



CMPT 413/713: Natural Language Processing

Adapting LLMs for tasks

Prompting and fine-tuning LLMs

Spring 2026
2026-02-09

Slides adapted from Anoop Sarkar

Using LLMs for tasks

- So your language model can complete a sentence, but you may want to do different things
 - Classify whether a email is SPAM or NOT SPAM
 - Answer a question: when was Albert Einstein born?
 - Extract information from text
- If I give it a piece of text, how do I tell it whether I want to translate it French, summarize it, or make it into a poem?

Using LLMs for tasks

Develop specialized model for your task (with LM as part)

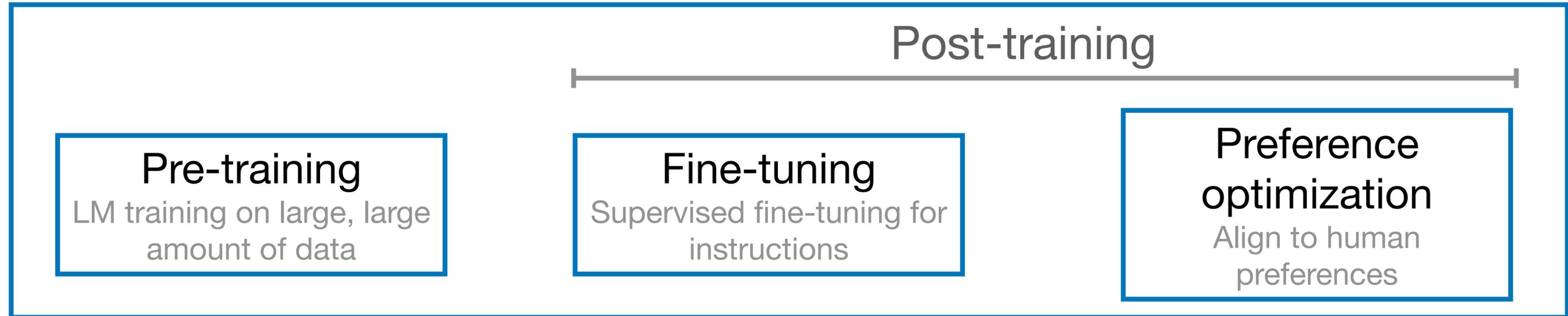
- Hookup appropriate inputs/outputs
- Fine-tuning parameters (include some LM parameters) for task

Try to use the LM network as it is (no extra network training)

- Zero-shot / few-shot prompting (in-context learning)

Try to have smaller LM to allow running on various devices

- Model distillation and pruning



Model compression
Reduce size of model for efficient deployment

Task specific fine-tuning
Custom domains, improved performance on specialized tasks

Prompting
Generate responses

Use to build LLM agents

Recipe for using LLMs

- Start with prompting
- Retrieval augmented prompting
- Use fine-tuning
 - Fine-tuning using APIs
 - Fine-tuning over open-source LLMs
 - Use PEFT
- Pre-train your own model - not recommended unless you are one of the big companies or a startup that have a lot of funding/resources

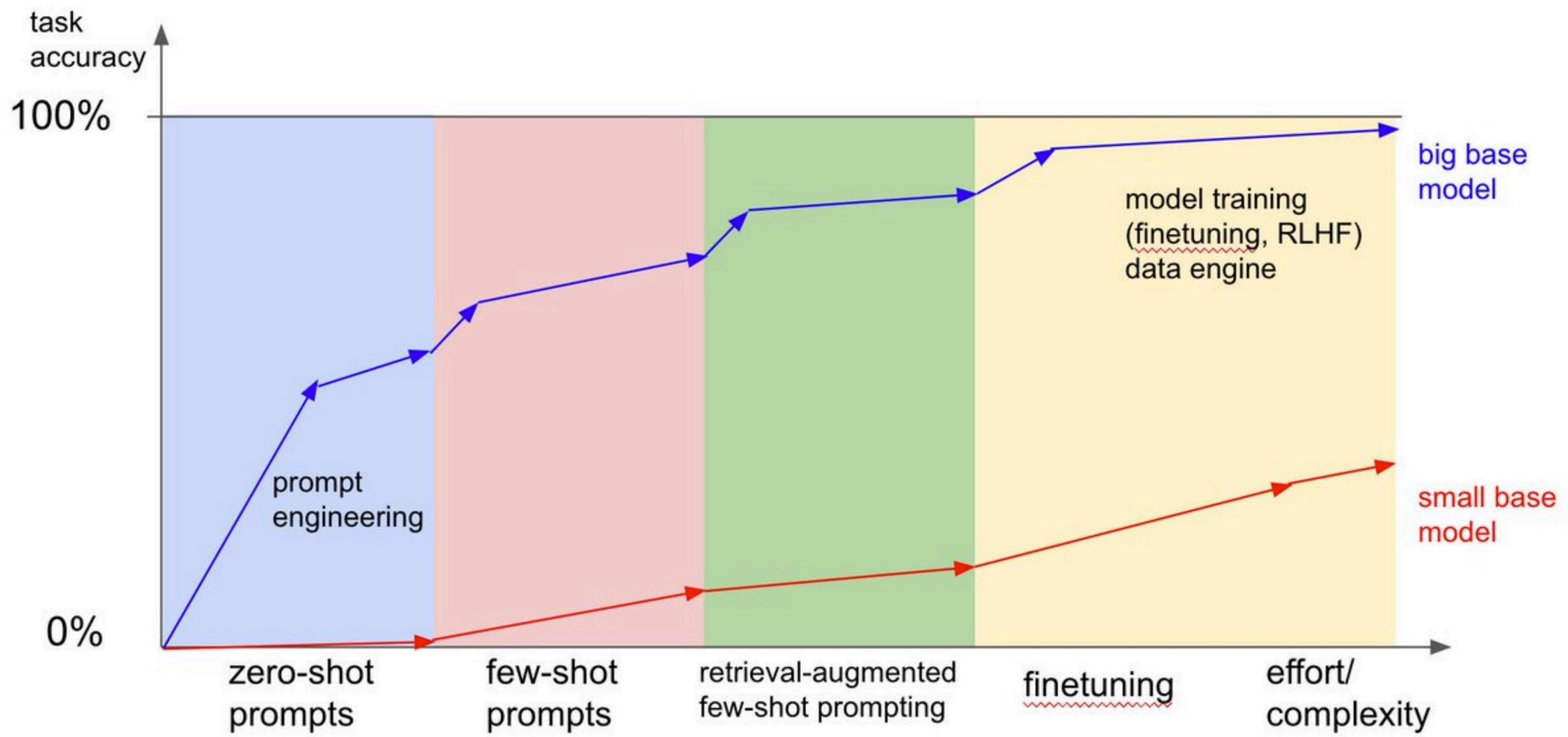


Figure credit: Andrej Karpathy

Few-shot learning and prompting

Language Models are Few-Shot Learners

Tom B. Brown*

Benjamin Mann*

Nick Ryder*

Melanie Subbiah*

Jared Kaplan[†]

Prafulla Dhariwal

Arvind Neelakantan

Pranav Shyam

Girish Sastry

Amanda Askell

Sandhini Agarwal

Ariel Herbert-Voss

Gretchen Krueger

Tom Henighan

Rewon Child

Aditya Ramesh

Daniel M. Ziegler

Jeffrey Wu

Clemens Winter

Christopher Hesse

Mark Chen

Eric Sigler

Mateusz Litwin

Scott Gray

Benjamin Chess

Jack Clark

Christopher Berner

Sam McCandlish

Alec Radford

Ilya Sutskever

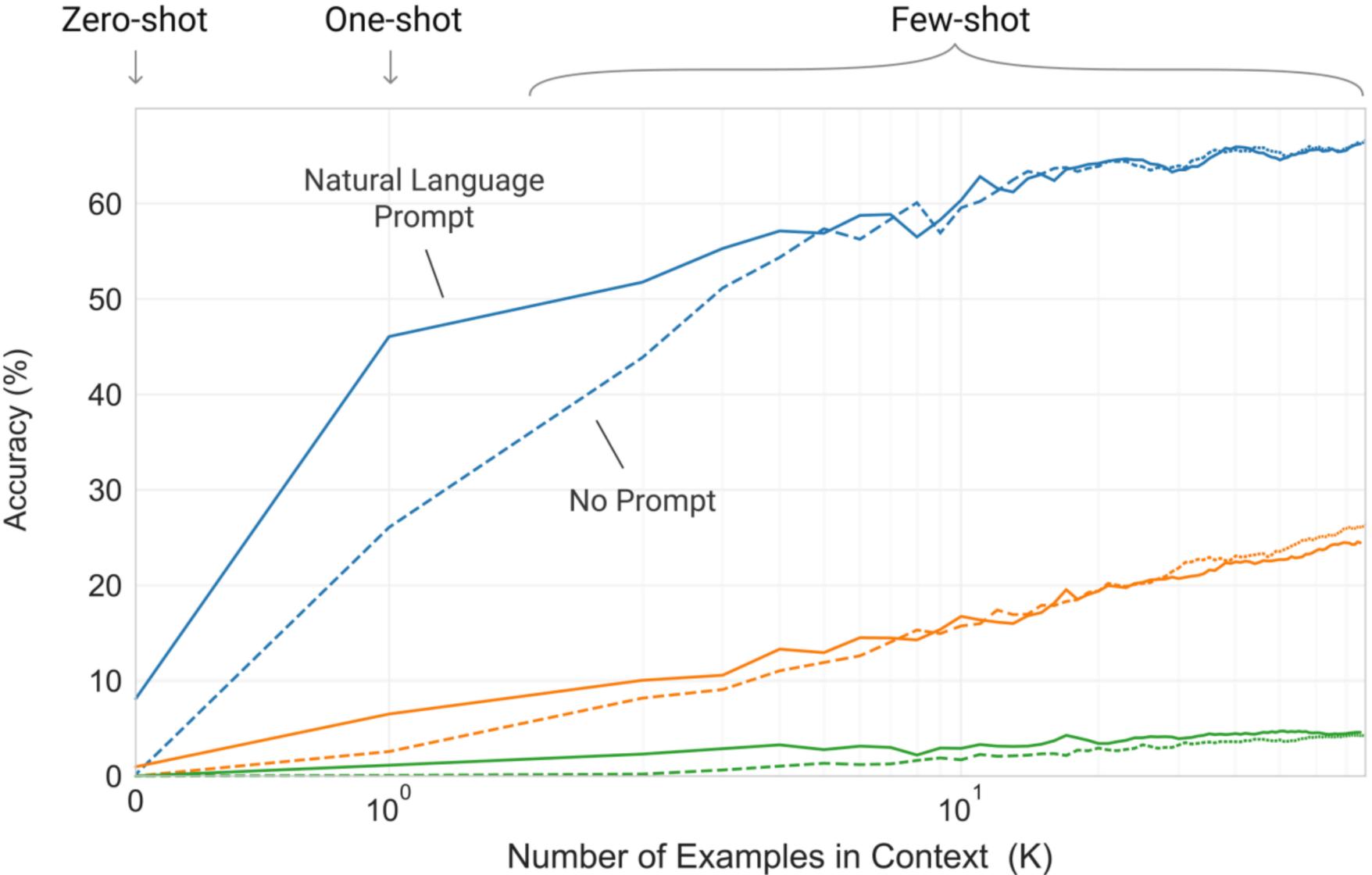
Dario Amodei

GPT-3: Few-shot learning

0 training examples

1 training examples

A few examples are provided at test time



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 ←
4 plush girafe => girafe peluche ←
5 cheese => ..... ← prompt
    
```

175B Params

13B Params

1.3B Params

Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



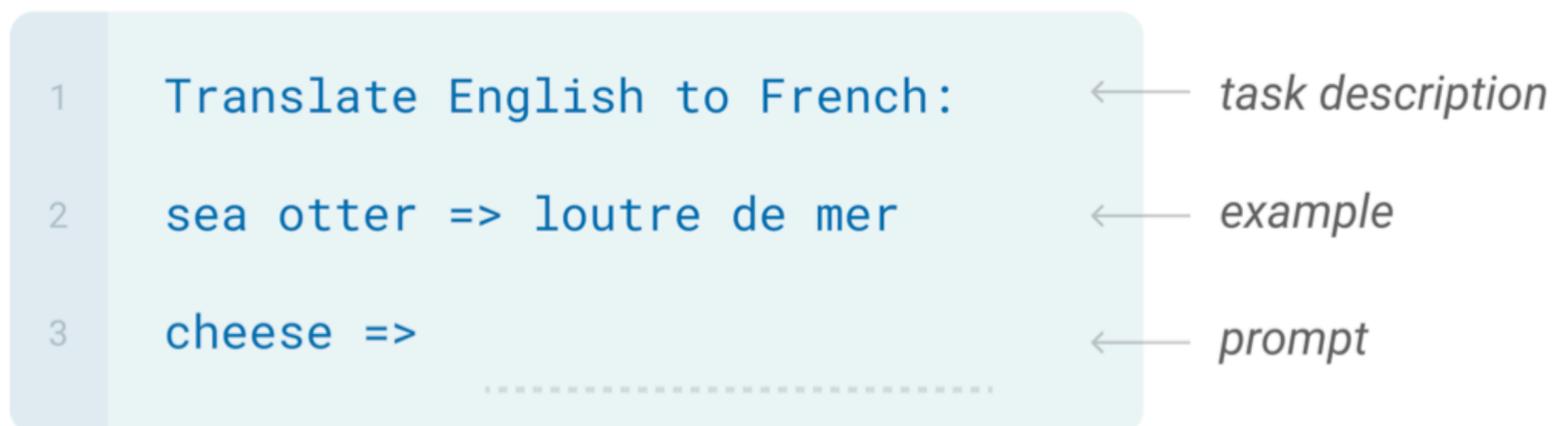
Zero-shot

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



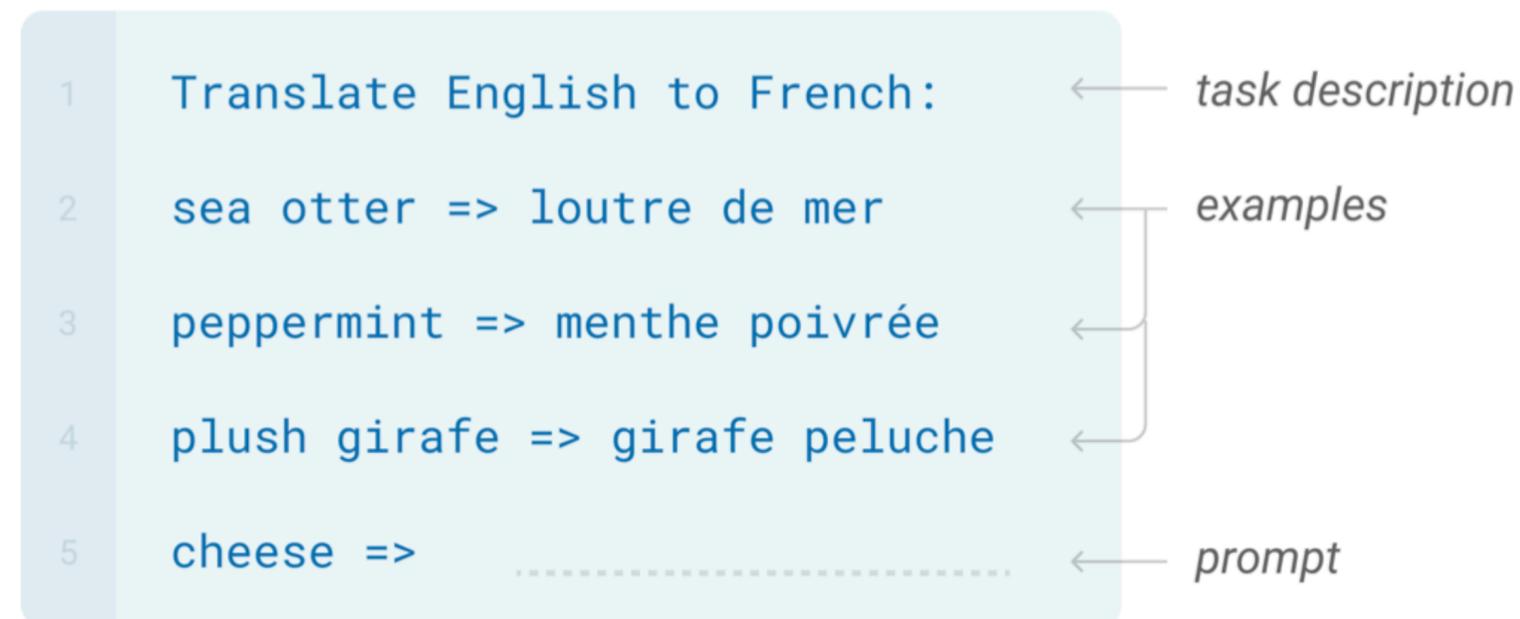
One-shot

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



Few-shot

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



Fine-tuning fails at scale

- LLMs $>10\text{B}$ parameters are very difficult to fine-tune and requires a big compute budget
- So in-context learning using a long prompt or prefix is needed to coax the answer from a "predict the next token" approach to solving multiple tasks
- Pre-training on web-scale text can observe many different tasks in-context during training in the inner loop (per batch)
- Gradient descent improves the model representations based on next token prediction over many batch updates in the outer loop

outer loop

Learning via SGD during unsupervised pre-training

inner loop

| | |
|---|------------|
| 1 | 5 + 8 = 13 |
| 2 | 7 + 2 = 9 |
| 3 | 1 + 0 = 1 |
| 4 | 3 + 4 = 7 |
| 5 | 5 + 9 = 14 |
| 6 | 9 + 8 = 17 |

↑
sequence #1

In-context learning

| | |
|---|----------------|
| 1 | gaot => goat |
| 2 | sakne => snake |
| 3 | brid => bird |
| 4 | fsih => fish |
| 5 | dcuk => duck |
| 6 | cmihp => chimp |

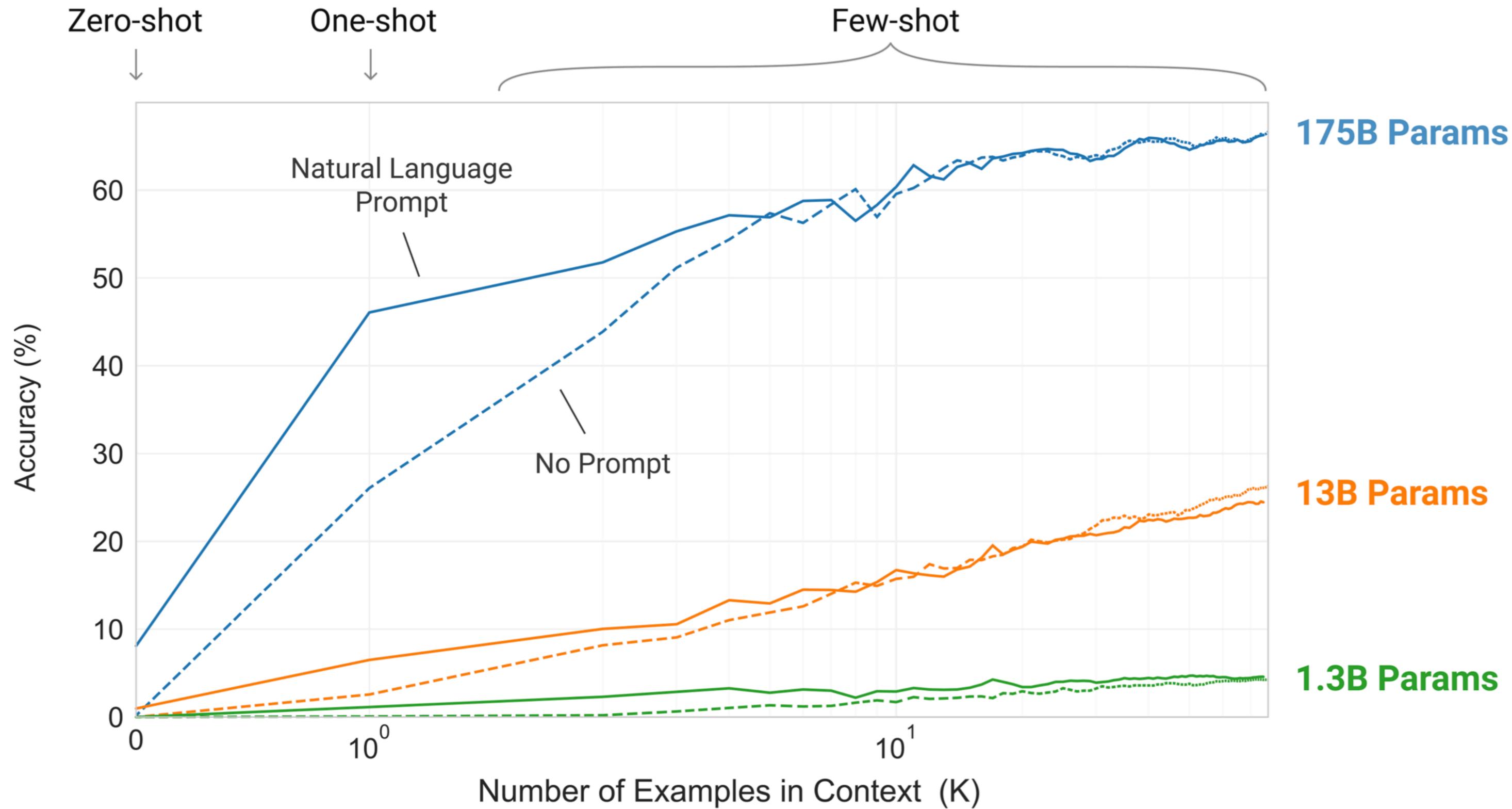
↑
sequence #2

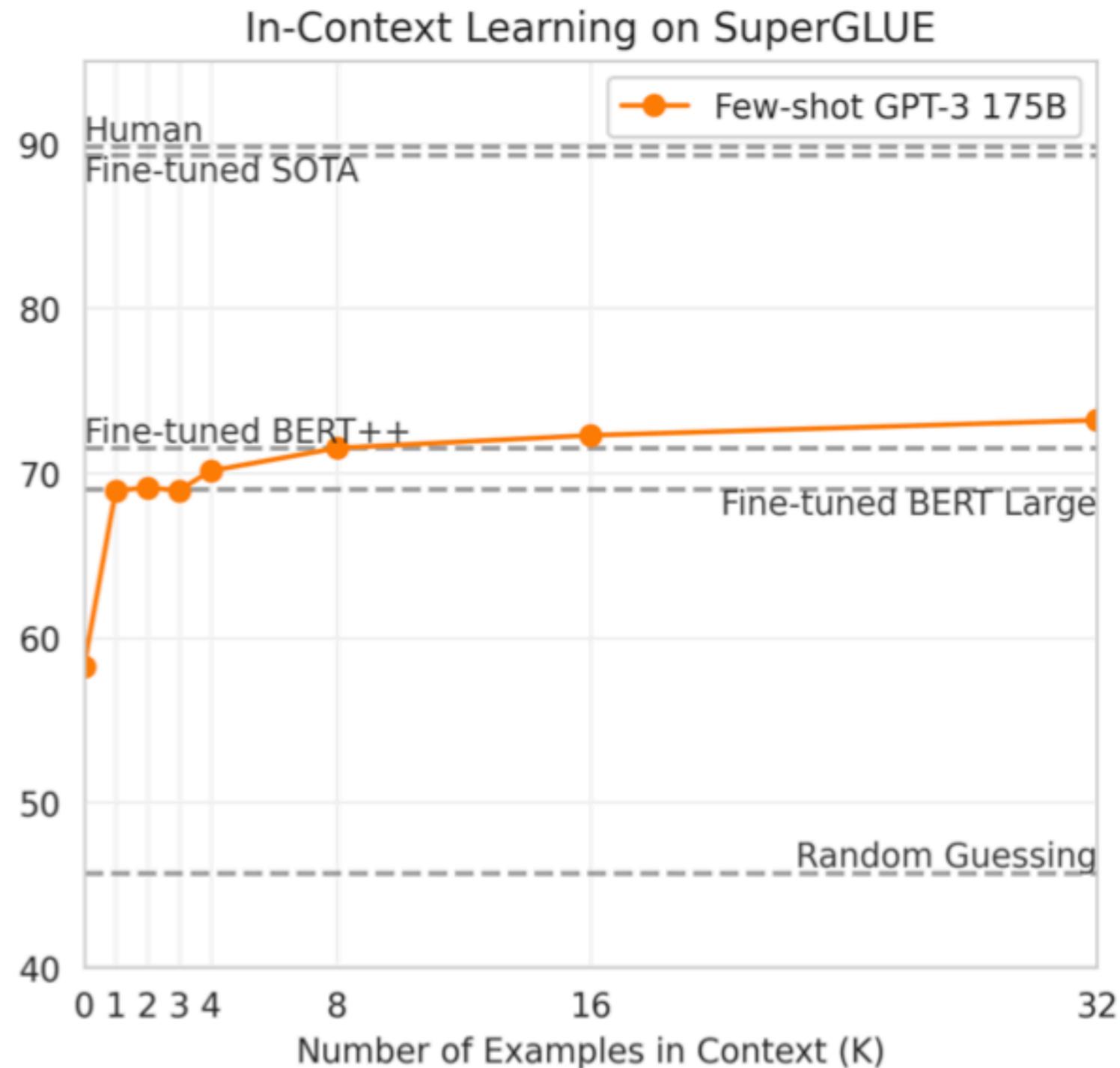
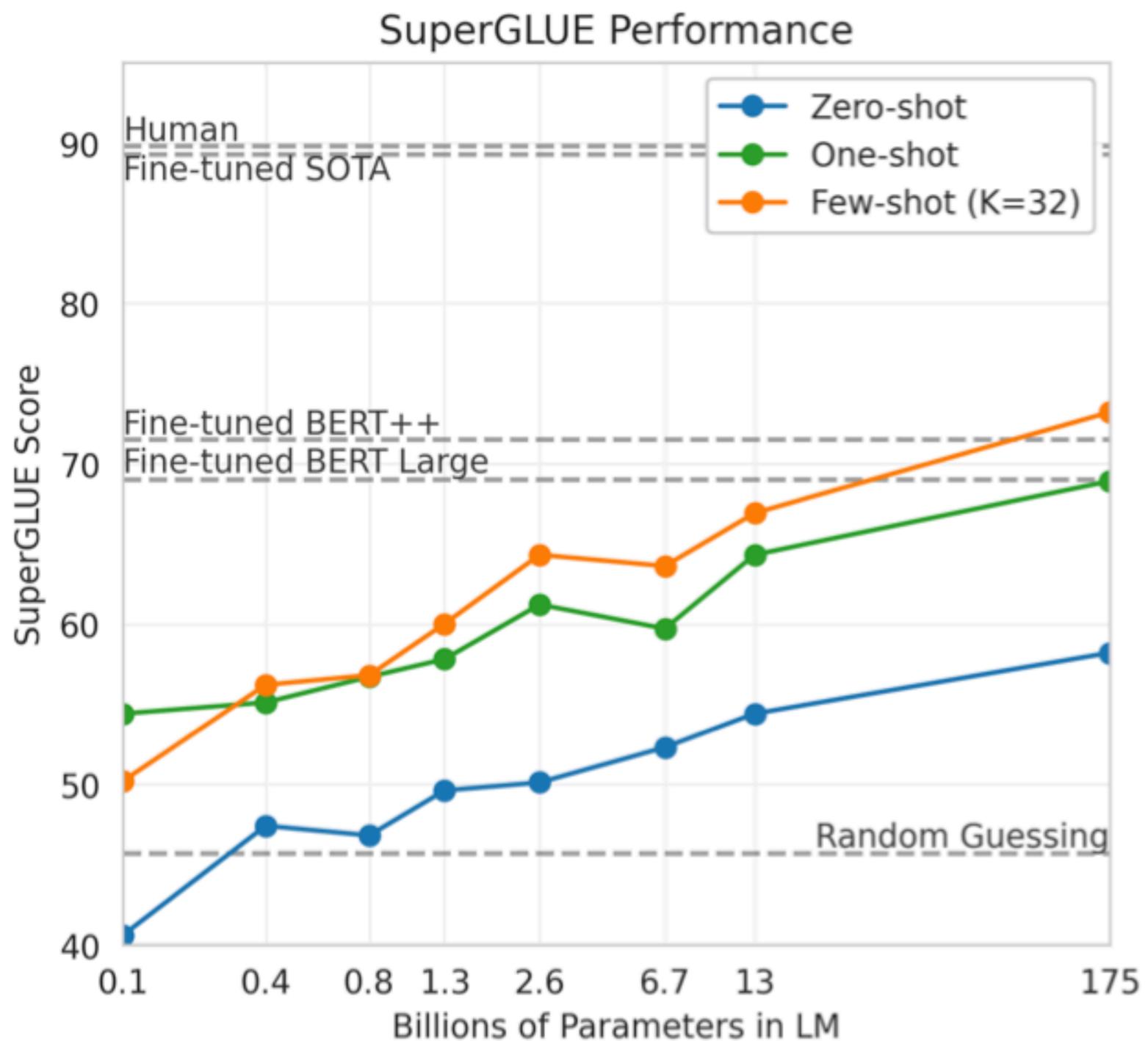
In-context learning

| | |
|---|------------------|
| 1 | thanks => merci |
| 2 | hello => bonjour |
| 3 | mint => menthe |
| 4 | wall => mur |
| 5 | otter => loutre |
| 6 | bread => pain |

↑
sequence #3

In-context learning





Performance on SuperGLUE increases with number of examples in context. We find the difference in performance between the BERT-Large and BERT++ to be roughly equivalent to the difference between GPT-3 with one example per context versus eight examples per context.

Prompting

- LLM responses depend on prompt provided
- Many different types of prompt designs
 - Direct prompting
 - Prompting with examples: few-shot
 - What examples will be most effective?
 - Prompting for reasoning: chain-of-thought
 - Prompting to check consistency and correctness: reflection
 - Prompting with retrieval: RAG

<https://www.promptingguide.ai/>

Basic prompting

- Provide textual string as input and ask the LM to complete it

Example input

When a dog sees a squirrel, it will usually

GPT-2 small

be afraid of anything unusual. As an exception, that's when a squirrel is usually afraid to bite.

GPT-2 XL

lick the squirrel. It will also touch its nose to the squirrel on the tail and nose if it can

Prompt templates and roles

Models are now often trained as chatbots

Example template from Salesforce

```
You're a copywriter at
<User.CompanyName__Merge_Field> and you
need to
create an account description for
<account.Name__Merge_Field>.
```

When I ask you to create the account description, you must strictly follow my instructions below.

Instructions:

```
"""
```

```
Write a description no longer than 500
characters.
```

```
"""
```

Now create the account description.

Example with roles (OpenAI messages format)

```
messages=[
  { "role": "system",
    "content": "Please classify movie
reviews as 'positive' or 'negative'." },
  { "role": "user",
    "content": "This movie is a banger." },
]
```

Roles:

“**system**”: message provided to the system to influence behavior

“**user**”: message input by the user

“**assistant**”: message output by the system

Chain of thought prompting

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models [Wei et al, 2022]

- How you prompt matters.
- For more complex problems, may need to provide prompts that **illustrate the reasoning** you expect

Standard Prompting

Model Input

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: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

Chain-of-Thought Prompting

Model Input

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.

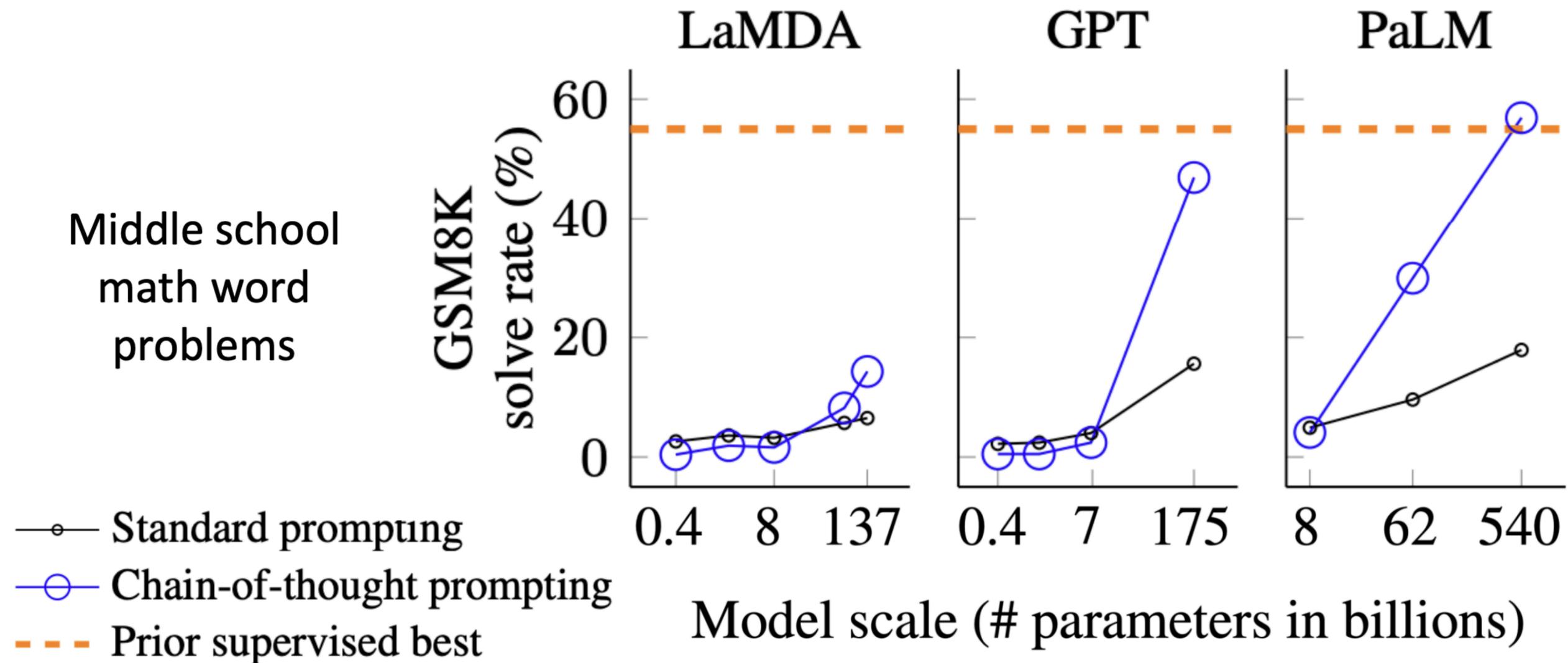
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅

Chain of thought prompting

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models [Wei et al, 2022]

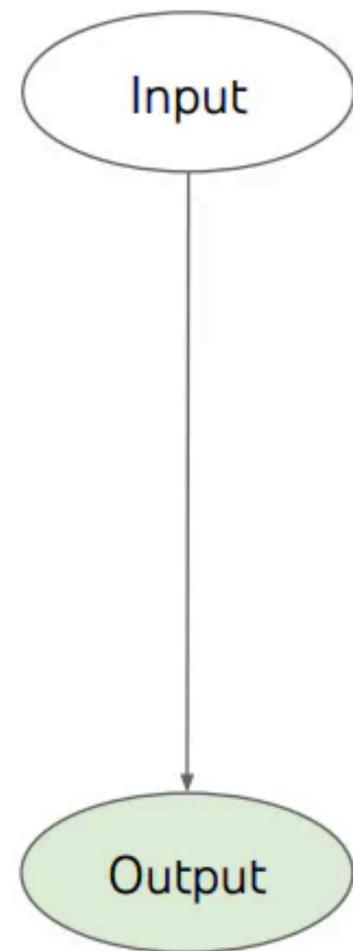


Tree of thoughts

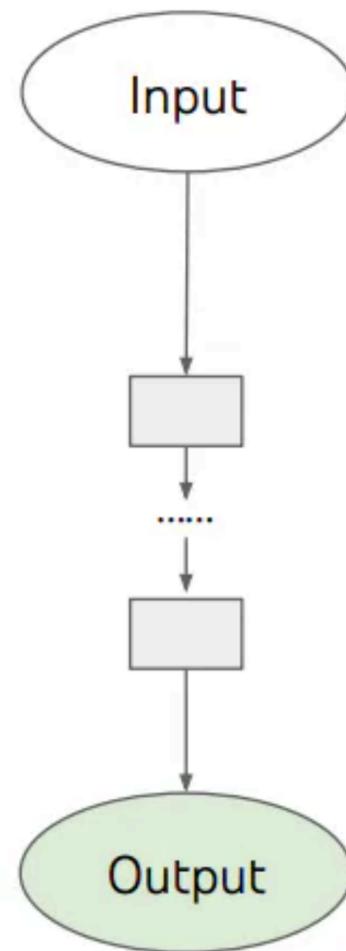
Tree of Thoughts: Deliberate Problem Solving with Large Language Models [Yao et al, 2023]

Large Language Model Guided Tree-of-Thought [Long 2023]

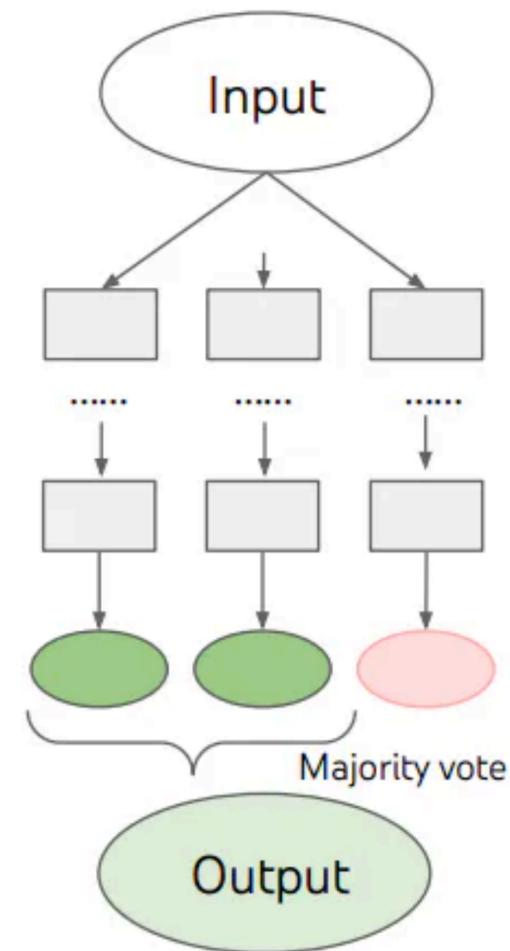
- Decision making by considering multiple paths of reasoning
- Consider different “thoughts” expressed in language
- Use to solve different types of problems



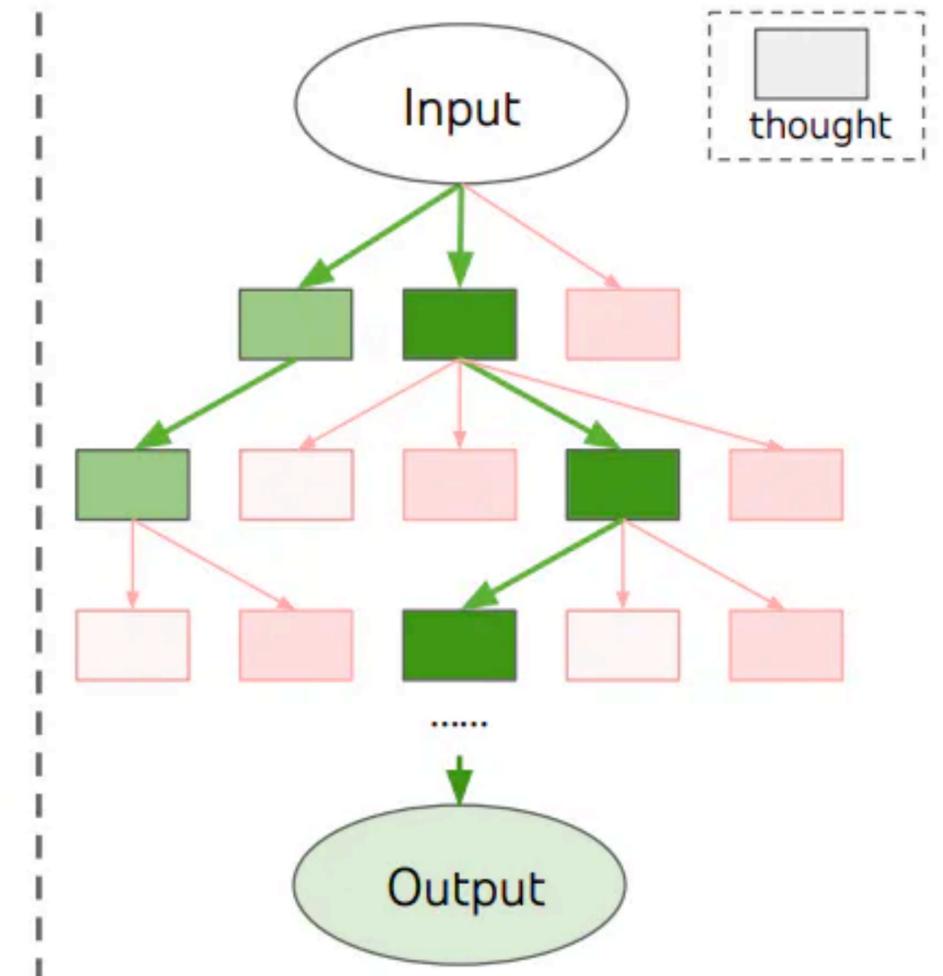
(a) Input-Output Prompting (IO)



(c) Chain of Thought Prompting (CoT)



(c) Self Consistency with CoT (CoT-SC)



(d) Tree of Thoughts (ToT)

Tree of thoughts

Tree of Thoughts: Deliberate Problem Solving with Large Language Models [Yao et al, 2023]

Large Language Model Guided Tree-of-Thought [Long 2023]

- Problem solving by considering multiple reasoning path over “thoughts”
- Decompose problem solving process into “thought” steps and states (partial solution)

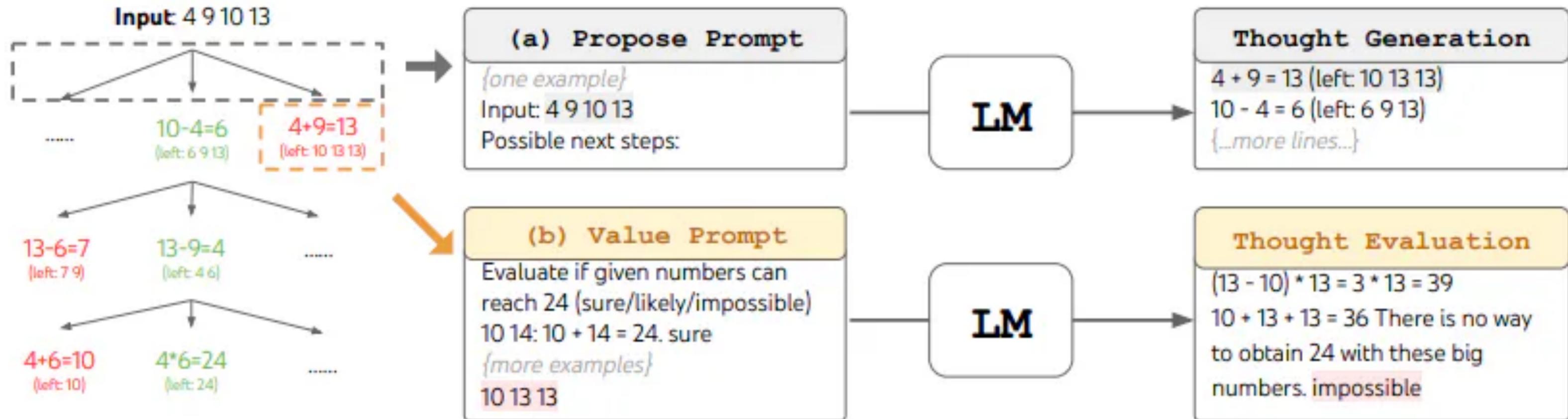
| | Game of 24 | Creative Writing | 5x5 Crosswords |
|-------------------|---|---|--|
| Input | 4 numbers (4 9 10 13) | 4 random sentences | 10 clues (h1. presented;..) |
| Output | An equation to reach 24 (13-9)*(10-4)=24 | A passage of 4 paragraphs ending in the 4 sentences | 5x5 letters: SHOWN; WIRRA; AVAIL; ... |
| Thoughts | 3 intermediate equations (13-9=4 (left 4,4,10); 10-4=6 (left 4,6); 4*6=24) | A short writing plan (1. Introduce a book that connects...) | Words to fill in for clues: (h1. shown; v5. naled; ...) |
| #ToT steps | 3 | 1 | 5-10 (variable) |

Table 1: Task overview. Input, output, thought examples are in blue.

Tree of thoughts

Tree of Thoughts: Deliberate Problem Solving with Large Language Models [Yao et al, 2023]

Example: want to get to number 24 from four input numbers



Use LLM to propose next steps and evaluate how good the state is

Tree of thoughts

Tree of Thoughts: Deliberate Problem Solving with Large Language Models [Yao et al, 2023]

| Method | Success |
|--------------------|------------|
| IO prompt | 7.3% |
| CoT prompt | 4.0% |
| CoT-SC (k=100) | 9.0% |
| ToT (ours) (b=1) | 45% |
| ToT (ours) (b=5) | 74% |
| IO + Refine (k=10) | 27% |
| IO (best of 100) | 33% |
| CoT (best of 100) | 49% |

Table 2: Game of 24 Results.

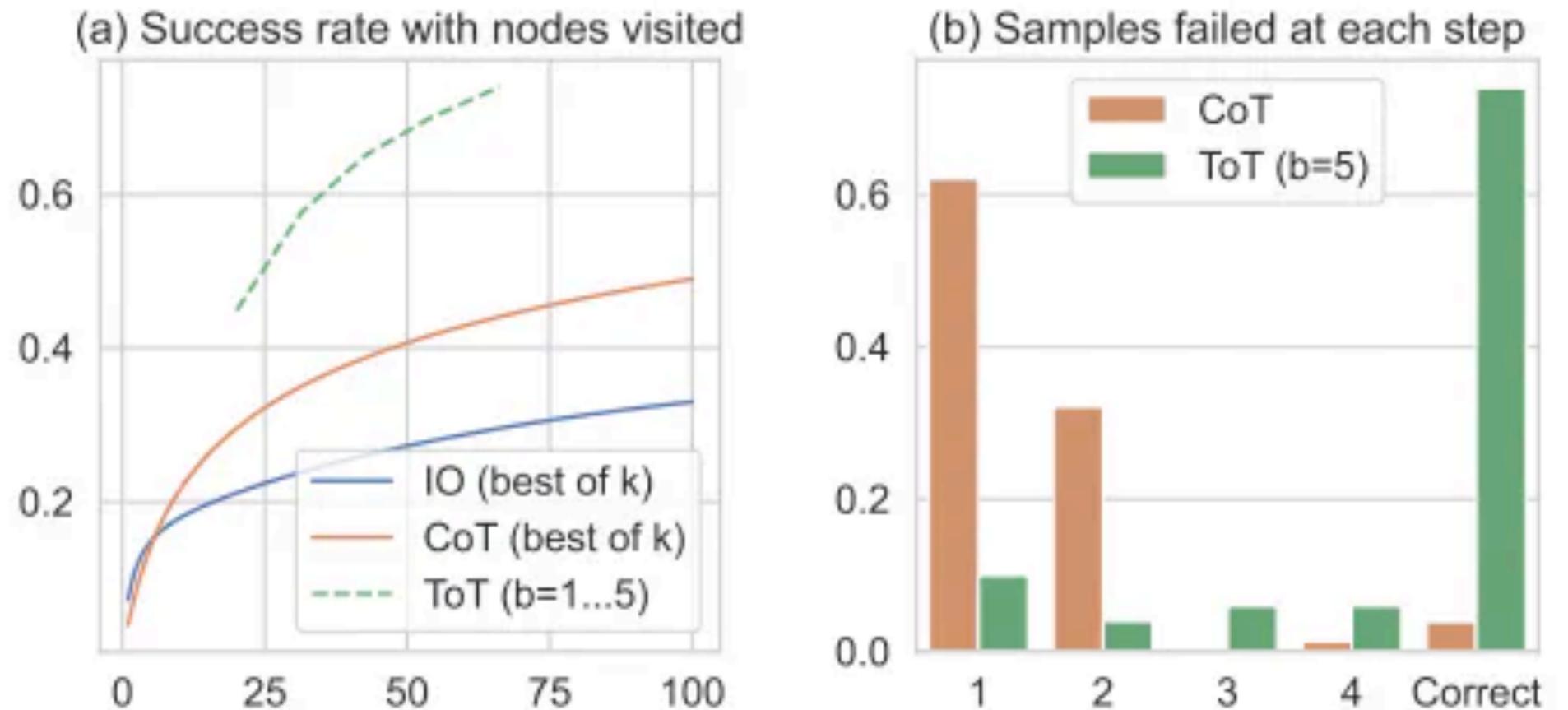


Figure 3: Game of 24 (a) scale analysis & (b) error analysis.

Different types of prompting: <https://www.promptingguide.ai/>

Issues with direct generation with LLMs

- Accuracy issues:
 - **Knowledge cutoffs:** parameters are usually only updated to a particular time
 - **Private data:** data stored in private text or data repositories not suitable for training
 - **Learning failures:** even for data that the model was trained on, it might not be sufficient to get the right answer
- Verifiability issues: It is hard to tell if the answer is correct
- **Solution:** combine LLM generation with ability to retrieve from a set of documents

Retrieval augmented generation (RAG)

- Augment text generation (often using LLM) with retrieved context from an external database
- Allows for incorporation of external information together with pretrained LLM
 - Up-to-date, current information
 - Specialized knowledge (e.g. company or domain specific information)

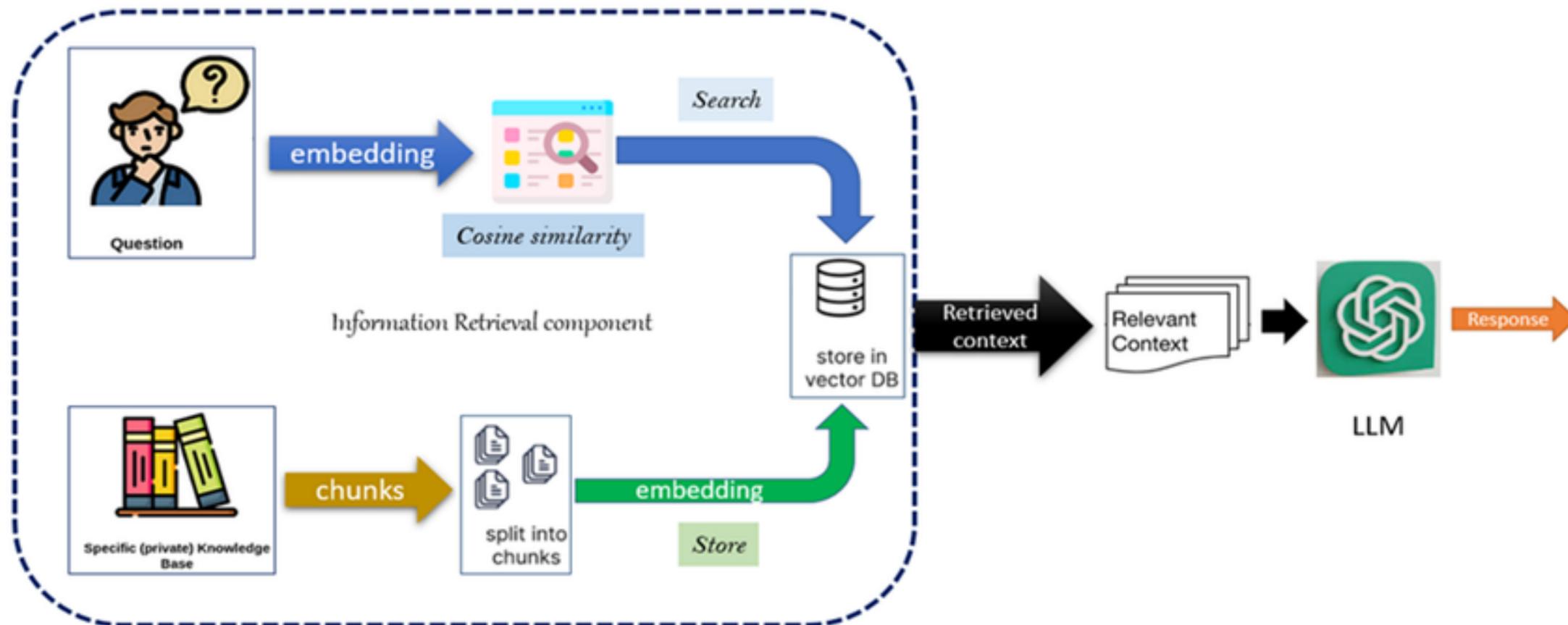
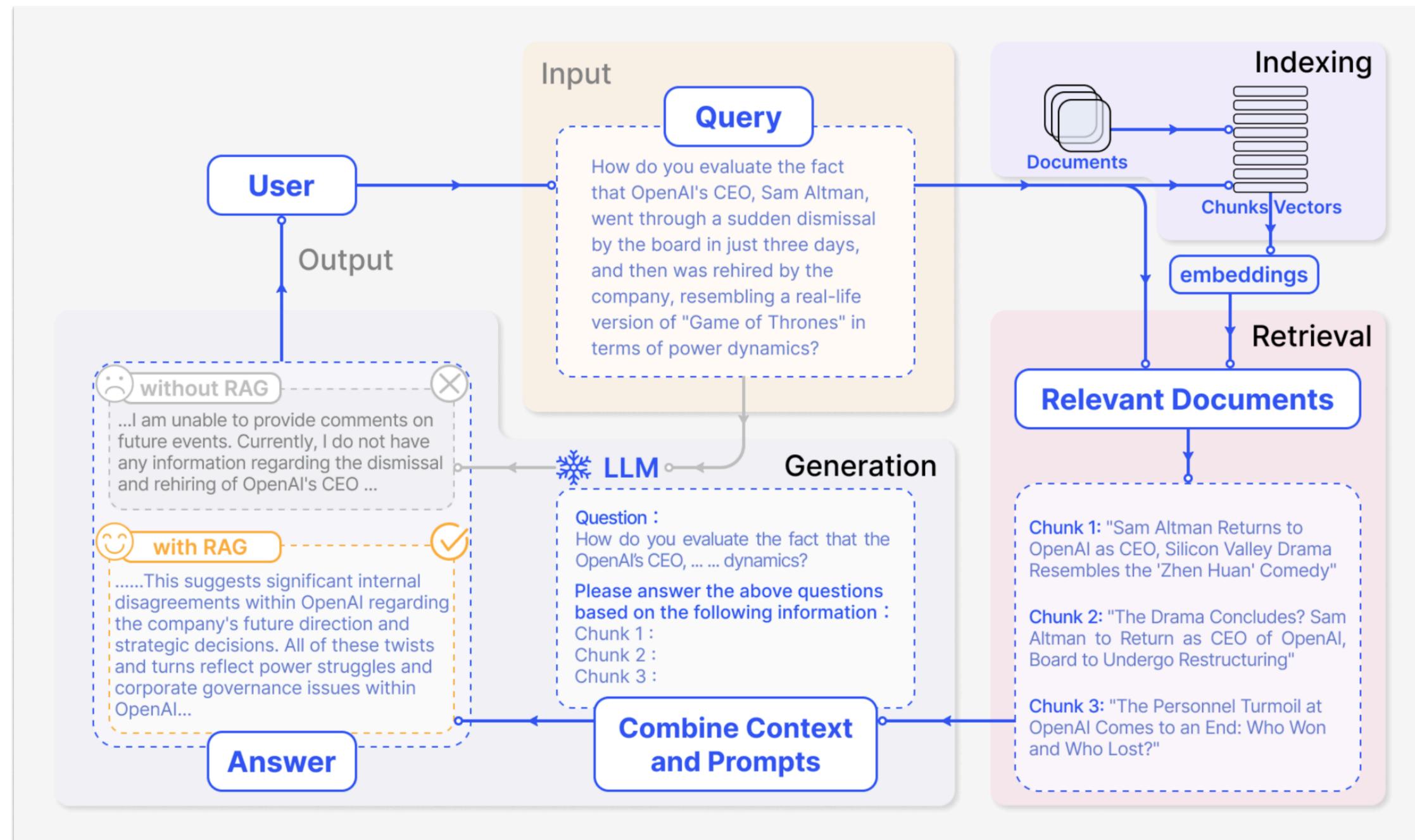


Figure credit: Semantic Embeddings for Arabic Retrieval Augmented Generation (ARAG) [Abdelazim et al. 2023]

Retrieval-augmented generation

- RAG for prompting: Retrieved documents / chunks are used to augment prompts



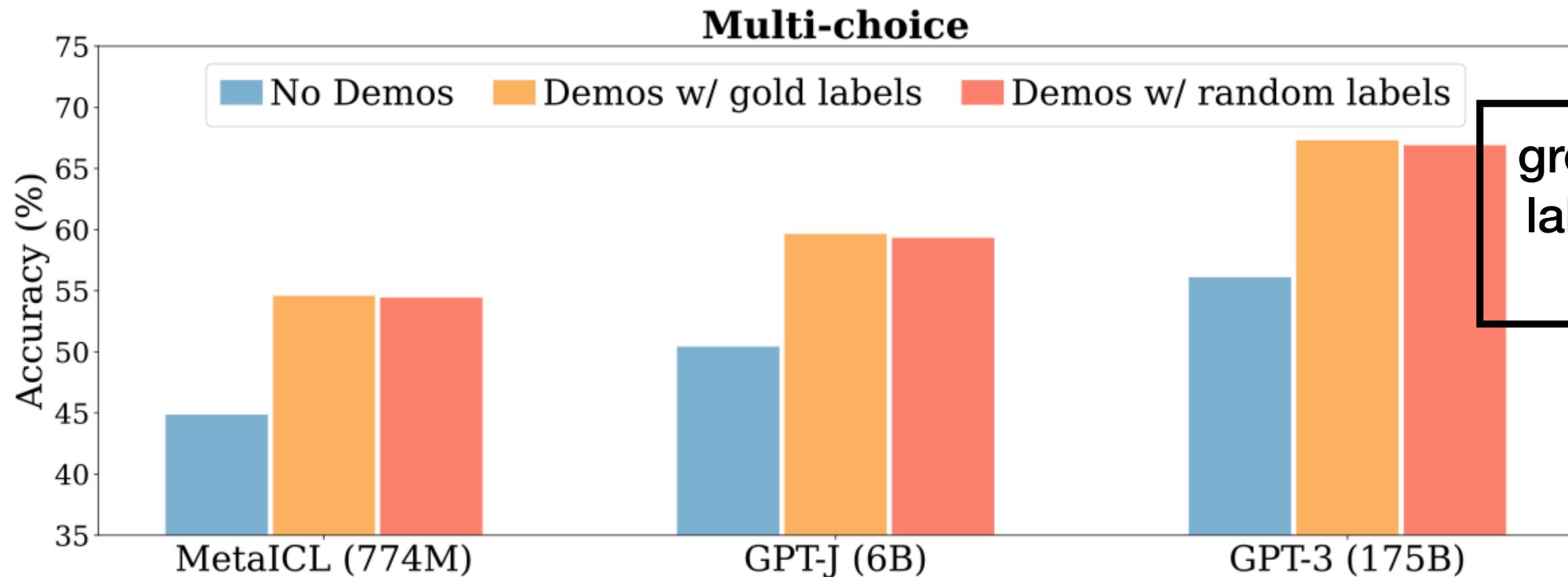
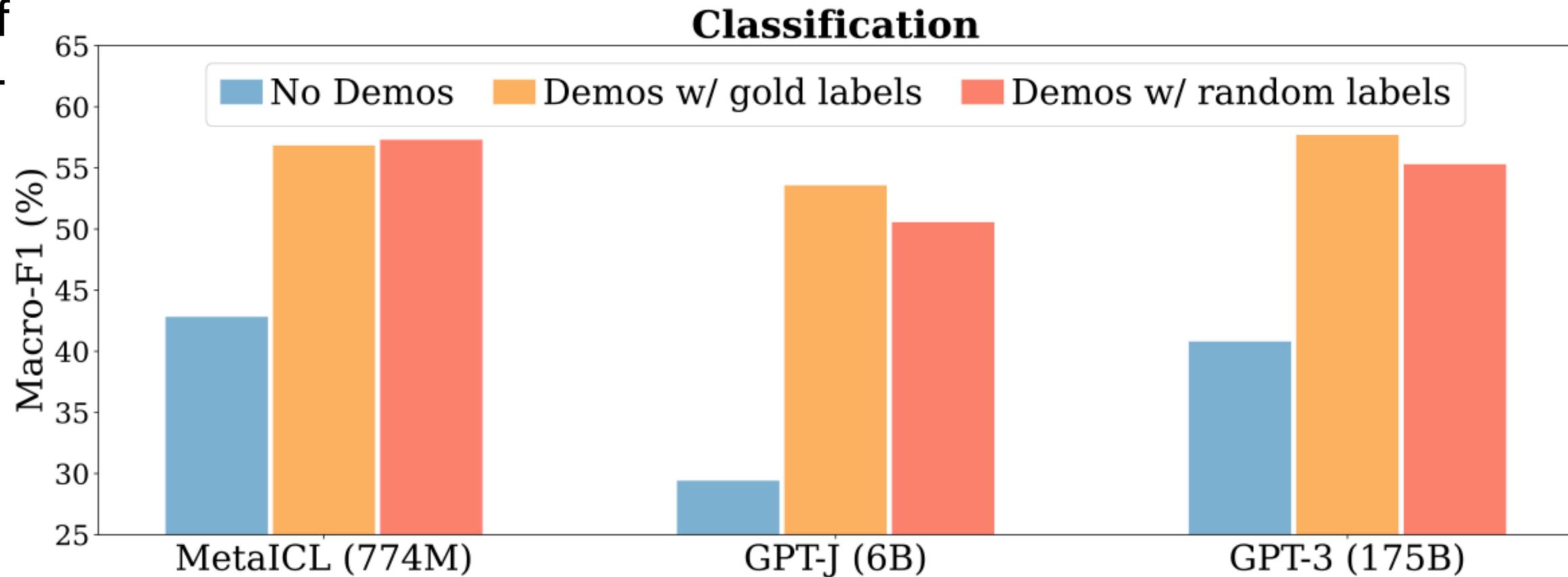
Retrieval-Augmented Generation for Large Language Models: A Survey [Gao et al. 2024]

Why does in-context learning work?

Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?

Sewon Min^{1,2} **Xinxi Lyu**¹ **Ari Holtzman**¹ **Mikel Artetxe**²
Mike Lewis² **Hannaneh Hajishirzi**^{1,3} **Luke Zettlemoyer**^{1,2}
¹University of Washington ²Meta AI ³Allen Institute for AI

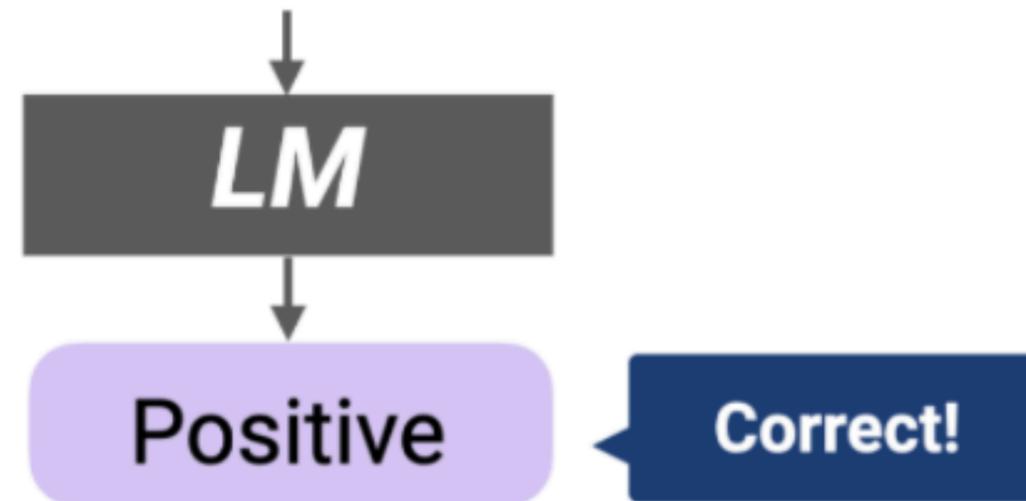
What happens if we provide few-shot examples with random labels?



ground truth labels don't matter!

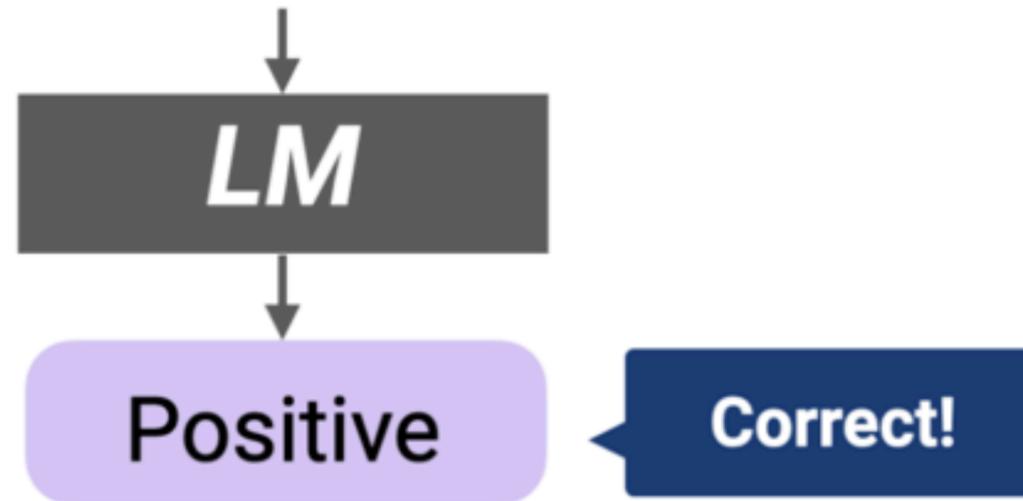
ground truth
labels

| | | |
|--|----|----------|
| 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 | _____ |



replace true labels with
random labels

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



Why does in-context learning work?

Four hypotheses

1. The **input-label mapping**, whether each input x_i is paired with the correct label y_i (not true)
2. The **distribution** that the input x_1, \dots, x_k are from (is it from a sports article, or business news?)
3. The **output label space** y_1, \dots, y_k
4. The **format of the demonstration**, e.g. $x // y$; Input: x Output: y ; etc.

Demonstrations

Distribution of inputs

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 |

*Format
(The use
of pairs)*

Test example

| | | |
|---|----|---|
| The acquisition will have an immediate positive impact. | \n | ? |
|---|----|---|

Input-label mapping



Colour-printed lithograph. Very good condition.

\n Neutral

Many accompanying marketing ... meaning.

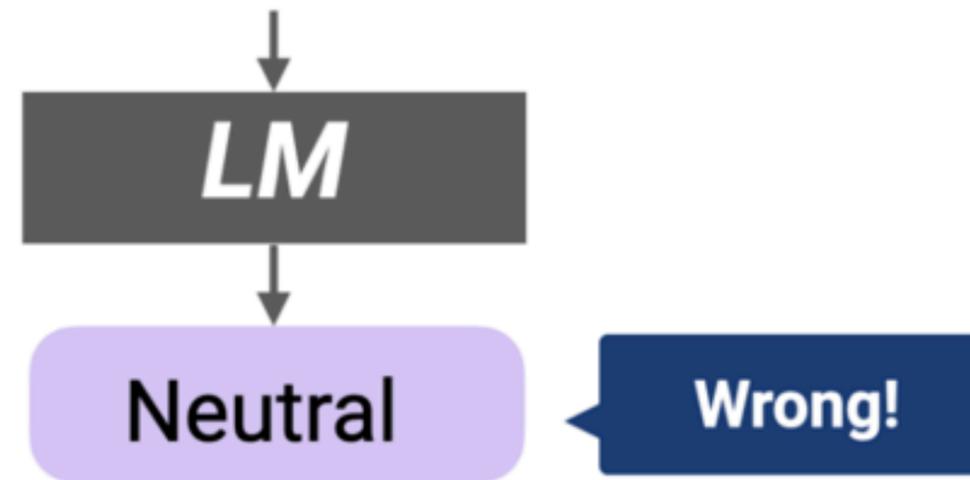
\n Negative

In case you are interested in learning more about ...

\n Positive

The company anticipated its operating profit to improve. \n _____

**Randomly Sampled from CC News*

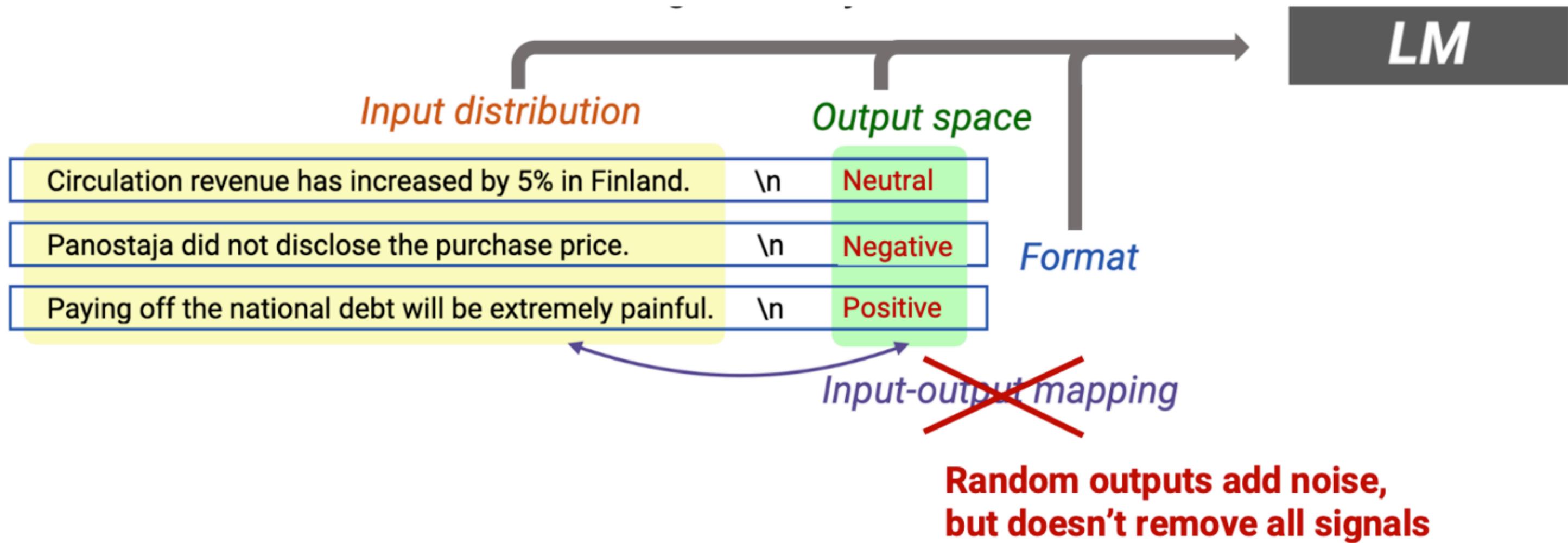


The **input distribution matters**: using inputs from an out of domain corpus causes a large performance drop

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



The **output distribution matters**: using labels that are random English unigrams causes a large performance drop



Training examples (truncated)

```
beet: sport  
golf: animal  
horse: plant/vegetable  
corn: sport  
football: animal
```



Test input and predictions

```
monkey: plant/vegetable ✓  
panda: plant/vegetable ✓  
cucumber: sport ✓  
peas: sport ✓  
baseball: animal ✓  
tennis: animal ✓
```

An example synthetic task with unusual semantics that GPT-3 can successfully learn. A modified figure from Rong.

IN-CONTEXT LEARNING LEARNS LABEL RELATIONSHIPS BUT IS NOT CONVENTIONAL LEARNING

Jannik Kossen^{1∇}

Yarin Gal^{1Δ}

Tom Rainforth^{2Δ}

¹ OATML, Department of Computer Science, University of Oxford

² Department of Statistics, University of Oxford

In-Context Learning (ICL)

- How does the **conditional label distribution** of ICL examples affect **accuracy**?
- ICL does incorporate in-context label information and can even learn truly novel tasks in-context.
- Analogies between ICL and conventional learning algorithms fall short in a variety of ways
 - Label relationships inferred from pre-training have a lasting effect that cannot be surmounted by in-context observations
 - Additional prompting can improve but likely not overcome this deficiency
 - ICL does not treat all information provided in-context equally and preferentially makes use of label information that appears closer to the query

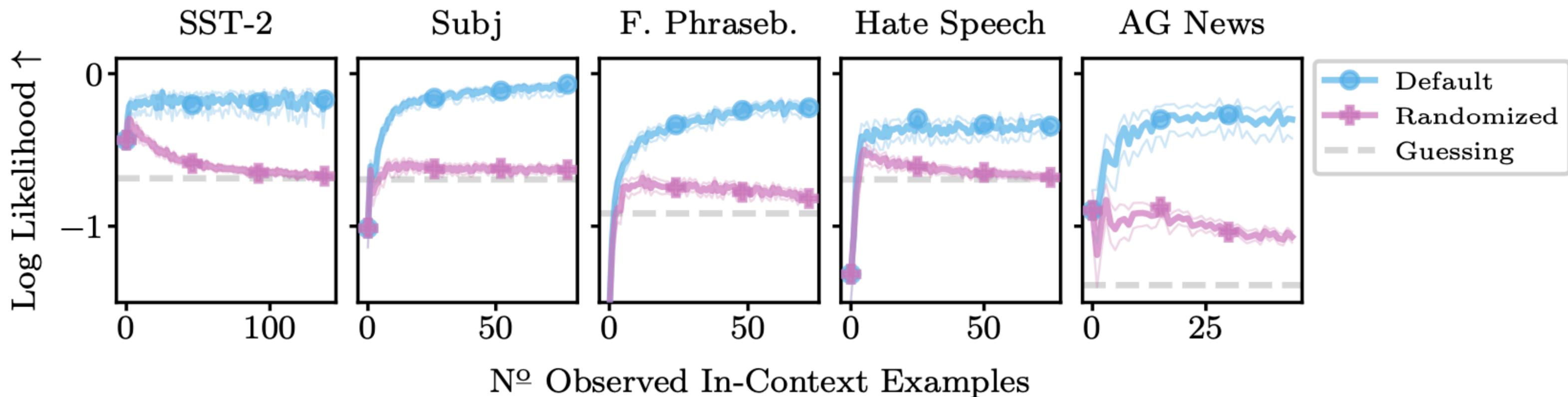


Figure 1: ICL predictions generally depend on the conditional label distribution of in-context examples: when in-context labels are **randomized**, average log likelihoods of label predictions decrease compared to ICL with **default** labels for LLaMa-2-70B across a variety of tasks. Results averaged over 500 in-context datasets and thin lines are 99% confidence intervals. See §5 for details.

Table 1: Average differences between ICL log likelihoods for default and randomized labels. Bold entries indicate differences are statistically significant. We can disregard lightgray entries: for them, default ICL performance is not significantly better than a random guessing baseline. Whenever default ICL outperforms the baseline, ICL almost always performs significantly worse (positive differences) for random labels. Averages over 500 runs at max. context size, standard errors in Table F.1.

| Δ Log Likelihood | SST-2 | Subj | FP | HS | AGN | MQP | MRPC | RTE | WNLI |
|-------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| LLaMa-2 7B | 0.42 | 0.39 | 0.57 | 0.18 | 0.53 | 0.03 | 0.02 | 0.03 | 0.02 |
| LLaMa-2 13B | 0.41 | 0.62 | 0.49 | 0.24 | 0.81 | 0.04 | 0.01 | 0.06 | 0.02 |
| LLaMa-2 70B | 0.51 | 0.53 | 0.57 | 0.34 | 0.80 | 0.29 | 0.04 | 0.22 | 0.18 |
| Falcon 7B | 0.20 | 0.19 | 0.25 | 0.06 | 0.31 | 0.01 | 0.01 | -0.01 | 0.01 |
| Falcon 7B Instr. | 0.13 | 0.08 | 0.11 | 0.03 | 0.15 | 0.03 | 0.02 | -0.00 | 0.00 |
| Falcon 40B | 0.34 | 0.35 | 0.31 | 0.18 | 0.90 | 0.06 | 0.01 | 0.01 | 0.02 |
| Falcon 40B Instr. | 0.25 | 0.37 | 0.27 | 0.02 | 0.77 | 0.06 | 0.02 | 0.02 | 0.04 |

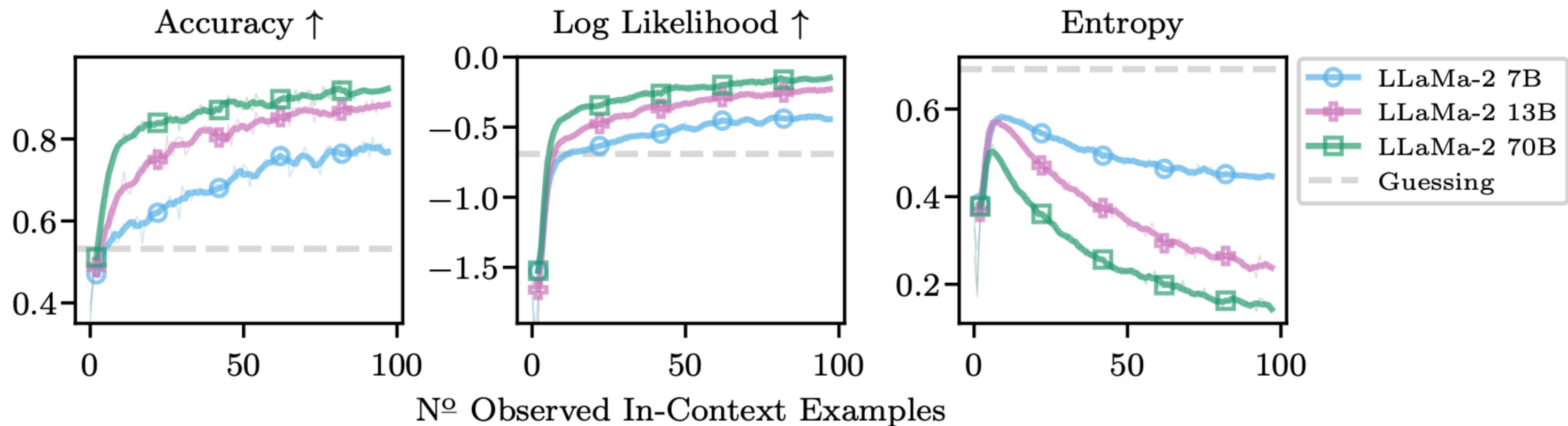


Figure 4: Few-shot ICL achieves accuracies significantly better than random guessing on our **novel author identification** task. Thus, LLMs can learn novel label relationships entirely in-context. Averages over 500 runs, thick lines with additional moving average (window size 5) for clarity.

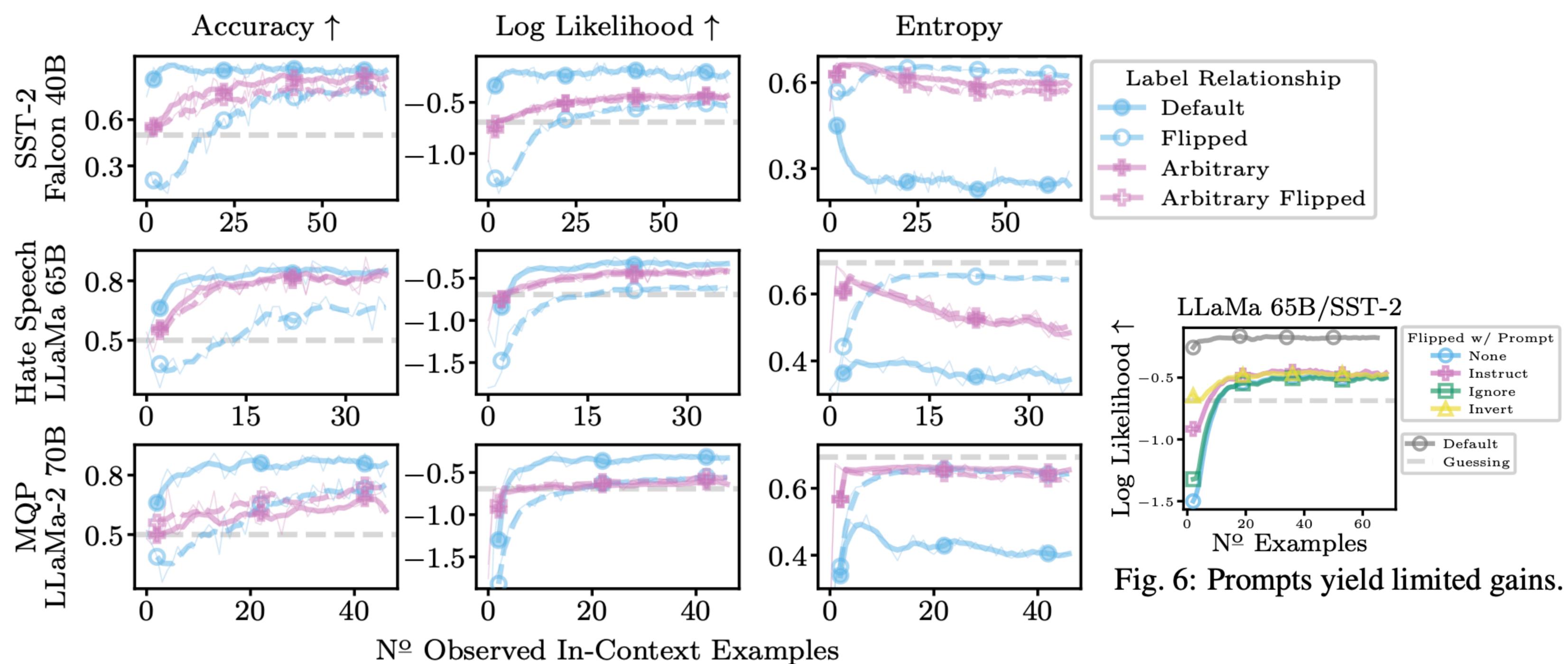


Fig. 6: Prompts yield limited gains.

Figure 5: Few-shot ICL with **replacement labels** for Falcon-40B on SST-2, LLaMa-2-65B on Hate Speech, and LLaMa-2-70B on MQP. Table 2 and §F contain results for all other models and tasks. ICL achieves better than guessing performance for all label relations and models. However, predictions for flipped labels (**dashed blue**) plateau at a higher entropies and lower likelihoods than those for the default label relation (**solid blue**). For arbitrary labels (**pink**), the model performs similarly for both label directions. Averages over 100 runs and thick lines with moving average (window size 5).

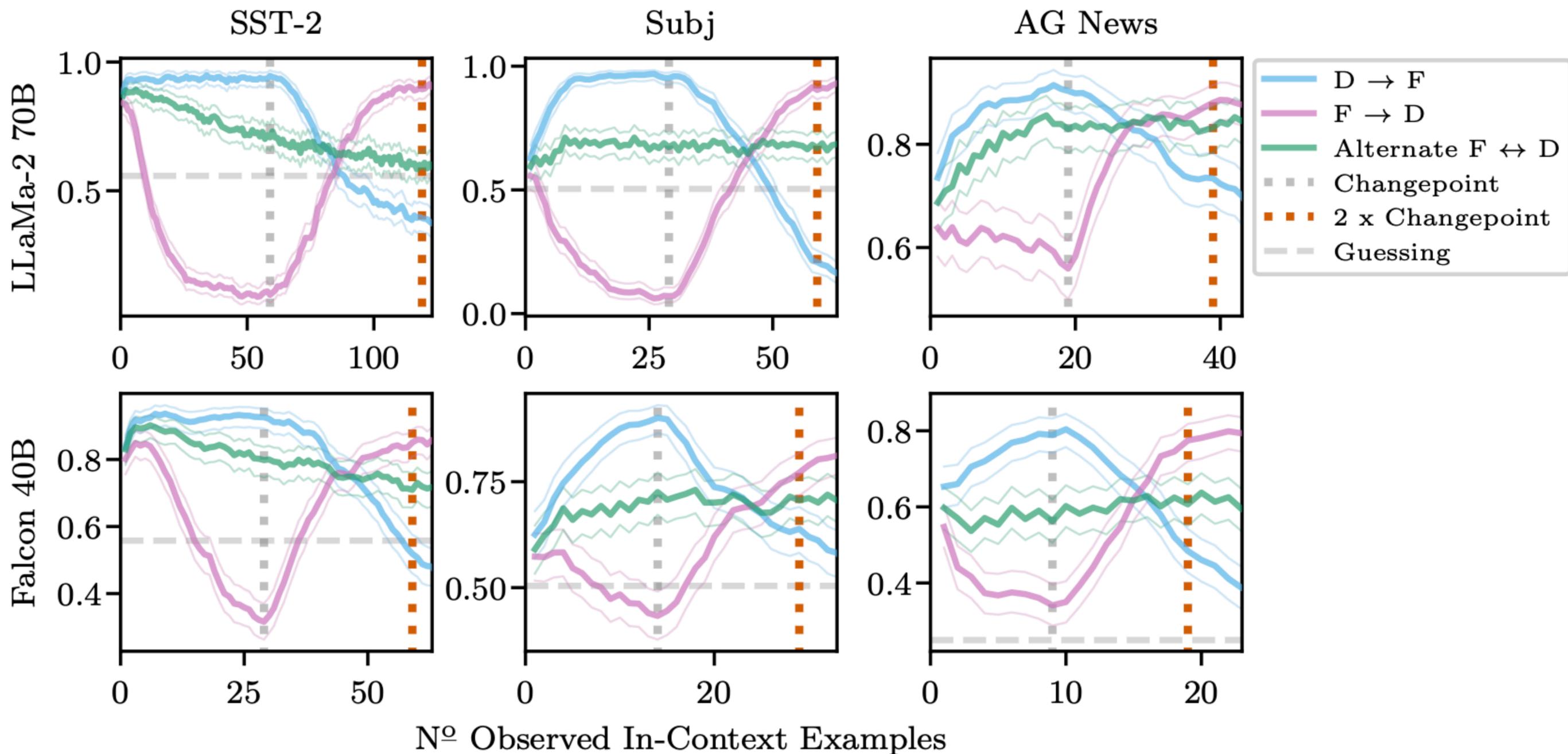


Figure 7: Few-shot ICL accuracies when the **label relationship changes throughout ICL**. For ($D \rightarrow F$), we start with default labels and change to flipped labels at the changepoint, for ($F \rightarrow D$) we change from flipped to the default labels at the changepoint, and for (Alternating $F \leftrightarrow D$) we alternate between the two label relationships after every observation. For all setups, at ‘2 x Changepoint’, the LLMs have observed the same number of examples for both label relations. If, according to NH3, ICL treats all in-context information equally, predictions should be equal at that point—but they are not. Bootstrapped 99% confidence intervals, moving averages (size 3), and 500 repetitions.

Fine-tuning and aligning LLMs

Different ways to fine-tune or align your model

Fine-tuning

- Full fine-tuning
- **Parameter efficient fine-tuning (PEFT)**

Aligning to instructions / human values:

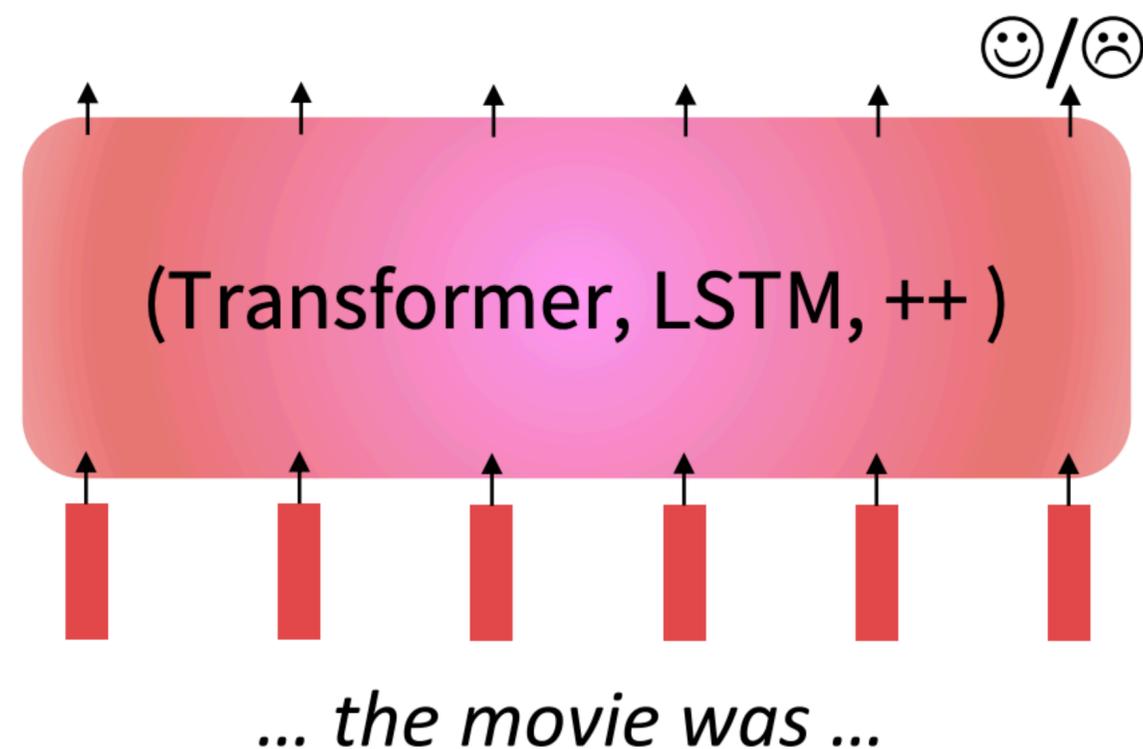
- Instruction tuning (fine-tune with instructions)
- Reinforcement learning with human feedback (train with modified objective that incorporates human preferences)

Full finetuning vs parameter efficient fine-tuning

- Finetuning every parameter in a pretrained model works well, but is memory-intensive.
- **Lightweight** finetuning methods adapt pretrained models in a constrained way.
- Leads to **less overfitting** and/or **more efficient finetuning and inference**.

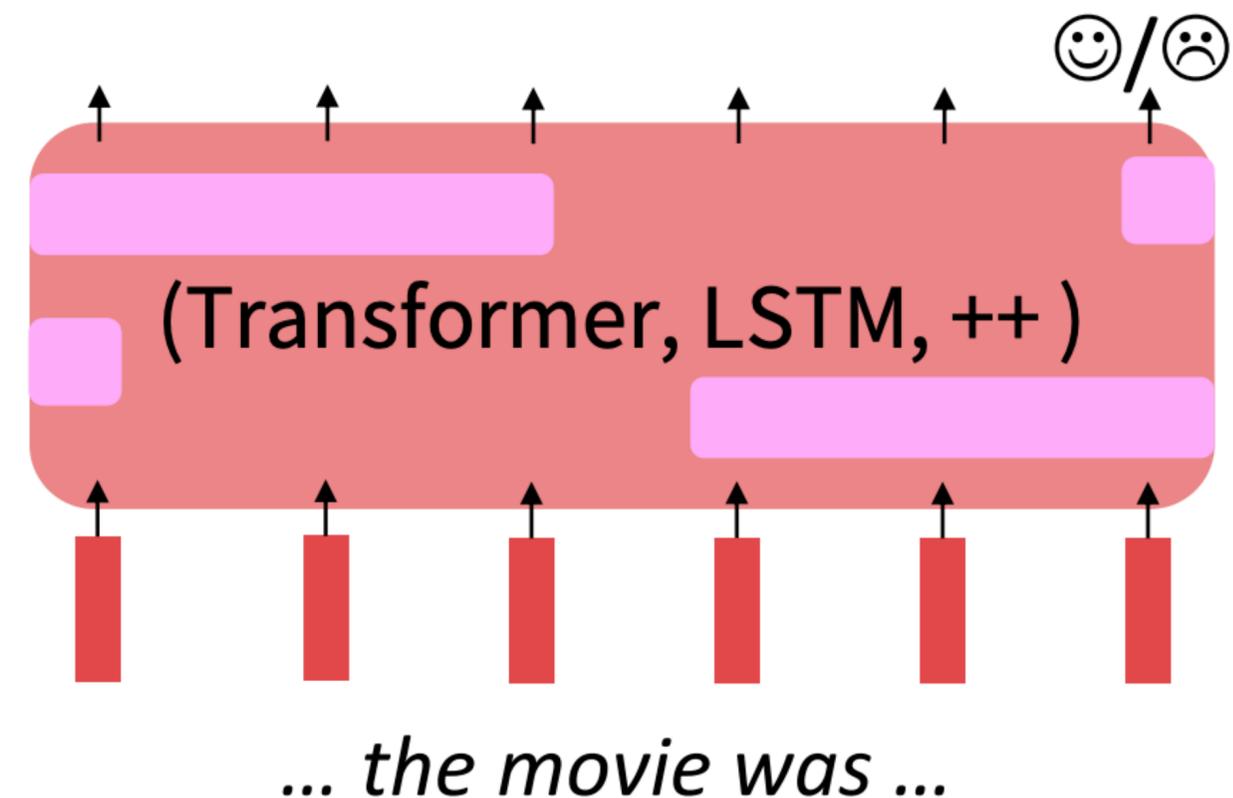
Full Finetuning

Adapt all parameters



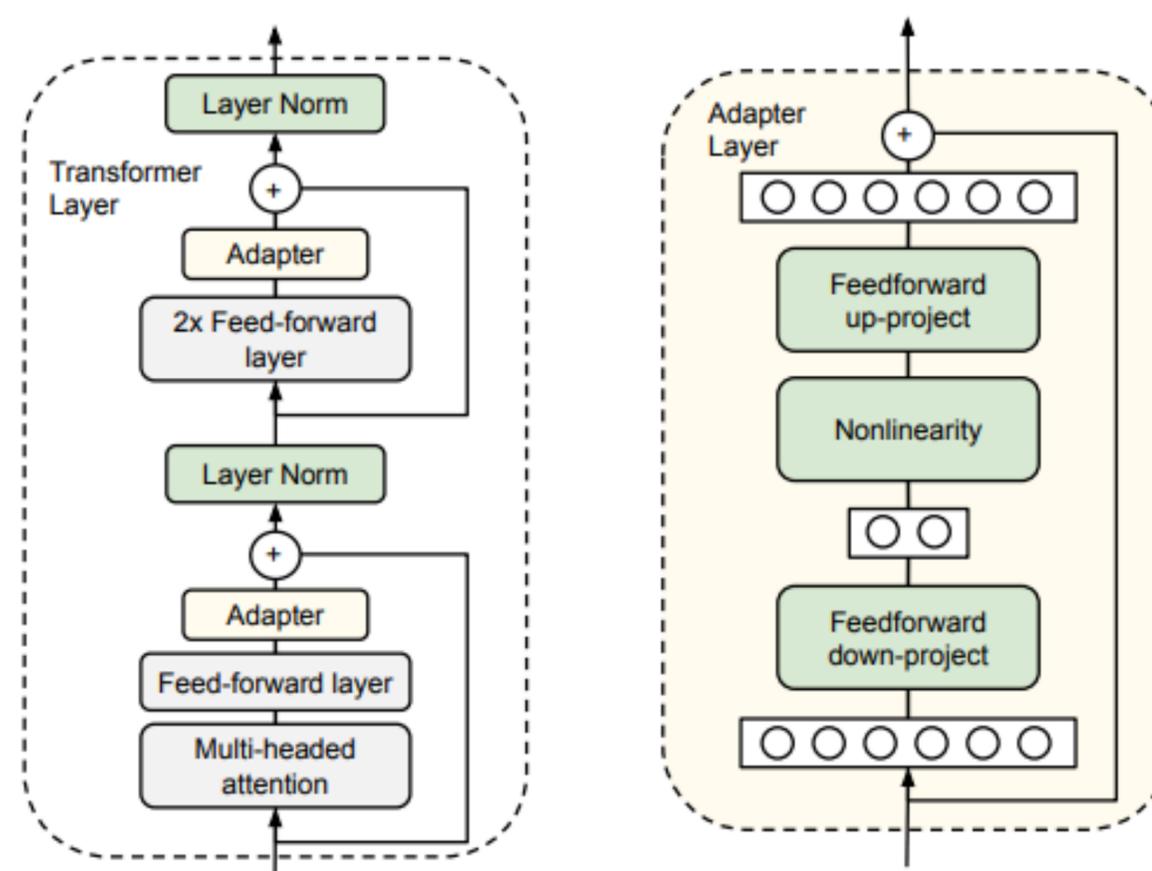
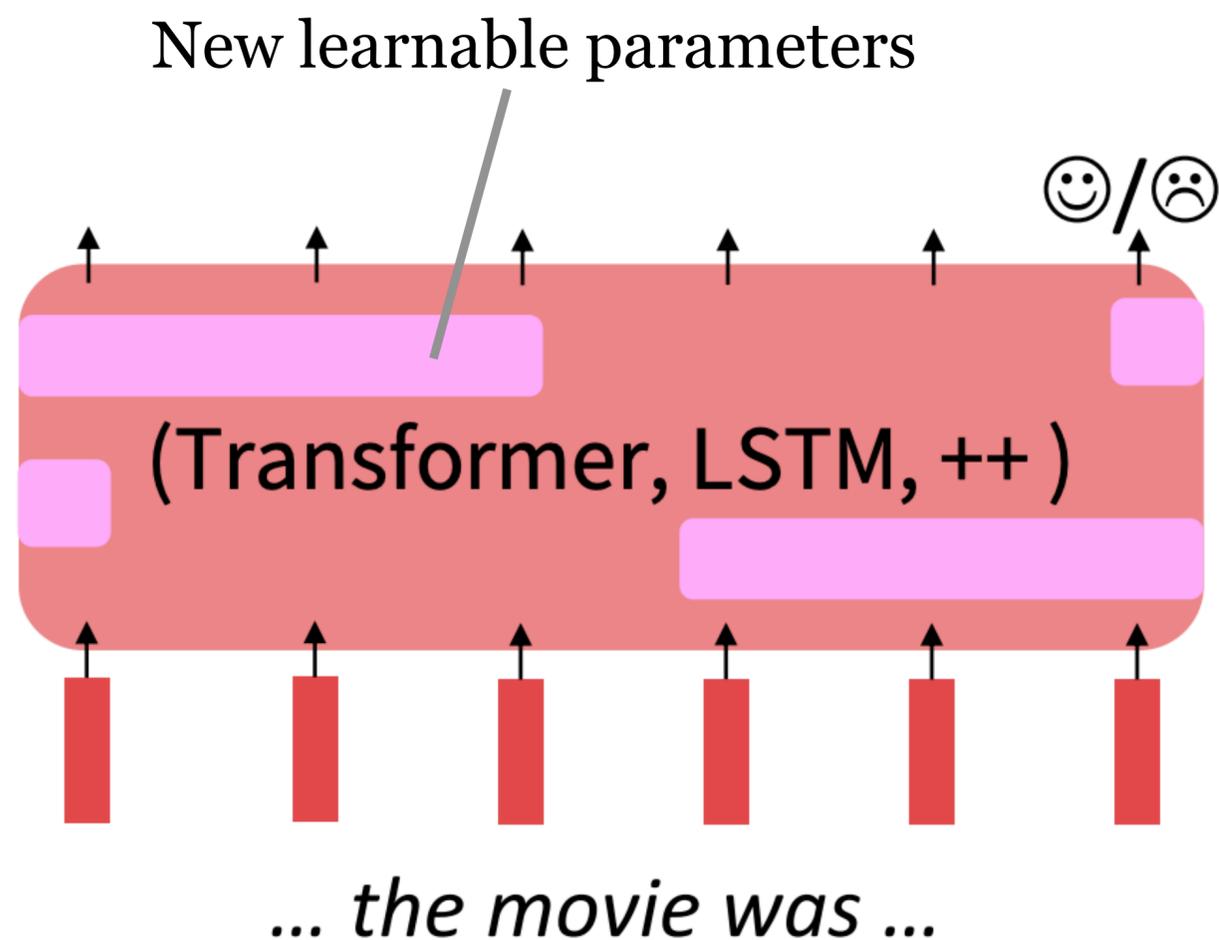
Lightweight Finetuning

Train a few existing or new parameters

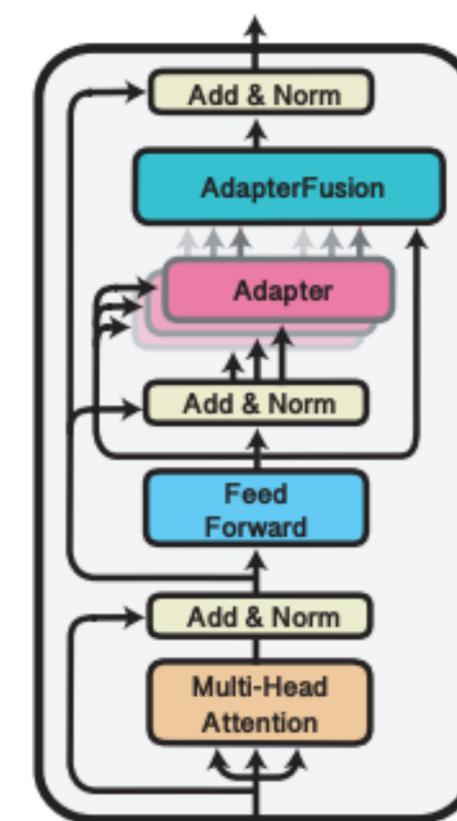


Parameter-Efficient Finetuning: Adapters

- Add lightweight network with new learnable parameters
- Only these parameters are fine-tuned, rest are frozen



[Houlsby et al., 2019]

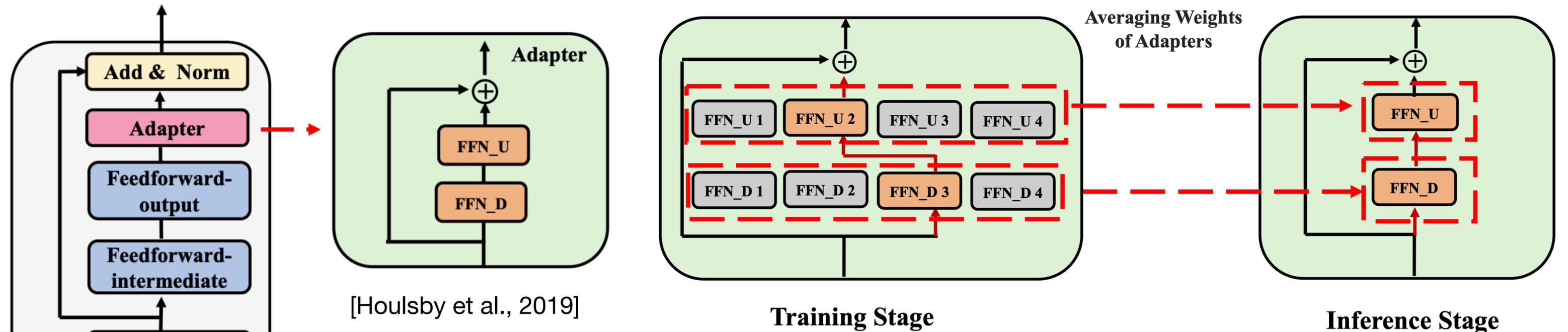


[Pfeiffer et al., 2021]

<https://github.com/adaptor-hub/adaptor-transformers>

Parameter-Efficient Finetuning: Adapters

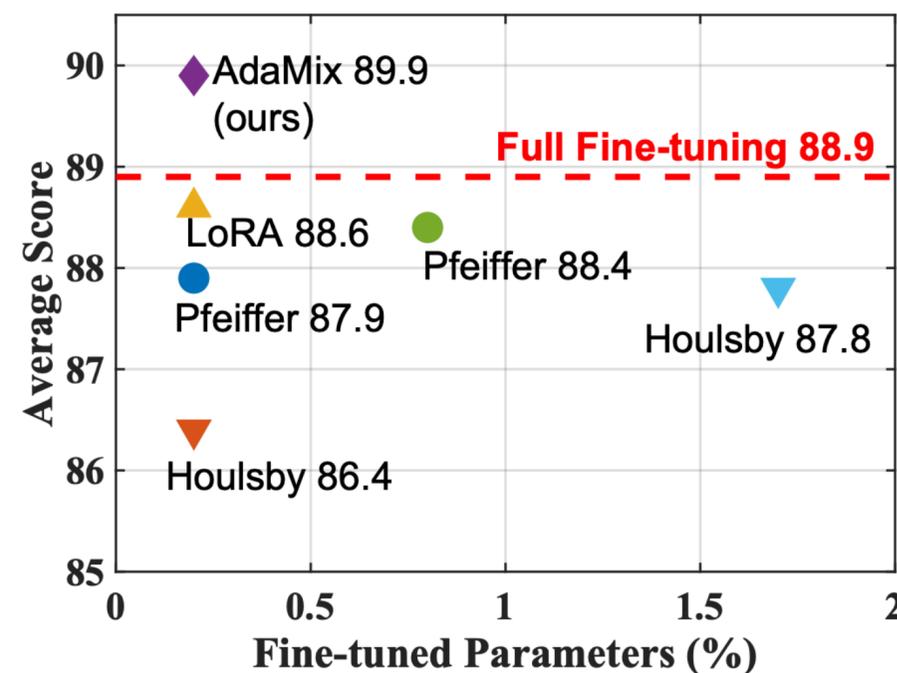
- **Mixture of adapters** - stochastically selected during training
- Average weights of adapters during inference



[Houlsby et al., 2019]

Training Stage

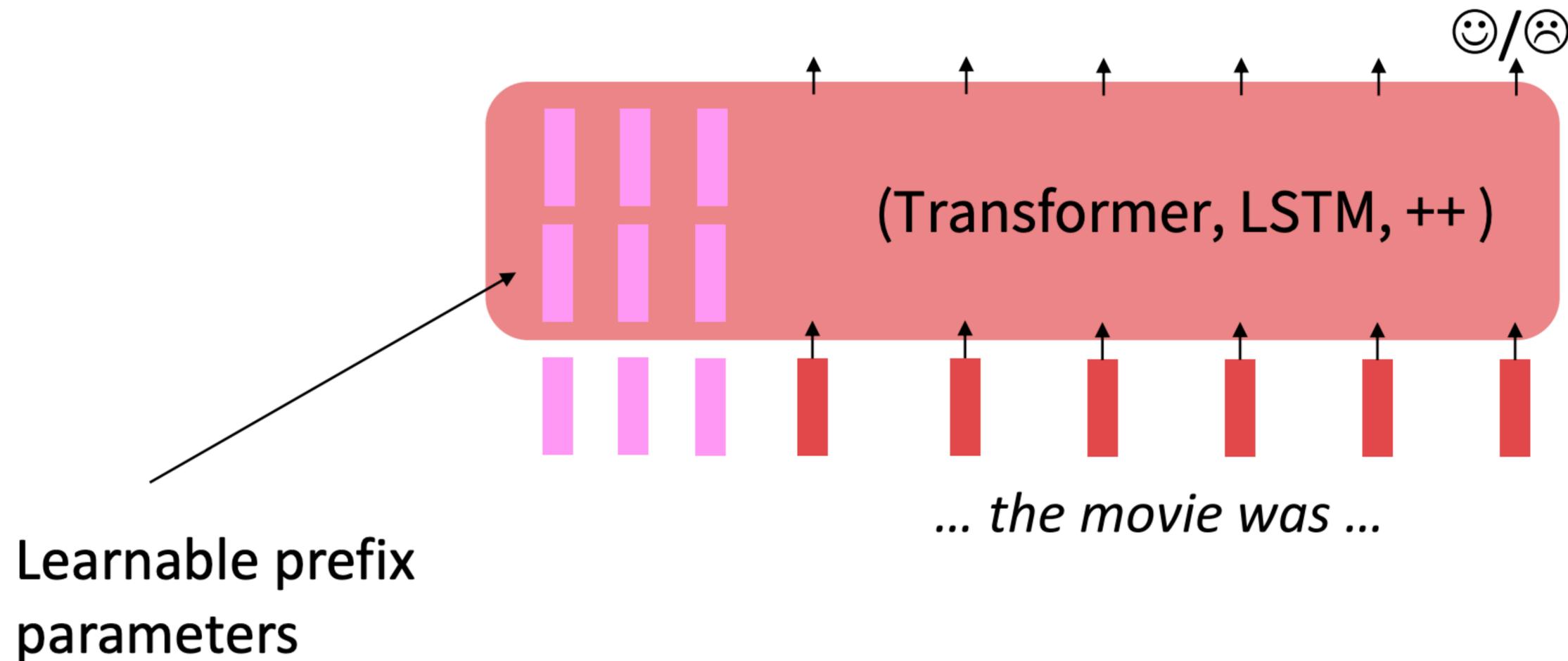
Inference Stage



Performance on GLUE, fine-tuning of RoBERTa-large

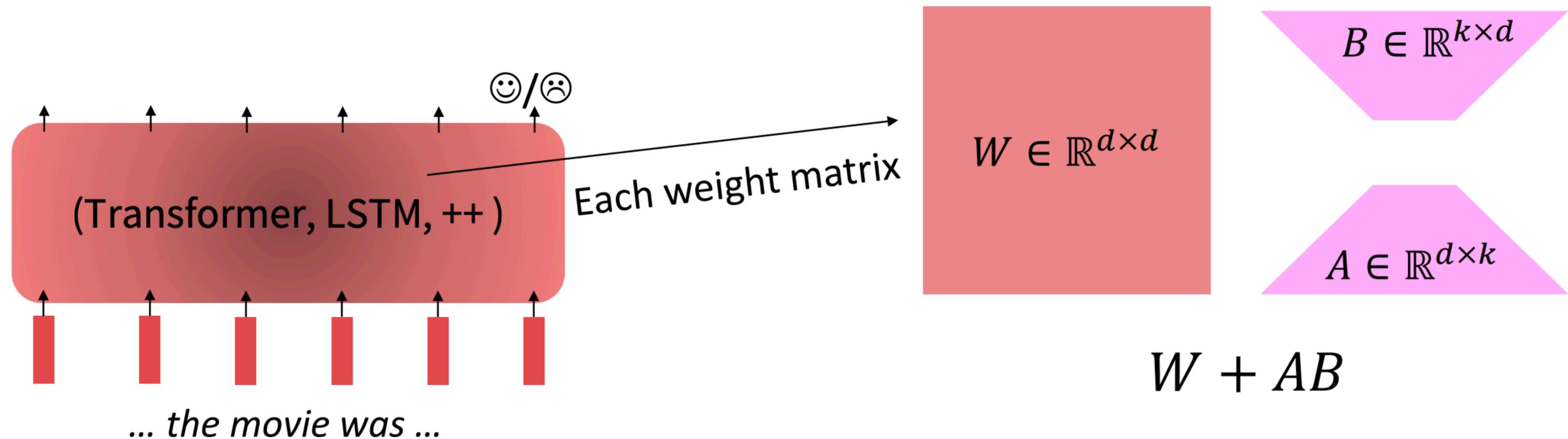
Parameter-Efficient Finetuning: Prefix-Tuning, Prompt tuning

- Prefix-Tuning adds a prefix of parameters, and freezes all pretrained parameters.
- The prefix is processed by the model just like real words would be.
- Advantage: each element of a batch at inference could run a different tuned model.



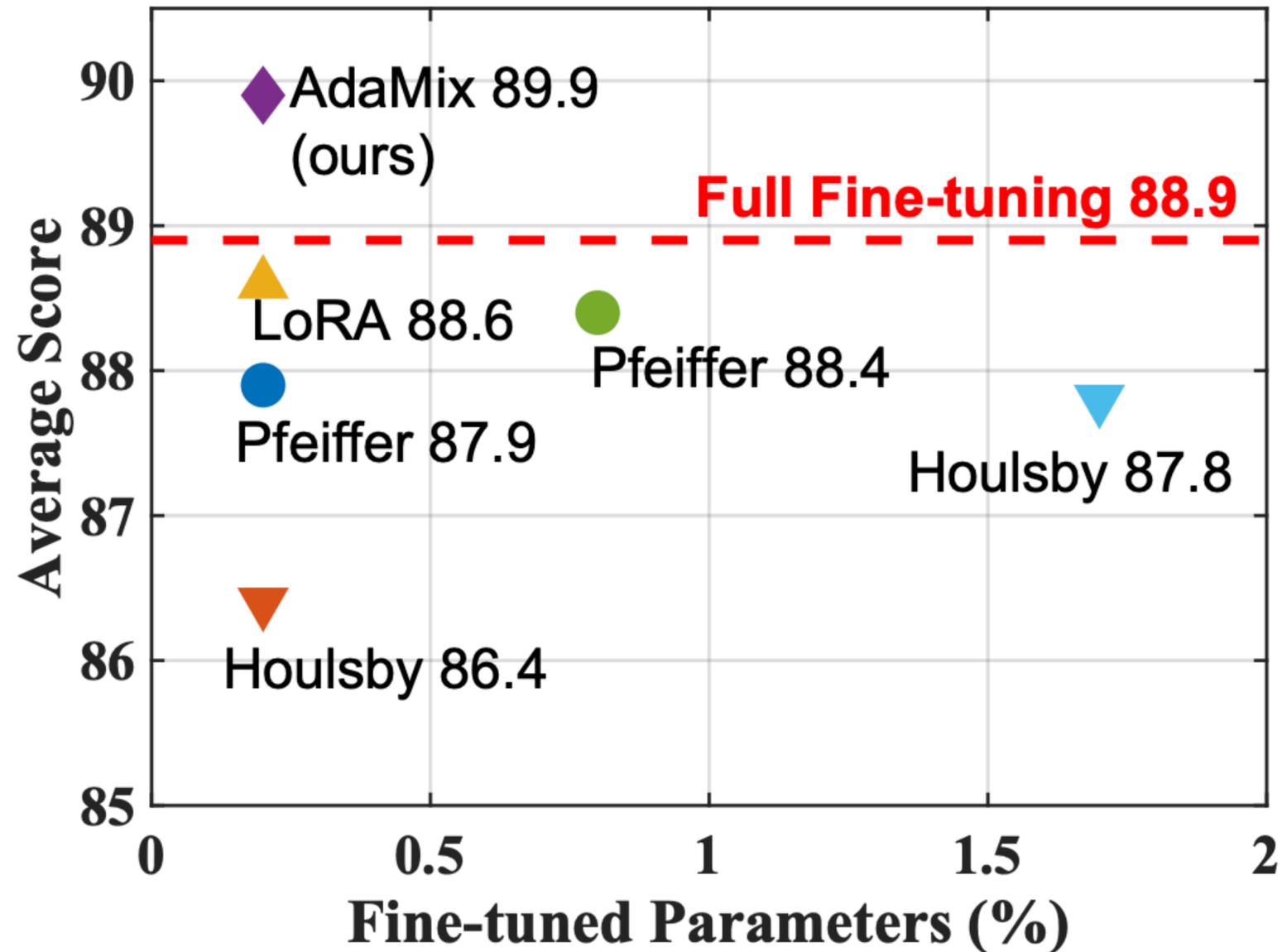
Parameter-Efficient Finetuning: Low-Rank Adaptation

- Low-Rank Adaptation learns a low-rank “diff” between the pretrained and finetuned weight matrices.
- Easier to learn than prefix-tuning



Parameter-Efficient Finetuning: Low-Rank Adaptation

| Model | #Param. | M |
|-------------------------------|---------|----|
| Full Fine-tuning [†] | 355.0M | 91 |
| Pfeiffer Adapter [†] | 3.0M | 91 |
| Pfeiffer Adapter [†] | 0.8M | 91 |
| Houlsby Adapter [†] | 6.0M | 81 |
| Houlsby Adapter [†] | 0.8M | 91 |
| LoRA [†] | 0.8M | 91 |
| AdaMix Adapter | 0.8M | 91 |



| S-B | Avg. |
|-----|------|
| 2.4 | 88.9 |
| 2.1 | 88.4 |
| 1.9 | 87.9 |
| 1.0 | 87.8 |
| 1.5 | 86.4 |
| 2.3 | 88.6 |
| 2.4 | 89.9 |

Good performance by tuning just a fraction of the weights

Going toward smaller powerful LMs

- Knowledge Distillation
 - DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. Sanh et al. NeurIPS Workshop 2019
 - TinyBERT: Distilling BERT for Natural Language Understanding. Jiao et al. Findings of ACL 2020
- Quantization
 - Q8BERT: Quantized 8bit BERT, Zafrir et al, NeurIPS Workshop 2019
- Model Pruning
 - Compressing BERT: Studying the effects of weight pruning on transfer learning. Gordon et al. Workshop of ACL 2020.