



Adapting Language Models

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

Adapting Language Models: Chapter Plan

You have a pre-trained language model that is pre-trained on massive amounts of data.

Now how to you use it to “adapt” for your use-case?

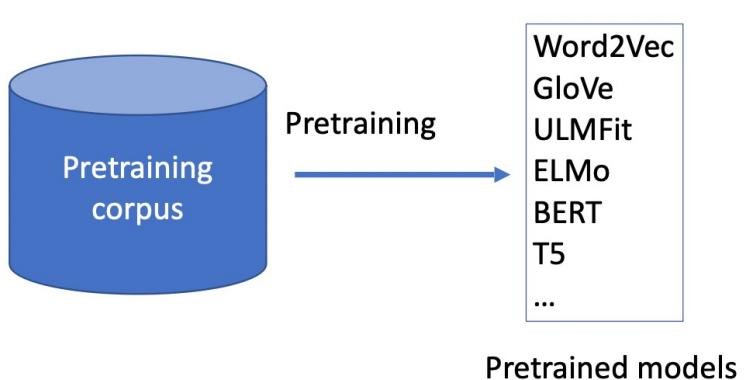
- **Prompting:** adapting model inputs (language statements)
- **Tuning:** adapting (modifying) model parameters

Adapting Language Models: Chapter Plan

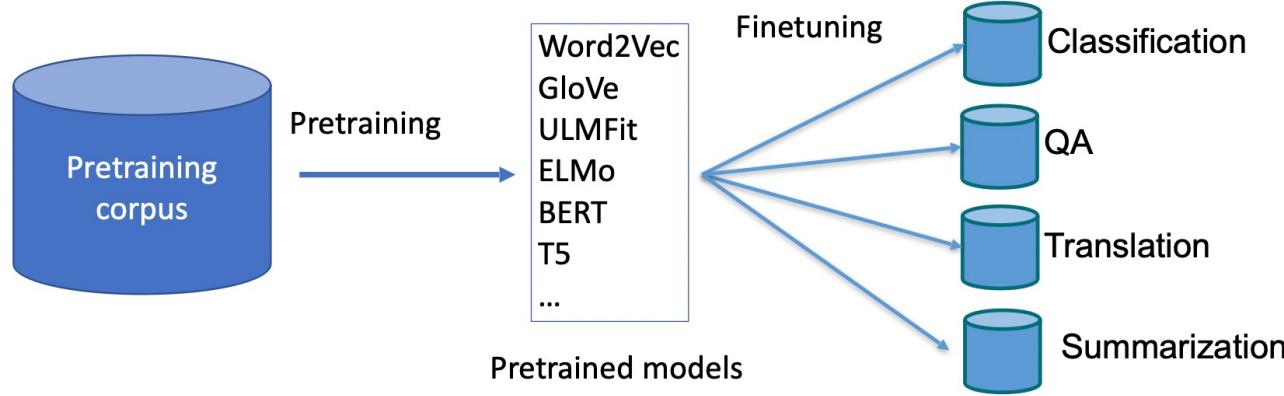
1. Adaptation via Fine-tuning
2. Parameter-efficient fine-tuning
3. Adaptation via In-Context Learning
4. In-context learning of multi-step problems.

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

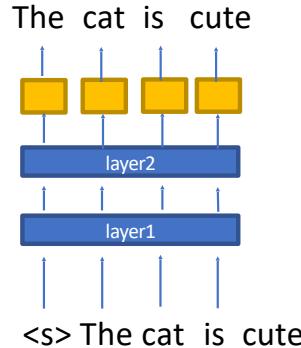
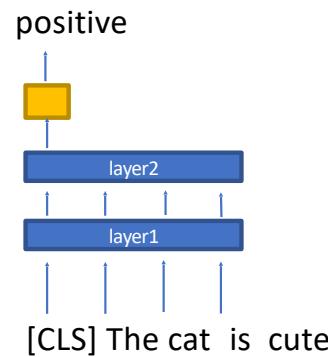
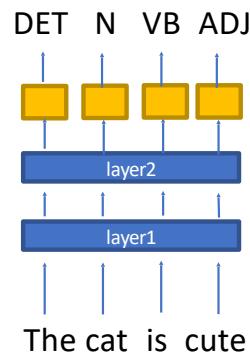
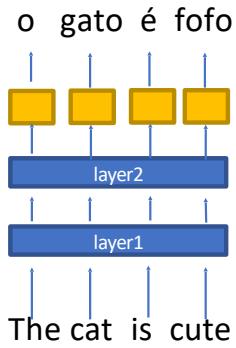
Adaptation via Fine-Tuning



Fine-Tuning for Tasks



Fine-Tuning for Tasks



Translation

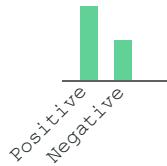
POS Tagging

Text classification

Language modeling

Fine-tuning Pre-trained Models

Classification Head



Language Model



Embeddings



Input

[CLS] A three-hour cinema master class.

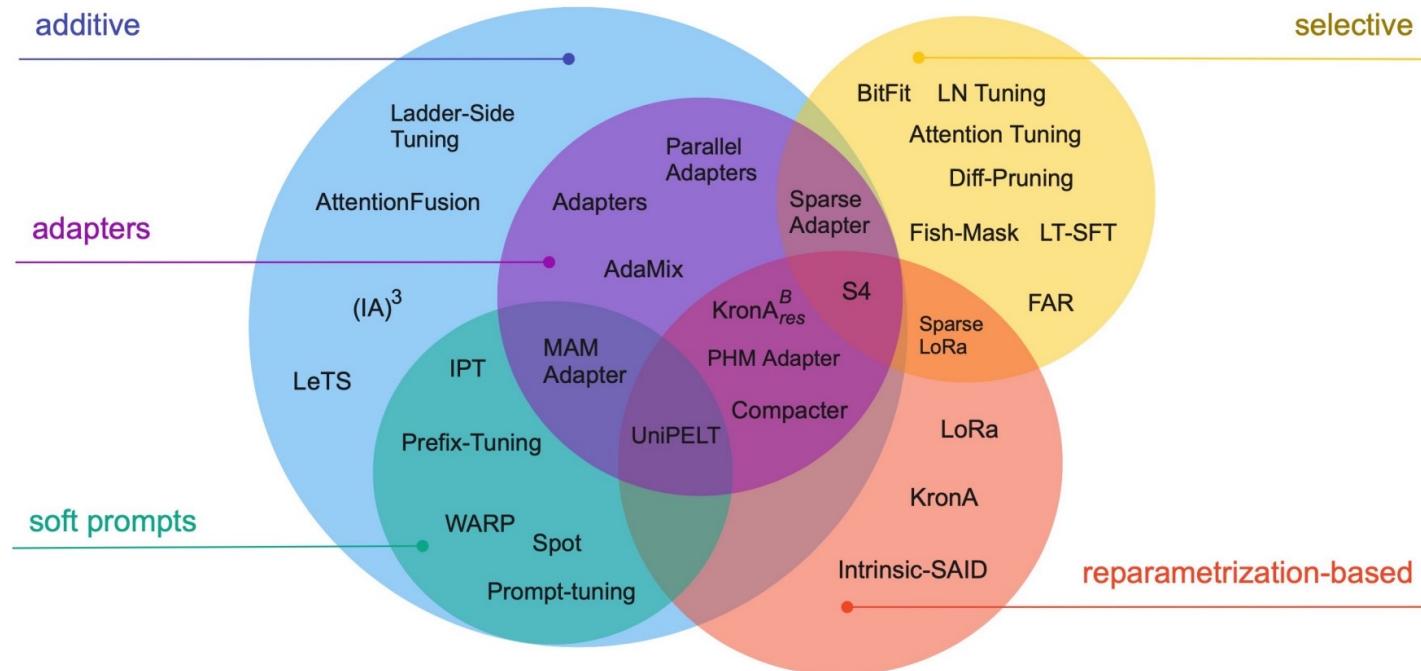
- Whole model tuning:
 - Run an optimization defined on your task data that updates **all** model parameters

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

Parameter-efficient Fine-tuning

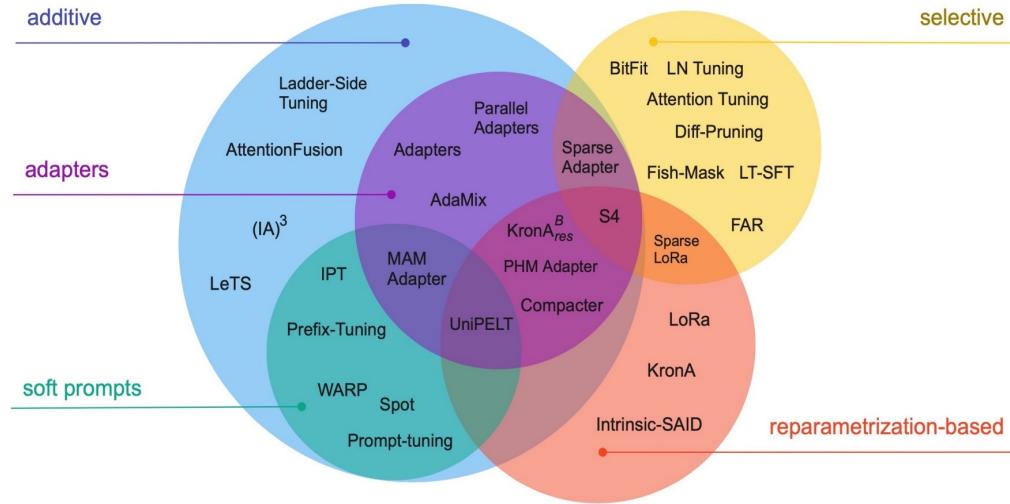
- In fine-tuning we need to updating and storing all the parameters of the LM
 - We would need to store a copy of the LM for each task
- With large models, storage management becomes difficult
 - E.g., A model of size 170B parameters requires ~340Gb of storage
 - If you fine-tune a separate model for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

Parameter-efficient Fine-tuning

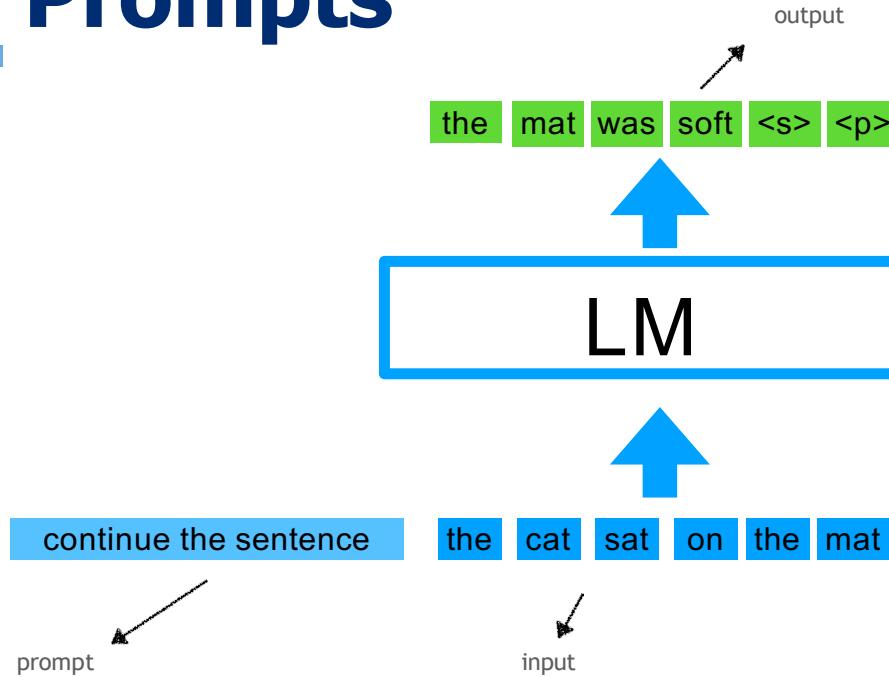


Parameter-efficient Fine-tuning: Adding Models

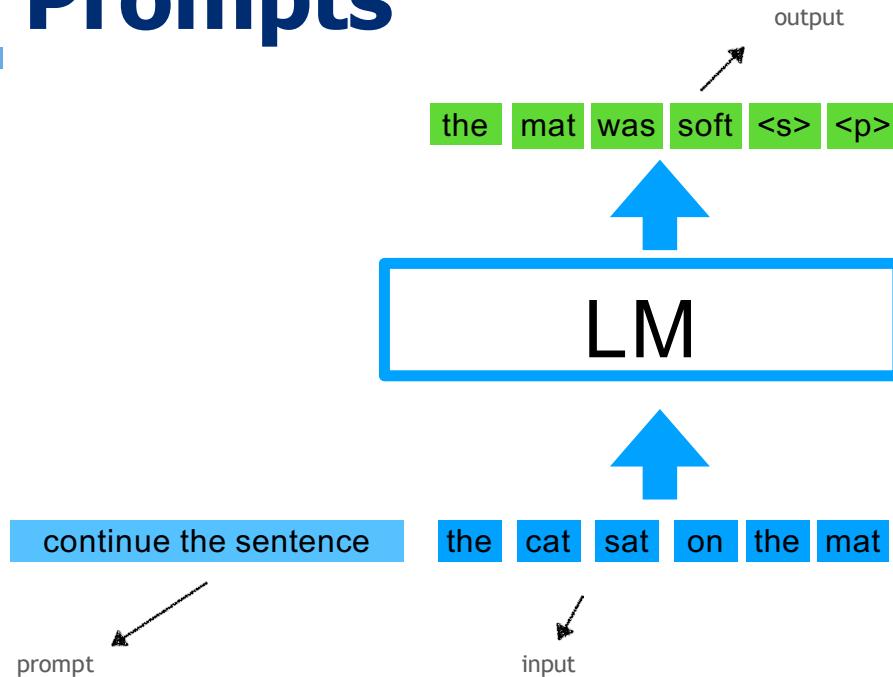
- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- Two commonly used methods:
 - **Soft prompts**
 - Adapters



Soft Prompts

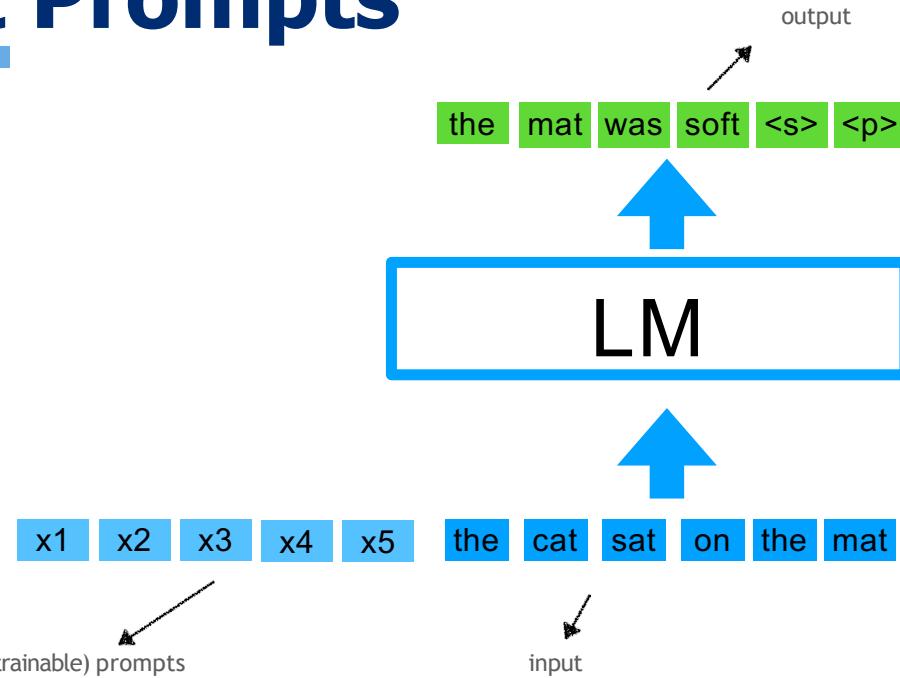


Soft Prompts



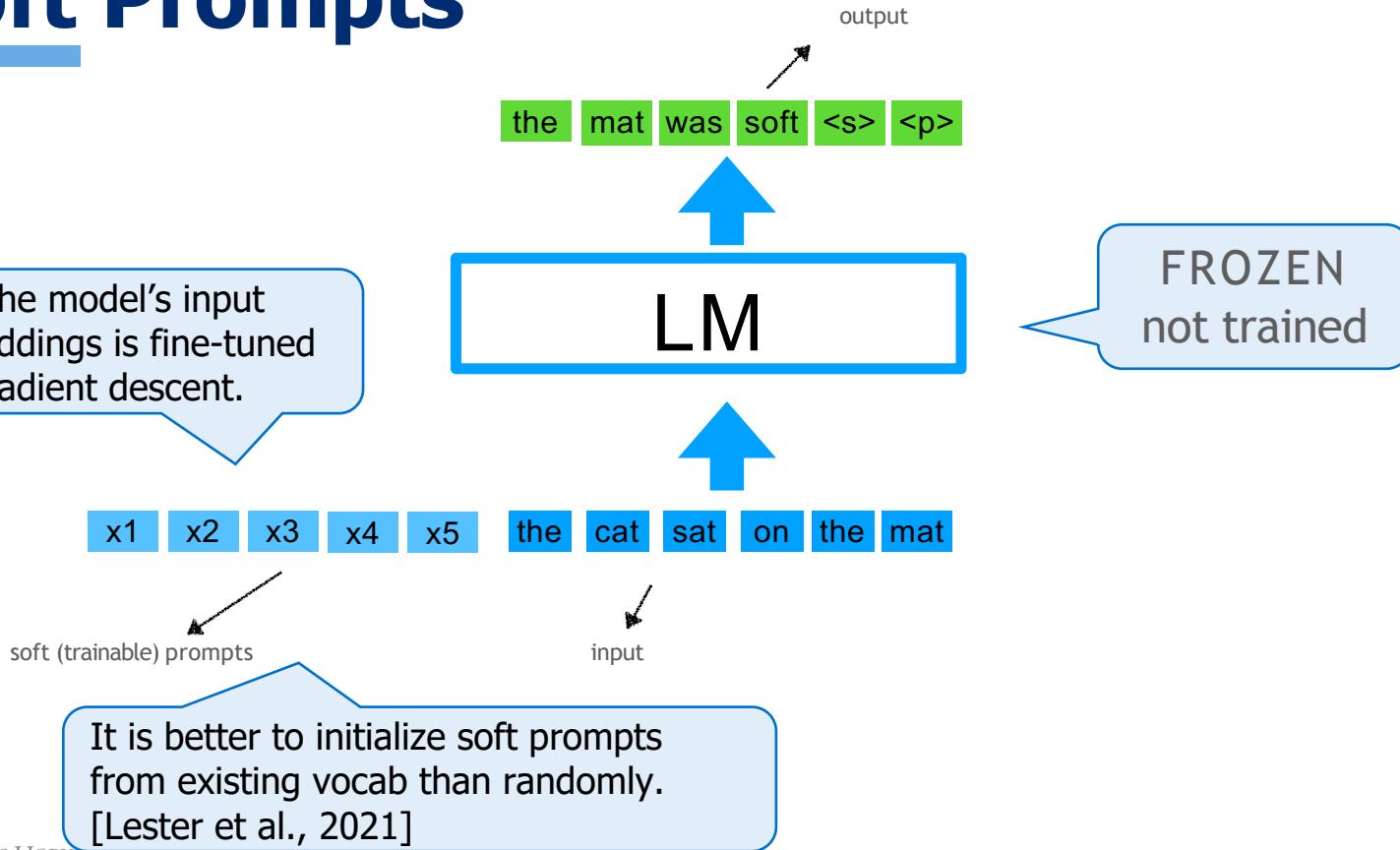
- Designing good prompts might be difficult for each task
- Maybe we can “learn” the prompts?

Soft Prompts



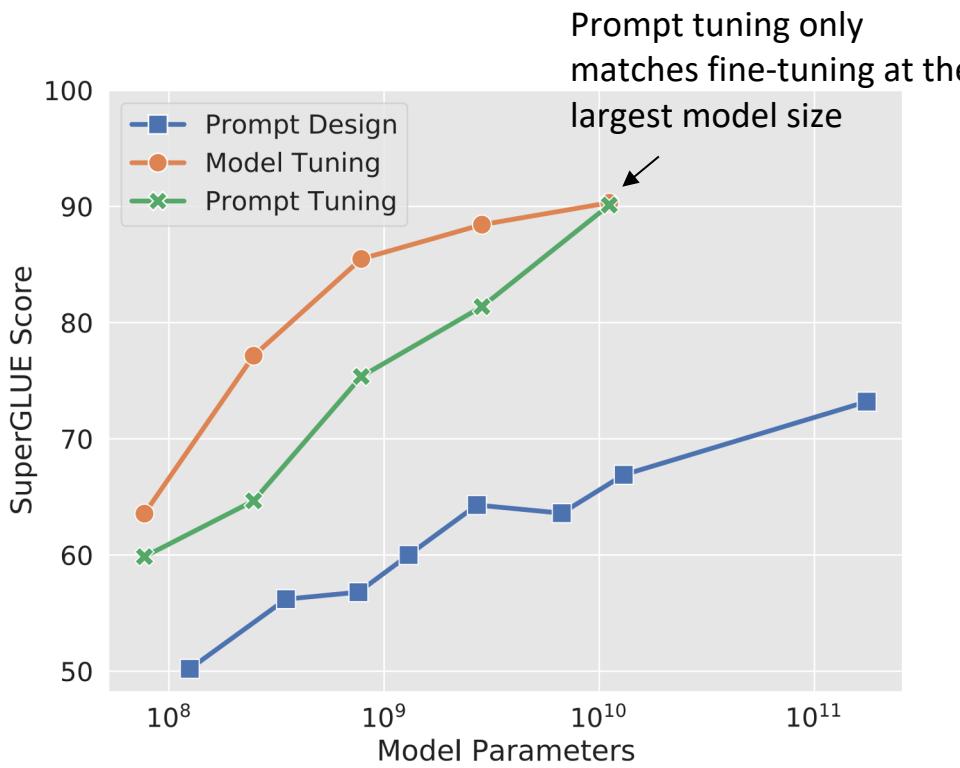
- Designing good prompts might be difficult for each task
- Maybe we can “learn” the prompts?

Soft Prompts



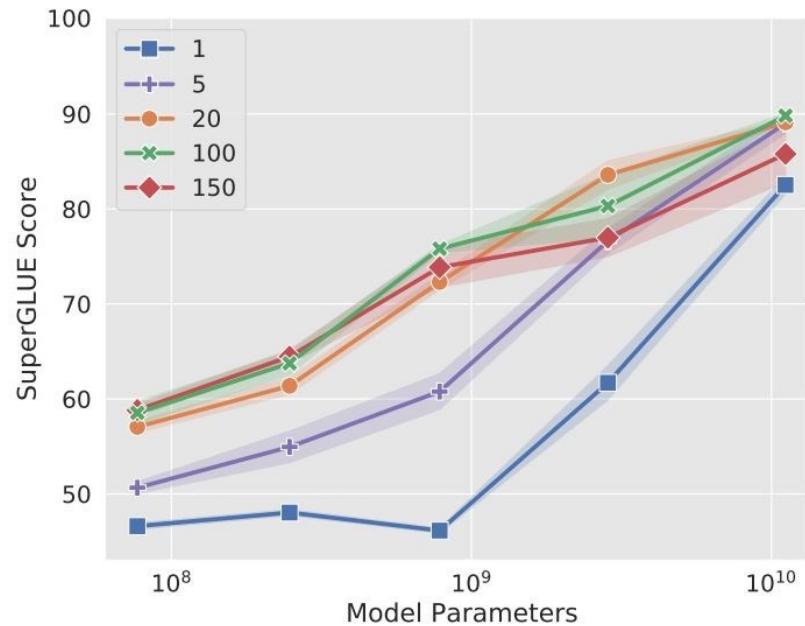
Prompt Tuning: Effect of Prompt Length

- 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

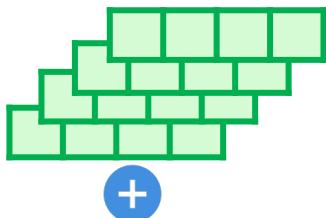


Are continuous prompts as interpretable as
Discrete (Language) prompts?

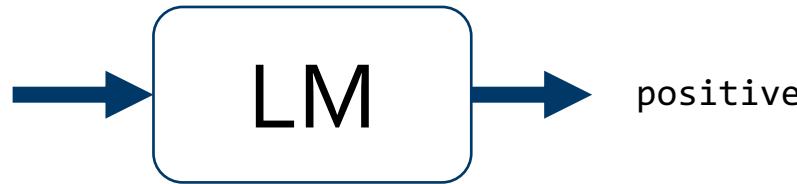
[Continuous] Prompt Waywardness

Prompt waywardness hypothesis, states that:

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



Sentence: That was a great fantasy movie.

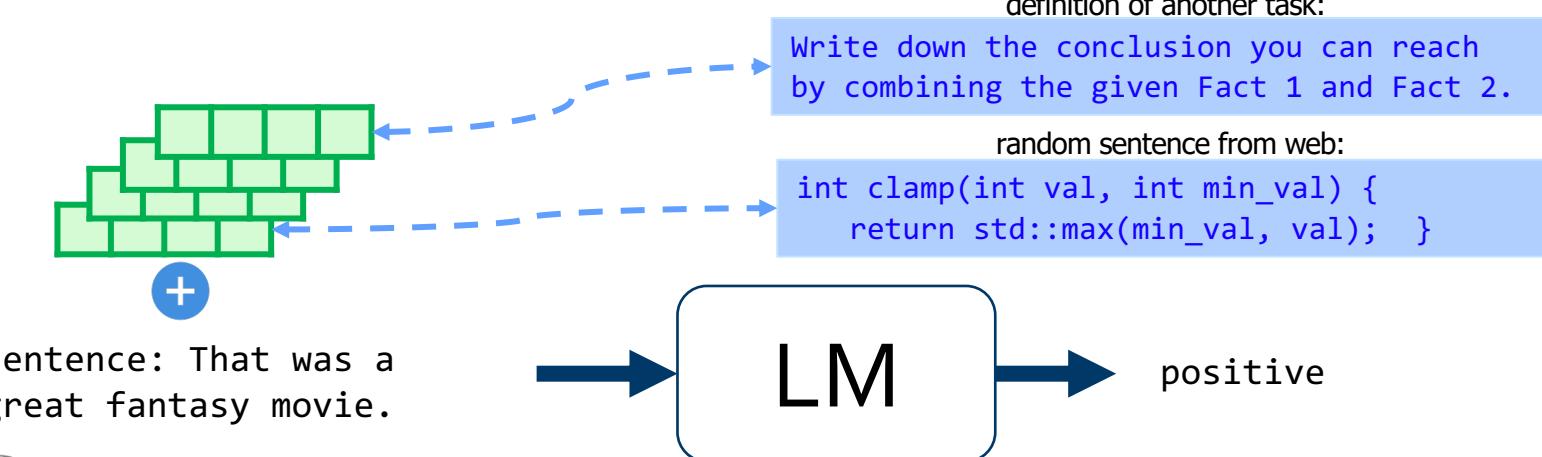


[Continuous] Prompt Waywardness

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)



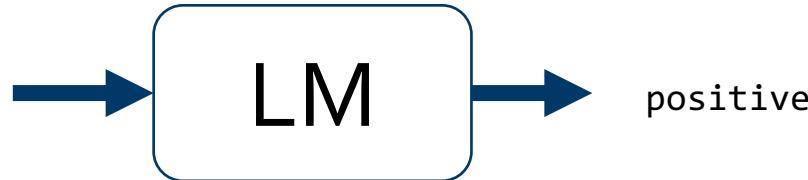
Sentence: That was a
great fantasy movie.

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

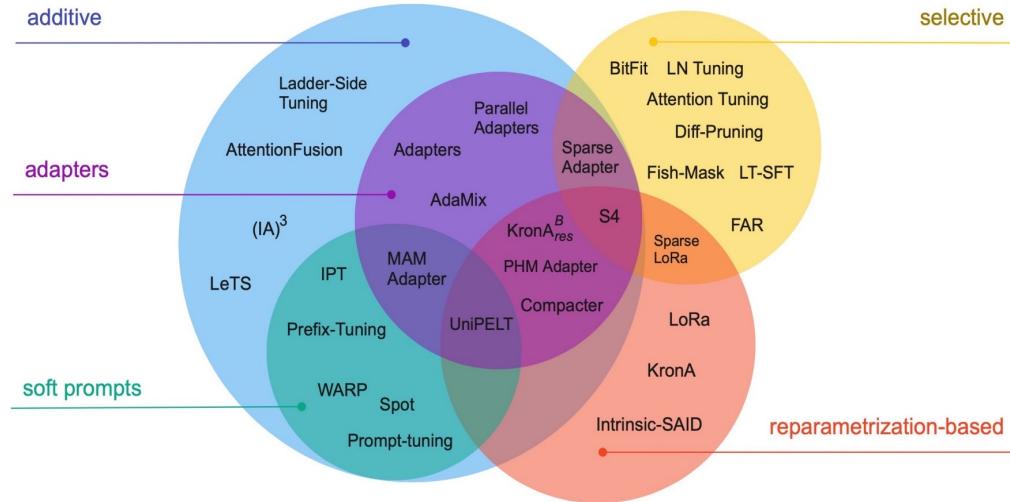


continuous prompts:
unclear how to interpret, but easy to optimize



Parameter-efficient Fine-tuning: Adding Models

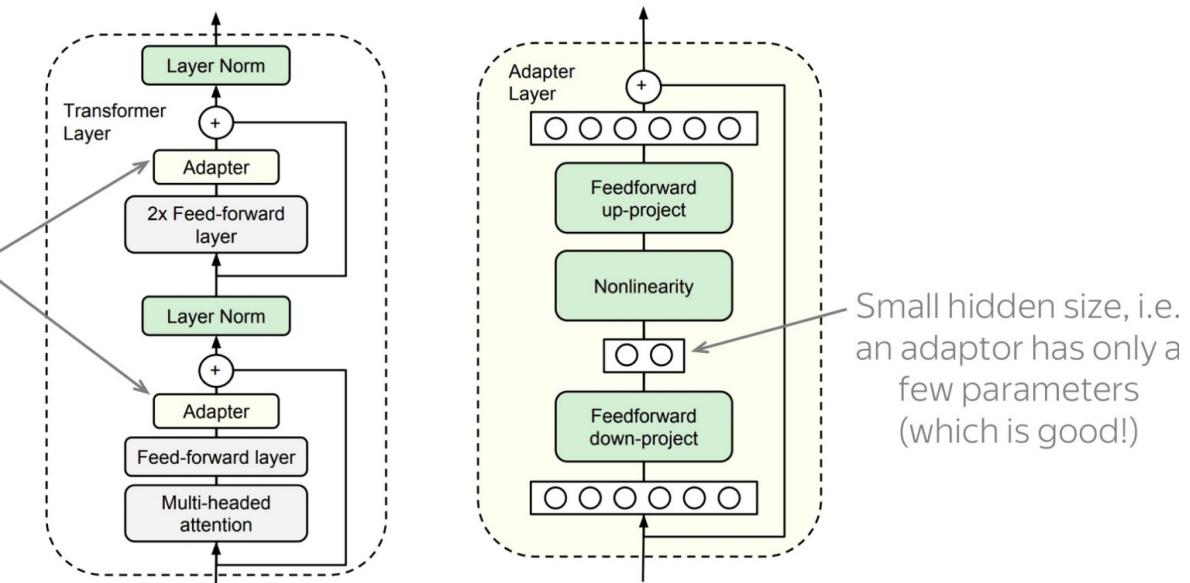
- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- Two commonly used methods:
 - Soft prompts
 - **Adapters**



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



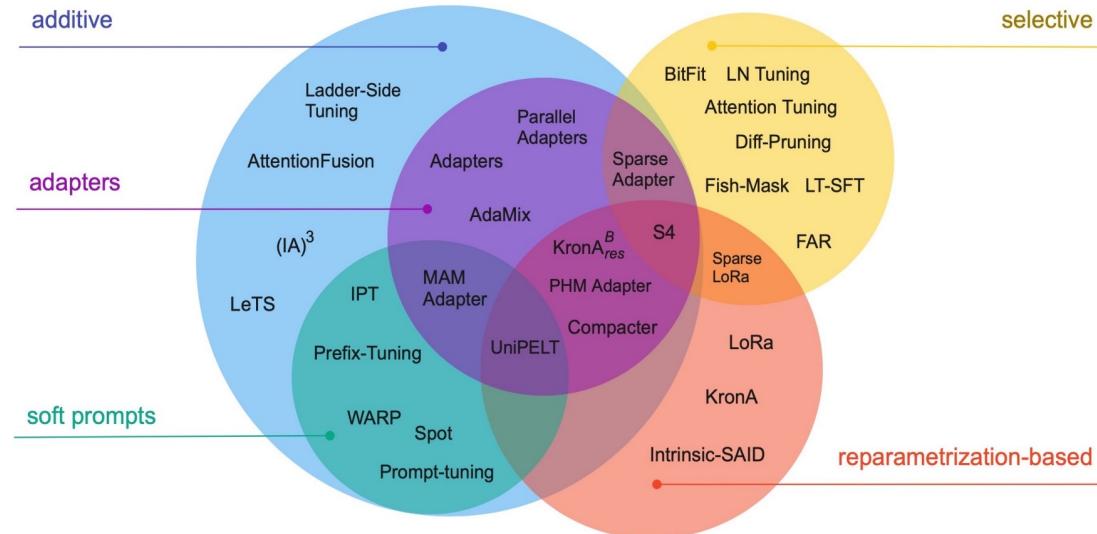
["Parameter-Efficient Transfer Learning for NLP", Houlsby et al., 2019.]

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

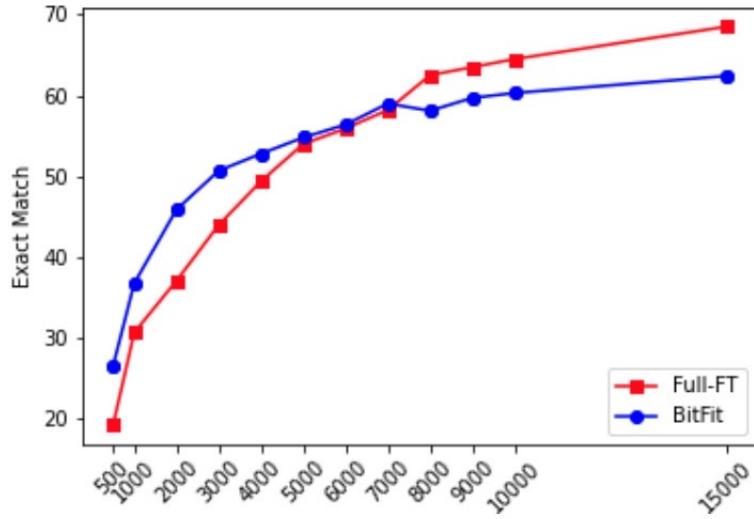
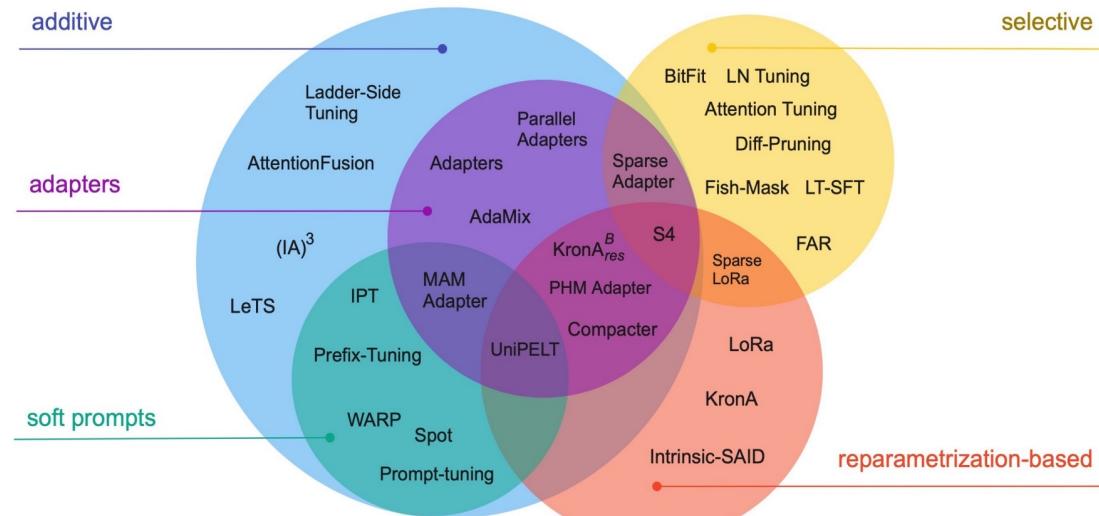


Figure 2: Comparison of BitFit and Full-FT with BERT_{BASE} exact match score on SQuAD validation set.

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

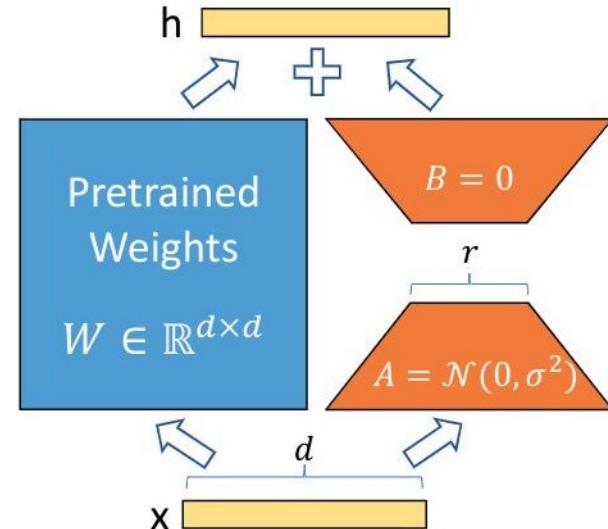
- Hypothesis: the intrinsic rank of the weight matrices in a large language model is low
- Parameter update for a weight matrix is decomposed into a product of two low-rank matrices

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

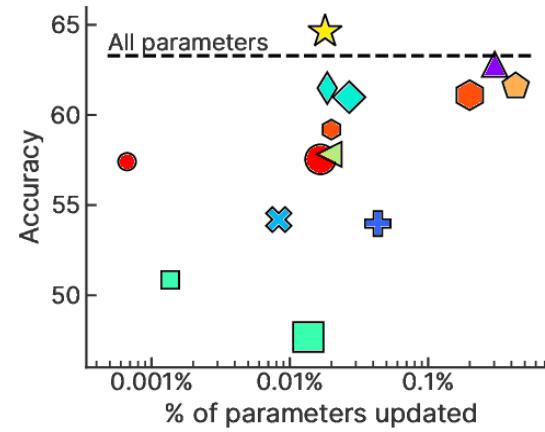
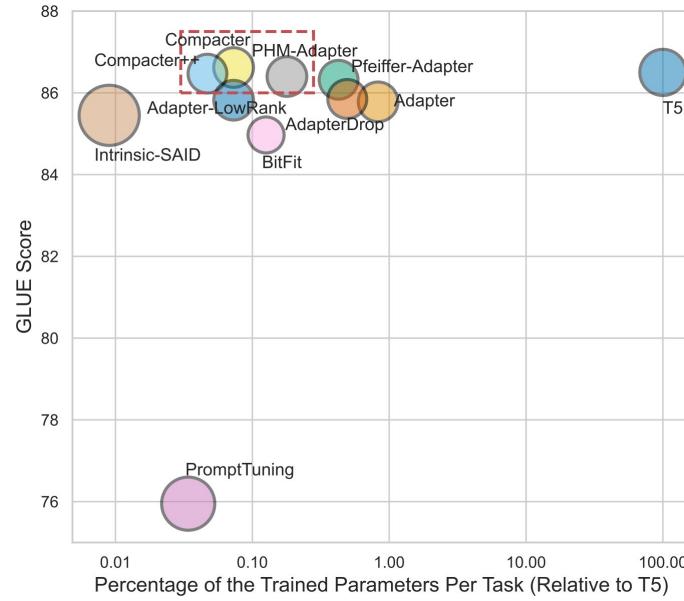
$$\Delta W = BA$$

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

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



Performance/compactness comparison

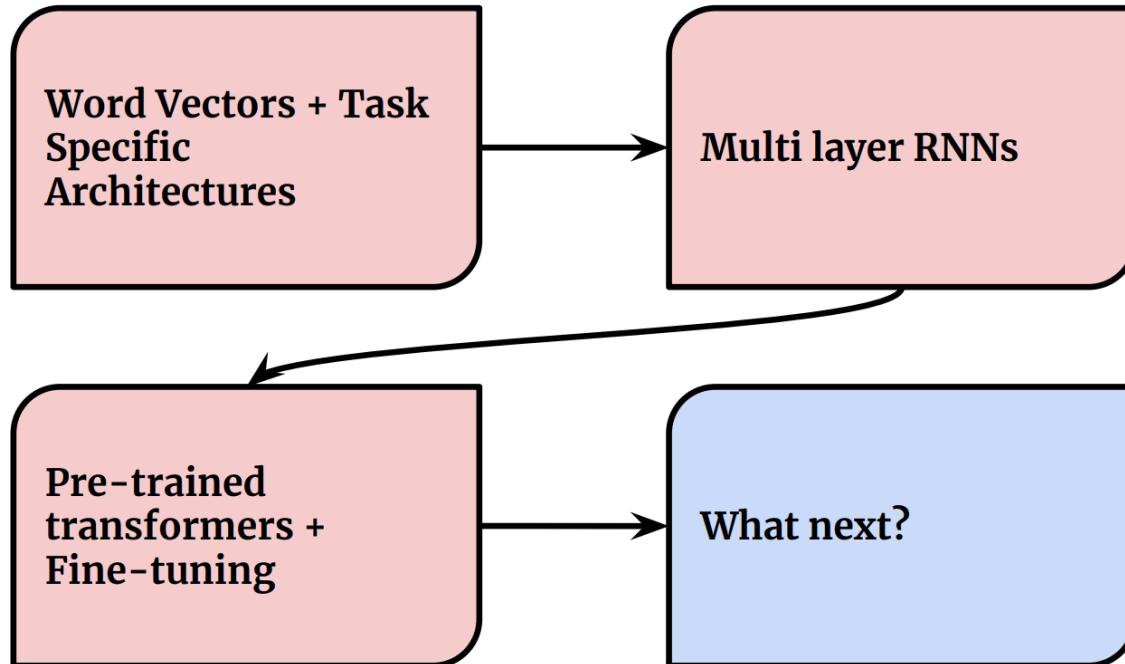


Summary

- 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.

In-Context Learning

The Phases of Paradigms



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

-- No one

Limitations of Pre-training -> Fine-tuning

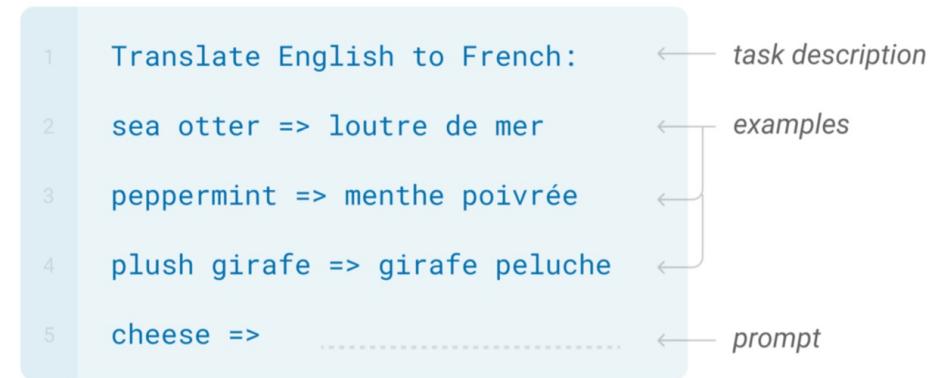
- Often you need a **large labeled data**
 - Though more pre-training can reduce the need for labeled data
- End up with **many copies** (or sub-copies) of the same model

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

In-context Learning

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

GPT3: Try it yourself!

<https://beta.openai.com/playground>

In-Context (Few Shot) Prompting

- Popularized by GPT-3 (but predates that model)
- Perform a task based on a few examples provided in the inference time.
- The model identifies patterns in examples and replicates it

GPT-3: Language Models are Few-Shot Learners

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French:      ← task description  
2 cheese => .....               ← prompt
```

One-shot

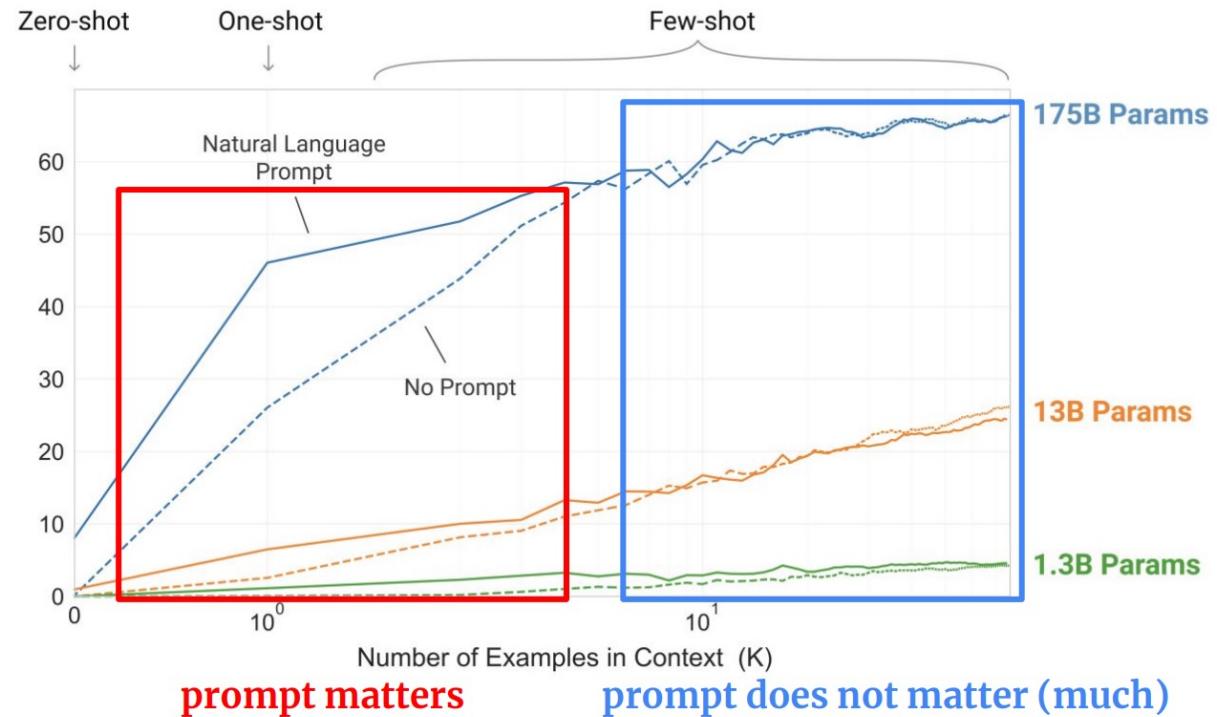
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1 Translate English to French:      ← task description  
2 sea otter => loutre de mer    ← example  
3 cheese => .....               ← prompt
```

Few-shot

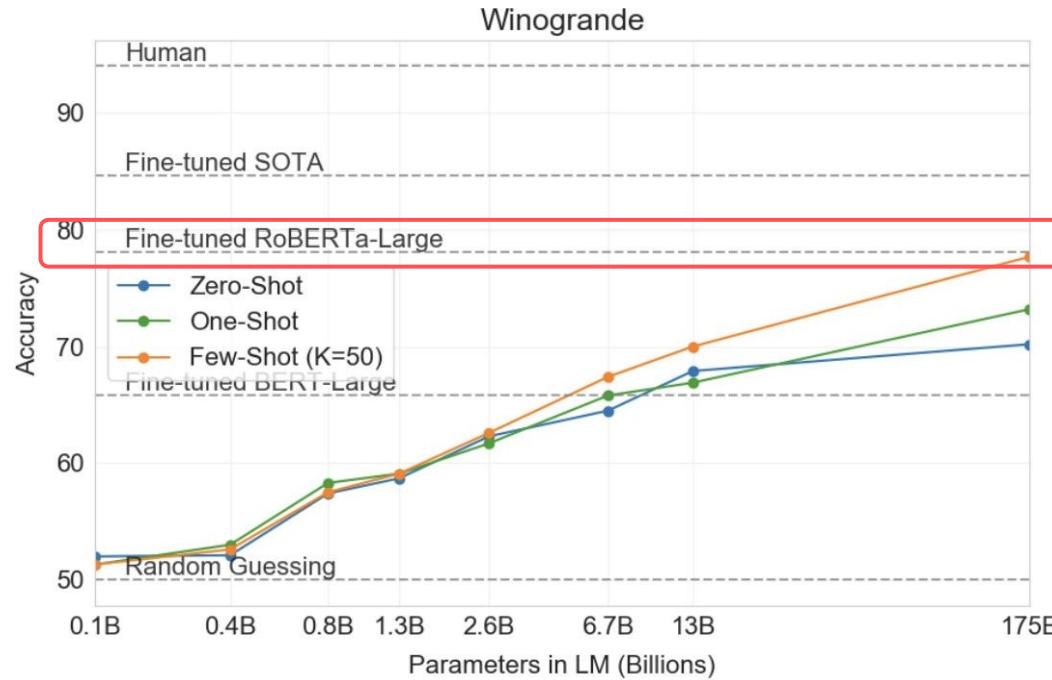
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

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



In-context learning results

[Brown et al. 2020.](#) "Language Models are Few-Shot Learners"



Robert woke up at 9:00am while Samuel woke up at 6:00am, so **he** had less time to get ready for school.
Robert woke up at 9:00am while Samuel woke up at 6:00am, so **he** had more time to get ready for school.

Robert / Samuel
Robert / Samuel

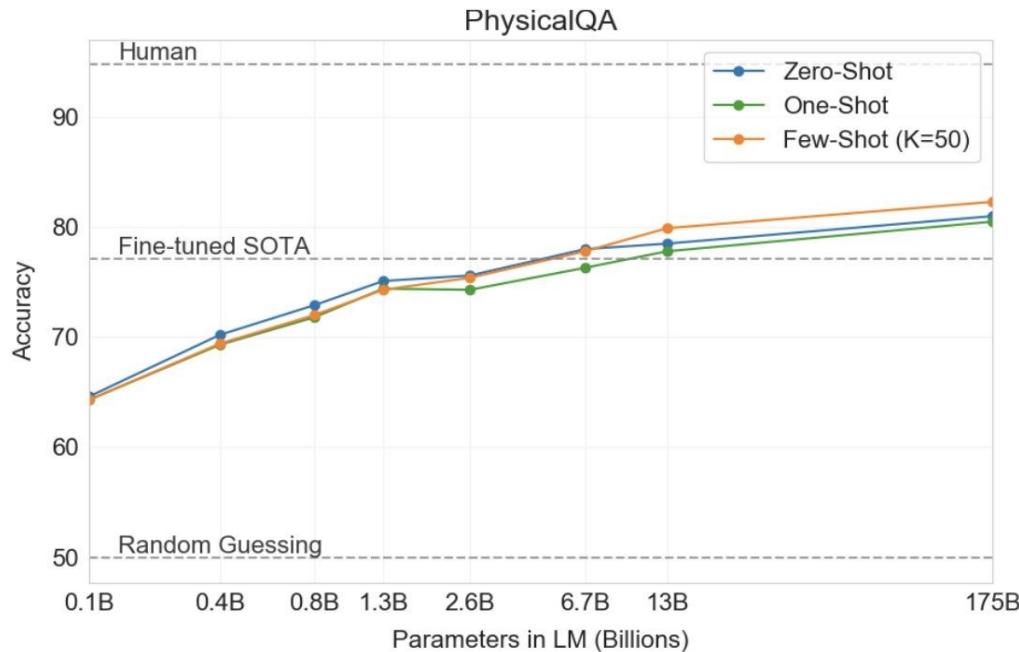
In-context learning results



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

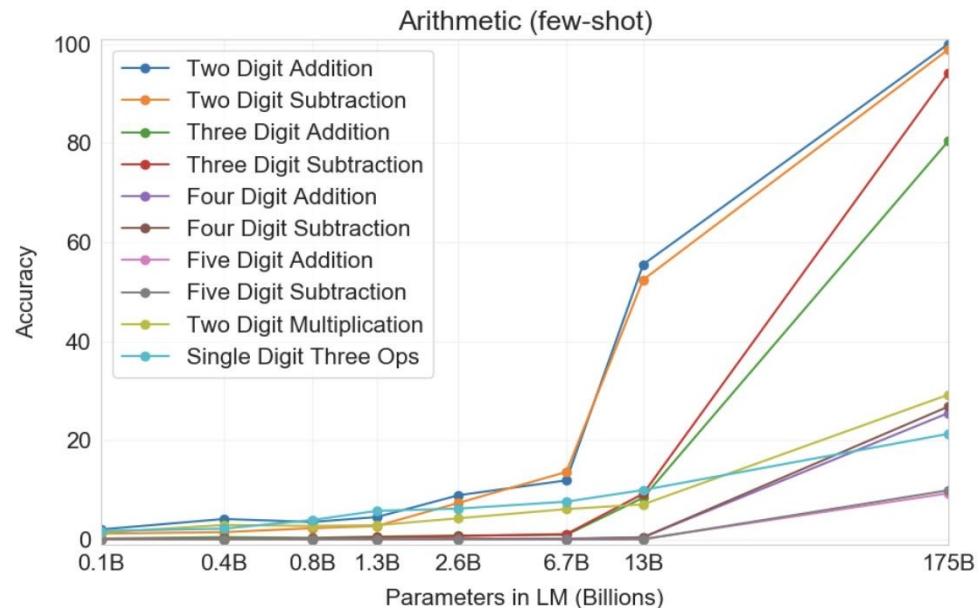
- a. **Squeeze** the water bottle and press it against the yolk. **Release**, which creates suction and lifts the yolk.

- 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!



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

Why Do We Care About Few-Shot Learning?

Practically Useful

Intellectually Intriguing

Practically Useful

Labeling data is costly

- May require domain expertise
 - Medical, legal, financial
- Inputs may be long/complex
 - Grammaticality
- Outputs may be complex
 - Semantic parsing

Practically Useful

Labeling data is costly

You want to do best with what you have

- You don't want to get more data
- Emergent, time-sensitive scenarios
 - Something new happened
 - Need to react quickly!

Practically Useful

Labeling data is costly

You want to do best with what you have

Finetuning can unstable

- Training is sensitive to hyperparams
- Not enough validation data
- We don't quite understand how finetuning works

Practically Useful

Labeling data is costly

You want to do best with what you have

Finetuning can be unstable

Finetuning large LMs is expensive

- Expensive to train, time and memory

Intellectually Intriguing

Potential test for “Intelligent Behavior”

- Generalization from few examples
 - Fundamental piece of intelligence
 - Often used in psychology
 - Quickly adjust to environment

Intellectually Intriguing

Potential test for “Intelligent Behavior”

“But... Deep learning is data hungry!”

- Long-standing criticism of DL
- Understand why it doesn’t work here
 - Or does it?
- What are the new limitations of DL?

Intellectually Intriguing

Potential test for “Intelligent Behavior”

“But... Deep learning is data hungry!”

Insights into Language Modeling

- What does an LLM “know”?
- What are the biases/limitations of LLMs?
- ...

Intellectually Intriguing

Potential test for “Intelligent Behavior”

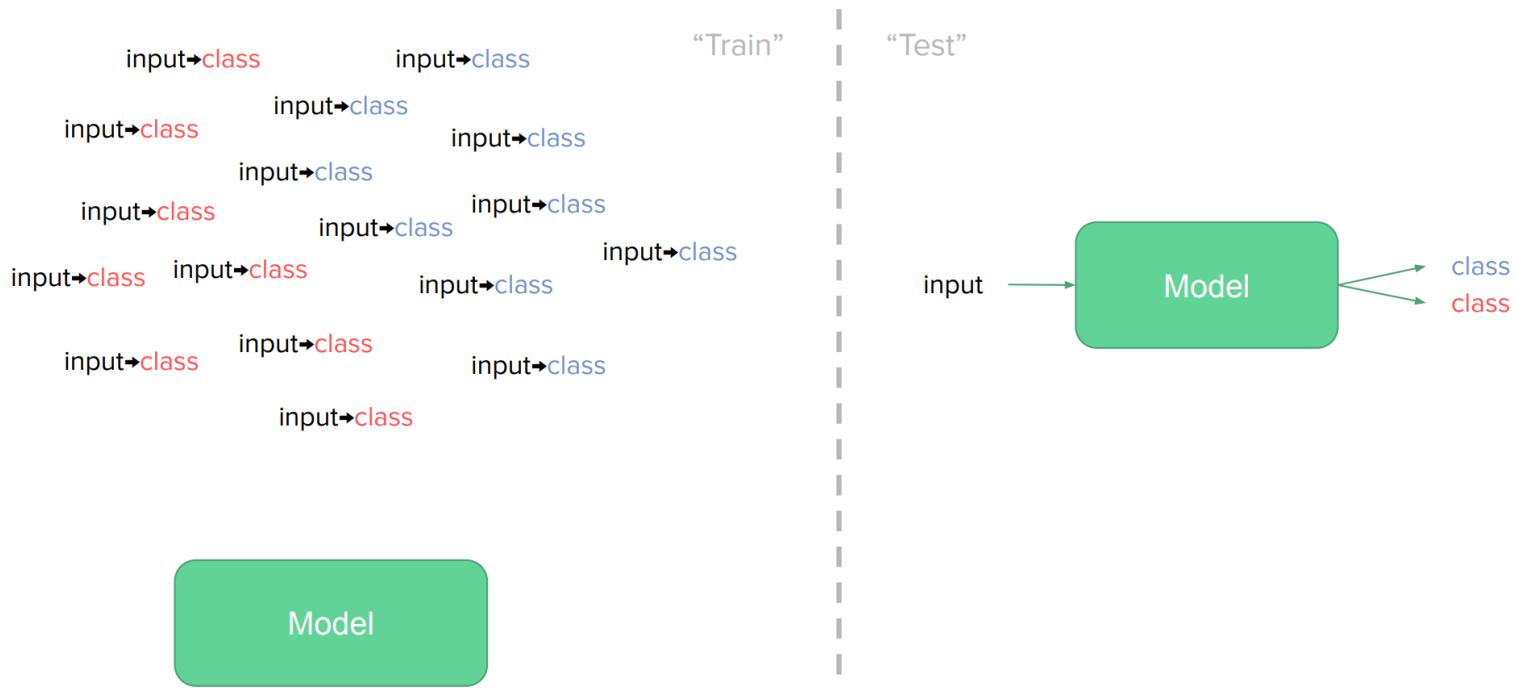
“But... Deep learning is data hungry!”

Insights into Language Modeling

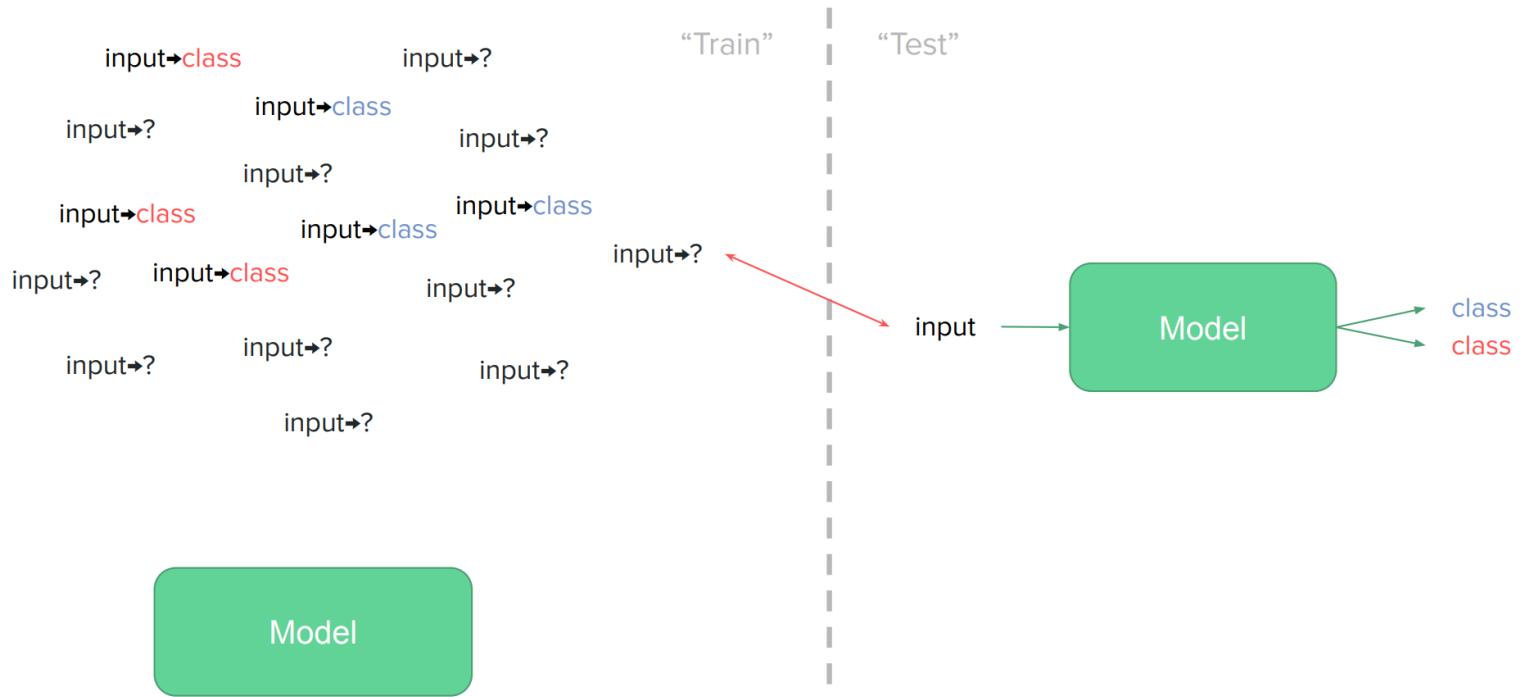
Because LARGE language models

- Training/inference/access is tough
- What else can we do? ;)

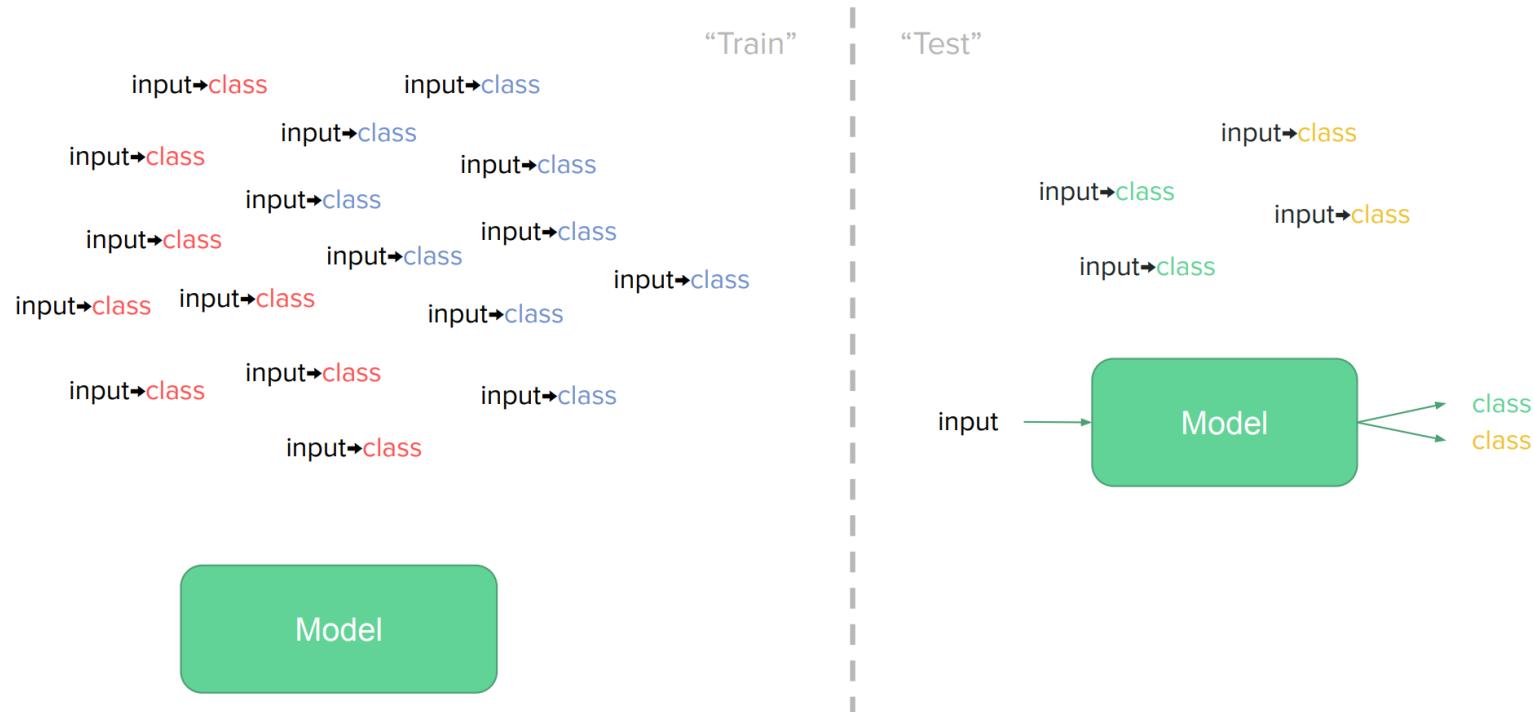
The Broader Context: Supervised Learning



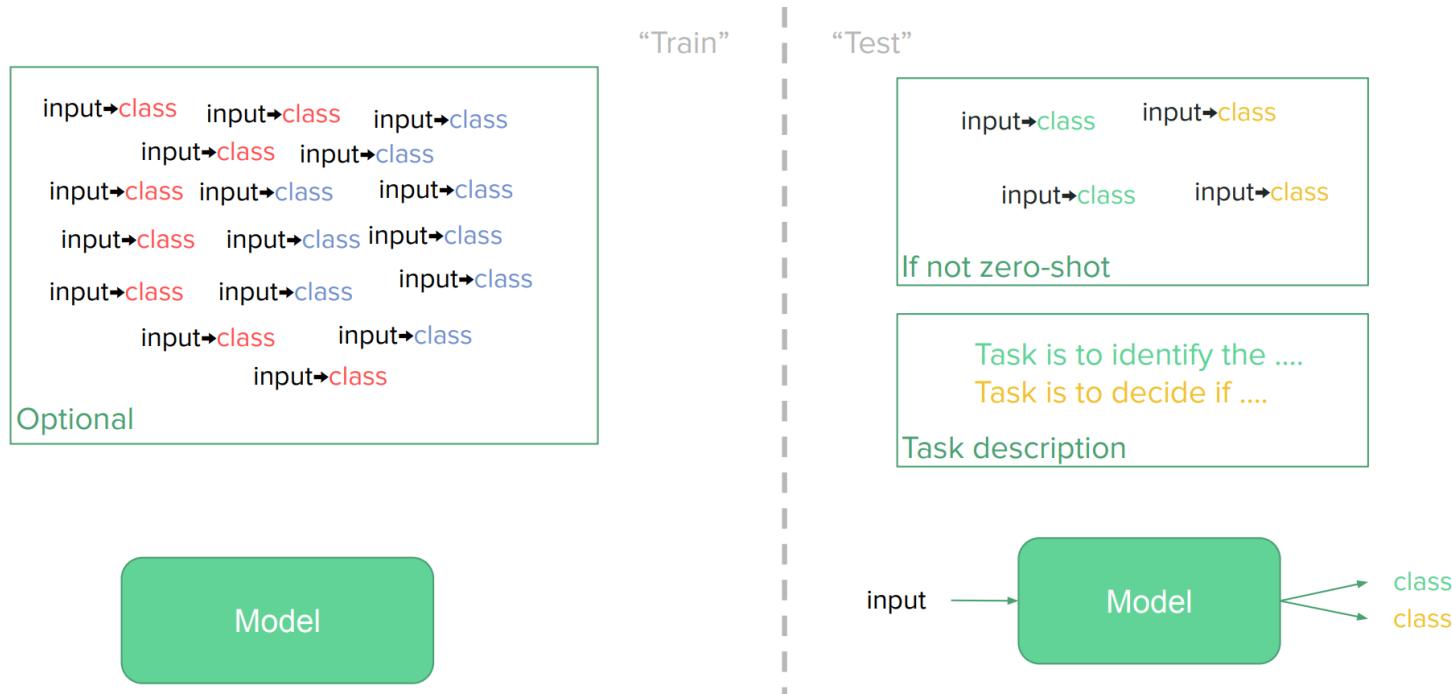
The Broader Context: Semi-Supervised Learning



The Broader Context: [Traditional] Few-shot Learning



The Broader Context: [Modern] Few-shot Learning



In-Context Prompting

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

$$p_1 = P(\text{It was great!} \mid \text{1st train input+output} \backslash n \text{ 2nd train input+output} \backslash n \text{ A three-hour cinema master class.})$$

$$p_2 = P(\text{It was terrible!} \mid \text{1st train input+output} \backslash n \text{ 2nd train input+output} \backslash n \text{ A three-hour cinema master class.})$$

$$\begin{array}{ll} p_1 > p_2 & \text{"positive"} \\ p_1 < p_2 & \text{"negative"} \end{array}$$

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

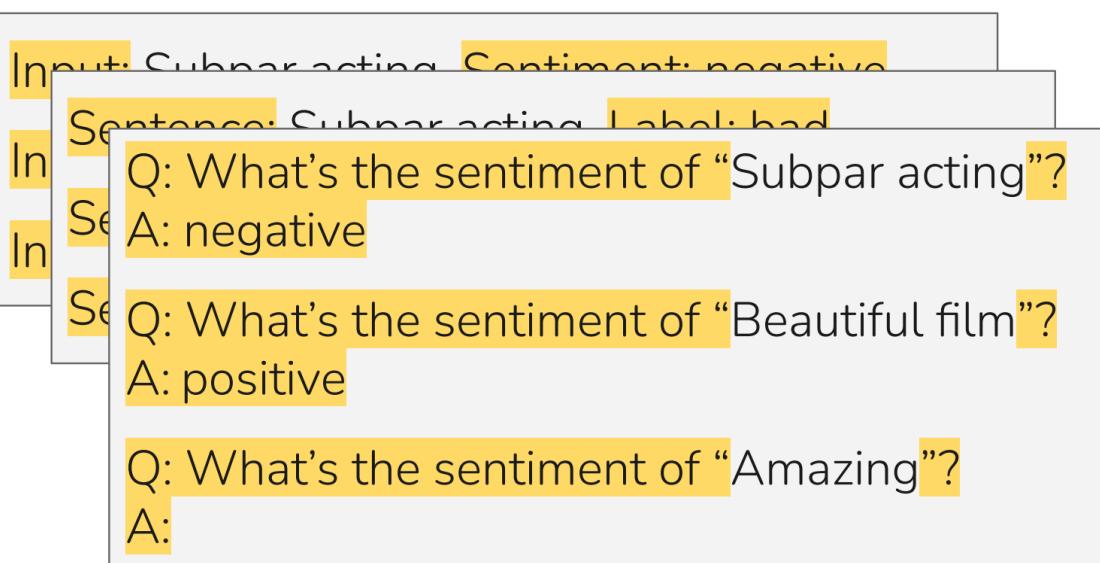
Sentence: Subpar acting. Label: bad

Sentence: Beautiful film. Label: good

Sentence: Amazing. Label:

LM Prompting: Choices of Encoding

Prompt



LM Prompting: Choices of Encoding

- **Pattern:** A function that encodes the inputs.
- **Verbalizer:** A function that encodes the output.

Pattern: $f(<\text{x}>)$ = "Input: < x >"

Verbalizer: $v(<\text{x}>)$ = "Label: < x >"

Pattern: $f(<\text{x}>)$ = "Q: What is the sentiment of < x >"

Verbalizer: $v(<\text{x}>)$ = "A: < x >"

Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Amazing. Sentiment:

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

A: negative

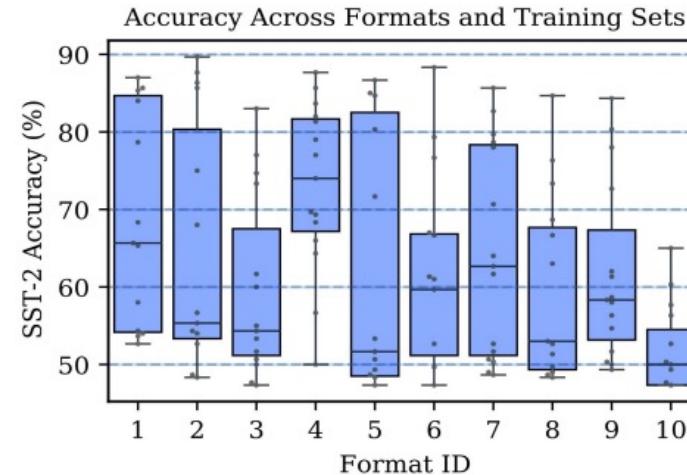
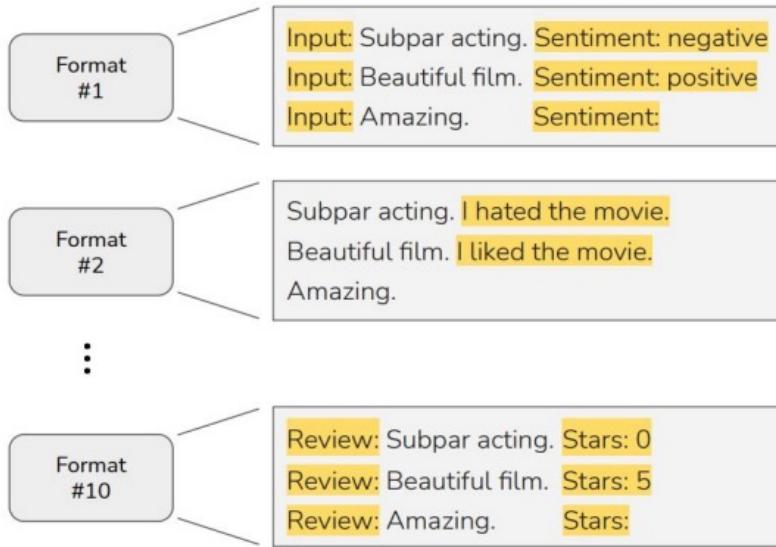
Q: What's the sentiment of "Beautiful film"?

A: positive

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

A:

In-Context Learning: Sensitivity to Encoding

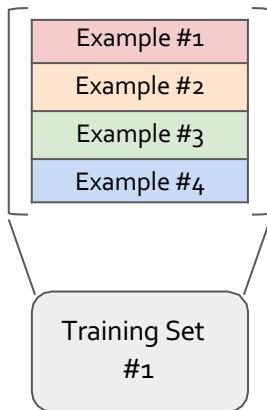


In-context learning is highly sensitive to prompt format (training sets and patterns/verbalizers)

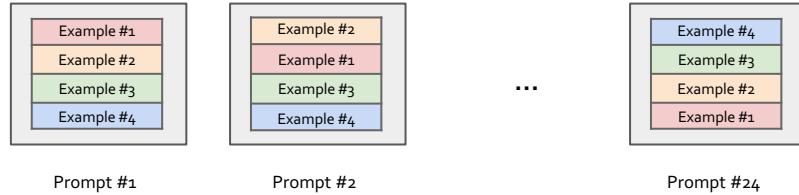
In-Context Learning: Sensitivity to Demo. Permutations

Training Set
#1

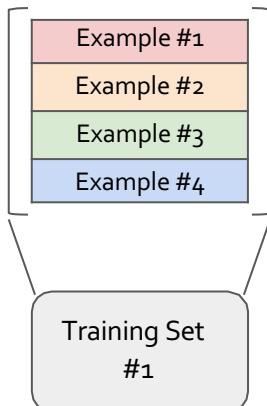
In-Context Learning: Sensitivity to Demo. Permutations



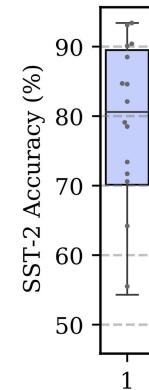
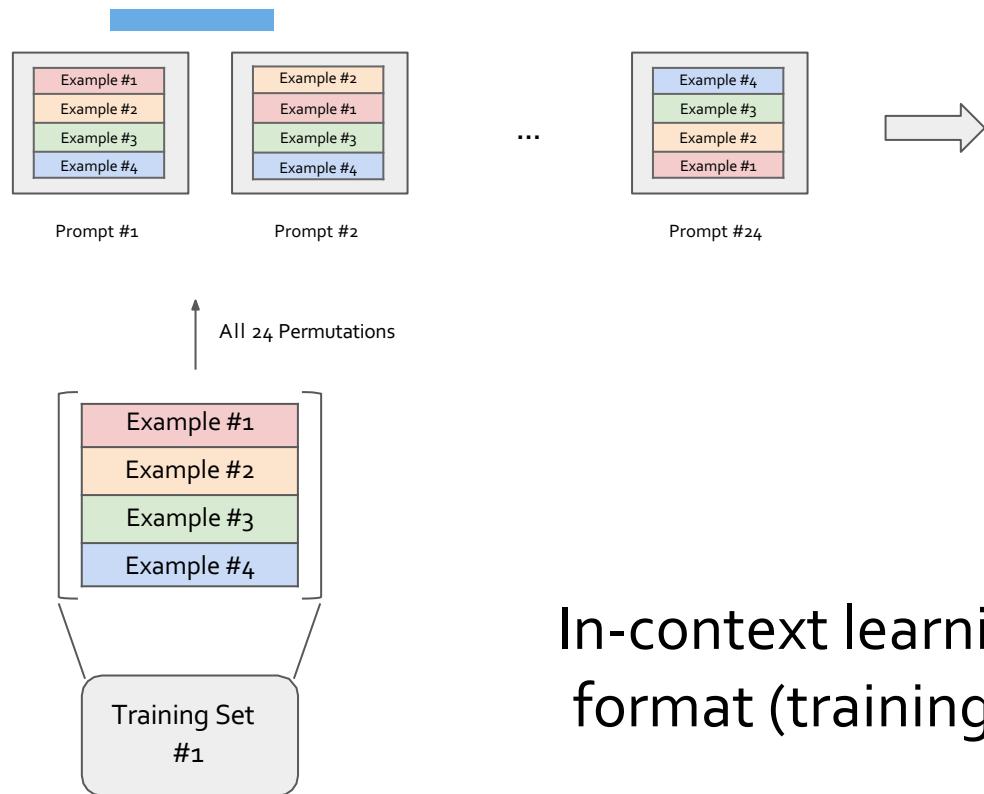
In-Context Learning: Sensitivity to Demo. Permutations



All 24 Permutations

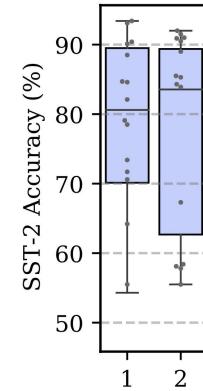
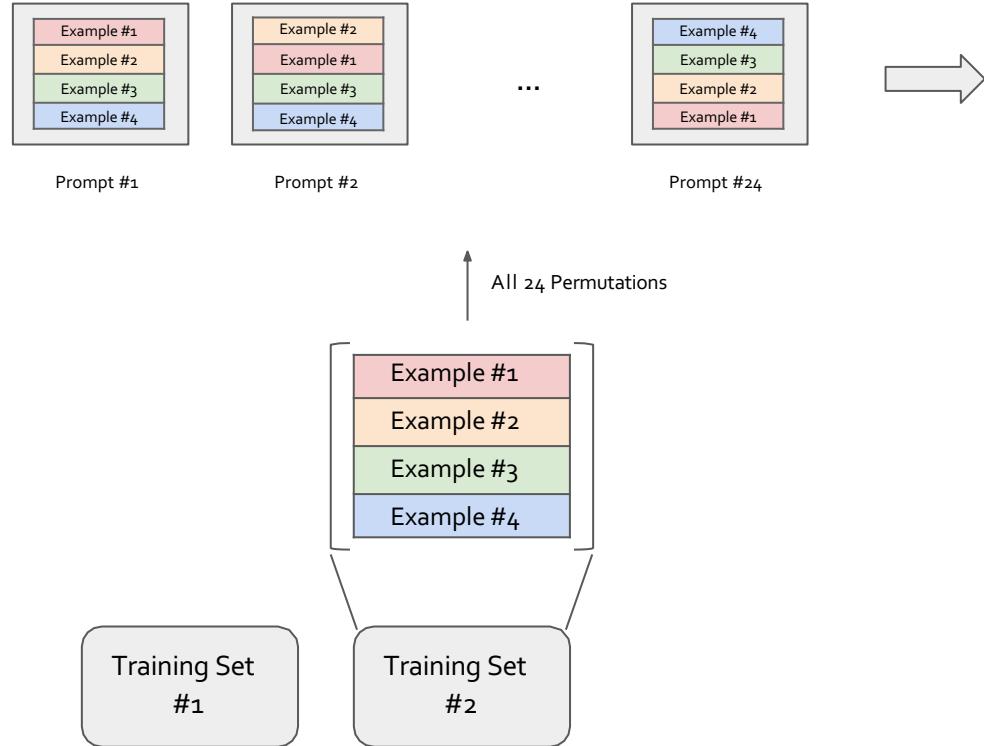


In-Context Learning: Sensitivity to Demo. Permutations

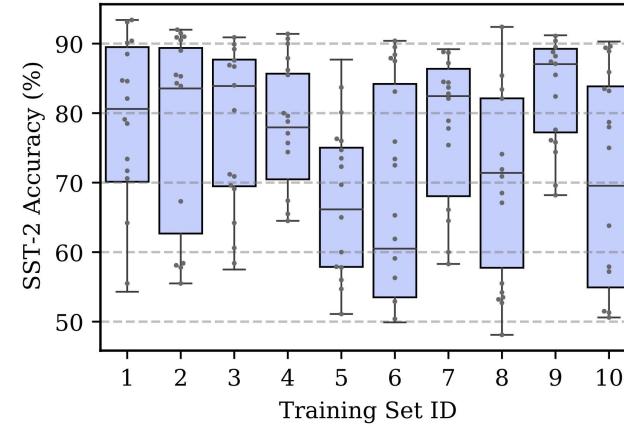
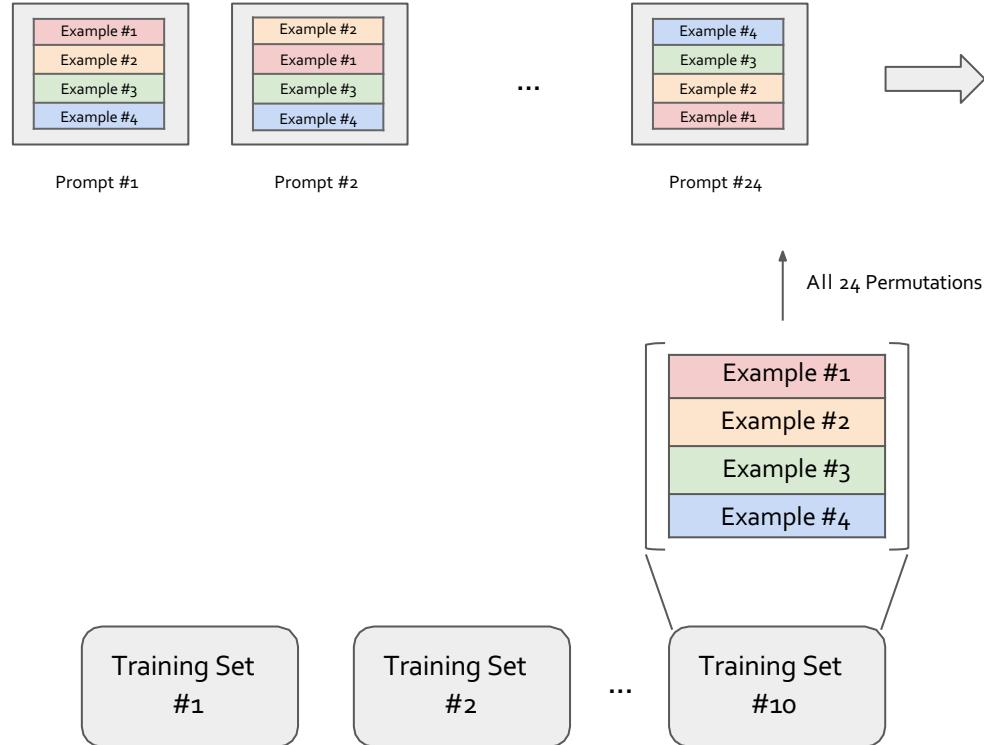


In-context learning is highly sensitive to prompt format (training sets and patterns/verbalizers)

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.

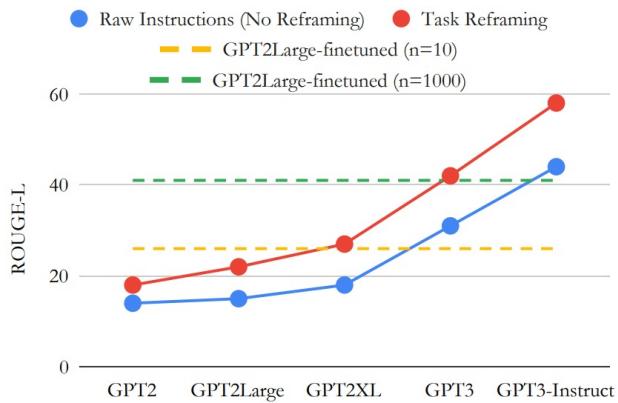


Figure 2: Across a variety of model sizes, **reframed prompts** consistently show considerable performance gain over **raw task instructions (no reframing)** in a few-shot learning setup. Since fine-tuning GPT3 is prohibitively expensive, we show the performance of fine-tuning smaller models (**horizontal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

- There are many possible ways to encode in-context examples of **a fixed task**
 - Many possible patterns/verbalizers
 - The choice of demonstrations
 - Ordering of the examples
 -
- It turns out there is a **huge variance** in performance depending on the encoding.
 - You can treat them as hyper-parameters
 - You should **not** choose these encodings based on the test data.
- Generally, it is better to use encoding that **makes the sequence closer to language modeling**—closer to what is observed during pretraining.

What Causes These Variances?

- Here we will provide several justifications ...

Majority Label Bias

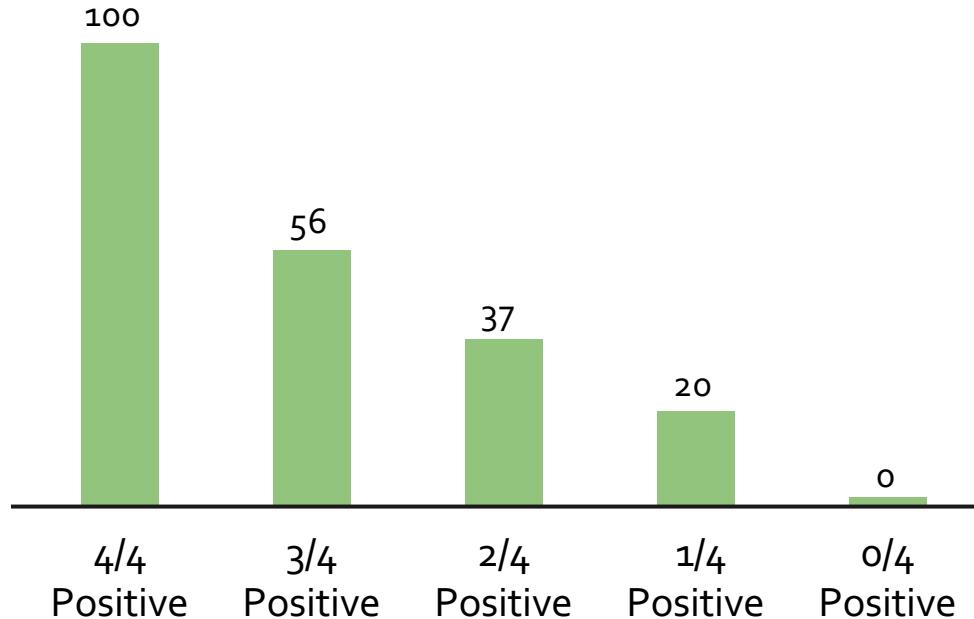
Majority Label Bias

Frequency of Positive Predictions

4/4 Positive	3/4 Positive	2/4 Positive	1/4 Positive	0/4 Positive
-----------------	-----------------	-----------------	-----------------	-----------------

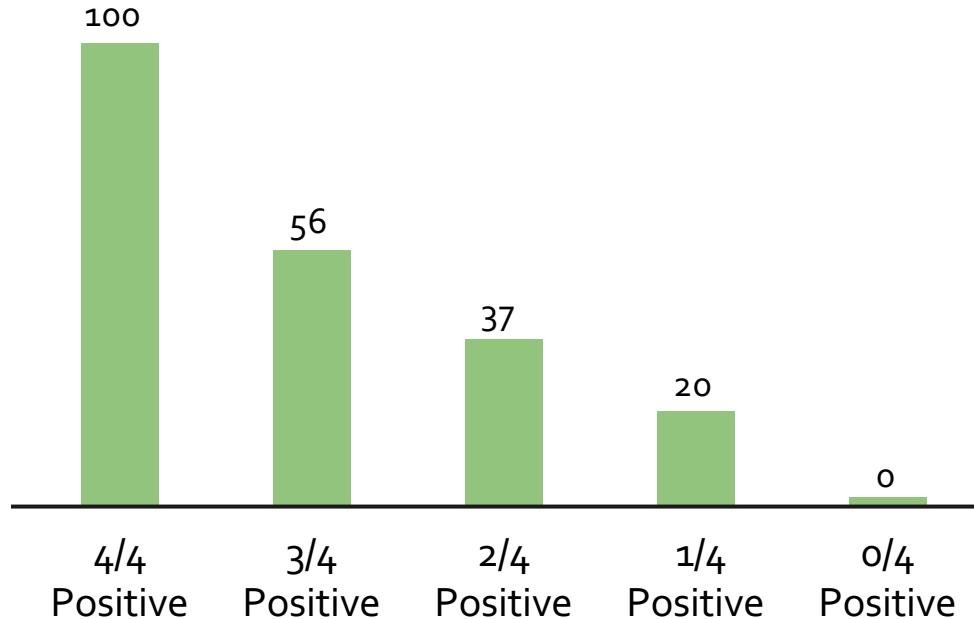
Majority Label Bias

Frequency of Positive Predictions



Majority Label Bias

Frequency of Positive Predictions



Majority label bias: frequent training answers dominate predictions
Explain some of the variances across example selections

Recency Bias

Recency Bias

Frequency of Positive Predictions

NPPP

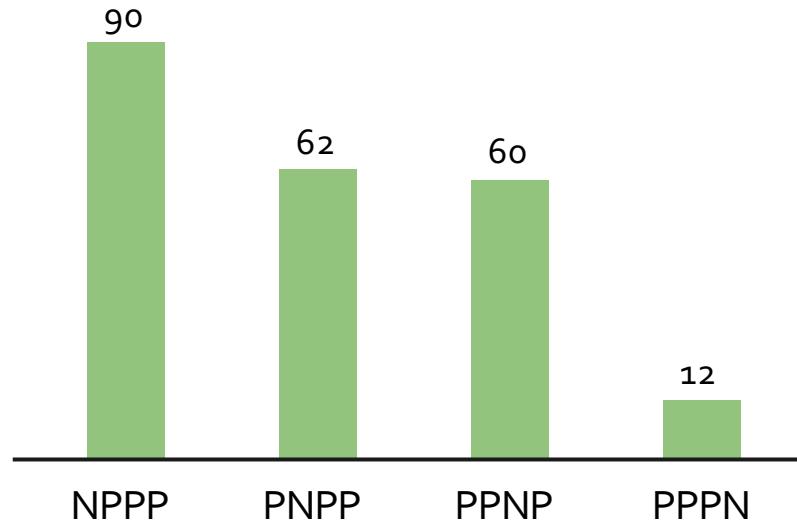
PNPP

PPNP

PPPN

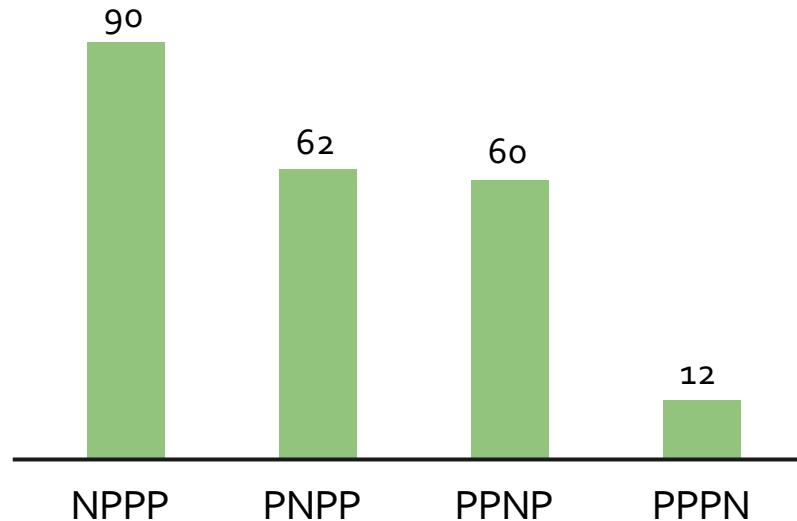
Recency Bias

Frequency of Positive Predictions



Recency Bias

Frequency of Positive Predictions



Recency bias: examples near end of prompt dominate predictions

- Explains variance across example permutations

Common Token Bias

Common Token Bias

Language Model



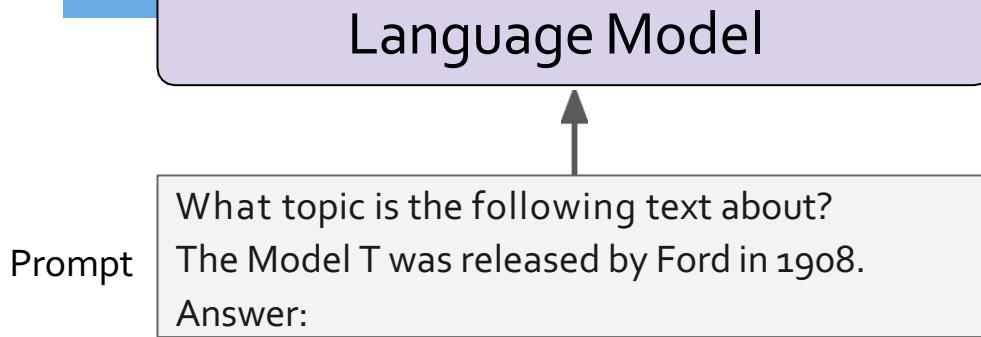
Prompt

What topic is the following text about?

The Model T was released by Ford in 1908.

Answer:

Common Token Bias

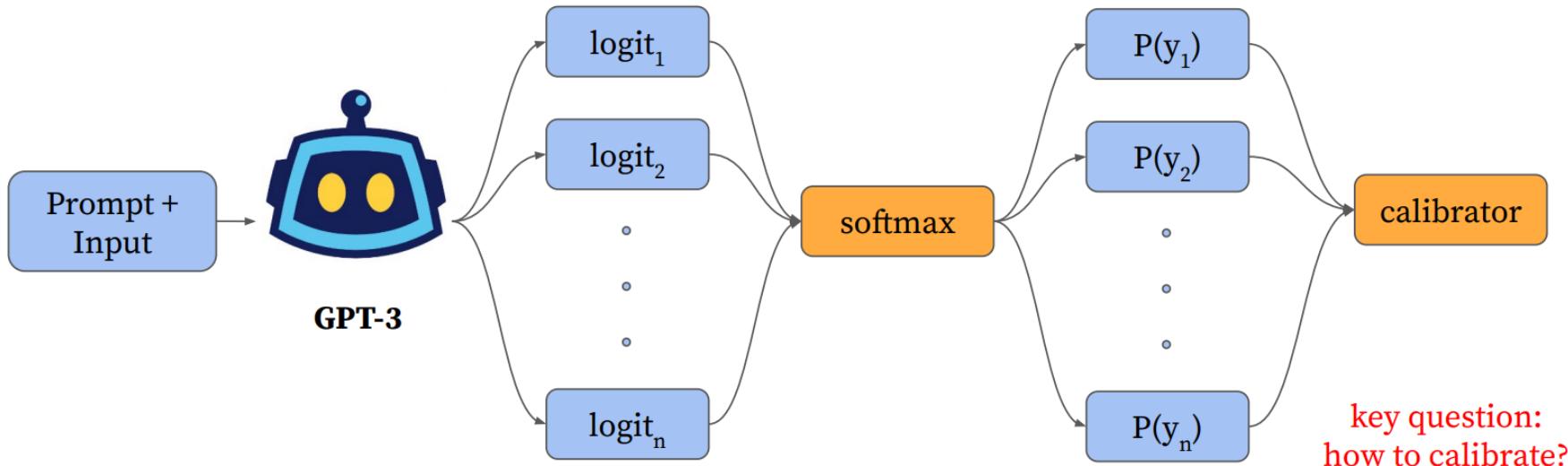


Model is biased towards predicting the incorrect frequent token "book" even when both "book" and "transportation" are equally likely labels in the dataset

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

Calibrating LM Probabilities



Calibrating LM Probabilities

Step 1: Estimate the bias

Calibrating LM Probabilities

Step 1: Estimate the bias

Insert “content-free” test input into prompt

Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Sentiment:

Calibrating LM Probabilities

Step 1: Estimate the bias

Insert “content-free” test input into prompt

Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Sentiment:

Get model’s prediction

<i>positive</i>	0.65
<i>negative</i>	0.35

Calibrating LM Probabilities

Step 1: Estimate the bias

Step 2: Counter the bias

Insert “content-free” test input into prompt

Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Sentiment:

Get model’s prediction

<i>positive</i>	0.65
<i>negative</i>	0.35

Calibrating LM Probabilities

Step 1: Estimate the bias

Insert “content-free” test input into prompt

Input: Subpar acting. Sentiment: negative

Input: Beautiful film. Sentiment: positive

Input: Sentiment:

Get model’s prediction

<i>positive</i>	0.65
<i>negative</i>	0.35

Step 2: Counter the bias (“Platt Scaling”)

“Calibrate” predictions with affine transformation

$$\hat{\mathbf{q}} = \text{softmax}(\mathbf{W}\hat{\mathbf{p}} + \mathbf{b})$$

Calibrating LM Probabilities

Step 1: Estimate the bias

Insert “content-free” test input into prompt

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

Get model’s prediction

<i>positive</i>	0.65
<i>negative</i>	0.35

Step 2: Counter the bias (“Platt Scaling”)

“Calibrate” predictions with affine transformation

$$\hat{q} = \text{softmax}(W\hat{p} + b)$$

The diagram illustrates the Platt Scaling equation. It shows the formula $\hat{q} = \text{softmax}(W\hat{p} + b)$. Two arrows point upwards from the terms in the equation to their respective components: one arrow from the softmax term to the label 'Calibrated probs', and another arrow from the linear term $W\hat{p} + b$ to the label 'Original probs'.

Calibrating LM Probabilities

Step 1: Estimate the bias

Insert “content-free” test input into prompt

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

Get model’s prediction

<i>positive</i>	0.65
<i>negative</i>	0.35

Step 2: Counter the bias (“Platt Scaling”)

“Calibrate” predictions with affine transformation

$$\hat{\mathbf{q}} = \text{softmax}(\mathbf{W}\hat{\mathbf{p}} + \mathbf{b})$$

Calibrated probs Original probs

Fit \mathbf{W} and \mathbf{b} so that outputs for the content-free input are **uniform**

Calibrating LM Probabilities

Step 1: Estimate the bias

Note, this calibration does **not** require any labeled data.

- **Classification tasks:** normalized scores of labels
- **Generation tasks:** probabilities of the first token of the generation over the entire vocabulary

Get model's prediction

<i>positive</i>	0.65
<i>negative</i>	0.35

Step 2: Counter the bias ("Platt Scaling")

"Calibrate" predictions with affine transformation

$$\hat{\mathbf{q}} = \text{softmax}(\mathbf{W}\hat{\mathbf{p}} + \mathbf{b})$$

↑
Calibrated probs Original probs

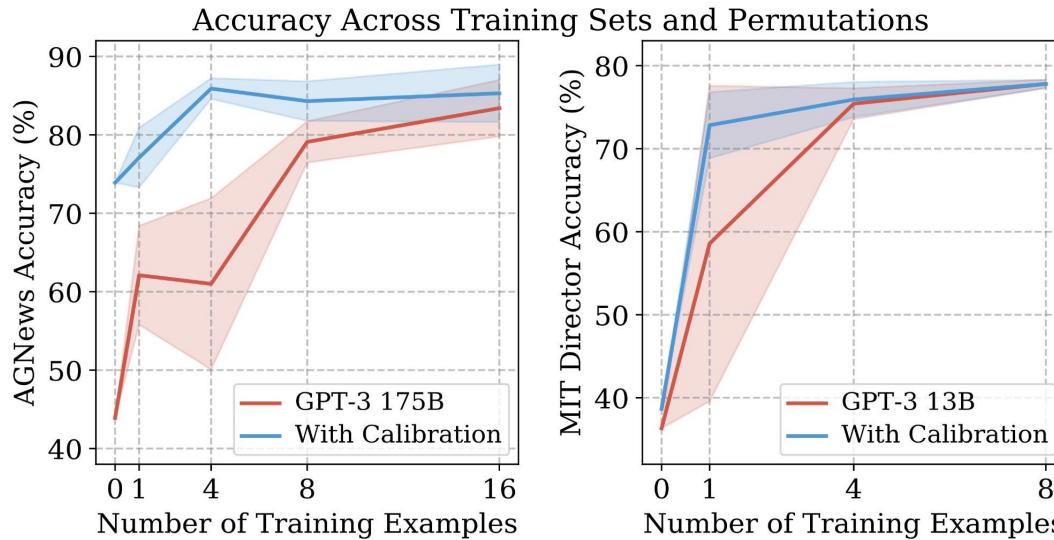
Fit \mathbf{W} and \mathbf{b} so that outputs for the content-free input are **uniform**

$$\mathbf{W} = \begin{pmatrix} \frac{1}{0.65} & 0 \\ 0 & \frac{1}{0.35} \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

W: diagonal matrix
b: bias set to zeros

Effect of Calibration

- Improves mean and worst-case accuracy
- Reduces variance across training sets and permutations



Surface Form Competition

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

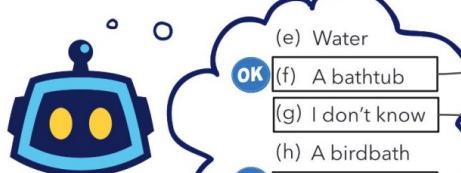
Humans select options



- (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

- LM prompting & In-context learning show promising results, but their performance is highly unstable/brittle.
- Better scoring: Calibration
- Other factors:
 - Better **formation** of demonstrations
 - Better **choice** of demonstrative examples
 - Better **ordering** of demonstrative examples

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



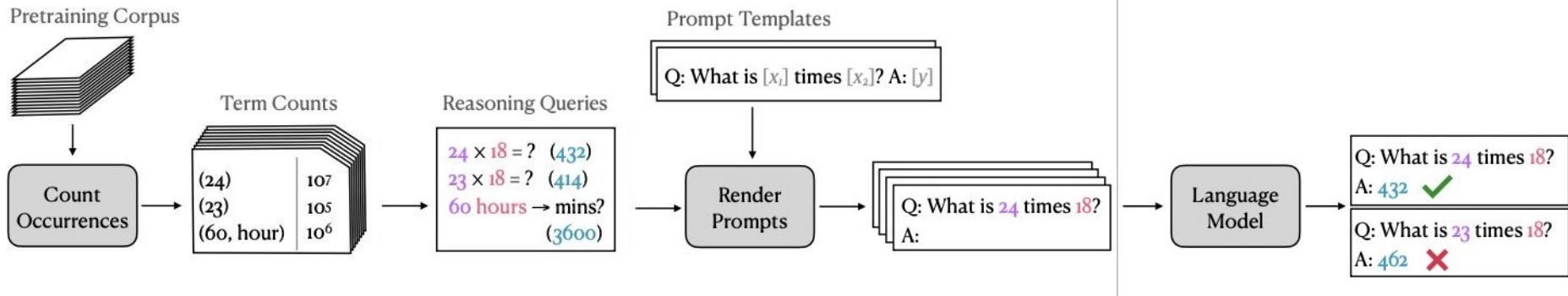
Language Model

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

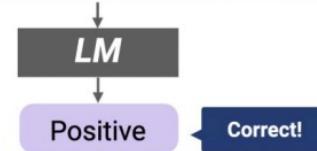
Impact of Input-Output Mapping

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

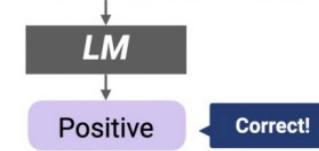
Circulation revenue has increased by 5% in Finland. \n Positive
Panostaja did not disclose the purchase price. \n Neutral
Paying off the national debt will be extremely painful. \n Negative
The company anticipated its operating profit to improve. \n _____



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 _____



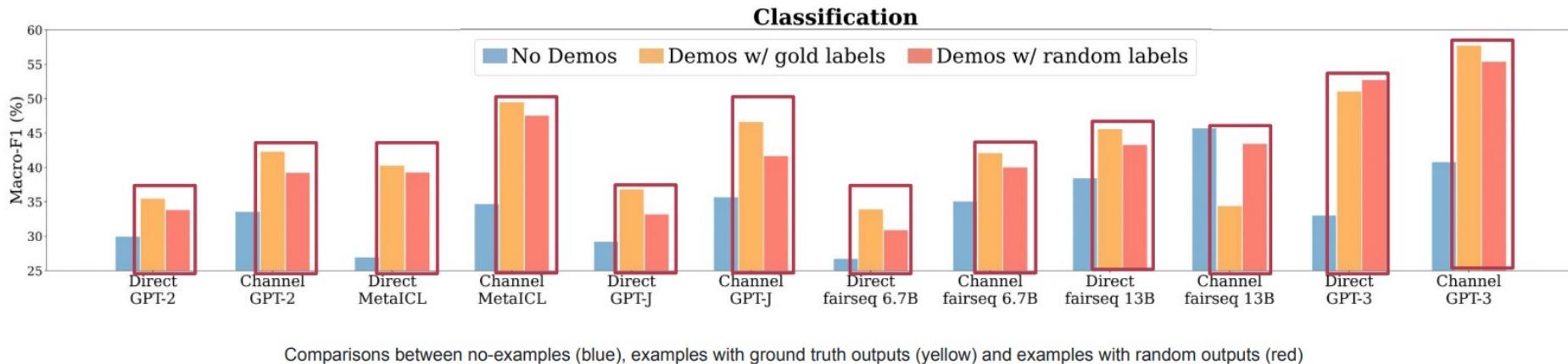
Prompt with true labels



Prompt with random labels

Impact of Input-Output Mapping

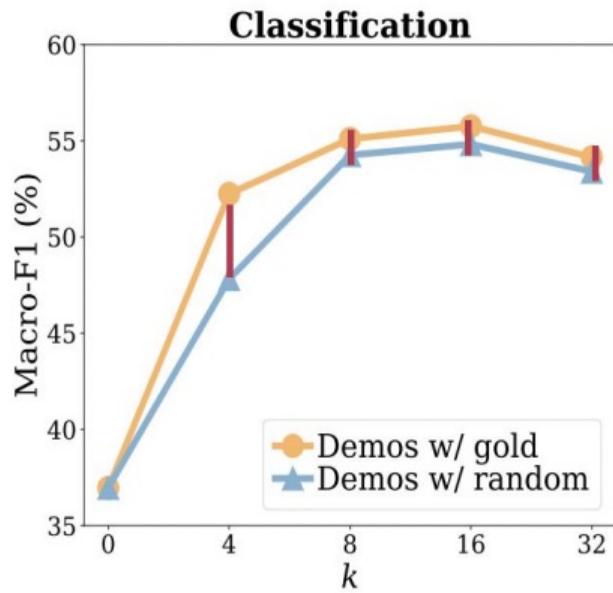
- Models see a small performance drop (0–5% absolute) with random labels



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

Impact of Input-Output Matching

- 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

CSCI 601 471/671
NLP: Self-Supervised Models

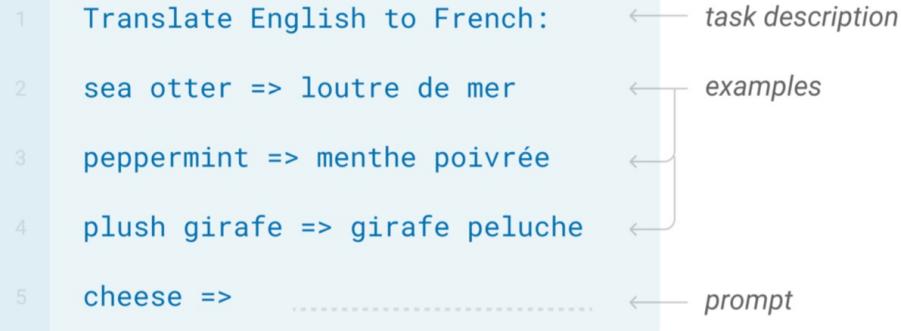
<https://self-supervised.cs.jhu.edu/sp2023/>



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

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?



How/Why does In-context Learning Work?

Any arbitrary task



Language Model

A few-shot learner

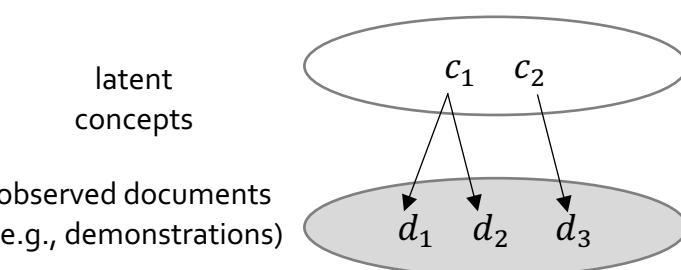


In-context Learning as Bayesian Inference

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

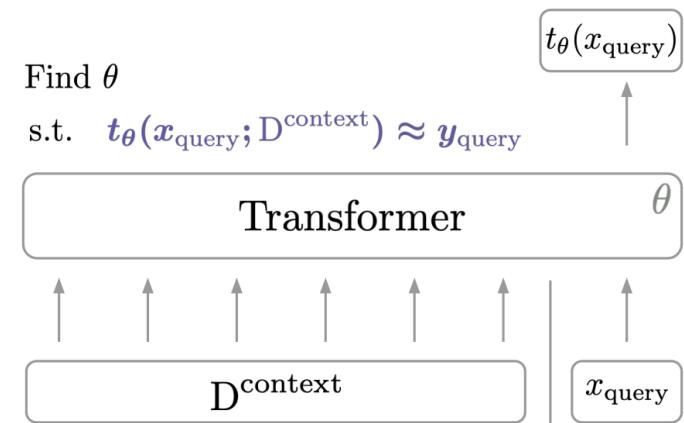
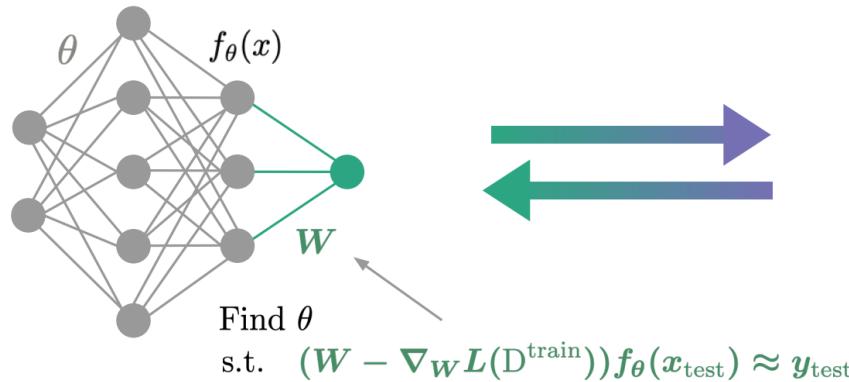
Idea:

- (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 for Multi-Step Reasoning

- 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)**.

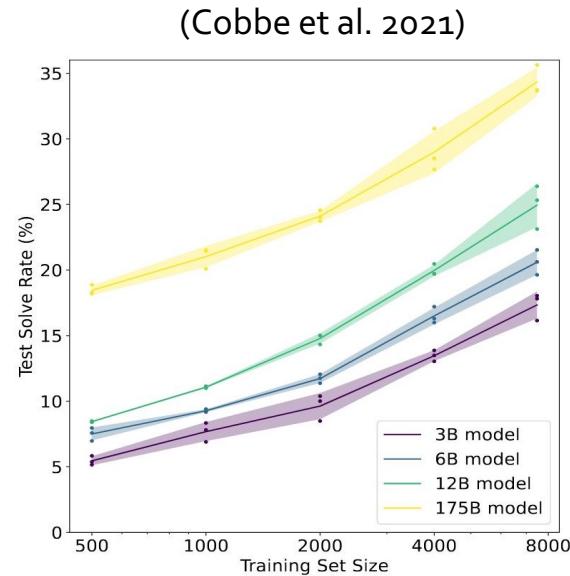
Arithmetic Reasoning (AR)
 $(+ - \times \div \dots)$

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

(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

(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

Two-stage Prompting
Step-by-step Answer

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

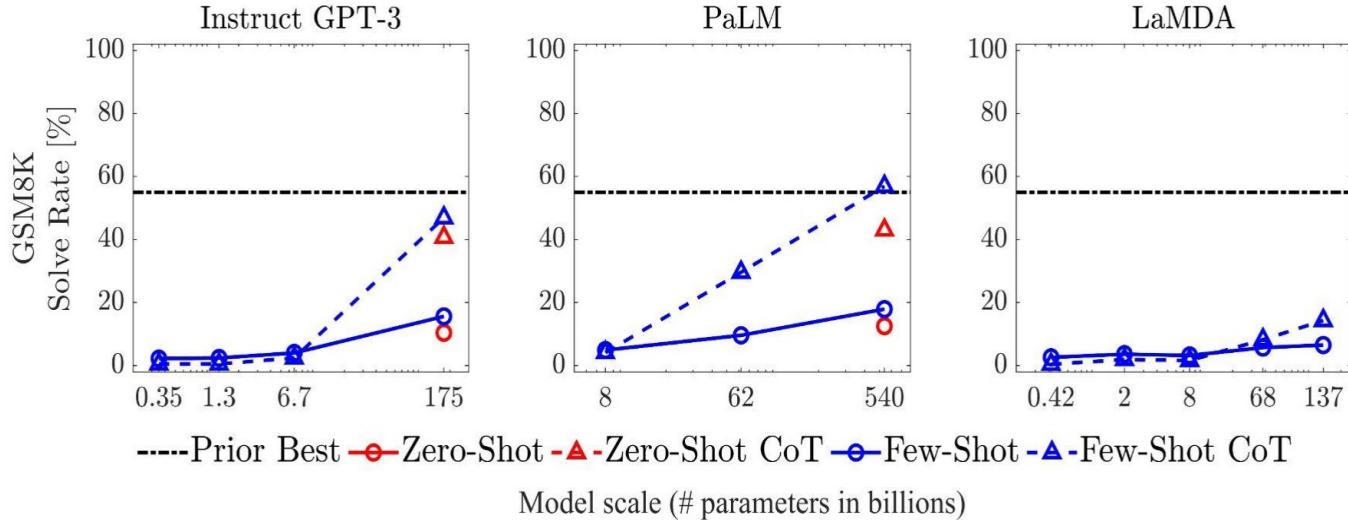
Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

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. ✓

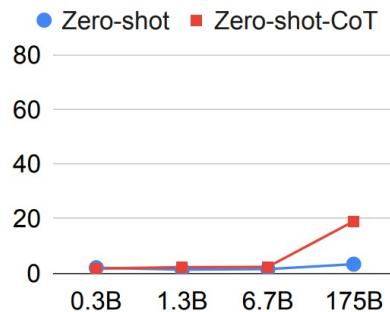
Multi-Step Prompting: Empirical Results

- **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**

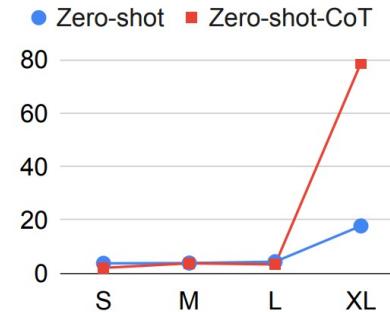


Multi-Step Prompting: Empirical Results

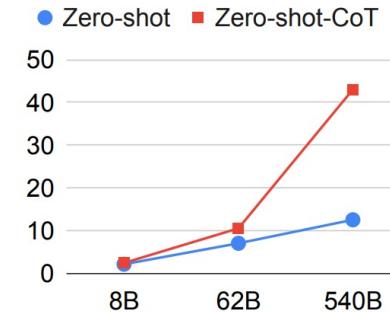
- **Setup:** show **demonstrations** that contain the **decompositions**
- The gains of multi-step prompting increases with scale.
- Prompting achieves **better perf than** [smaller] models that are fine-tuned on a lot more data.



(a) MultiArith on Original GPT-3



(b) MultiArith on Instruct GPT-3

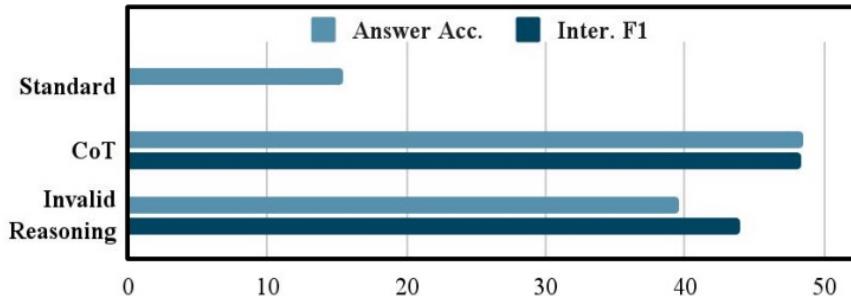


(c) GMS8K on PaLM

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. X
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. ✓

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

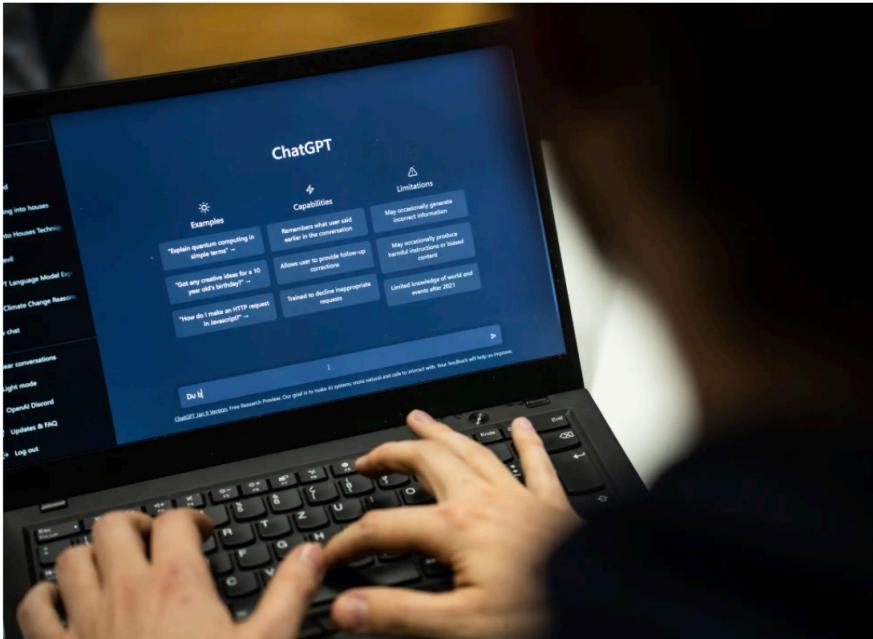
- Prompting language models is a powerful way to adapt them to our desired tasks.
- They also serve as a gateway to understand the underlying dynamics inside models.
- Lots of activity in this area and room for a lot of research progress.

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