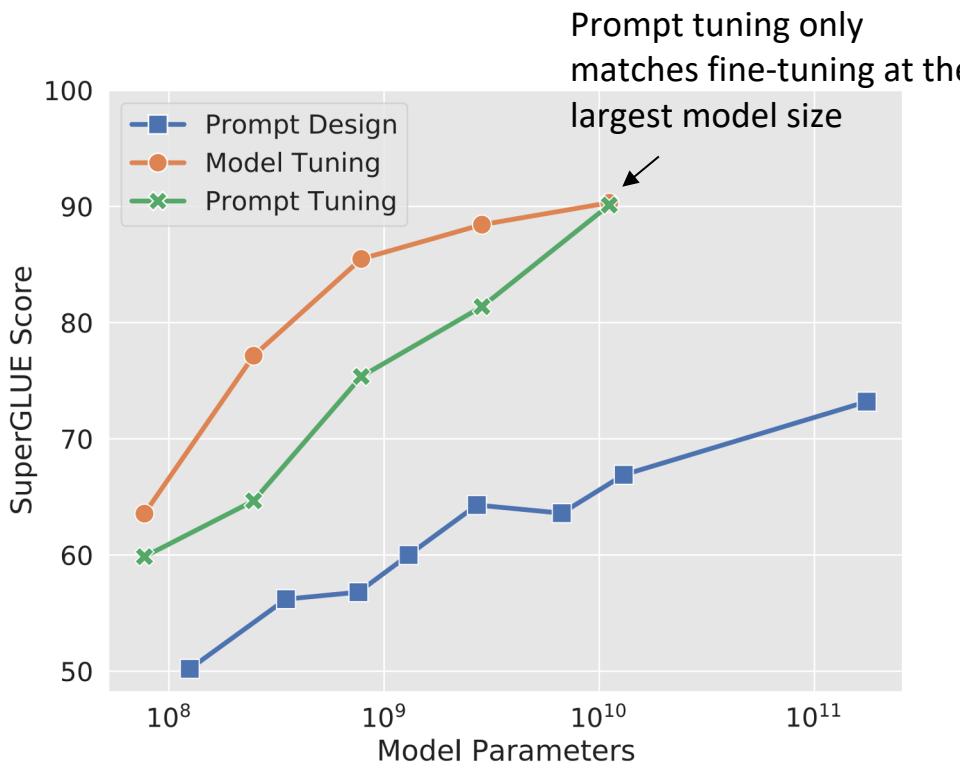
- Designing good prompts might be difficult for each task
- Maybe we can “learn” the prompts?

# Soft Prompts



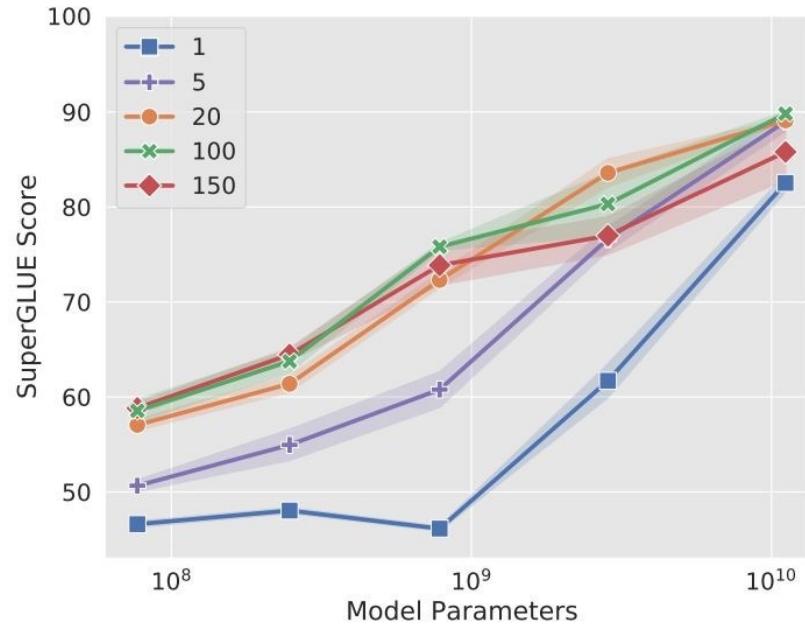
# Prompt Tuning: Effect of Prompt Length

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



# 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

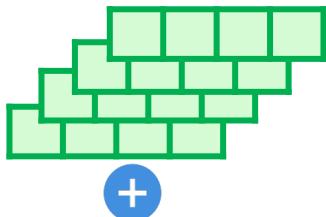


Are continuous prompts as interpretable as  
Discrete (Language) prompts?

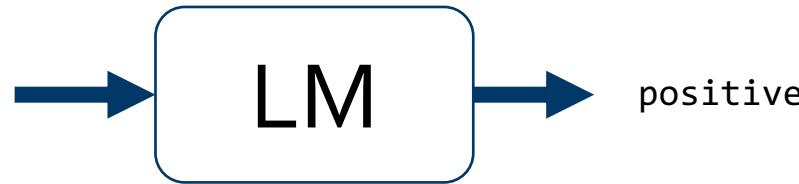
# [Continuous] Prompt Waywardness

Prompt waywardness hypothesis, states that:

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



Sentence: That was a great fantasy movie.

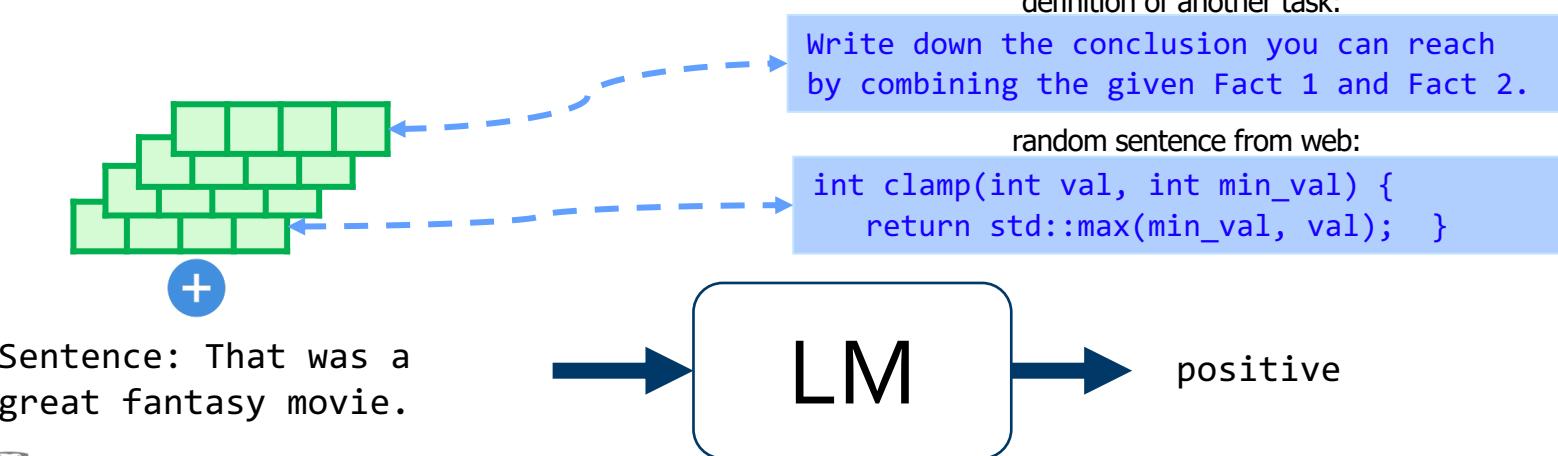


# [Continuous] Prompt Waywardness

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



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



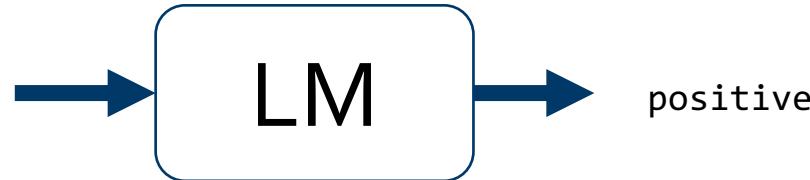
Sentence: That was a  
great fantasy movie.

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



Sentence: That was a  
great fantasy movie.

**discrete (text) prompts:**  
easy to interpret, but not easy to optimize

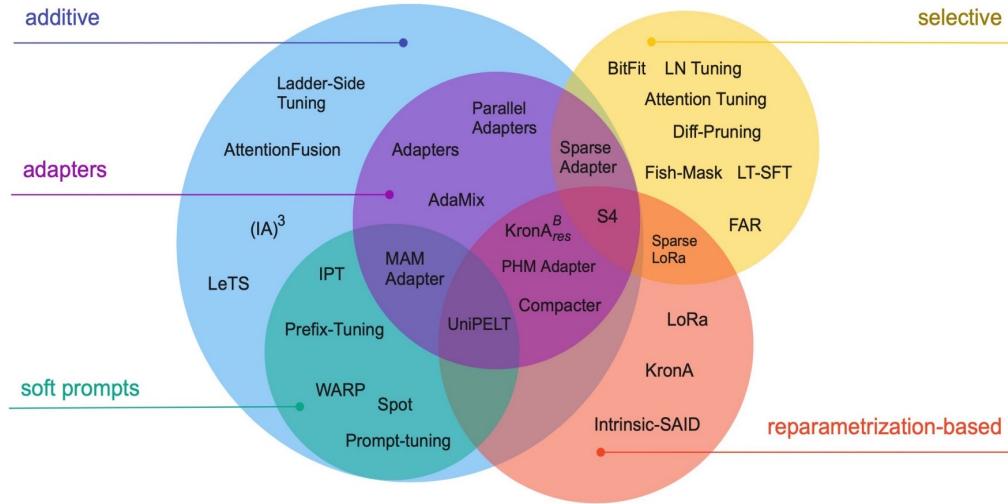


**continuous prompts:**  
unclear how to interpret, but easy to optimize



# Parameter-efficient Fine-tuning: Adding Models

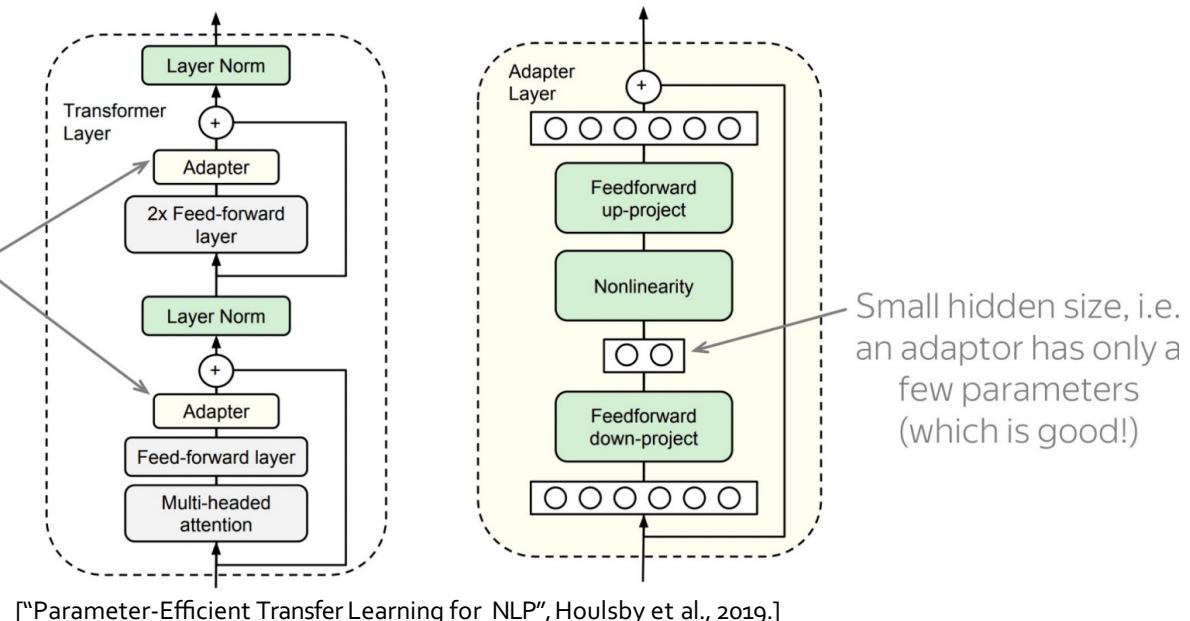
- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- Two commonly used methods:
  - Soft prompts
  - **Adapters**



# 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



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

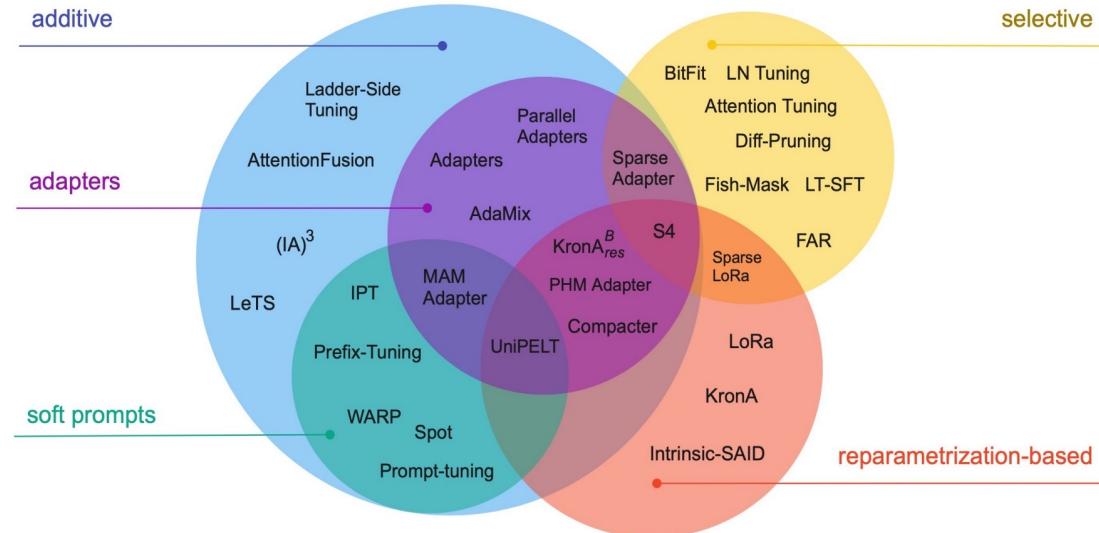
# Question

---

- 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.

# 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.



# BitFit

---

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

$$\mathbf{h}_2^\ell = \text{Dropout}(\mathbf{W}_{m_1}^\ell \cdot \mathbf{h}_1^\ell + \mathbf{b}_{m_1}^\ell) \quad (1)$$

$$\mathbf{Q}^{m,\ell}(\mathbf{x}) = \mathbf{W}_q^{m,\ell} \mathbf{x} + \mathbf{b}_q^{m,\ell} \quad (2)$$

$$\mathbf{K}^{m,\ell}(\mathbf{x}) = \mathbf{W}_k^{m,\ell} \mathbf{x} + \mathbf{b}_k^{m,\ell} \quad (3)$$

$$\mathbf{V}^{m,\ell}(\mathbf{x}) = \mathbf{W}_v^{m,\ell} \mathbf{x} + \mathbf{b}_v^{m,\ell} \quad (4)$$

$$\text{out}^\ell = \mathbf{g}_{LN_2}^\ell \odot \frac{(\mathbf{h}_5^\ell + \mathbf{h}_3^\ell) - \mu}{\sigma} + \mathbf{b}_{LN_2}^\ell \quad (5)$$

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

# BitFit

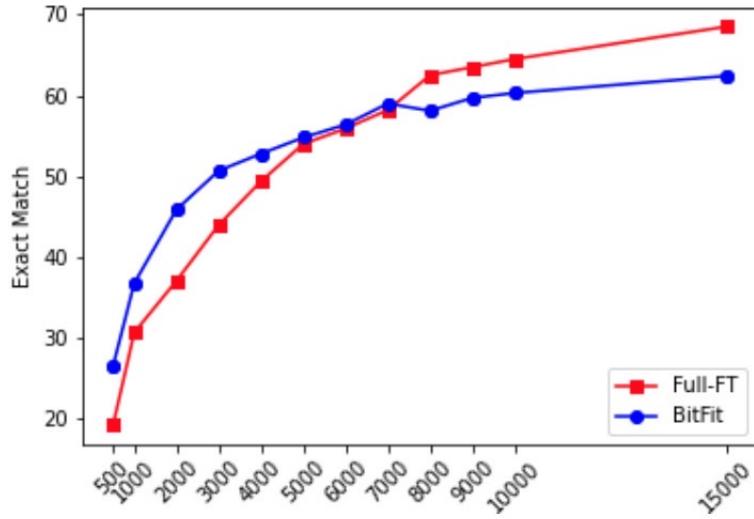
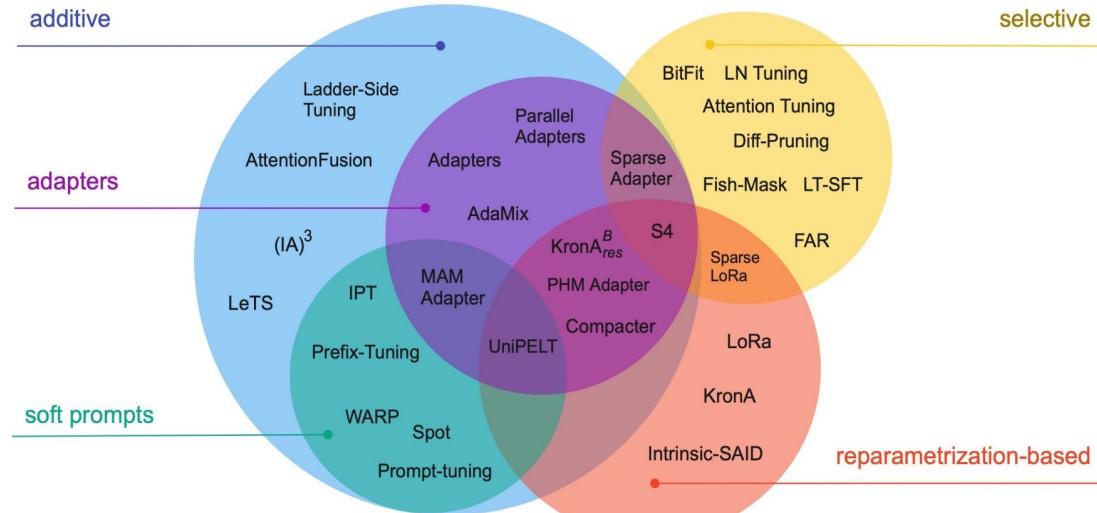


Figure 2: Comparison of BitFit and Full-FT with BERT<sub>BASE</sub> exact match score on SQuAD validation set.

# 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

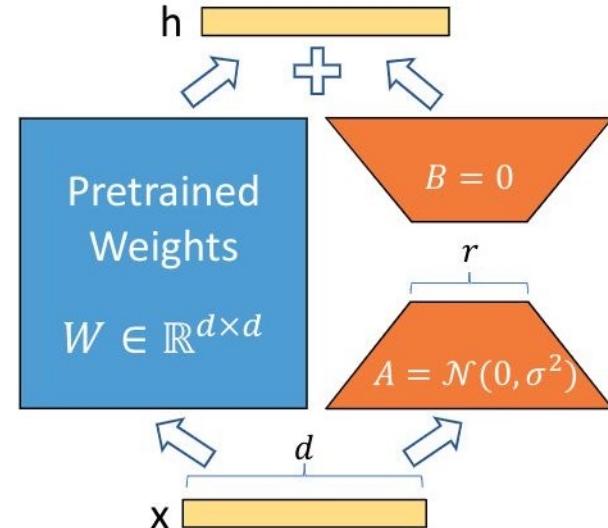
- Hypothesis: the intrinsic rank of the weight matrices in a large language model is low
- Parameter update for a weight matrix is decomposed into a product of two low-rank matrices

$$W \leftarrow W + \Delta W$$

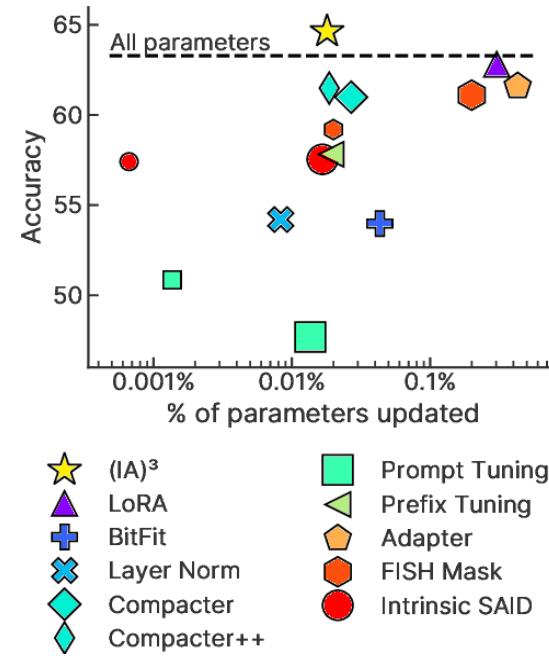
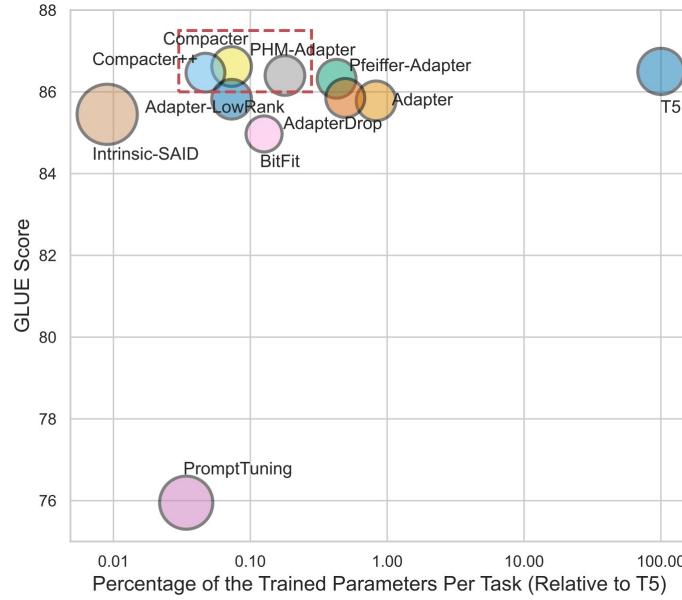
$$\Delta W = BA$$

$$B \in \mathbb{R}^{d,r}, A \in \mathbb{R}^{r,k}, r \ll \min(k, d)$$

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



# Performance/compactness comparison



# Summary

- 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.

# Prompting and In-Context Learning

“I have an extremely large  
collection of clean labeled data”

-- No one

# Limitations of Pre-training, then Fine-tuning

---

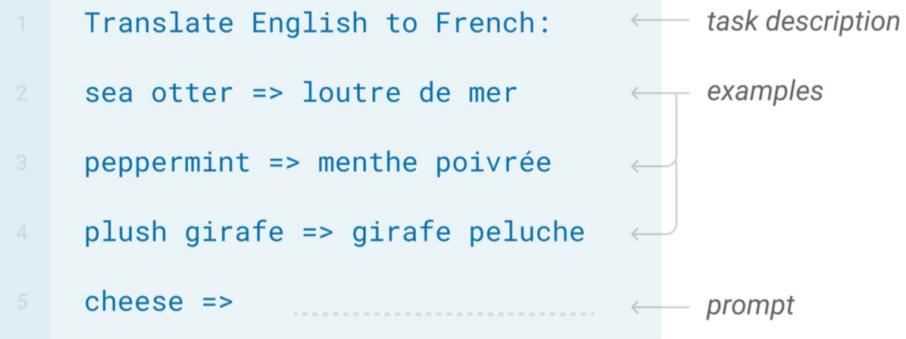
- Often you need a **large labeled data**
  - Though more pre-training can reduce the need for labeled data
- End up with **many copies** (or sub-copies) of the same [sub]model

# In-Context Learning

- 
- 1 Translate English to French: ← *task description*
- 2 sea otter => loutre de mer ← *examples*
- 3 peppermint => menthe poivrée ←
- 4 plush girafe => girafe peluche ←
- 5 cheese => ..... ← *prompt*

# In-Context Learning

- 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

# 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.)
```

$p1 > p2 \rightarrow \text{"positive"}$   
 $p1 < p2 \rightarrow \text{"negative"}$

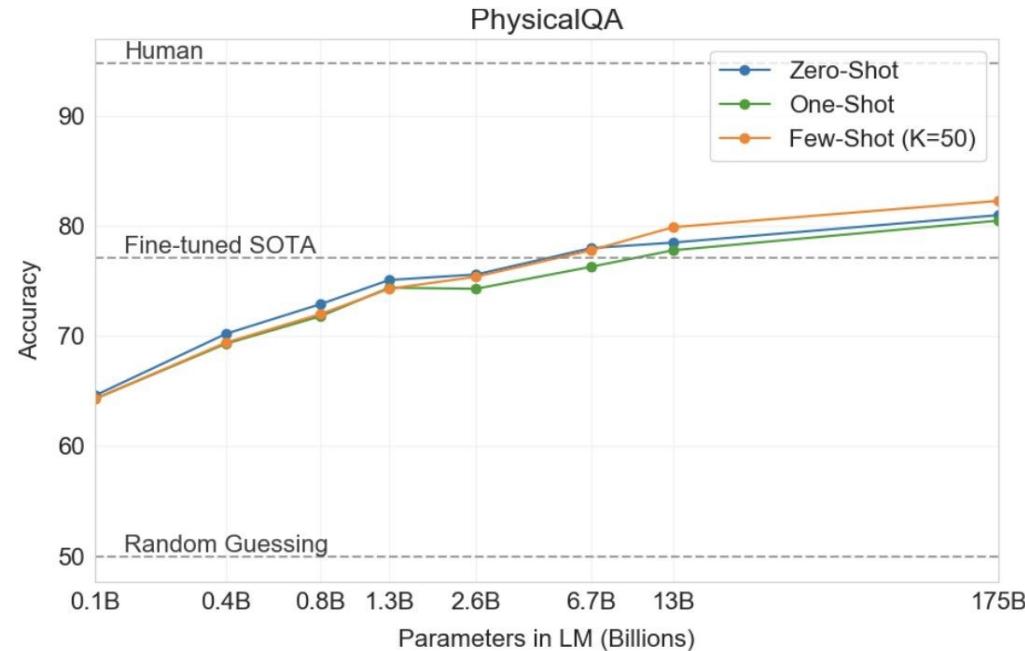
# In-Context learning results



To separate egg whites from the yolk using a water bottle, you should...

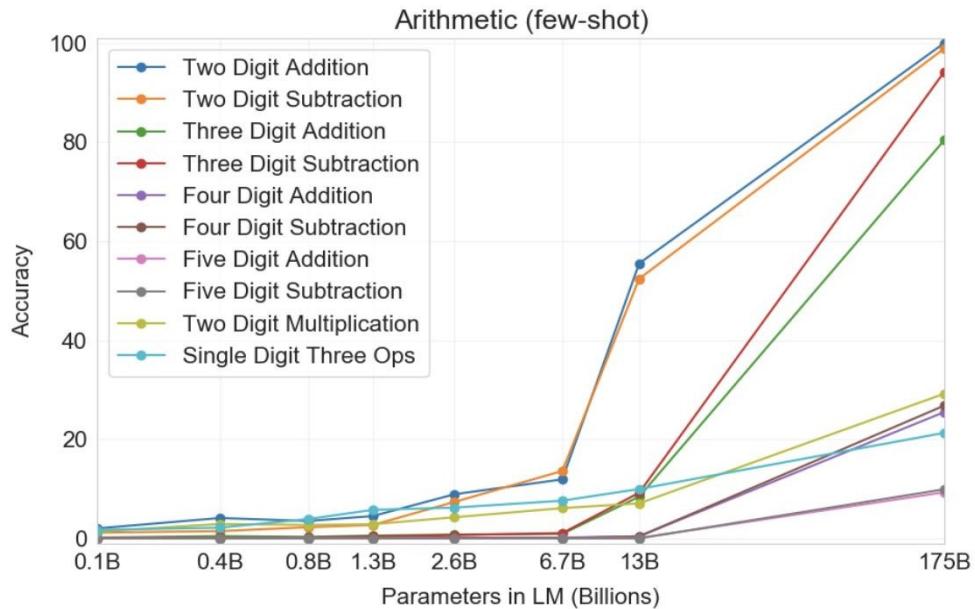
- a. **Squeeze** the water bottle and press it against the yolk. **Release**, which creates suction and lifts the yolk.

- b. **Place** the water bottle and press it against the yolk. **Keep pushing**, which creates suction and lifts the yolk.



# 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!



# Why Do We Care About In-Context Learning?

Practically Useful

Intellectually Intriguing

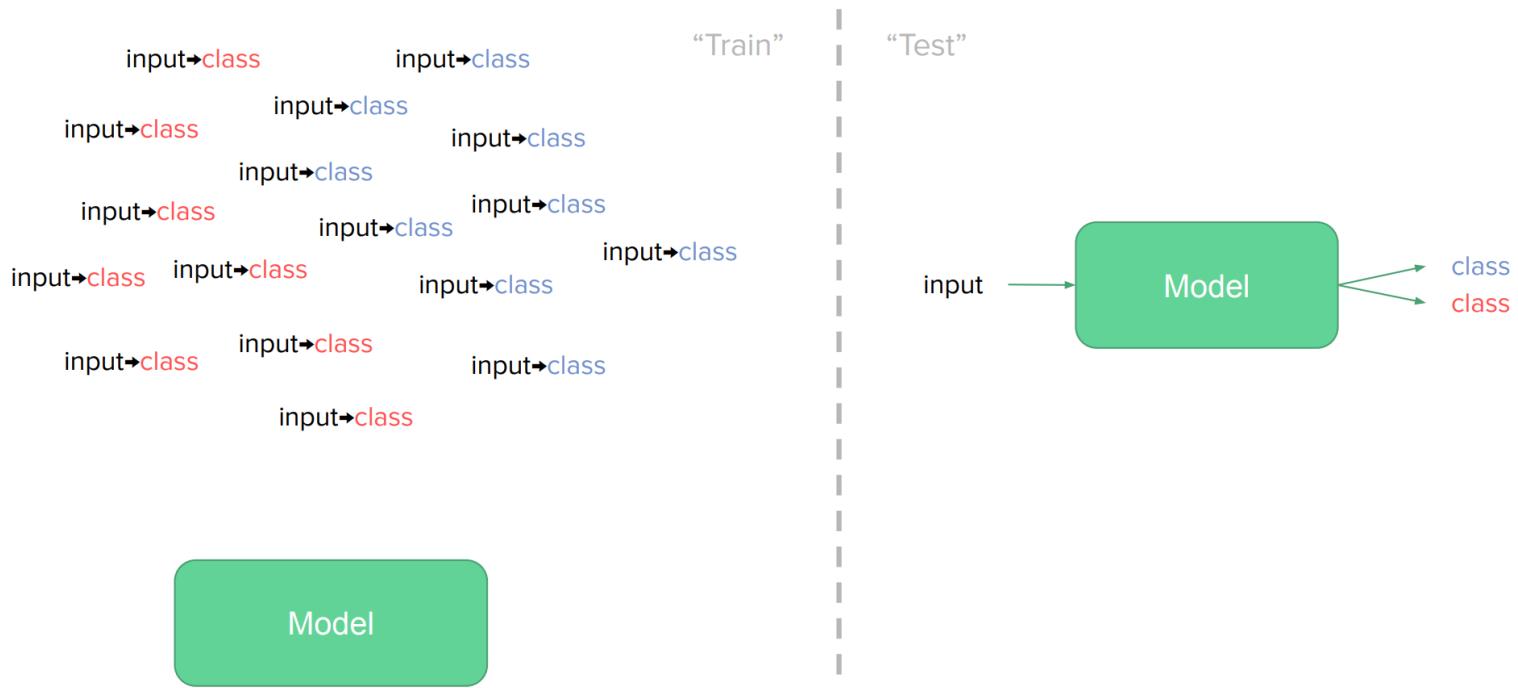
# In-Context Learning: Practically Useful

- 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

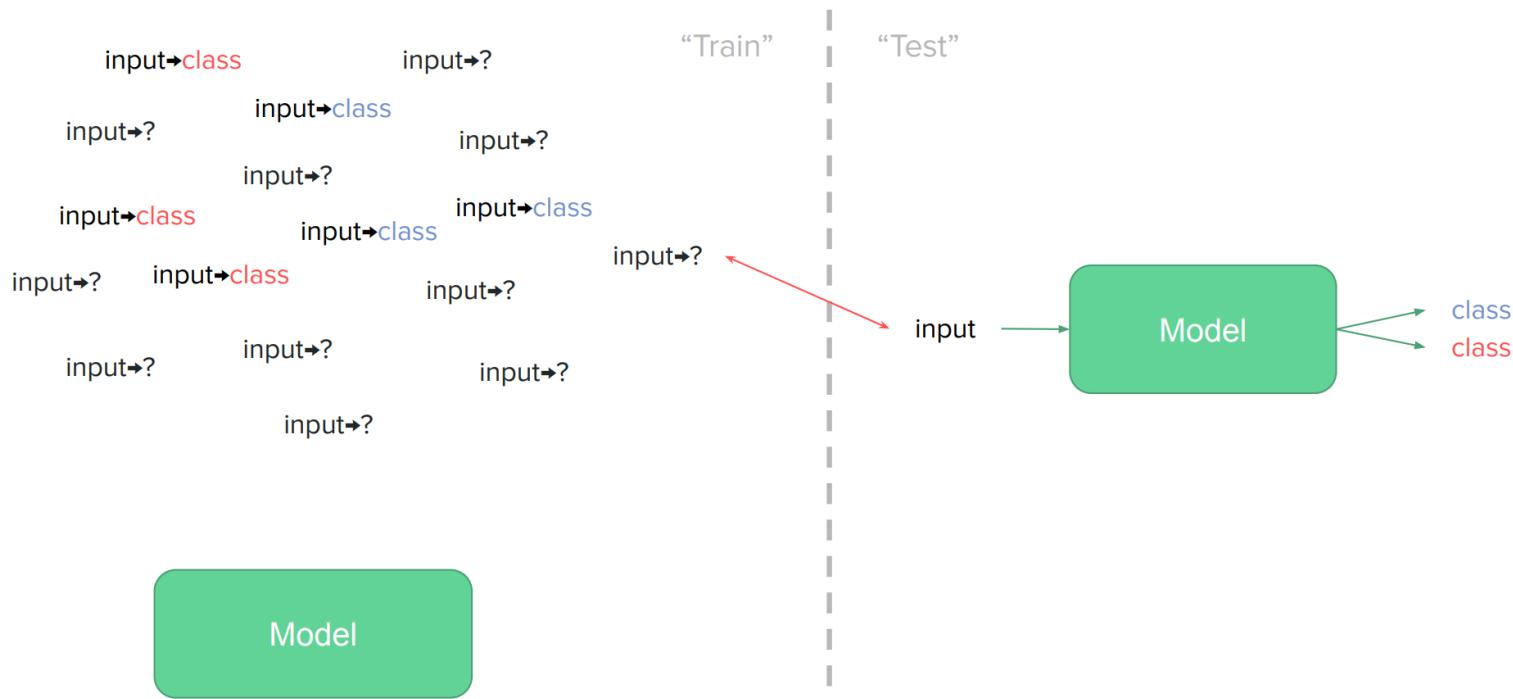
# In-Context Learning: Intellectually Intriguing

- Potential test for “Intelligent Behavior”
  - Generalization from few examples
    - Fundamental piece of intelligence
    - Often used in psychology
    - Quickly adjust to environment
  - What does an LLM “know”?
  - What are the biases/limitations of LLMs?
  - ...
- Insights into Language Modeling

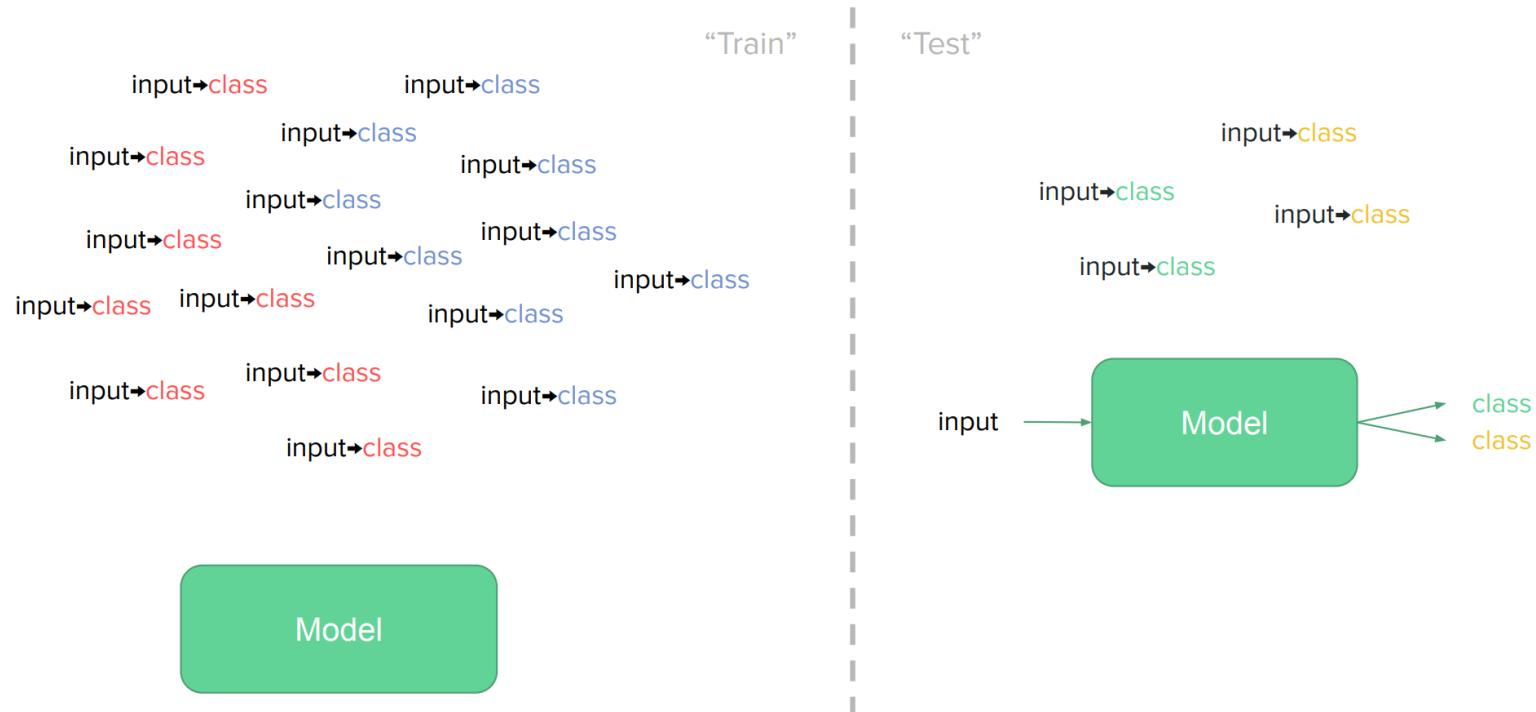
# The Broader Context: Supervised Learning



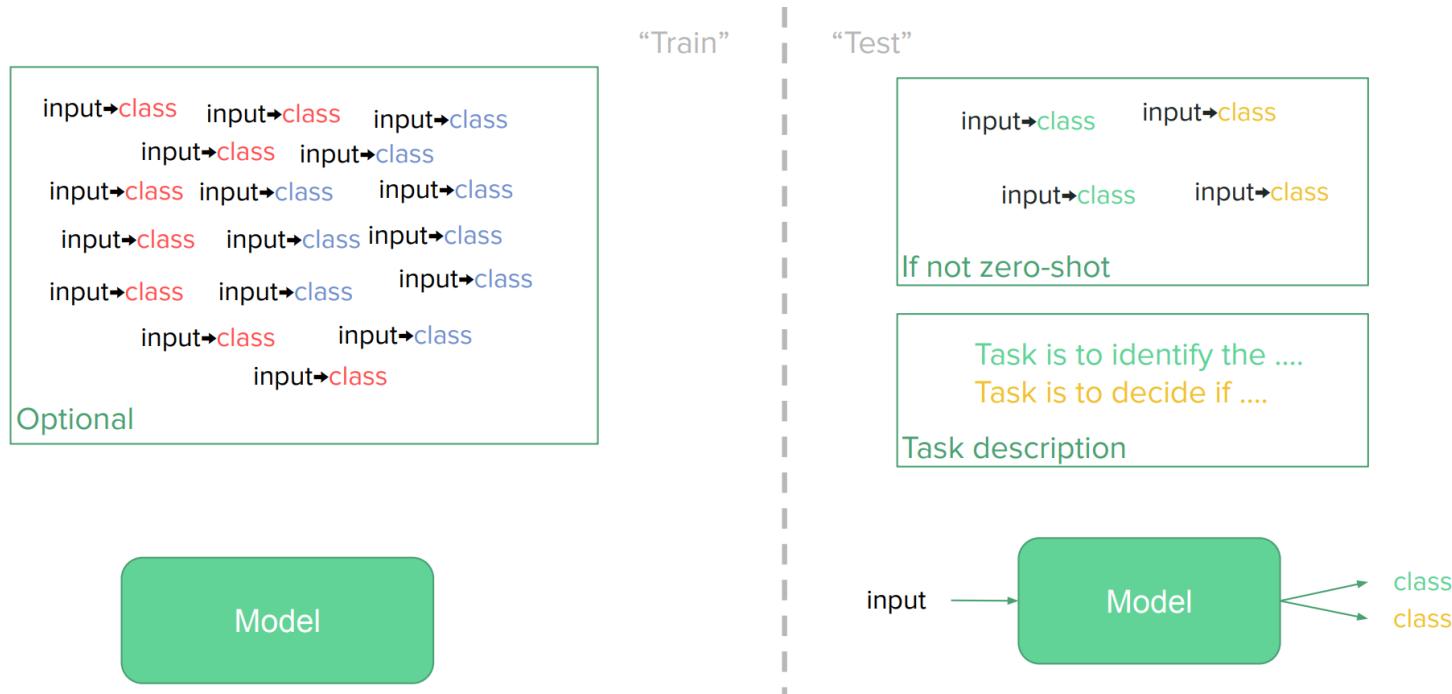
# The Broader Context: Semi-Supervised Learning



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



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



# LM Prompting: Choices of Encoding

Prompt

Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

# LM Prompting: Choices of Encoding

Prompt

Input: Subpar acting. Contimont: negative

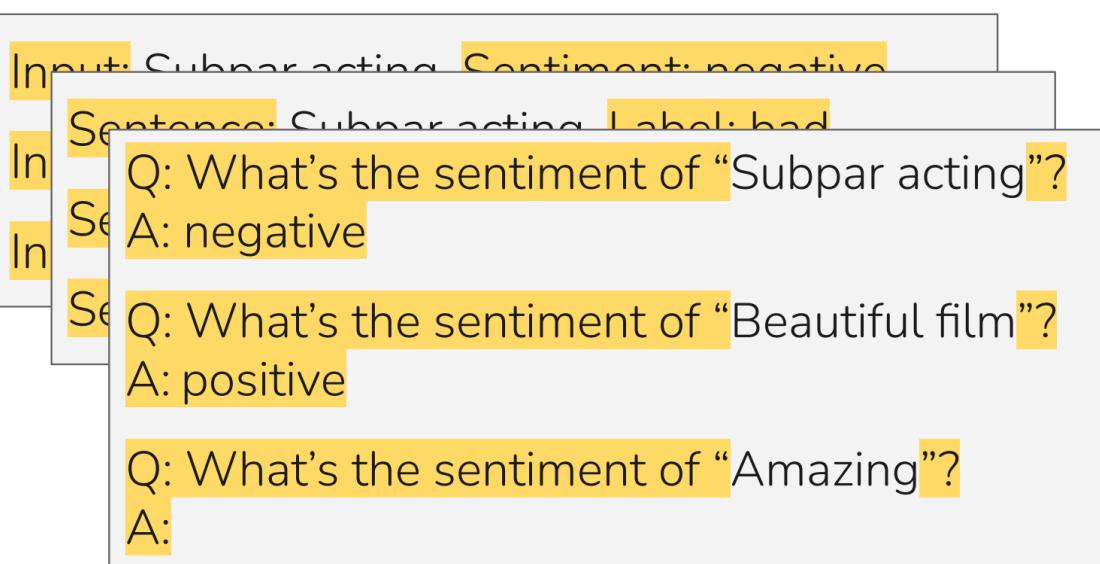
Sentence: Subpar acting. Label: bad

Sentence: Beautiful film. Label: good

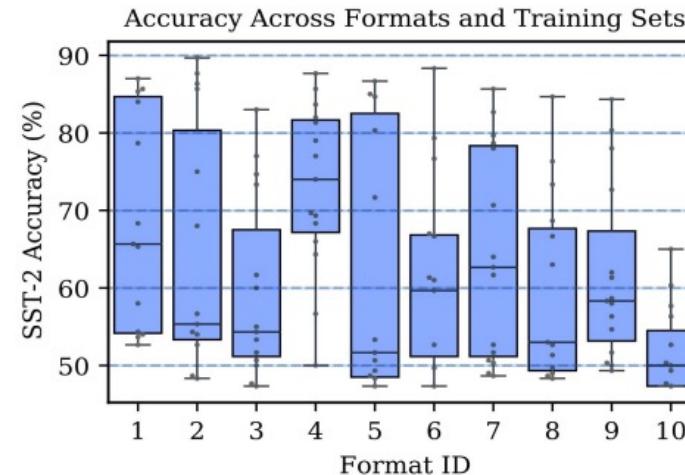
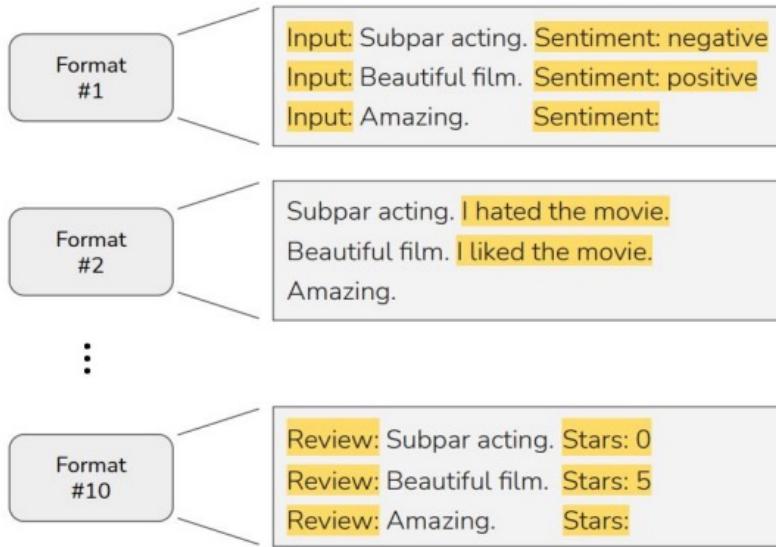
Sentence: Amazing. Label:

# LM Prompting: Choices of Encoding

Prompt



# In-Context Learning: Sensitivity to Encoding



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

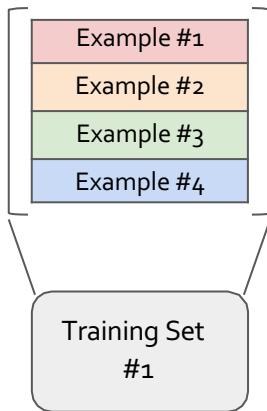
# In-Context Learning: Sensitivity to Demo. Permutations

---

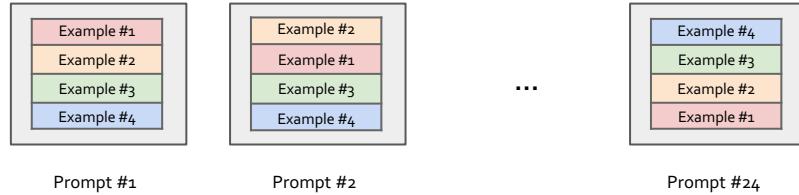
Training Set  
#1

# In-Context Learning: Sensitivity to Demo. Permutations

---



# In-Context Learning: Sensitivity to Demo. Permutations



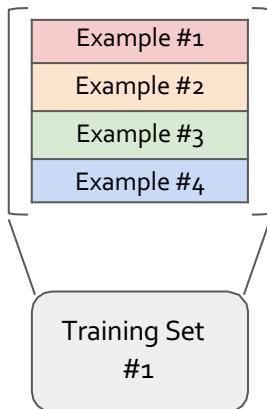
Prompt #1

Prompt #2

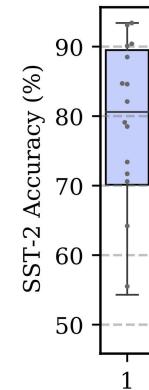
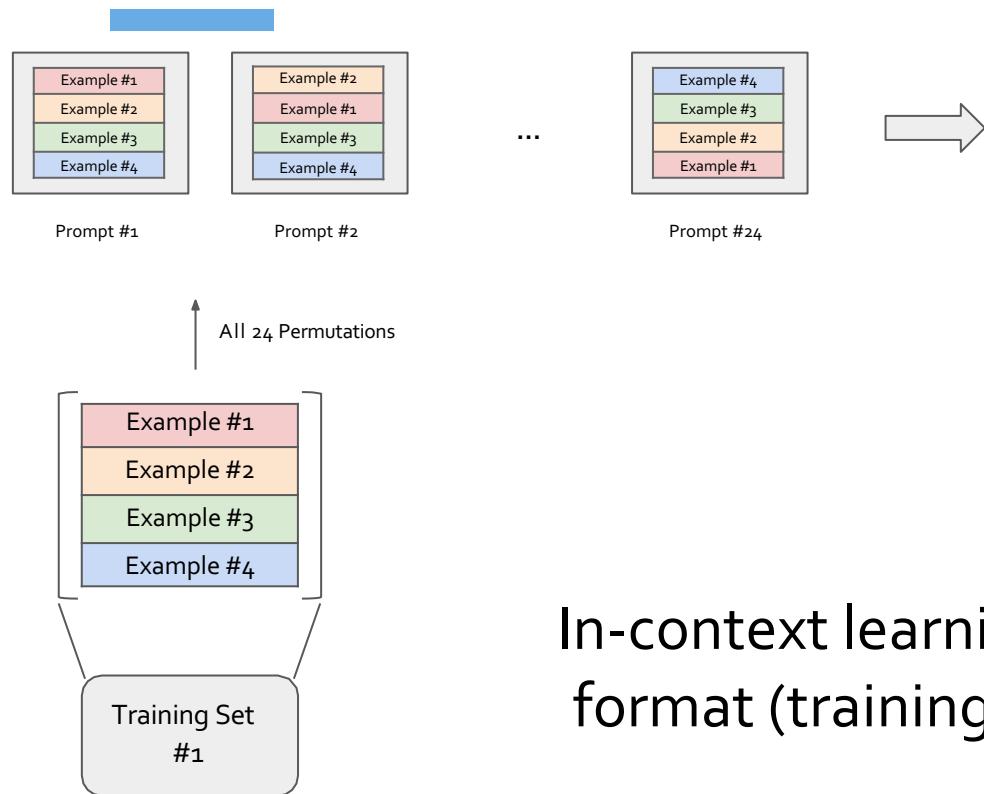
...

Prompt #24

All 24 Permutations

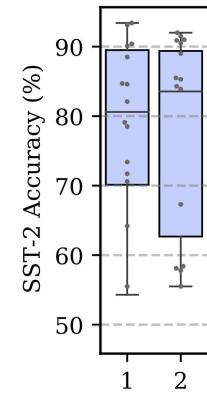
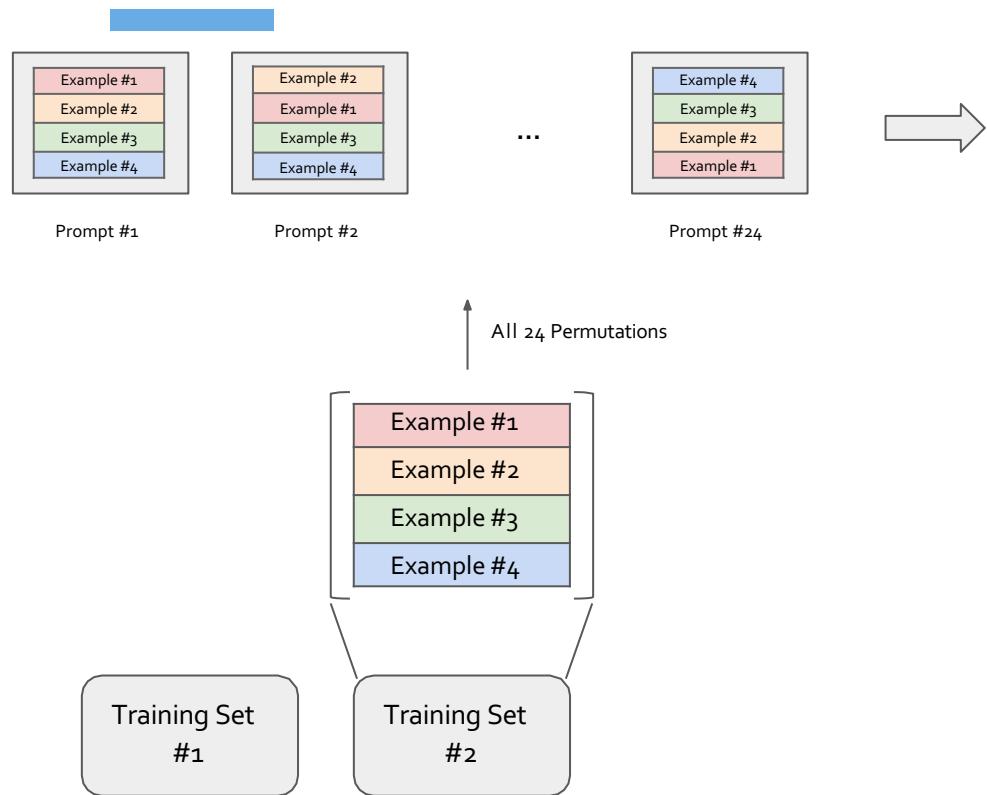


# In-Context Learning: Sensitivity to Demo. Permutations

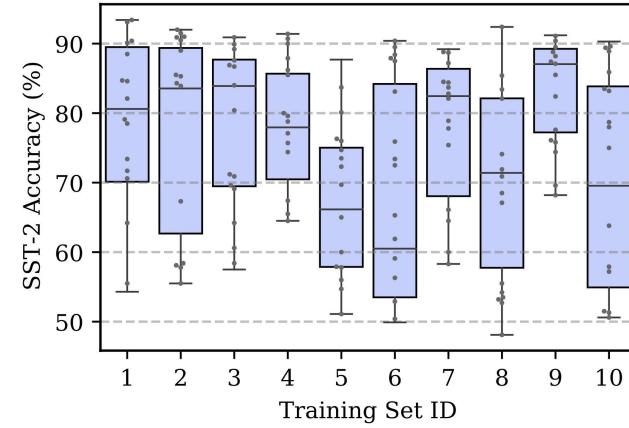
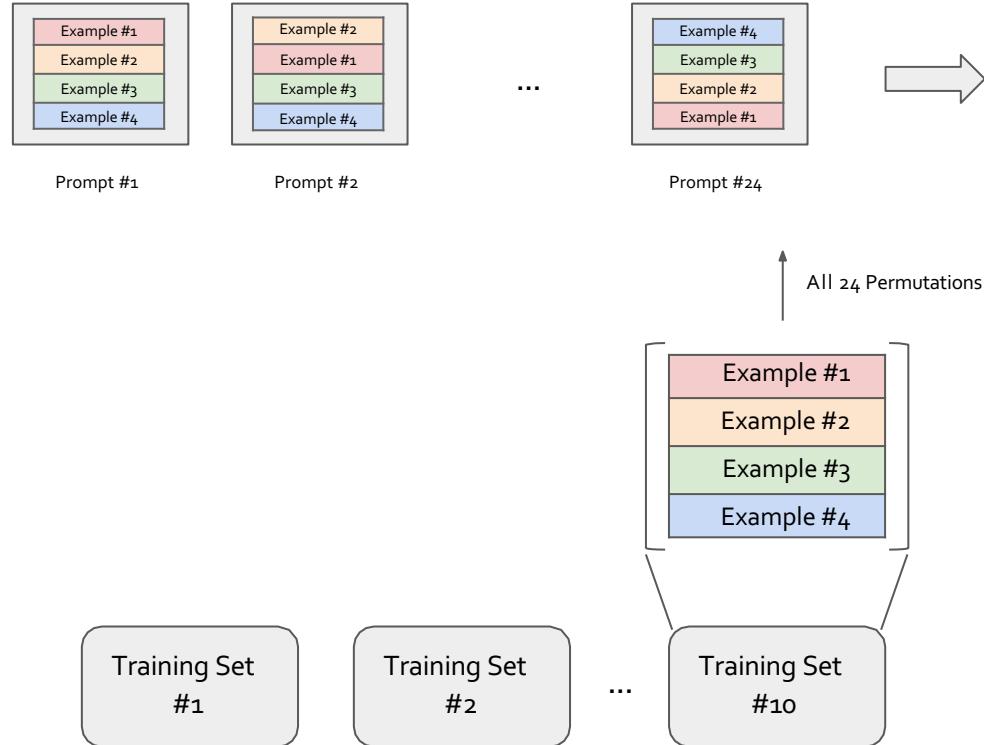


In-context learning is highly sensitive to prompt format (training sets and patterns/verbalizers)

# In-Context Learning: Sensitivity to Demo. Permutations



# In-Context Learning: Sensitivity to Demo. Permutations



The choice of demonstrations and their order is quite important.

# 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.

# 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.

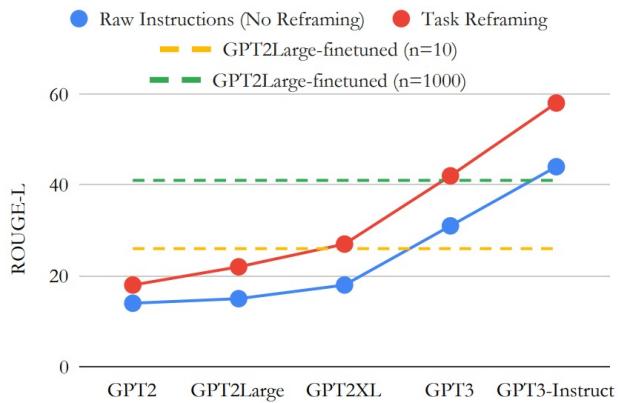


Figure 2: Across a variety of model sizes, **reframed prompts** consistently show considerable performance gain over **raw task instructions (no reframing)** in a few-shot learning setup. Since fine-tuning GPT3 is prohibitively expensive, we show the performance of fine-tuning smaller models (**horizontal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

# 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

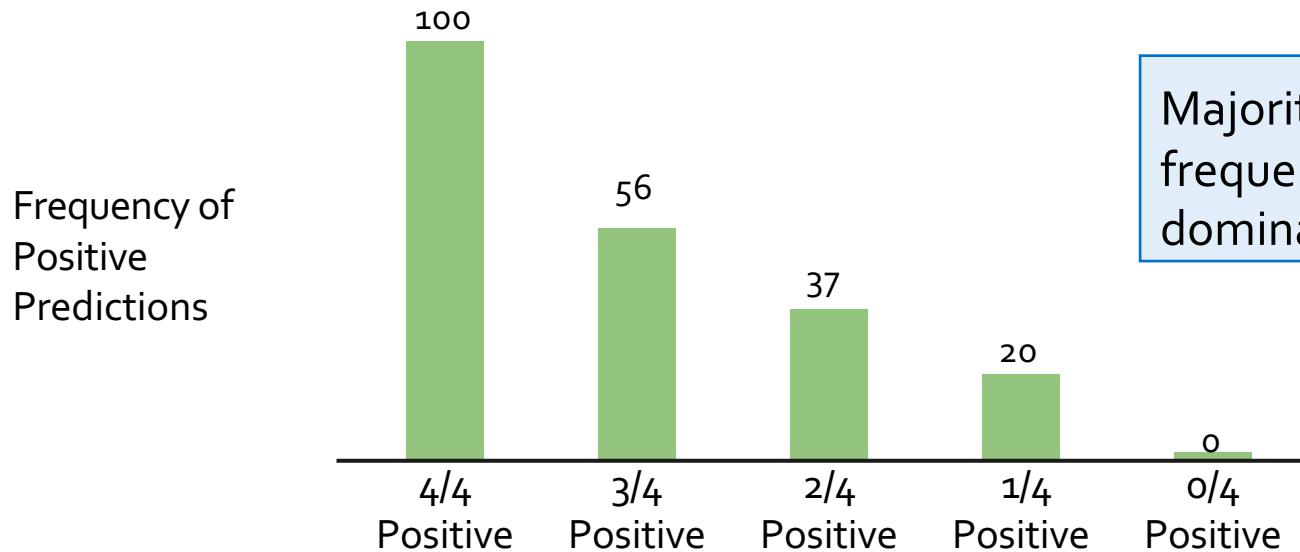
Prompt

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:  
frequent training answers  
dominate predictions.

# 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

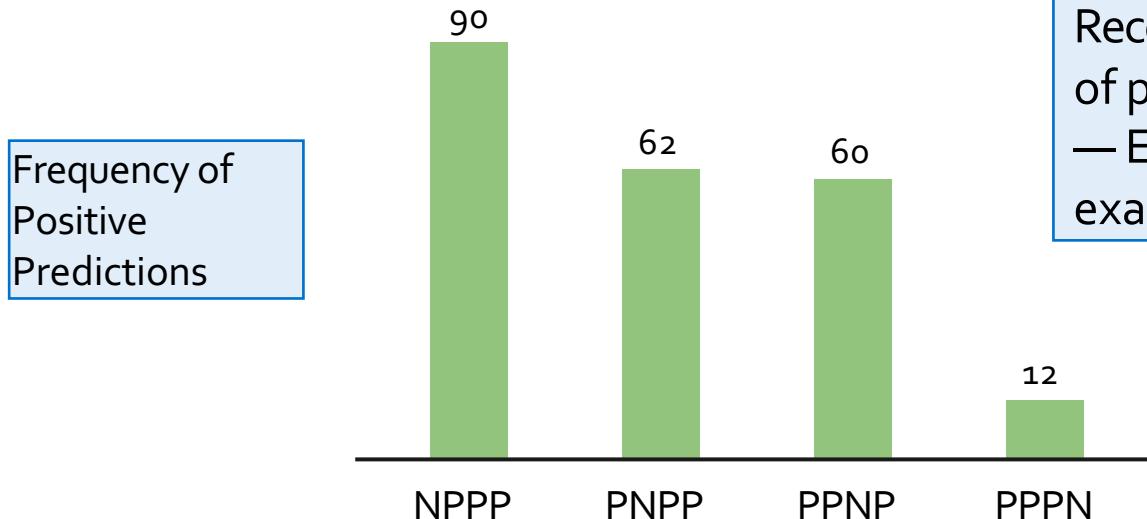
Prompt

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.



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

# Common Token Bias

---

# Common Token Bias

## Language Model



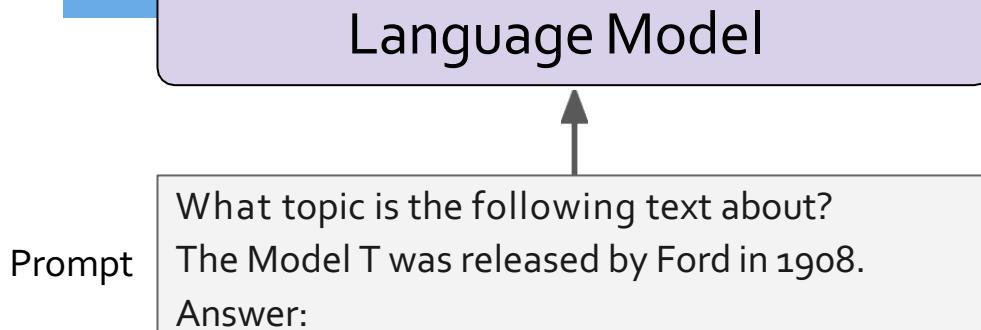
Prompt

What topic is the following text about?

The Model T was released by Ford in 1908.

Answer:

# Common Token Bias



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

# Surface Form Competition

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

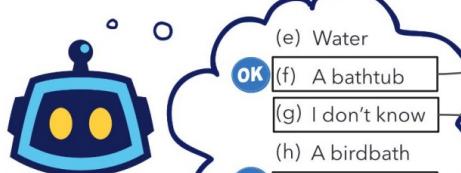
Humans select options



- (a) Coffee cup
- (b) Whirlpool bath
- (c) Cup
- (d) Puddle

$$P(\text{Bathtub} \mid x) = 0.8 \rightarrow P(\text{Whirlpool bath} \mid x) \leq 0.2$$

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

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 — skew the probabilities.  
There are some ideas on how to calibrate for these issues (see [Holtzman et al 2021](#)).

# Summary Thus Far

- In-context learning, while being fascinating, is highly unstable/brittle.
- Next: why does ICL emerge?

# Why Does ICL Emerge?

# How/Why does In-context Learning Work?

Any arbitrary task



Language Model

A few-shot learner



# How/Why does In-context Learning Work?

Any arbitrary task



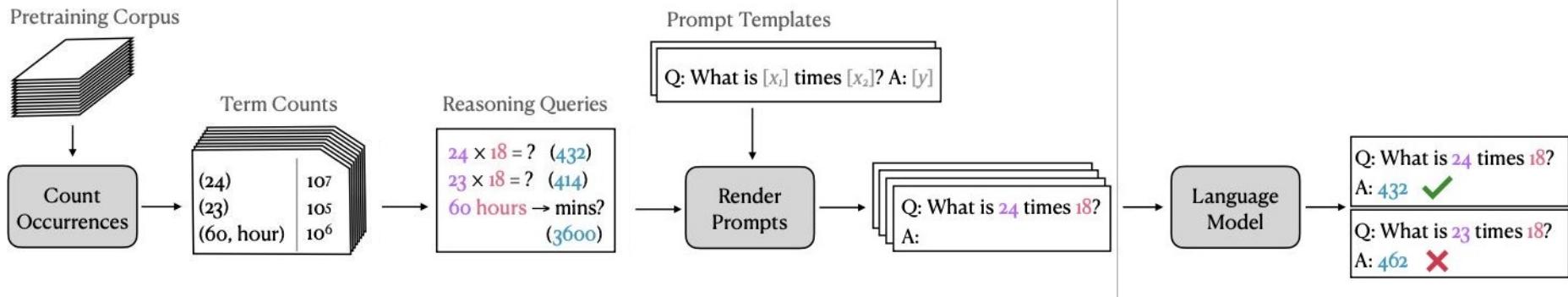
Language Model

A few-shot learner



# 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

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

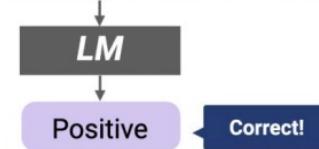
# Impact of Input-Output Mapping

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

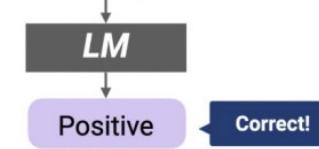
Circulation revenue has increased by 5% in Finland. \n Positive  
Panostaja did not disclose the purchase price. \n Neutral  
Paying off the national debt will be extremely painful. \n Negative  
The company anticipated its operating profit to improve. \n \_\_\_\_\_



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 \_\_\_\_\_



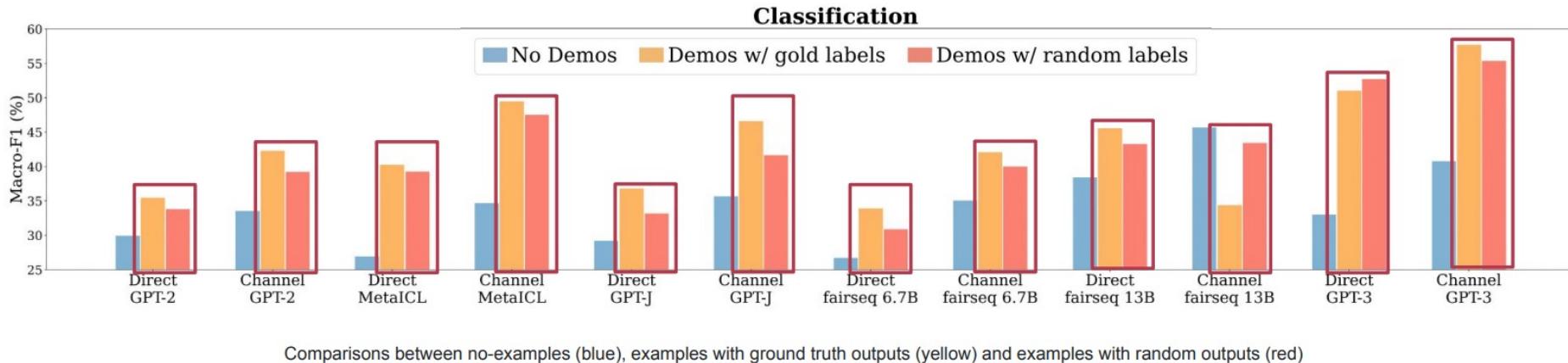
Prompt with true labels



Prompt with random labels

# Impact of Input-Output Mapping

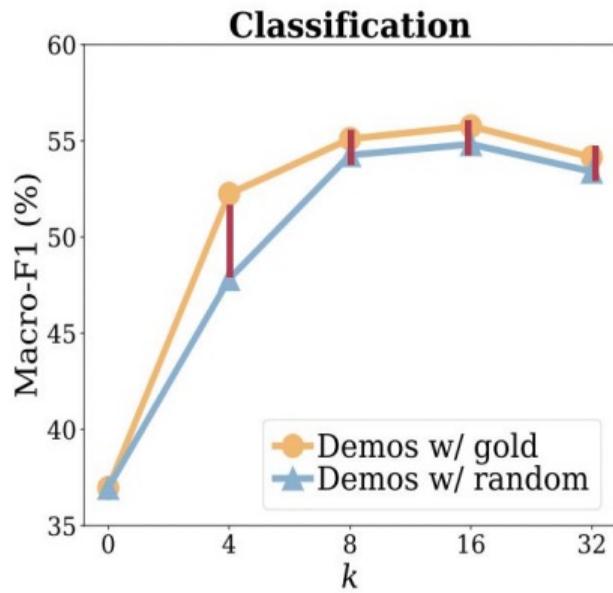
- Models see a small performance drop (0–5% absolute) with random labels



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

# Impact of Input-Output Margin

- Vary number of demonstrations
- **Takeaway:**
  - Performance drop from using gold labels to random labels is consistently small across varying  $k$ , ranging from 0 to 32.
  - Using small number of examples with random labels leads to similar performance as using gold labels.

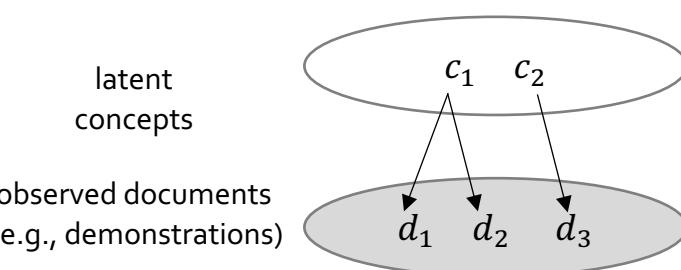


# In-context Learning as Bayesian Inference

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

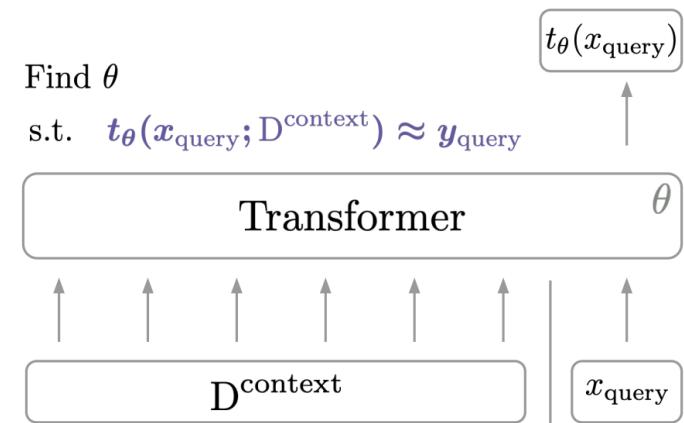
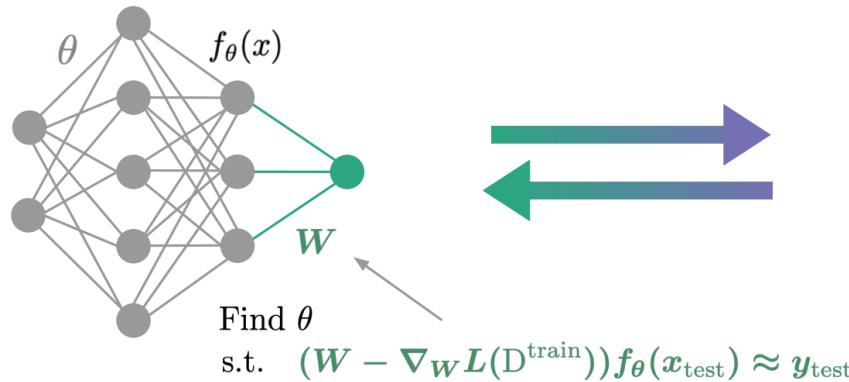
## Idea:

- (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



# In-context Learning as Gradient Descent

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



# Summary & Open questions

- In-context learning has been a promising few-shot learning approach
  - No need for gradient updates → Much easier to use large models!
- Better calibration, better scoring of model outputs, and better formation of demonstrations lead to great improvements
  - How to make it less sensitive?
  - How to scale it (longer context, more training examples, wider range of tasks)?
- Still in progress ...
  - Understanding how/why it works,
  - Disentangling looking up task location vs learning a new task
  - Can we predict whether in-context learning would work on a given task or not?

# Prompting Multi-step Problems

- Useful slides: [ACL 2023 Tutorial: Complex Reasoning in Natural Language \(wenting-zhao.github.io\)](https://wenting-zhao.github.io/)
- [Teach-LLMs-to-Reason-7-2023 \(dennyzhou.github.io\)](https://dennyzhou.github.io/Teach-LLMs-to-Reason-7-2023)

# 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**

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

A: The answer is **nk**.

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

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

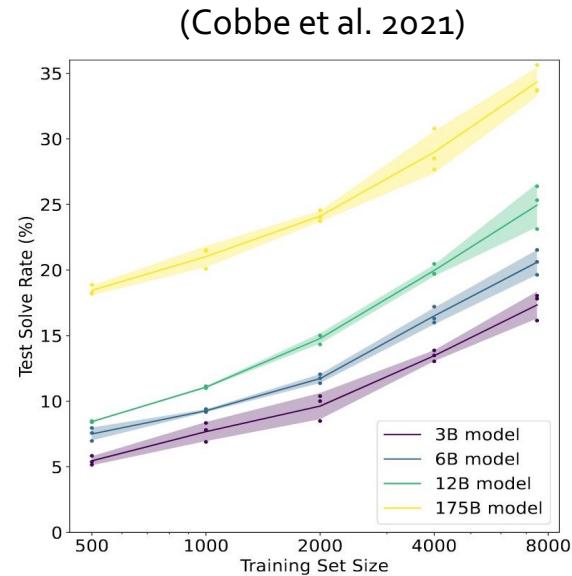
Arithmetic Reasoning (AR)  
 $(+ - \times \div \dots)$

Symbolic Reasoning (SR)

Commonsense Reasoning (CR)

# Reasoning Problems

- 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?



# 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.

# Reasoning Problems via Multi-Step Prompting

## (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*

# Reasoning Problems via Multi-Step Prompting

(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. ✓

Step-by-step demonstration

Step-by-step Answer

# Reasoning Problems via Multi-Step Prompting

(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. ✓

Step-by-step demonstration

Step-by-step Answer

(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

# Reasoning Problems via Multi-Step Prompting

(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. ✓

Step-by-step demonstration

Step-by-step Answer

(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

(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?

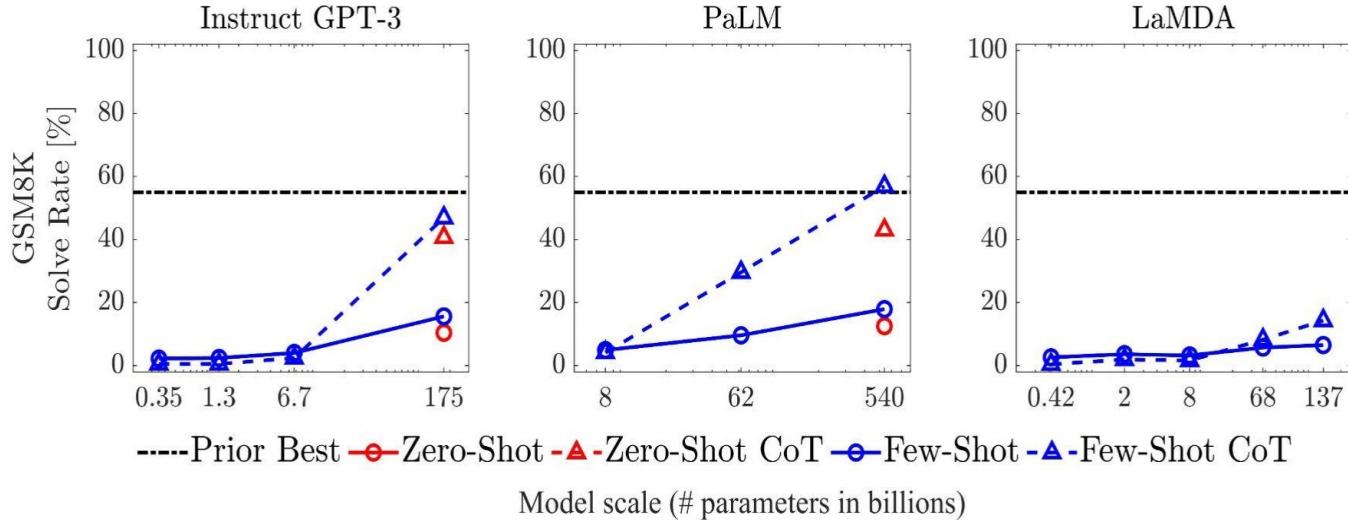
A: Let's think step by step.

(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. ✓

Two-stage Prompting  
Step-by-step Answer

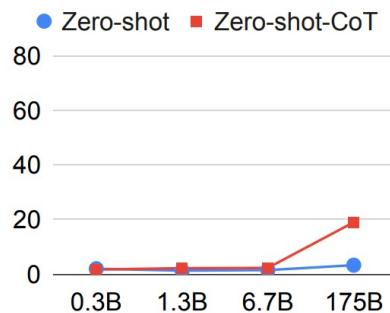
# Multi-Step Prompting: Empirical Results

- **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**

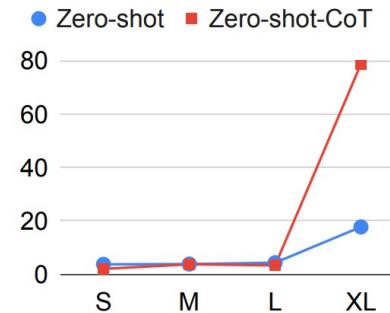


# Multi-Step Prompting: Empirical Results

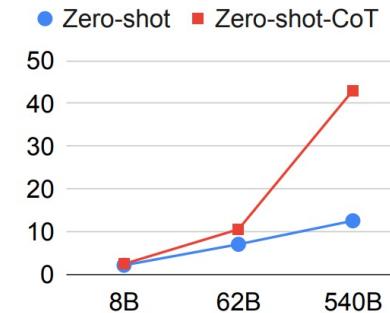
- **Setup:** show **demonstrations** that contain the **decompositions**
- The gains of multi-step prompting increases with scale.
- Prompting achieves **better perf than** [smaller] models that are fine-tuned on a lot more data.



(a) MultiArith on Original GPT-3



(b) MultiArith on Instruct GPT-3

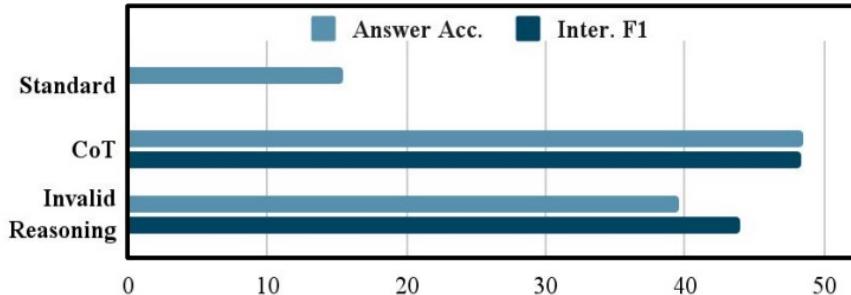


(c) GMS8K on PaLM

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.

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

- It is possible even with invalid demonstrations
- Prompting with invalid reasoning steps can act 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. <span style="color:red;">X</span>
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. <span style="color:green;">✓</span>
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. <span style="color:green;">✓</span>

# 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 given references in the proofs?
  - How do use external “tools” (e.g., logic, calculator, Python) in forming proofs?

# Summary

---

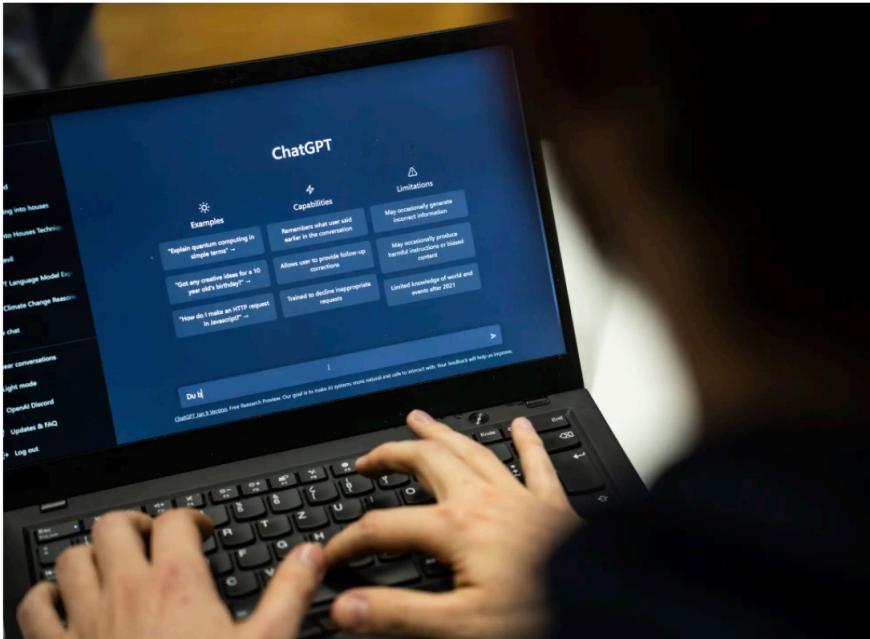
- Prompting language models is a powerful way to adapt them to our desired tasks.
- They also serve as a gateway to understand the underlying dynamics inside models.
- Lots of activity in this area and room for a lot of research progress.

# 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



Read in app



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

# The Phases of Our Understanding

---

“Language modeling is a useful **subtask** for many NLP tasks”  
– everyone, pre-2018

“Language modeling is a useful **supertask** for many NLP tasks”  
– everyone, post-2018