SFU NatLangLab

CMPT 413/713: Natural Language Processing

Adapting LLMs for tasks
Prompting and fine-tuning LLMs

Spring 2026
2026-02-23

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

Training stages for modern LLMs

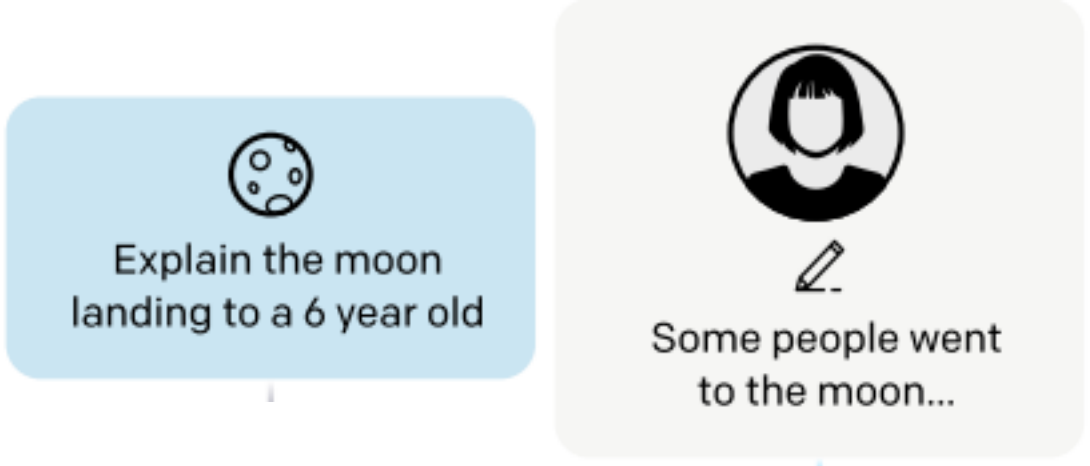
Piles of unlabeled text!

Self-supervised training LM objective

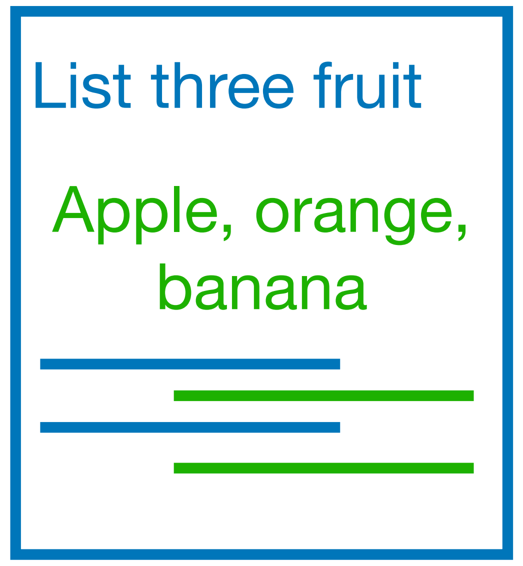
$p(w_t | w_{<t})$

Pre-training
LM training on large, large amount of data

Pre-training can be broken into stages (mid-training)



Text with "instructions" and "responses"

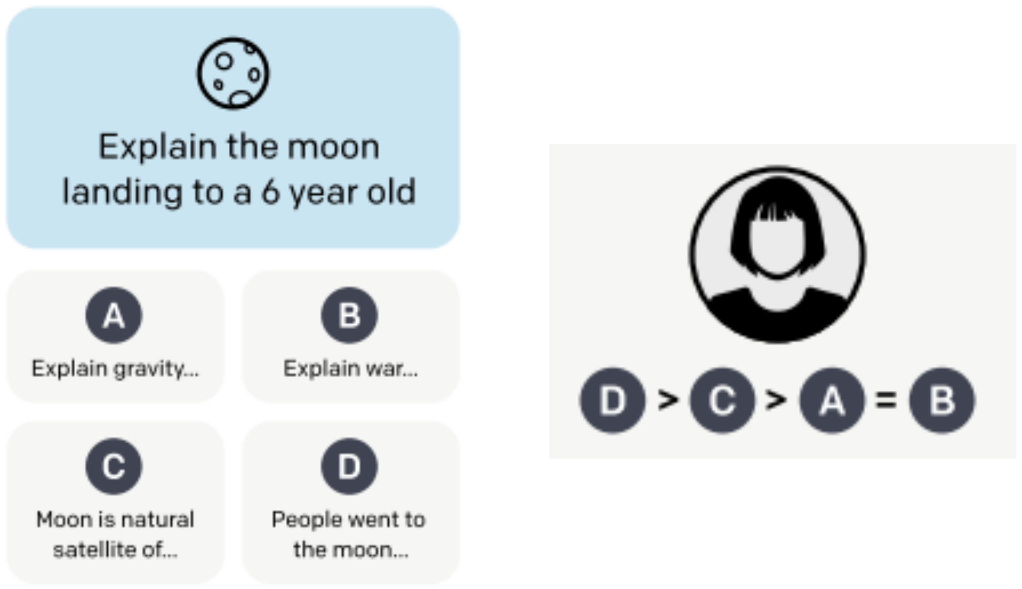


Supervised training LM objective

$p(w_t | w_{<t};\ \text{prompt})$

Instruction-tuning
Supervised fine-tuning for instructions

Human preference data
Data about what people prefer

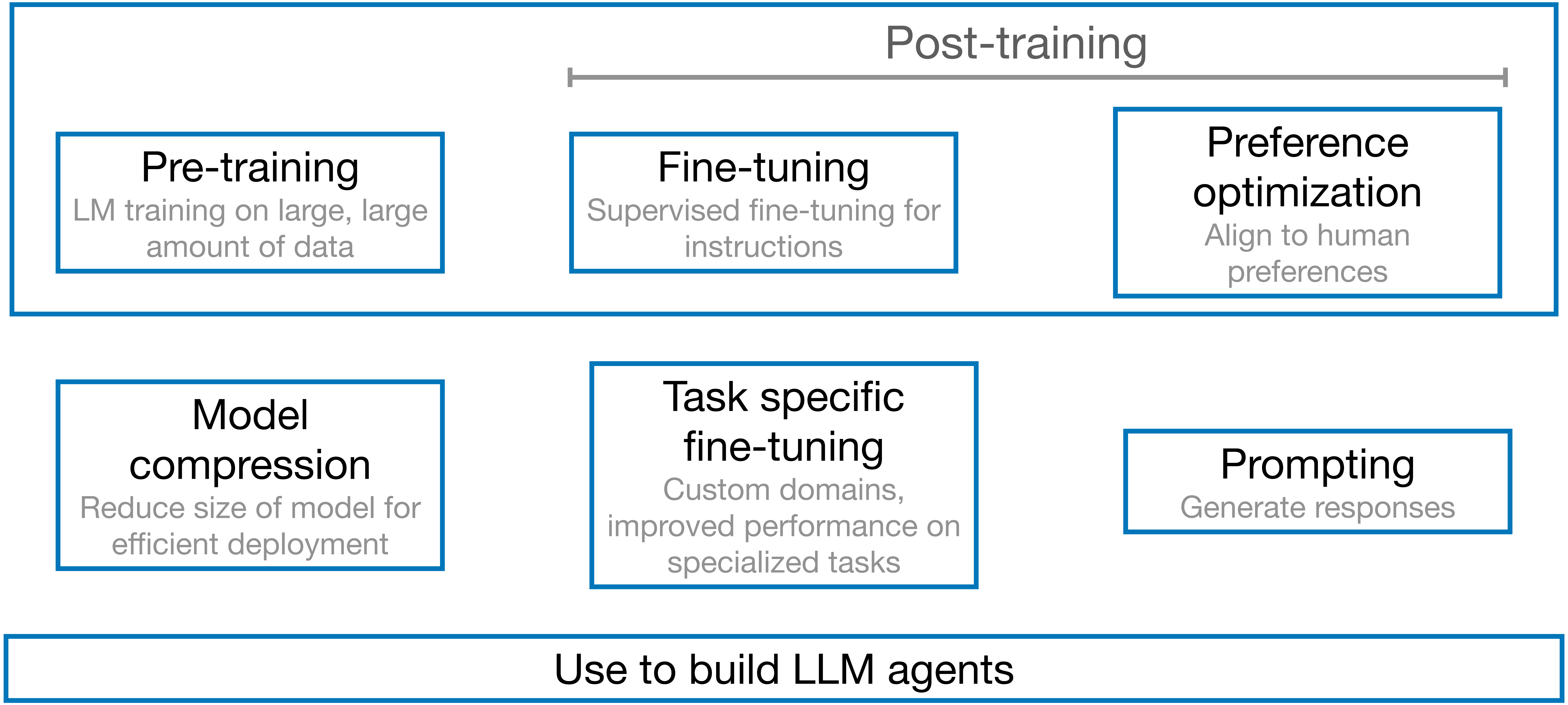


Reinforcement learning

Preference optimization
Align to human preferences

Post-training

(Can have more iterations)


Post-training
Pre-training
LM training on large, large amount of data
Fine-tuning
Supervised fine-tuning for instructions
Preference optimization
Align to human preferences
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

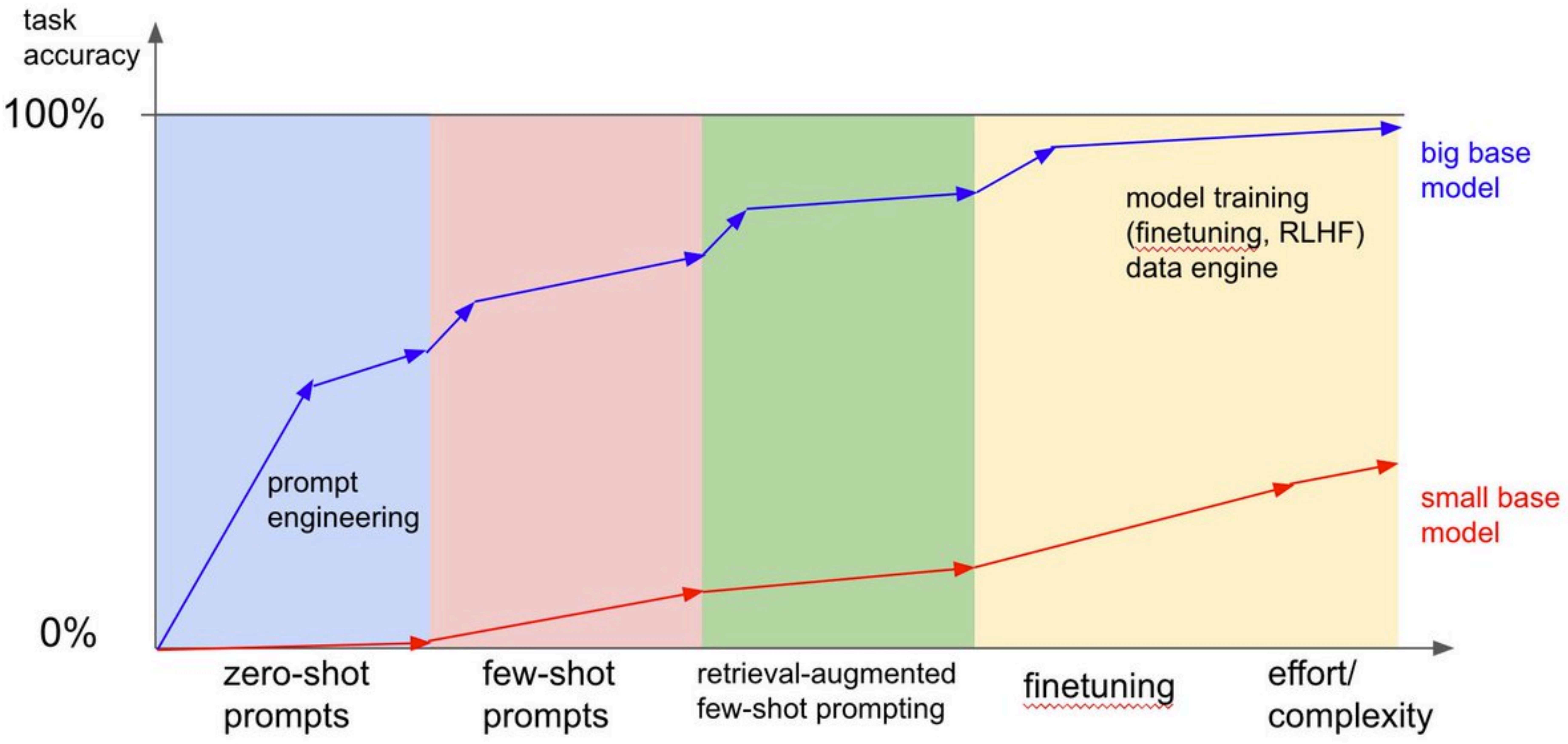


Figure credit: Andrej Karpathy

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**

https://arxiv.org/abs/2005.14165

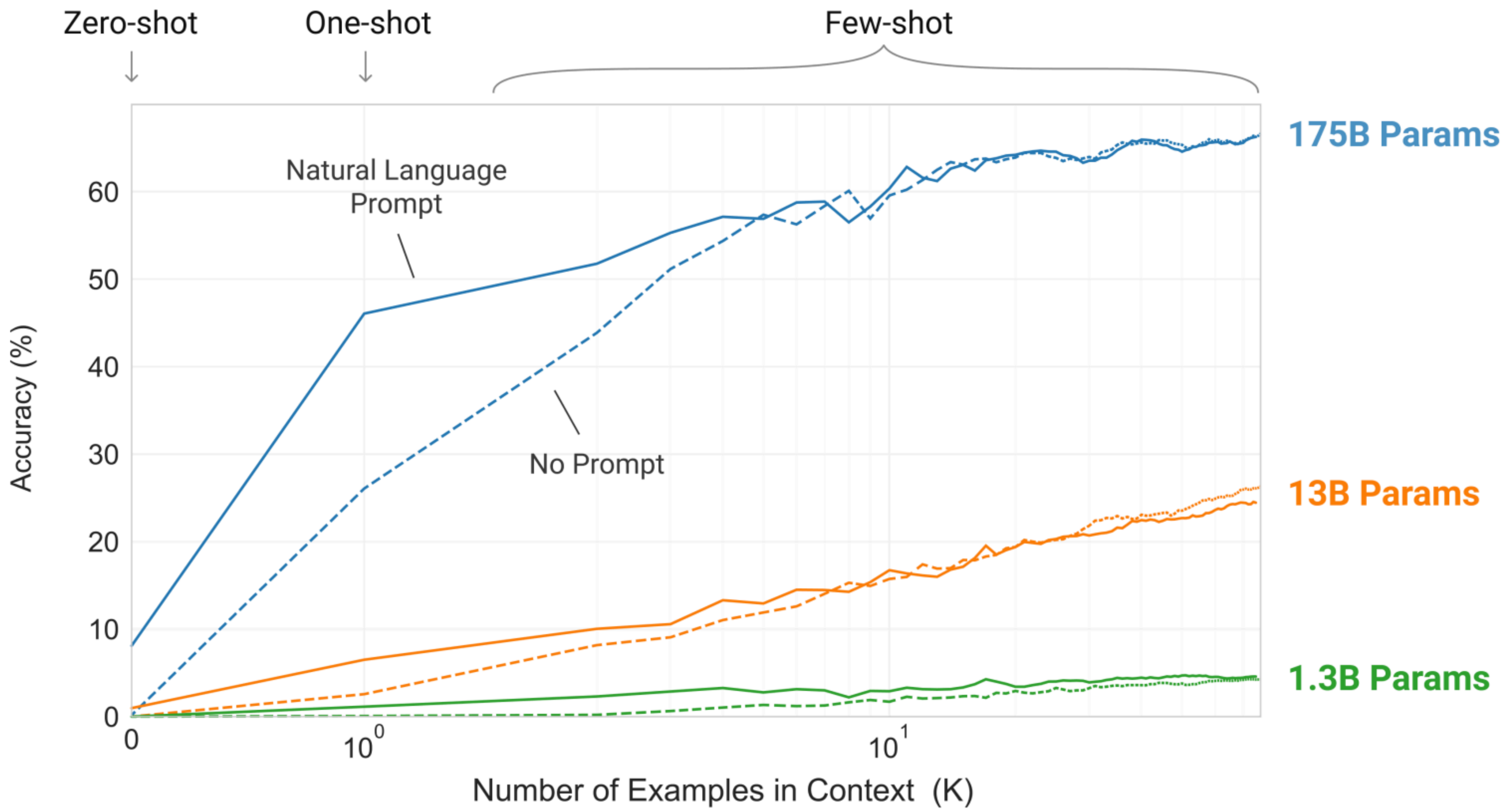
NeurIPS 2020, Vancouver, BC

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.

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

One-shot

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

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

Few-shot

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

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

Language Models are Few-Shot Learners (Brown et al. OpenAI, 2020)

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.

`Translate English to French:`	*task description*
`cheese =>`	*prompt*

One-shot

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

`Translate English to French:`	*task description*
`sea otter => loutre de mer`	*example*
`cheese =>`	*prompt*

Few-shot

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

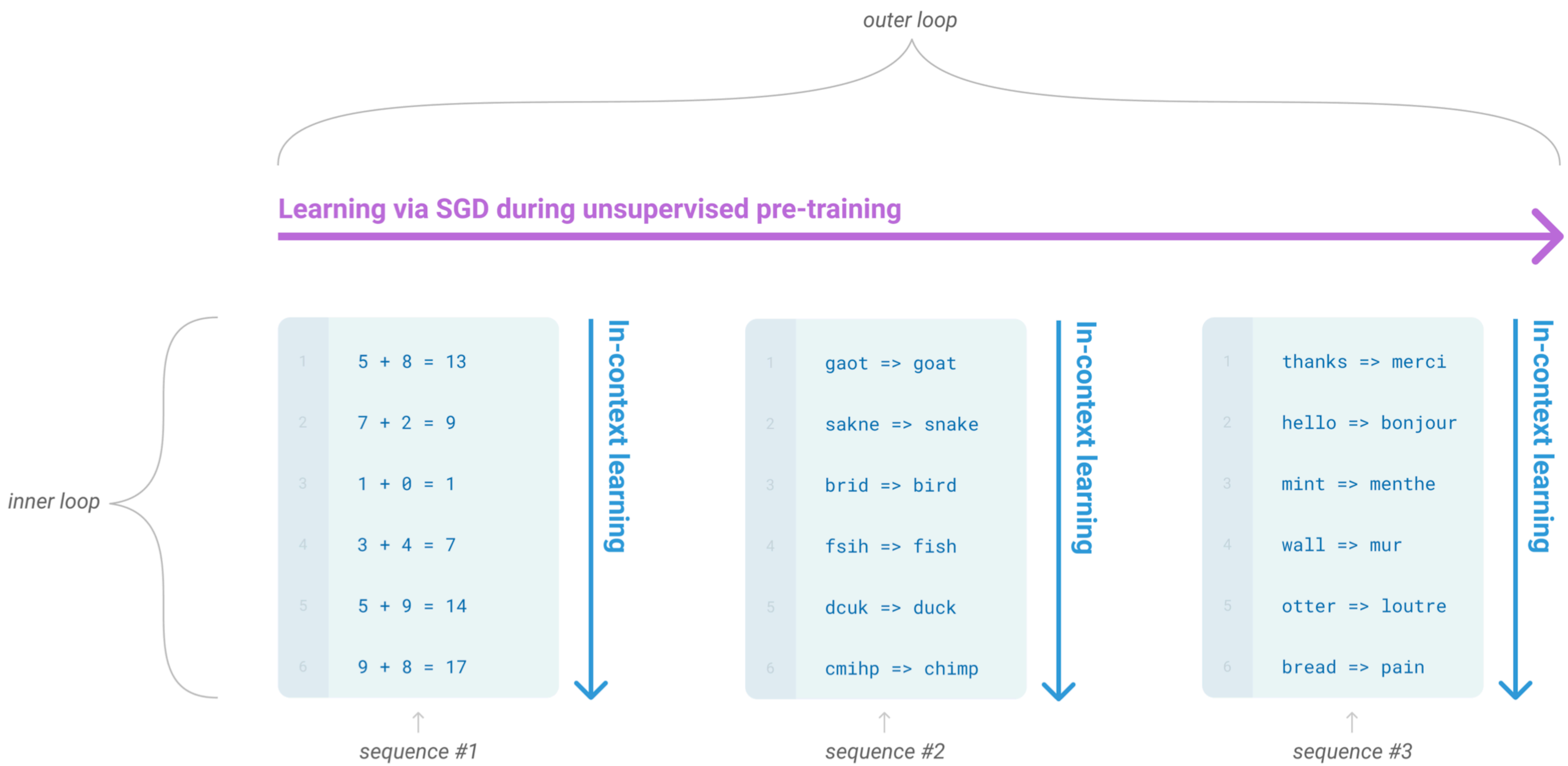
`Translate English to French:`	*task description*
`sea otter => loutre de mer`	*examples*
`peppermint => menthe poivrée`	
`plush girafe => girafe peluche`	
`cheese =>`	*prompt*

Fine-tuning for large models

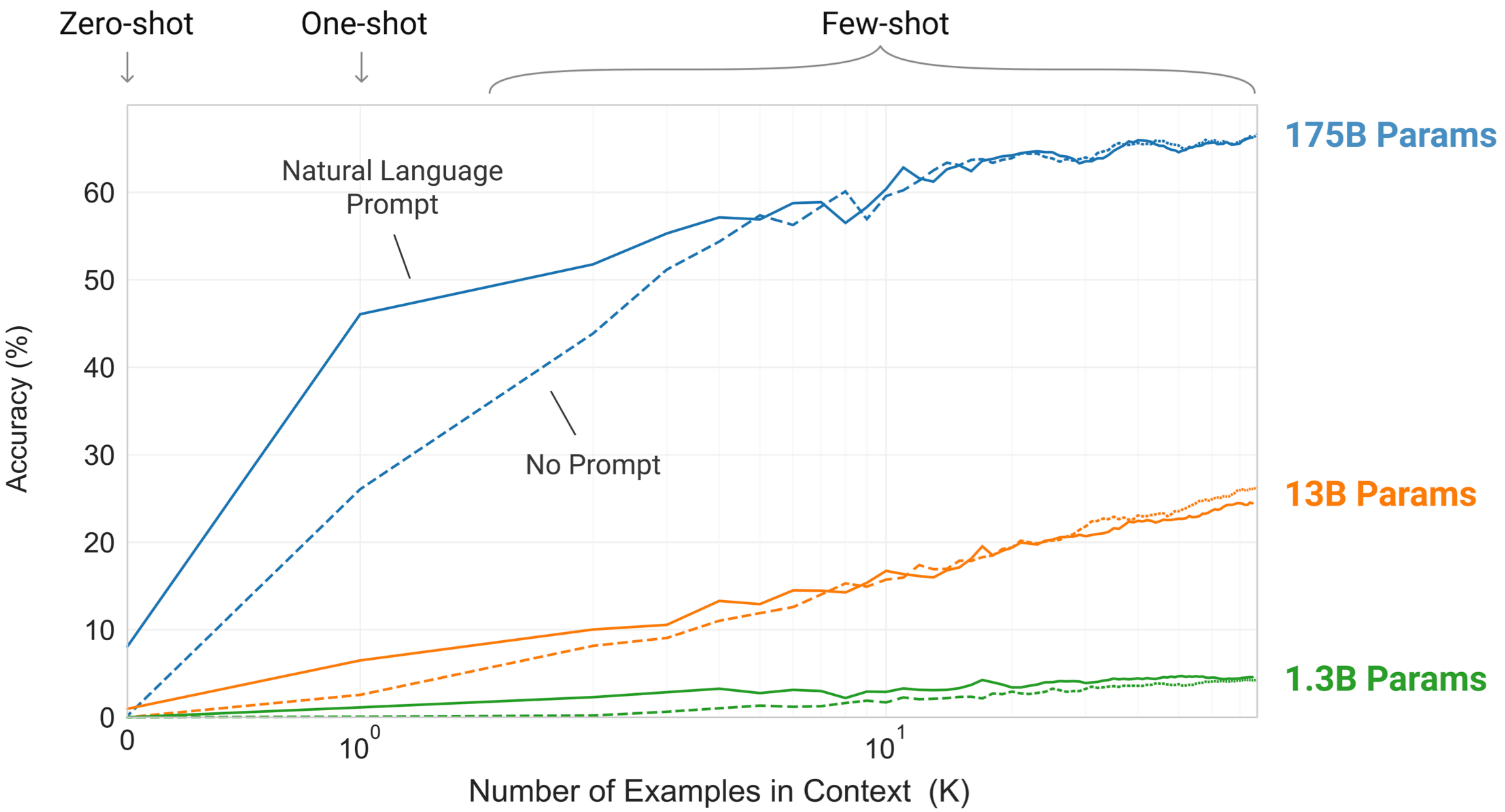
- LLMs >10B parameters are very difficult to fine-tune and requires a big compute budget

Learning via examples provided in the prompt

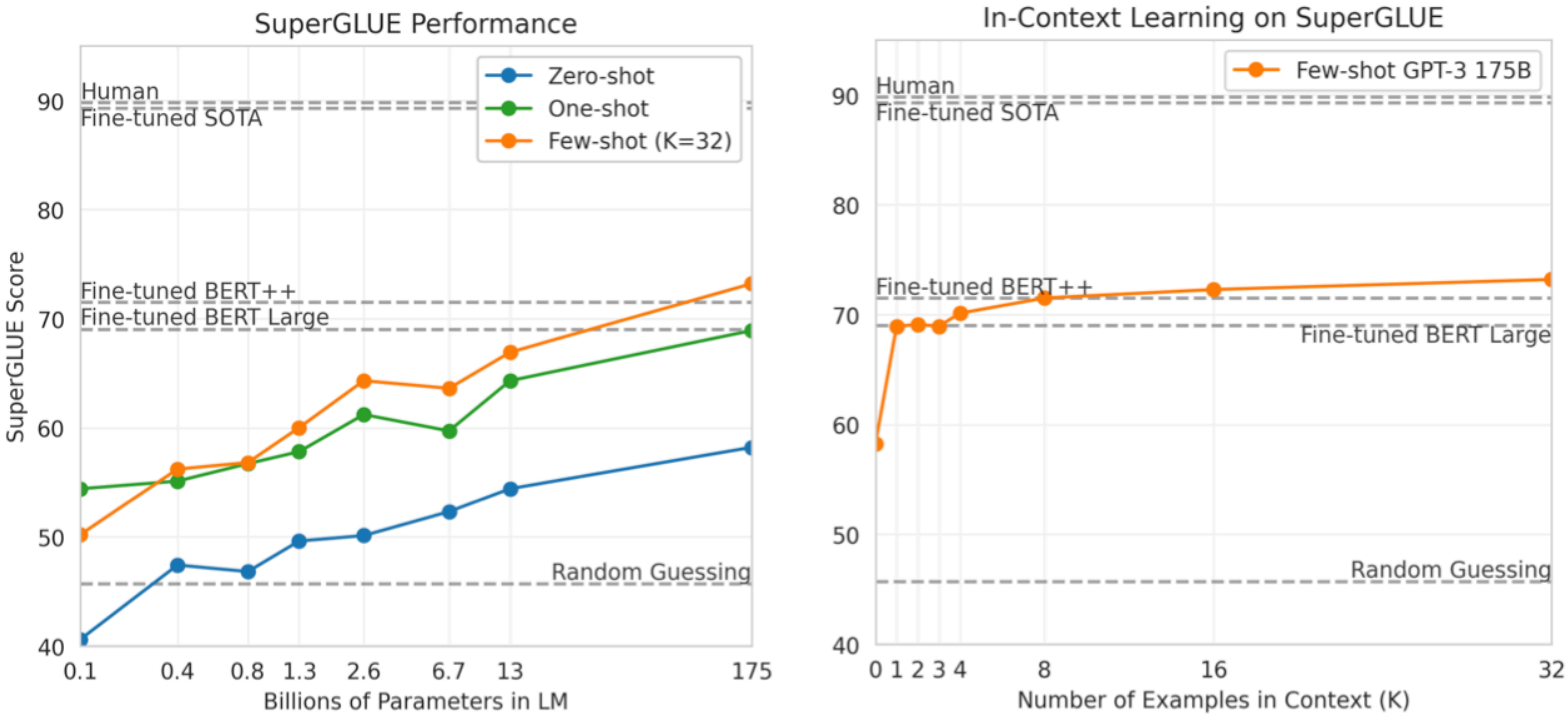
- So **in-context learning** using a long prompt or prefix is needed to coax the answer from a "predict the next token" approach to solving multiple tasks
- Pre-training on web-scale text can observe many different tasks in-context during training in the inner loop (per batch)
- Gradient descent improves 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
5 + 8 = 13
7 + 2 = 9
1 + 0 = 1
3 + 4 = 7
5 + 9 = 14
9 + 8 = 17
In-context learning
gaot => goat
sakne => snake
brid => bird
fsih => fish
dcuk => duck
cmihp => chimp
In-context learning
thanks => merci
hello => bonjour
mint => menthe
wall => mur
otter => loutre
bread => pain
In-context learning
sequence #1
sequence #2
sequence #3



Zero-shot
One-shot
Few-shot
Natural Language Prompt
No Prompt
175B Params
13B Params
1.3B Params
Accuracy (%)
0
10
20
30
40
50
60
0
10^0
10^1
Number of Examples in Context (K)



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 strategies

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 with **roles** and **templates**
  - Prompting for reasoning: **chain-of-thought**
  - Prompting to **reflect** to check **consistency** and **correctness**
  - Prompting with **multiple steps**: breaking it down
  - 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

Example Credit: CMU CS11-711, Graham Neubig

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." },
  ]
```

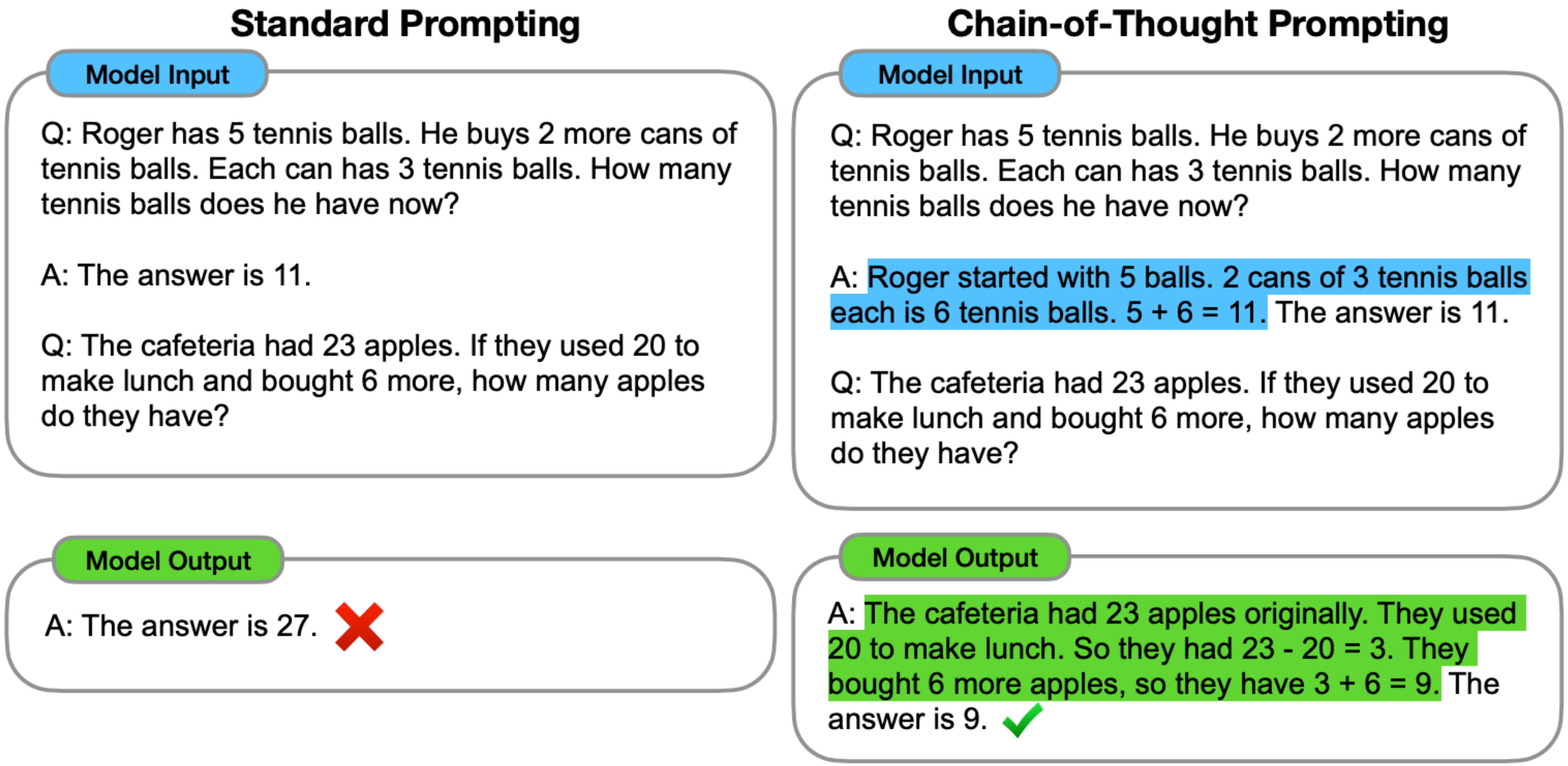
Roles:
"system": message provided to the system to influence behavior
"user": message input by the user
"assistant": message output by the system

Example Credit: CMU CS11-711, Graham Neubig

Chain of thought prompting

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

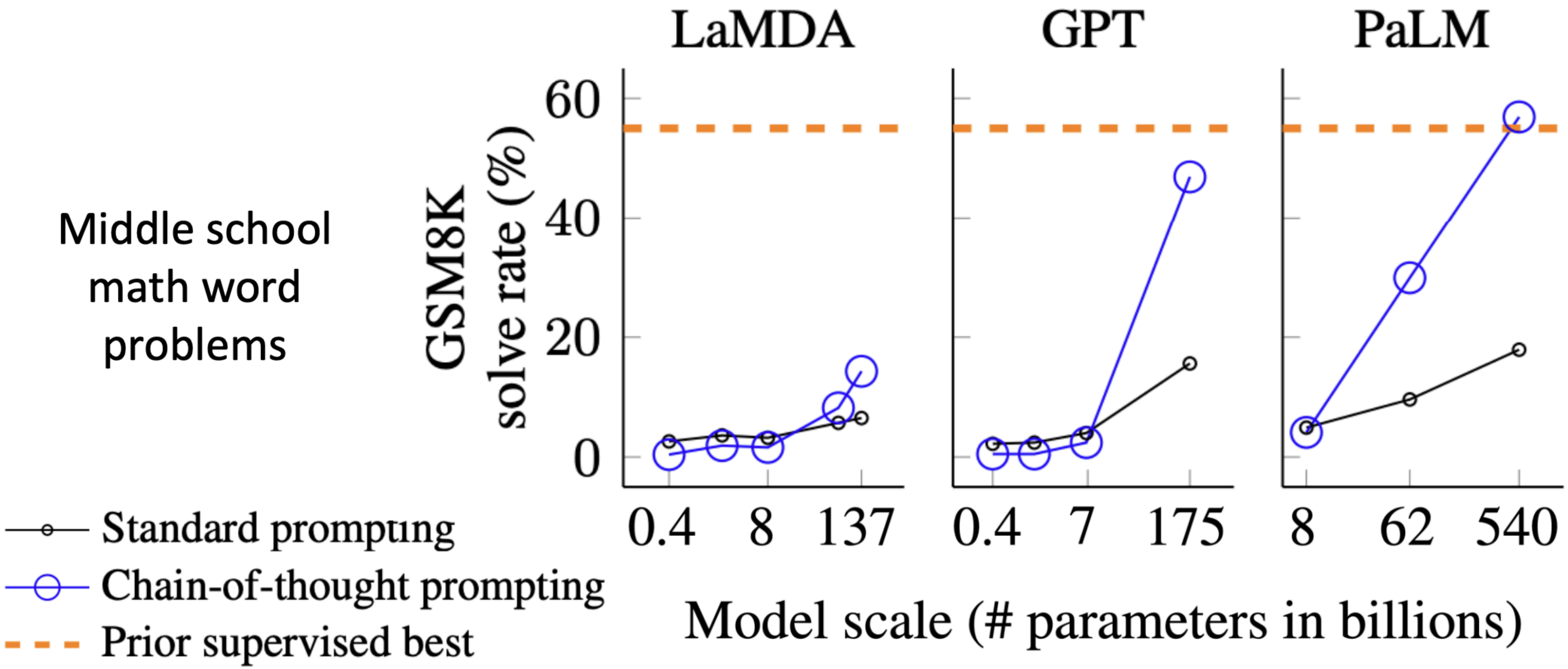
- How you prompt matters.
- For more complex problems, may need to provide prompts that illustrate the reasoning you expect



Chain of thought prompting

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

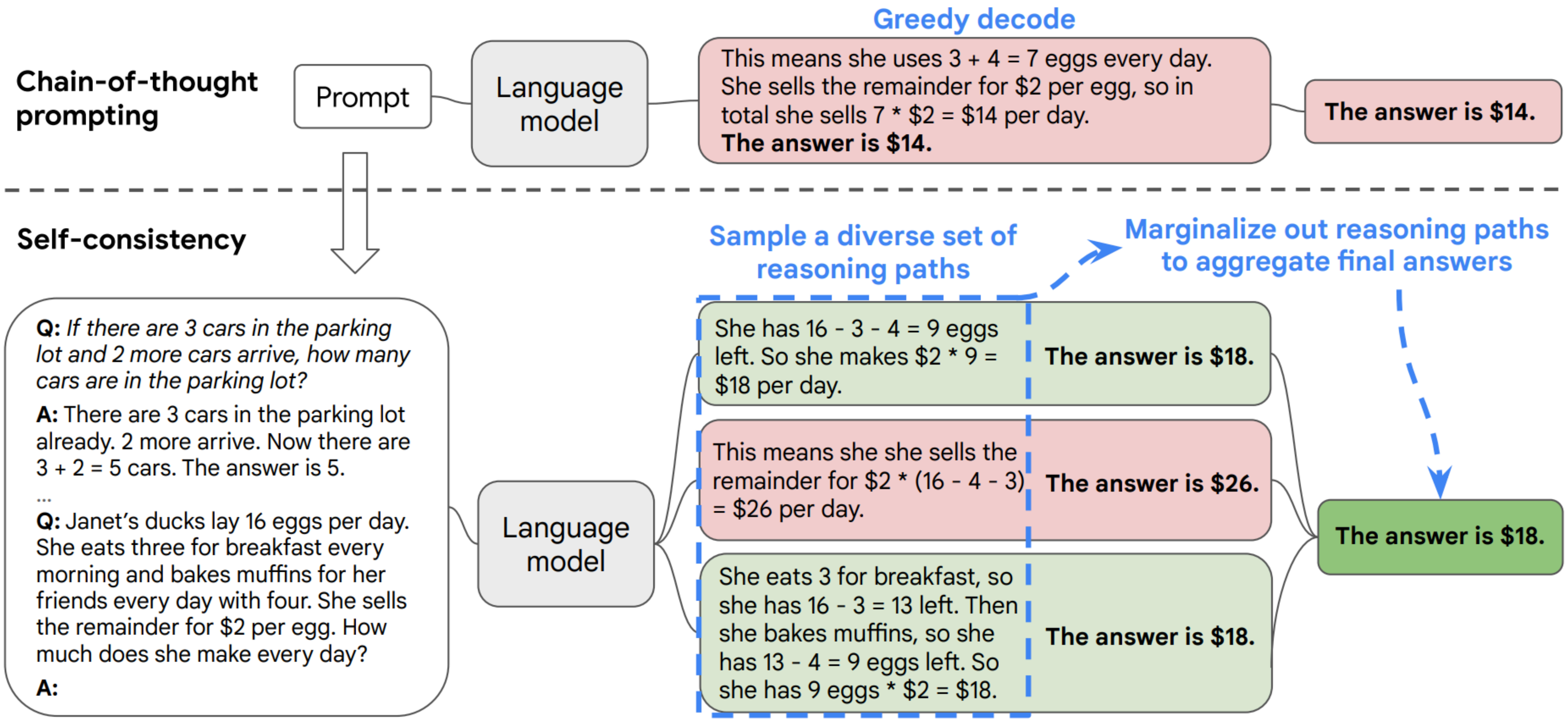
Middle school math word problems



Self-consistency

Self-Consistency Improves Chain of Thought Reasoning in Language Models [Wang et el. 2023]

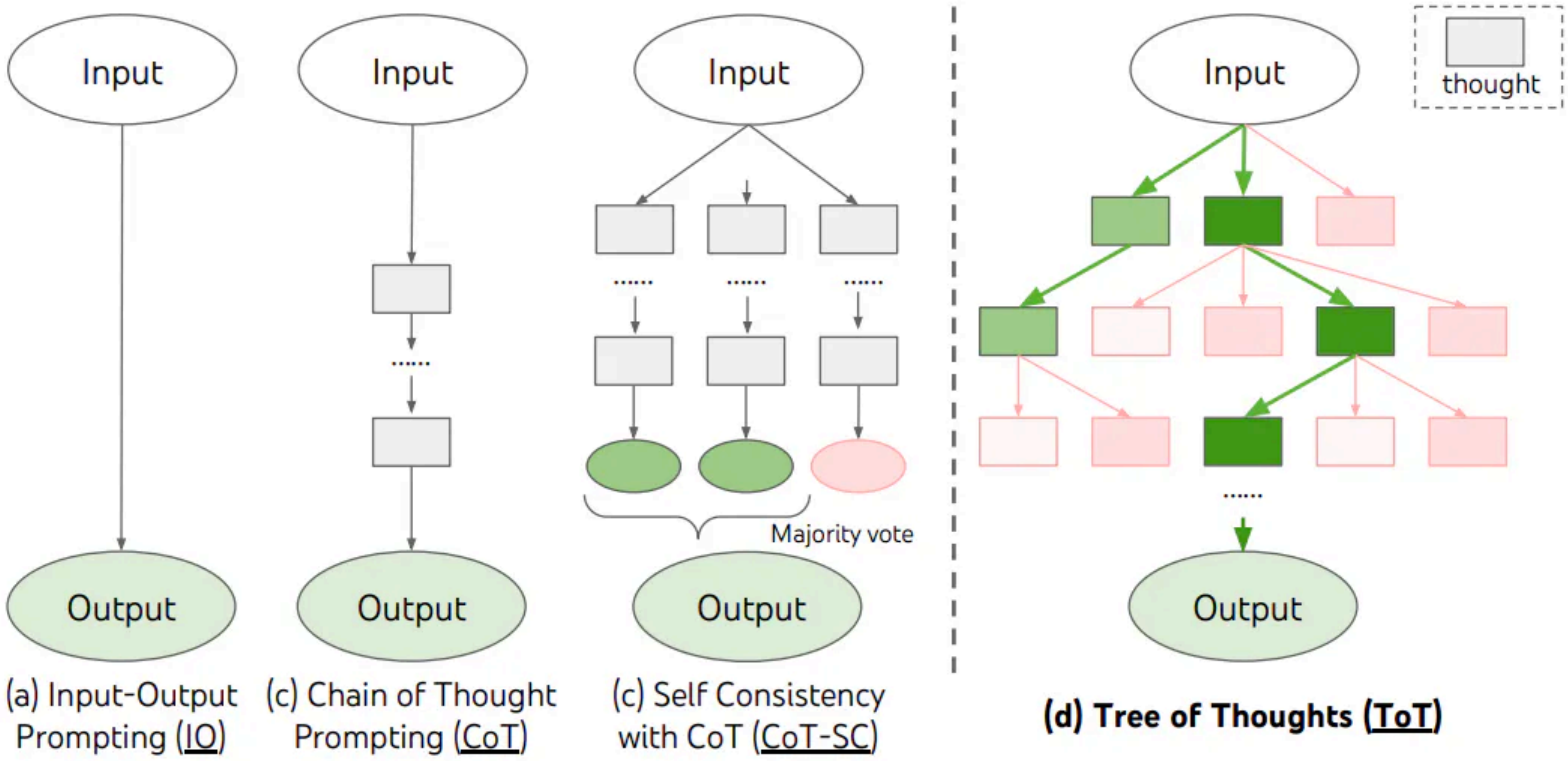
- Sample multiple paths
- Select solution based on majority vote



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



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)

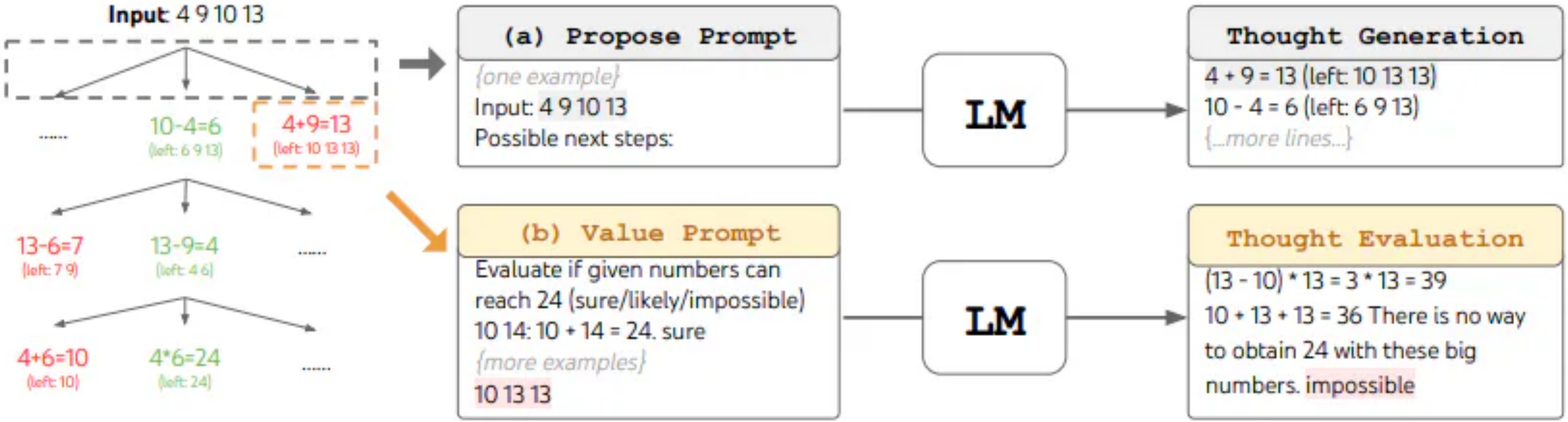
	Game of 24	Creative Writing	5x5 Crosswords
Input	4 numbers (4 9 10 13)	4 random sentences	10 clues (h1. presented;..)
Output	An equation to reach 24 (13-9)*(10-4)=24	A passage of 4 paragraphs ending in the 4 sentences	5x5 letters: SHOWN; WIRRA; AVAIL; ...
Thoughts	3 intermediate equations (13-9=4 (left 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.

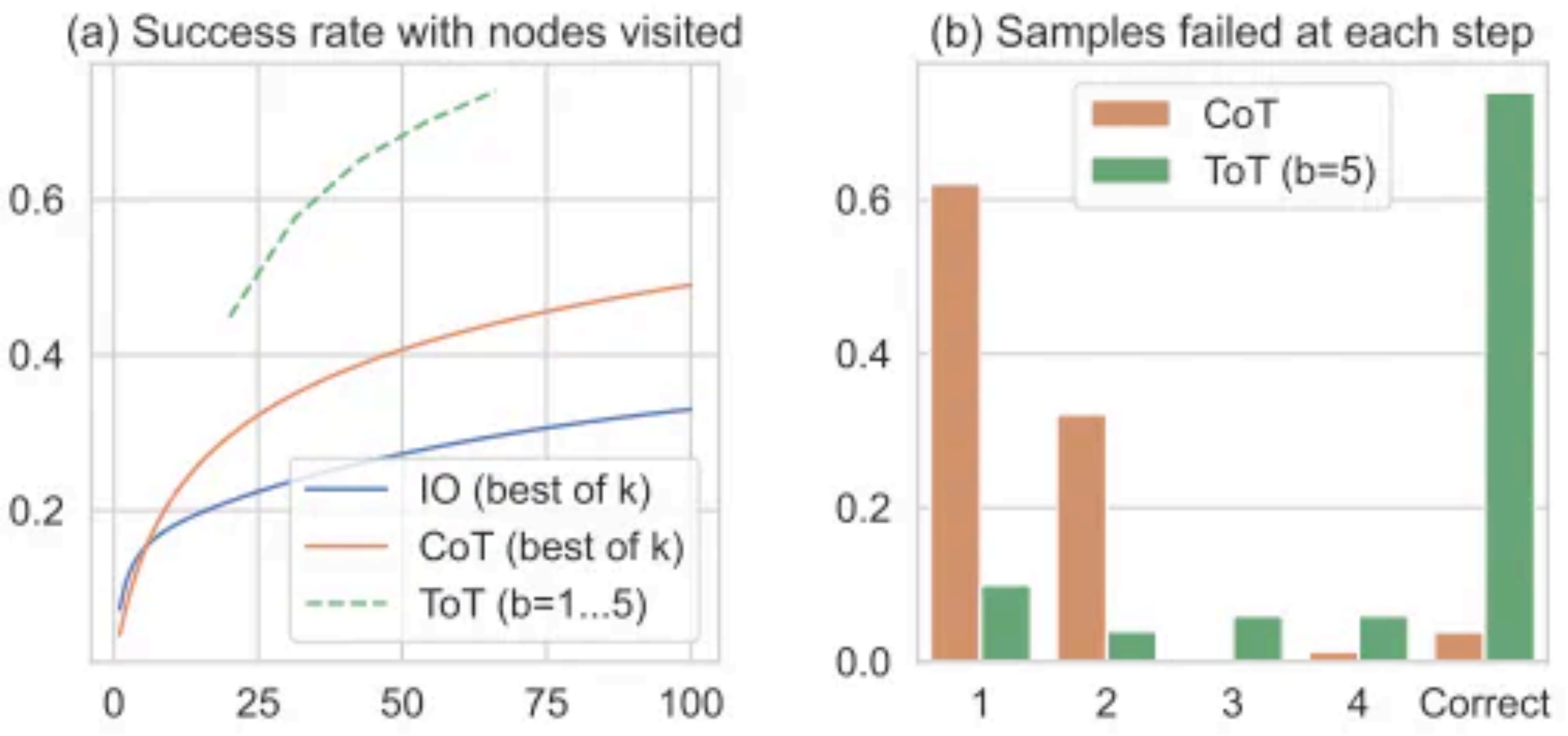


Figure 3: Game of 24 (a) scale analysis & (b) error analysis.

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 or ability to use tools (agents)

Self-Reflection

- ReAct: interleave thought, action, and observation

- Prompting to combine reasoning with actions

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports',' Front Row (software) ', …]

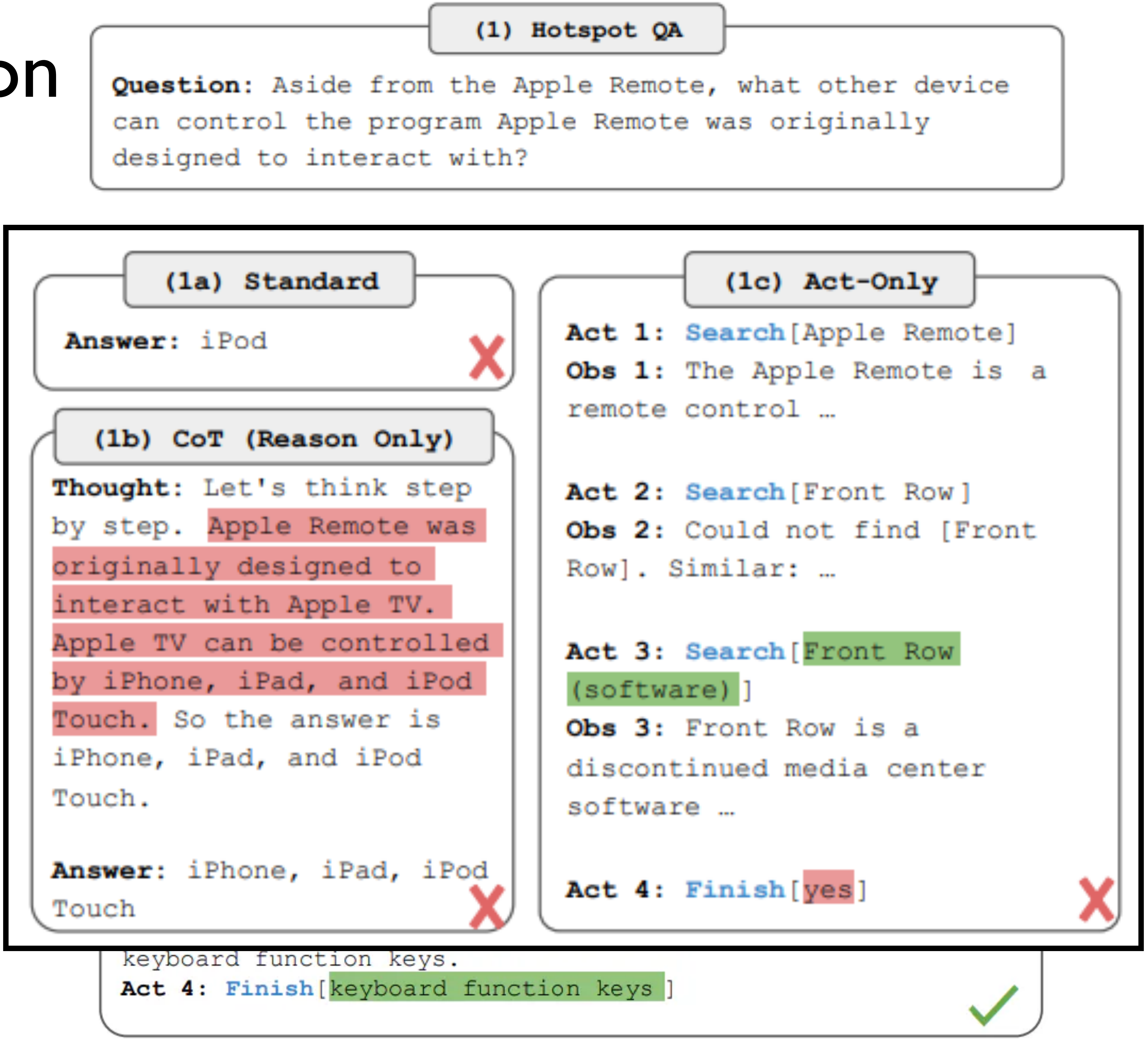
Thought 3: Front Row is not found. I need to search Front Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.
Act 4: Finish[keyboard function keys]

ReAct: Synergizing Reasoning and Acting in Language Models [Yao et al. 2022]

Self-Reflection

- Comparison with other ways of prompting



ReAct: Synergizing Reasoning and Acting in Language Models [Yao et al. 2022]

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)

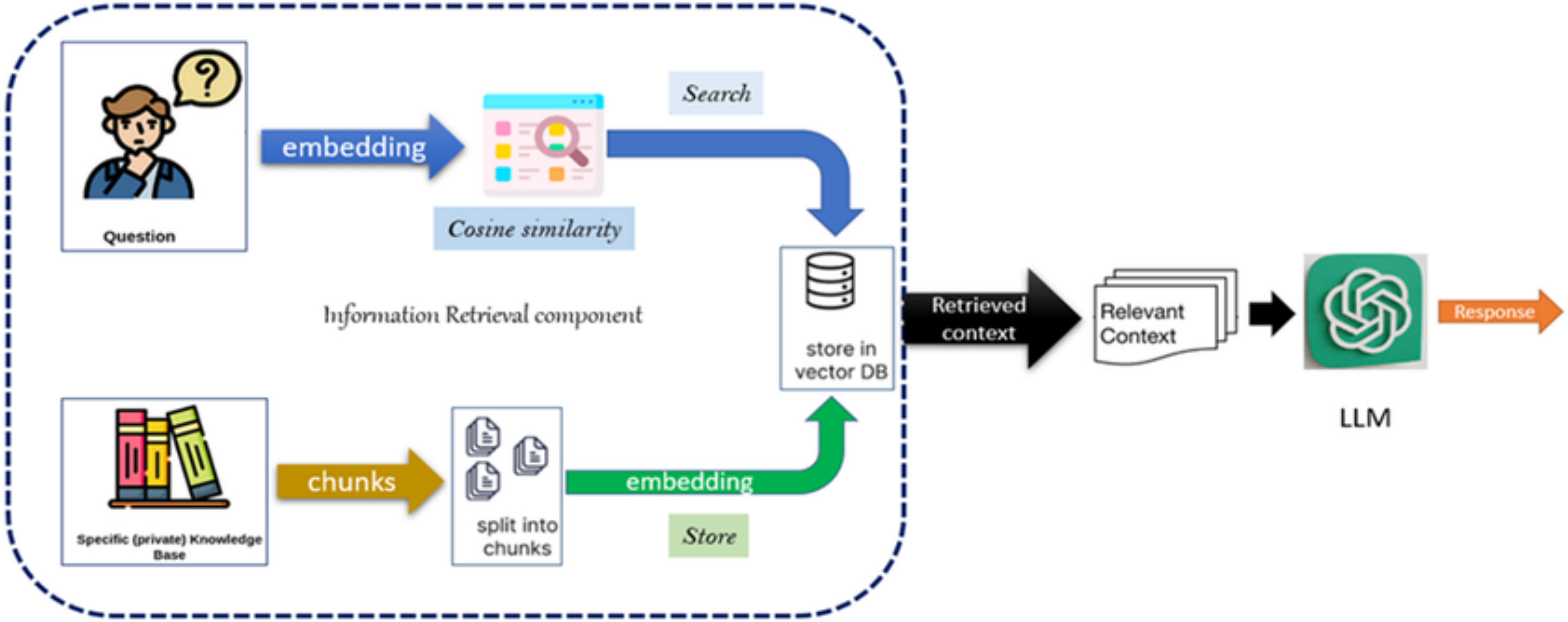
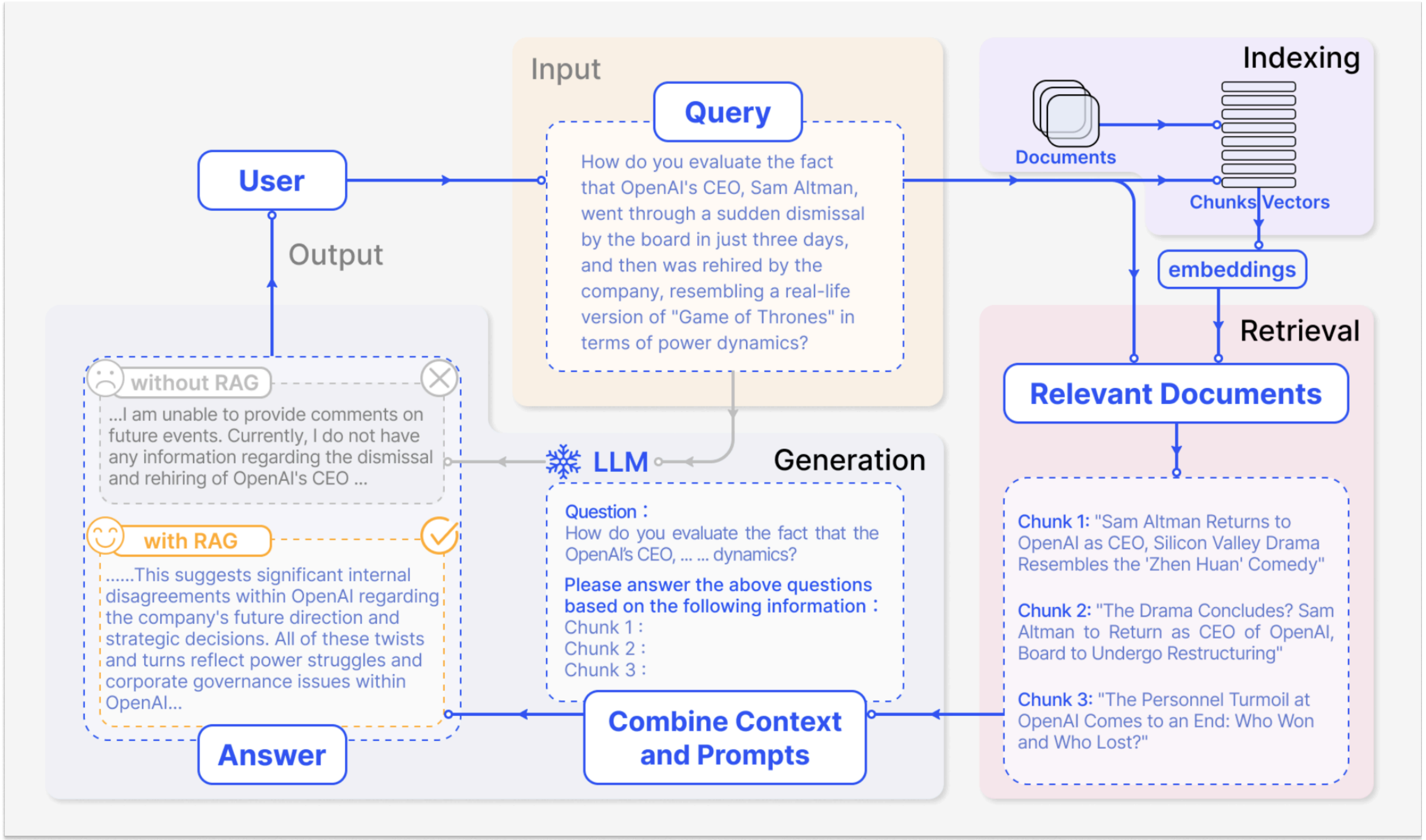


Figure credit: Semantic Embeddings for Arabic Retrieval Augmented Generation (ARAG) [Abdelazim et al. 2023]

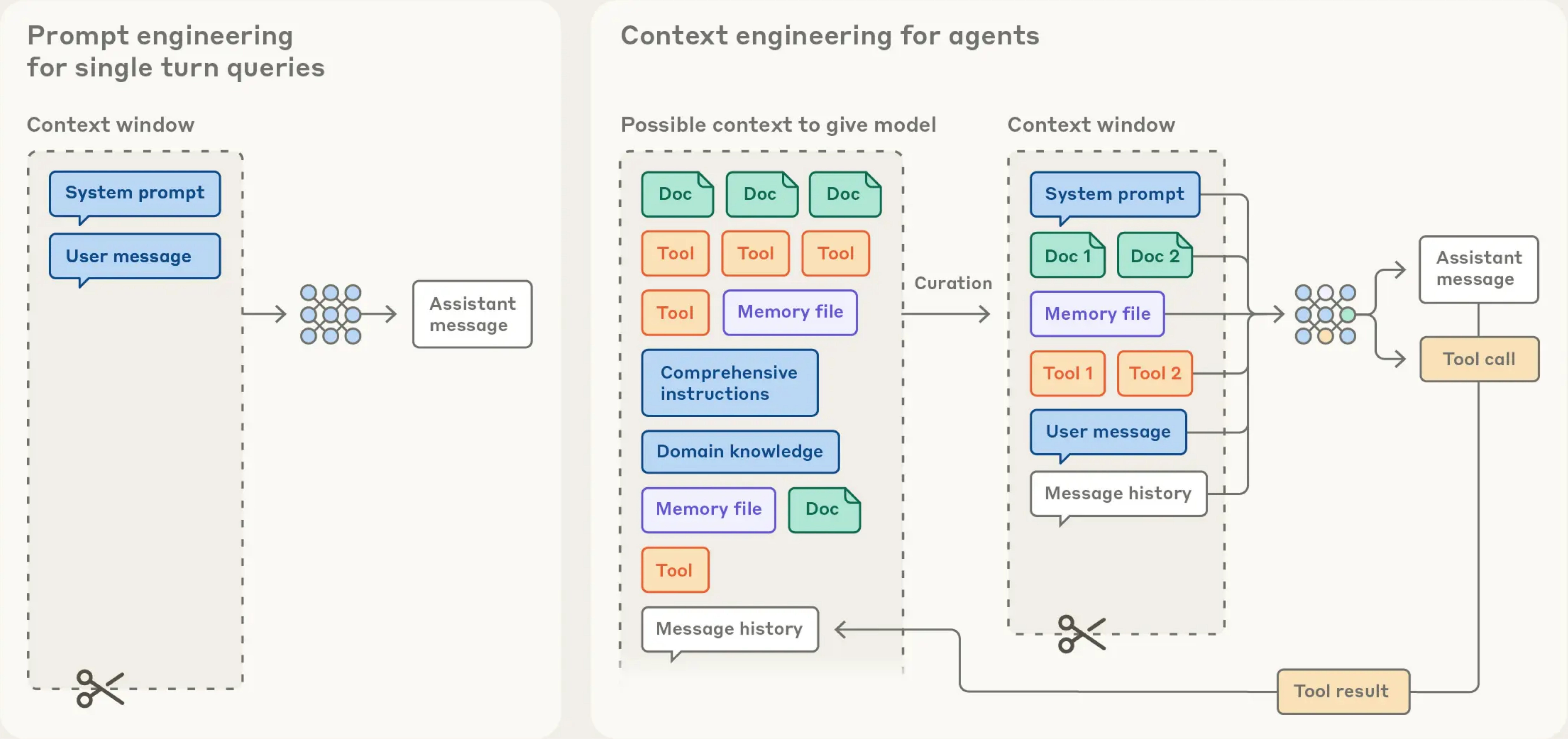
Retrieval-augmented generation

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



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

Prompt engineering vs. context engineering



https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents

Prompt Engineering Guides

- https://www.promptingguide.ai/
- OpenAI: https://developers.openai.com/api/docs/guides/prompt-engineering/
- Google: https://cloud.google.com/discover/what-is-prompt-engineering
- Anthropic (Claude): https://platform.claude.com/docs/en/build-with-claude/prompt-engineering/overview

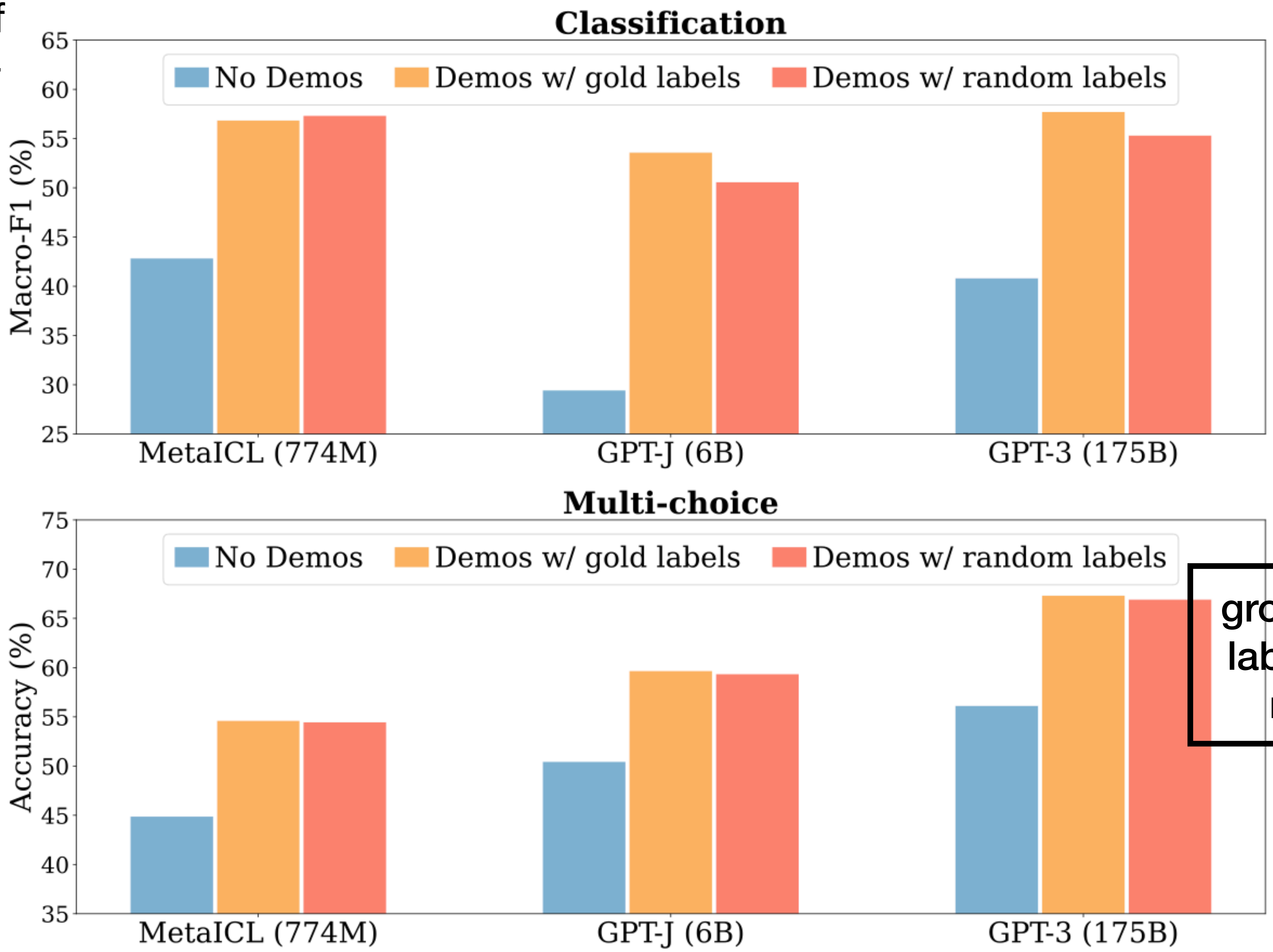
Why does in-context learning work?

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

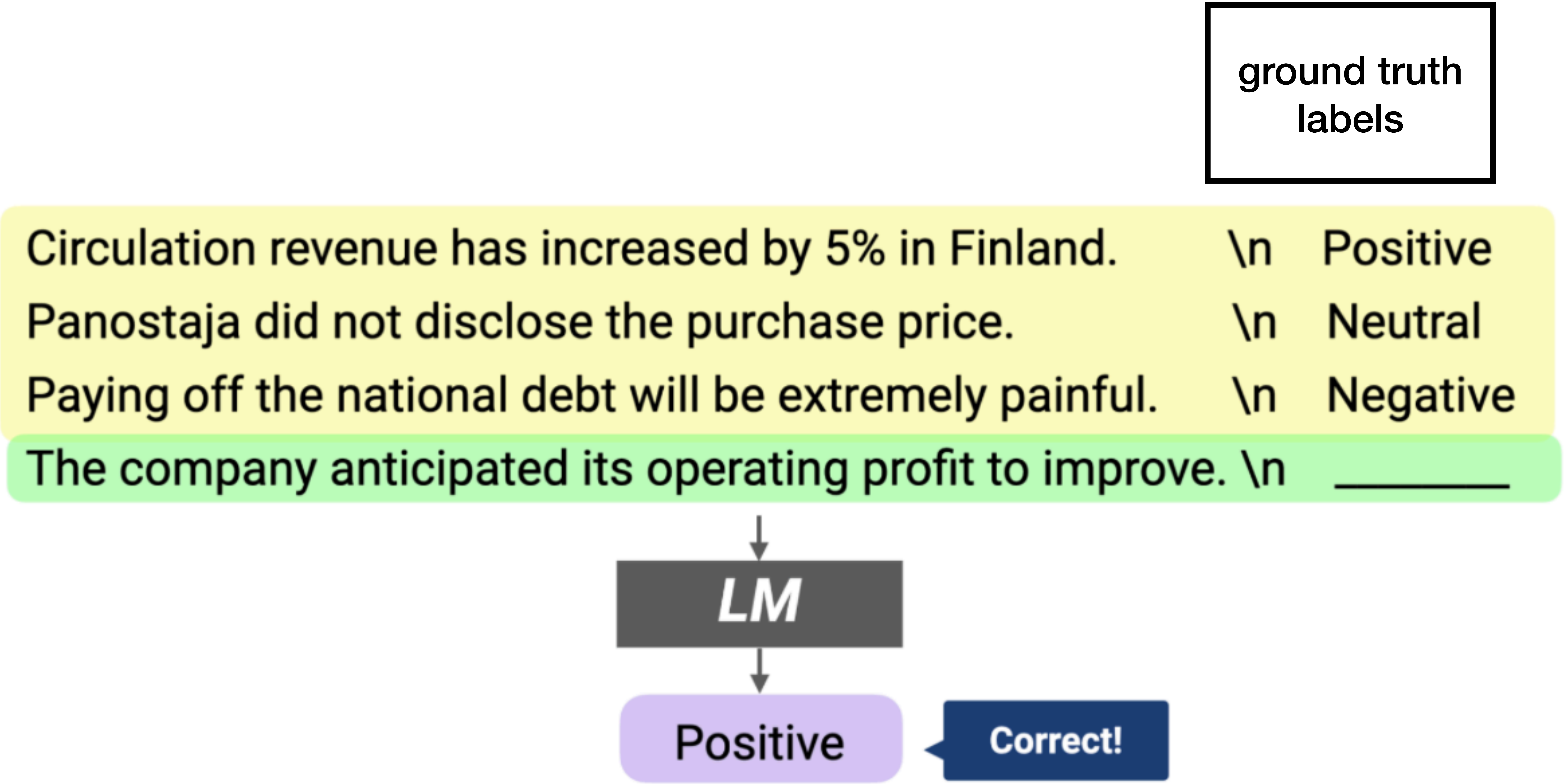
Sewon Min[1,2] **Xinxi Lyu**[1] **Ari Holtzman**[1] **Mikel Artetxe**[2]
Mike Lewis[2] **Hannaneh Hajishirzi**[1,3] **Luke Zettlemoyer**[1,2]
[1]University of Washington [2]Meta AI [3]Allen Institute for AI

https://arxiv.org/abs/2202.12837

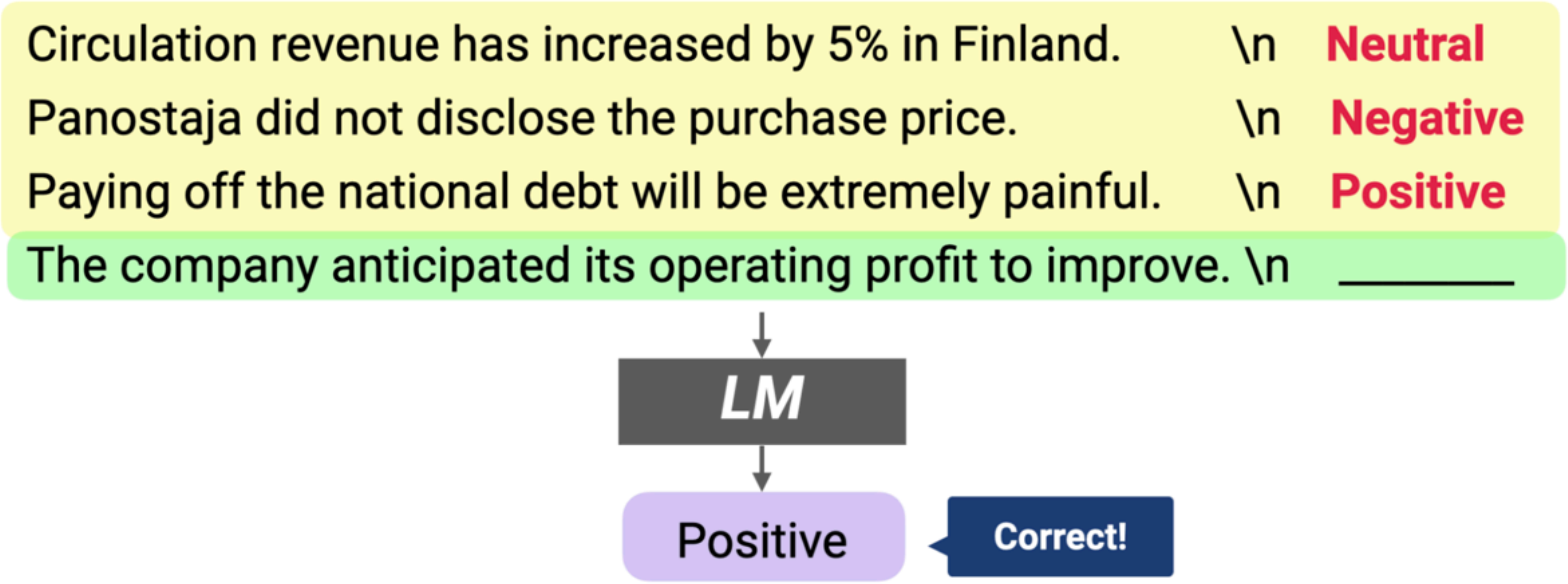
What happens if we provide few-shot examples with random labels?



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 ________
LM
Positive
Correct!

replace true labels with random labels



Why does in-context learning work?

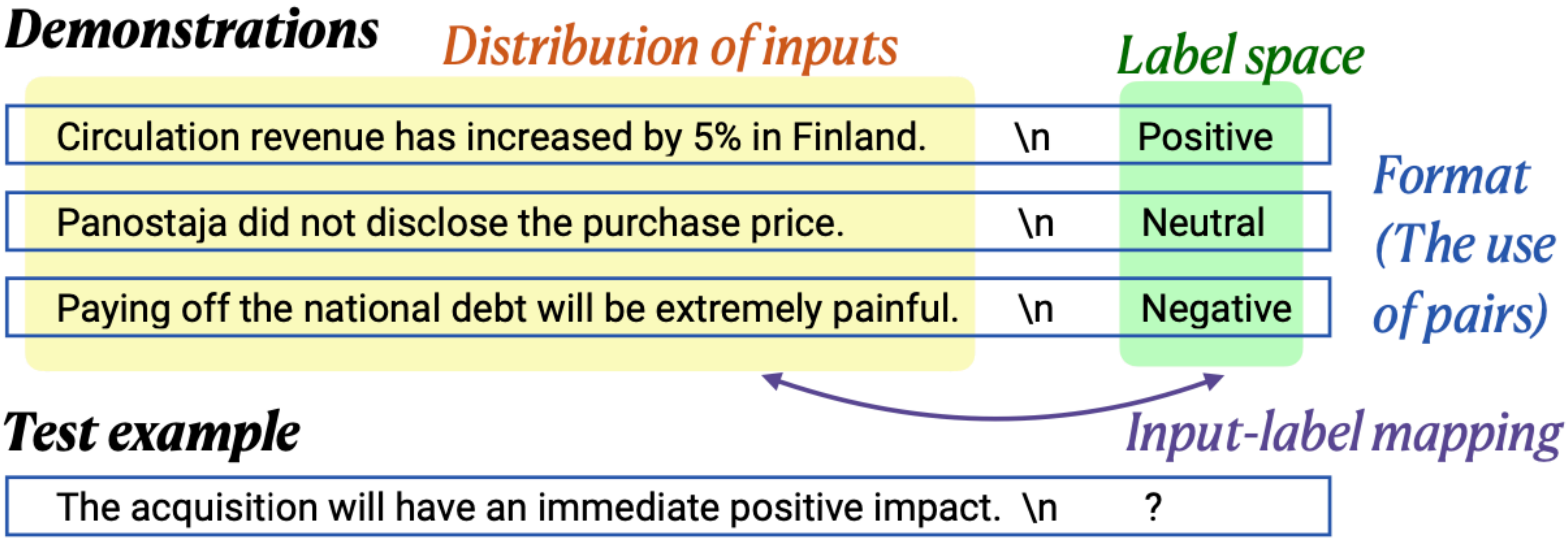
Four hypotheses

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 $x_1, \ldots, x_k$ are from (is it from a sports article, or business news?)

3. The **output label space** $y_1, \ldots, y_k$

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
Input-label mapping
The acquisition will have an immediate positive impact. \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 $x_1, \ldots, x_k$ are from (is it from a sports article, or business news?)

3. The **output label space** $y_1, \ldots, y_k$

4. The **format of the demonstration**, e.g. `x // y`; `Input: x Output: y`; etc.

Why does in-context learning work?

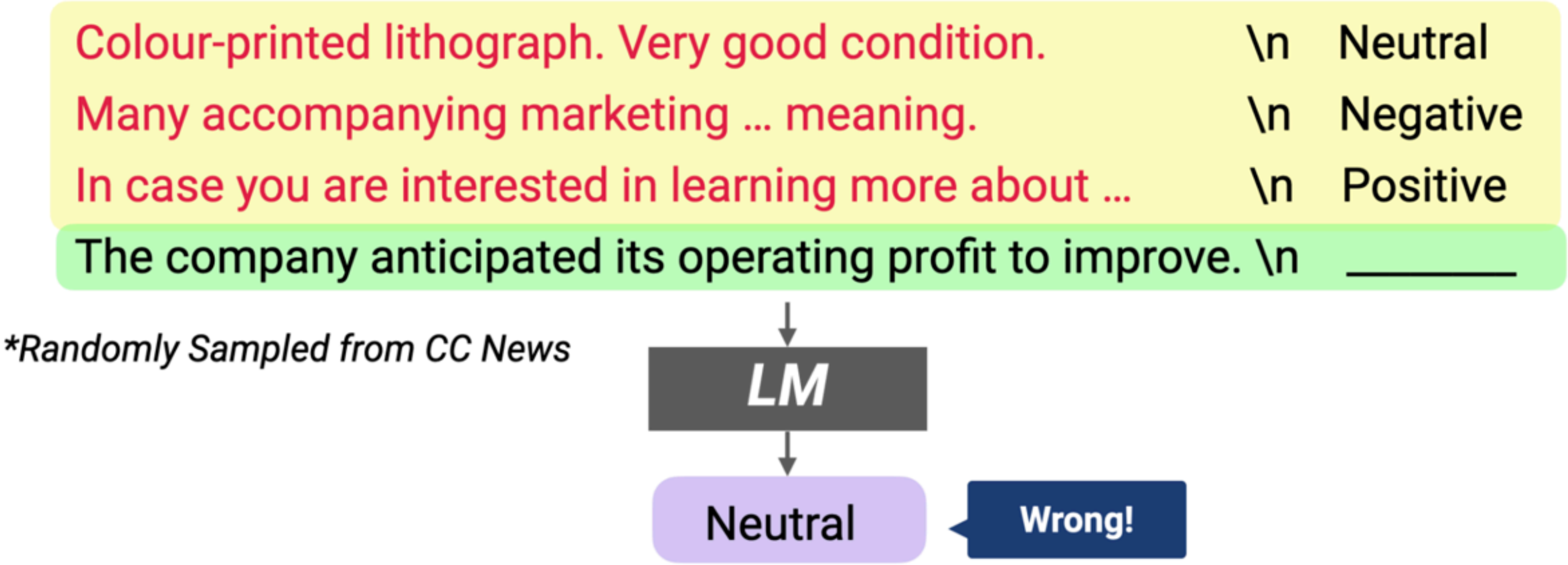
Four hypotheses

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 $x_1, \ldots, x_k$ are from (is it from a sports article, or business news?)

3. The **output label space** $y_1, \ldots, y_k$

4. The **format of the demonstration**, e.g. `x // y`; `Input: x Output: y`; etc.



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

Why does in-context learning work?

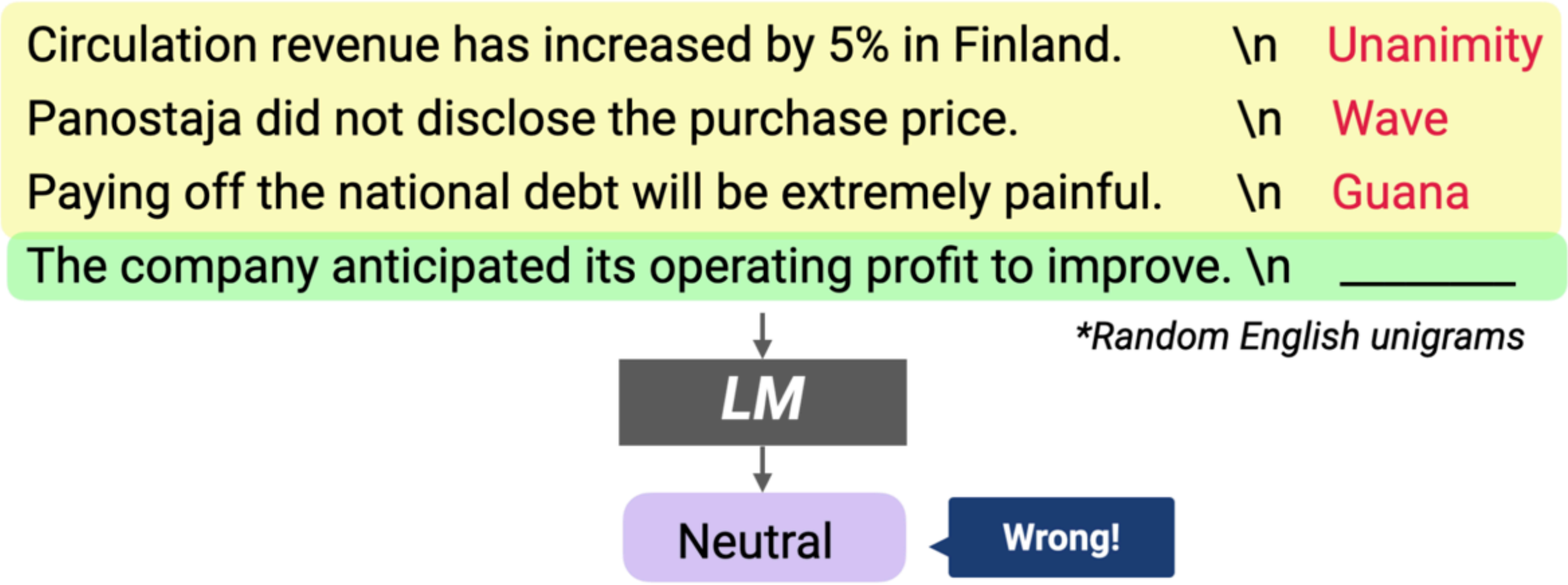
Four hypotheses

1. The **input-label mapping**, whether each input x_i is paired with the correct label y_i (not true)

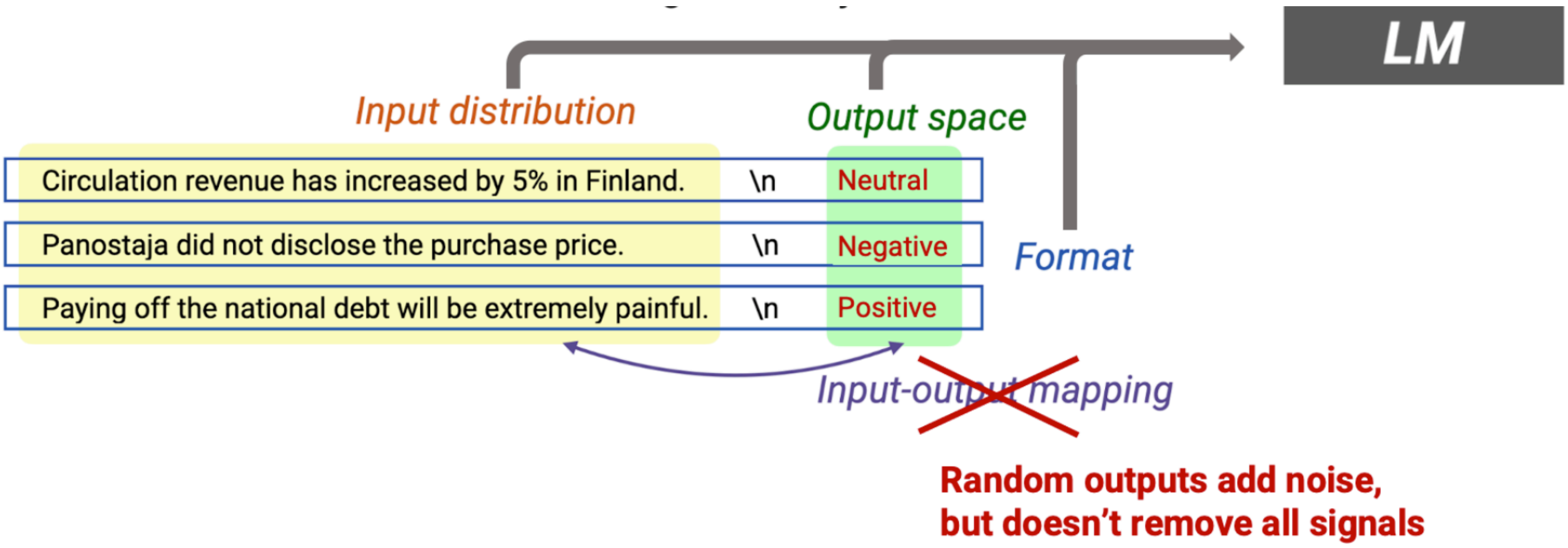
2. The **distribution** that the input $x_1, \ldots, x_k$ are from (is it from a sports article, or business news?)

3. The **output label space** $y_1, \ldots, y_k$

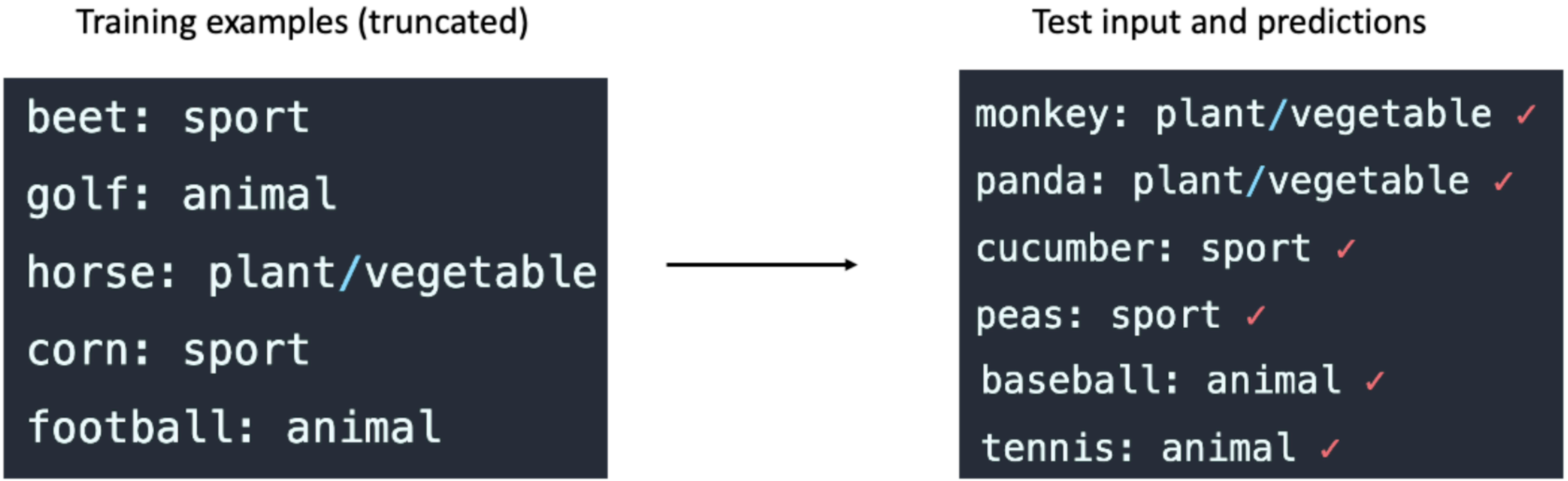
4. The **format of the demonstration**, e.g. `x // y`; `Input: x Output: y`; etc.



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



LM
Input distribution
Output space
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
Format
~~Input-output mapping~~
Random outputs add noise,
but doesn't remove all signals



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**[1△] **Tom Rainforth**[2△]

[1] OATML, Department of Computer Science, University of Oxford
[2] Department of Statistics, University of Oxford

https://arxiv.org/pdf/2307.12375

Opposite conclusion from Min et al. 2022!

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

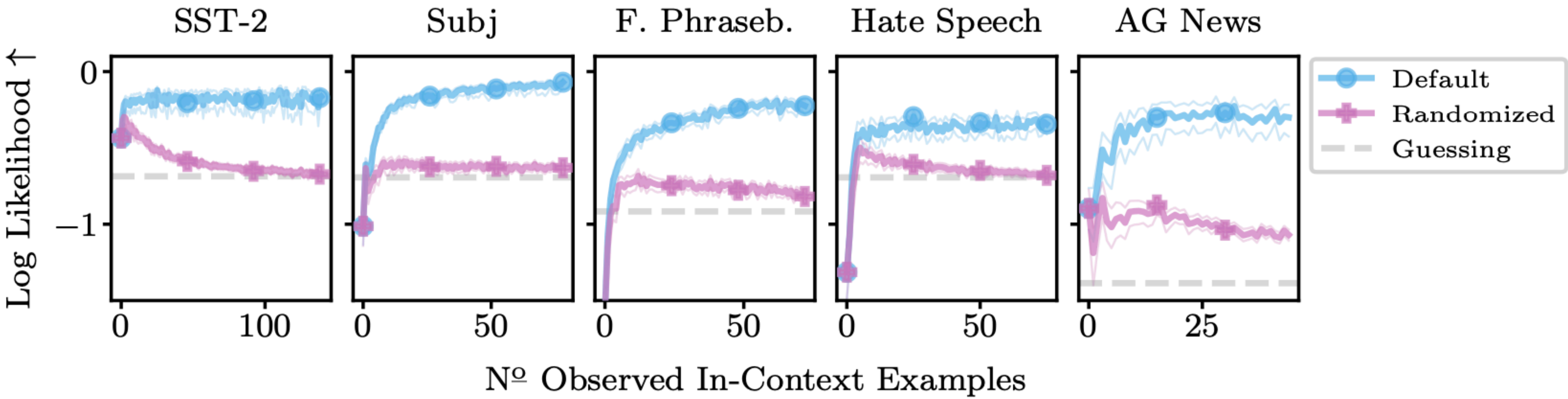


Figure 1: ICL predictions generally depend on the conditional label distribution of in-context examples: when in-context labels are randomized, average log likelihoods of label predictions decrease compared to ICL with default labels for LLaMa-2-70B across a variety of tasks. Results averaged over 500 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**

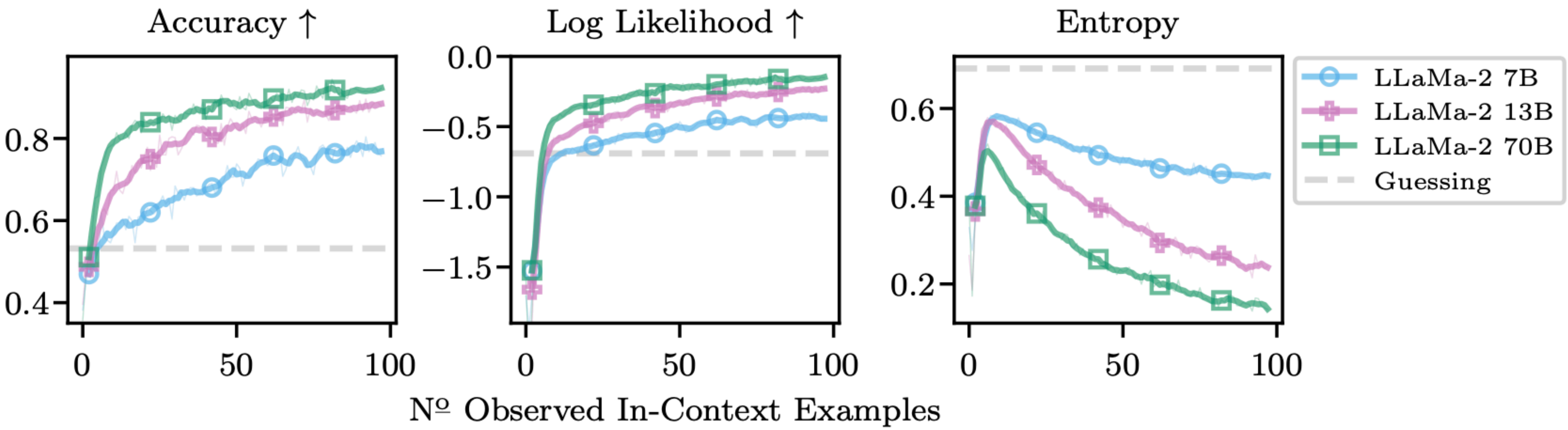
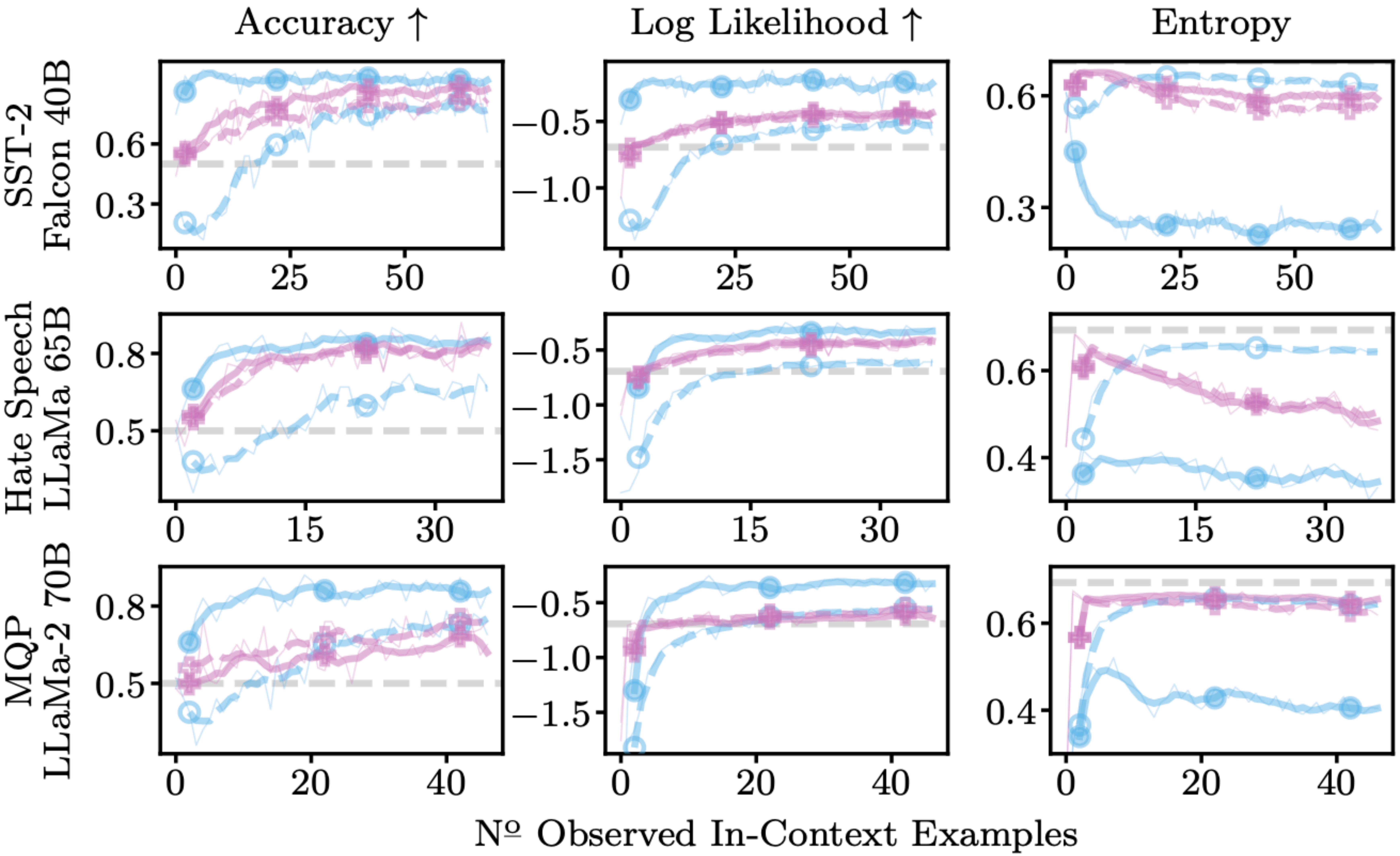


Figure 4: Few-shot ICL achieves accuracies significantly better than random guessing on our **novel author identification** task. Thus, LLMs can learn novel label relationships entirely in-context. Averages over 500 runs, thick lines with additional moving average (window size 5) for clarity.



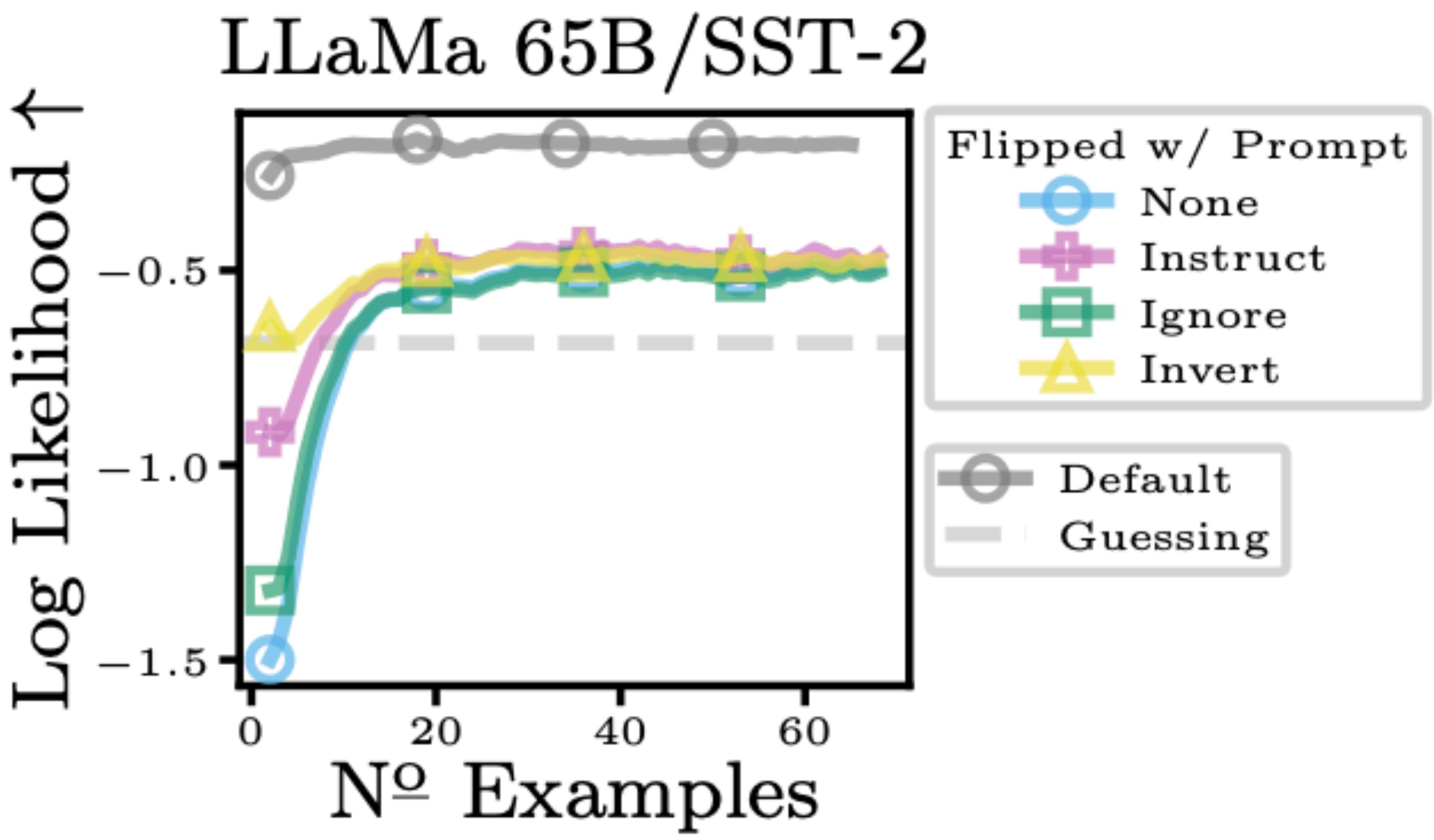


Fig. 6: Prompts yield limited gains.

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).

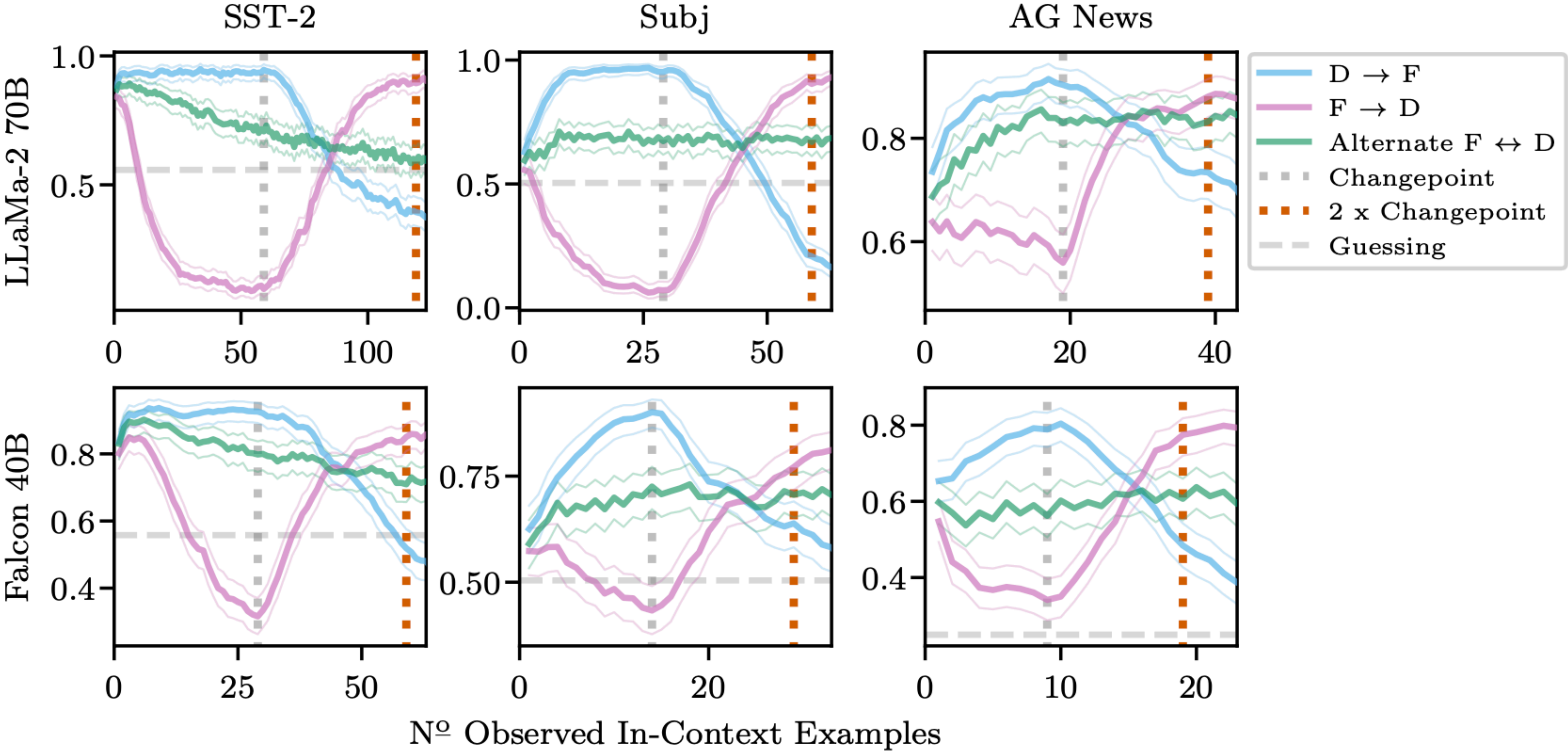


Figure 7: Few-shot ICL accuracies when the **label relationship changes throughout ICL**. For (D → F), we start with <u>d</u>efault labels and change to <u>f</u>lipped labels at the changepoint, for (F → D) we change from flipped to the default labels at the changepoint, and for (Alternating F ↔ D) we 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:

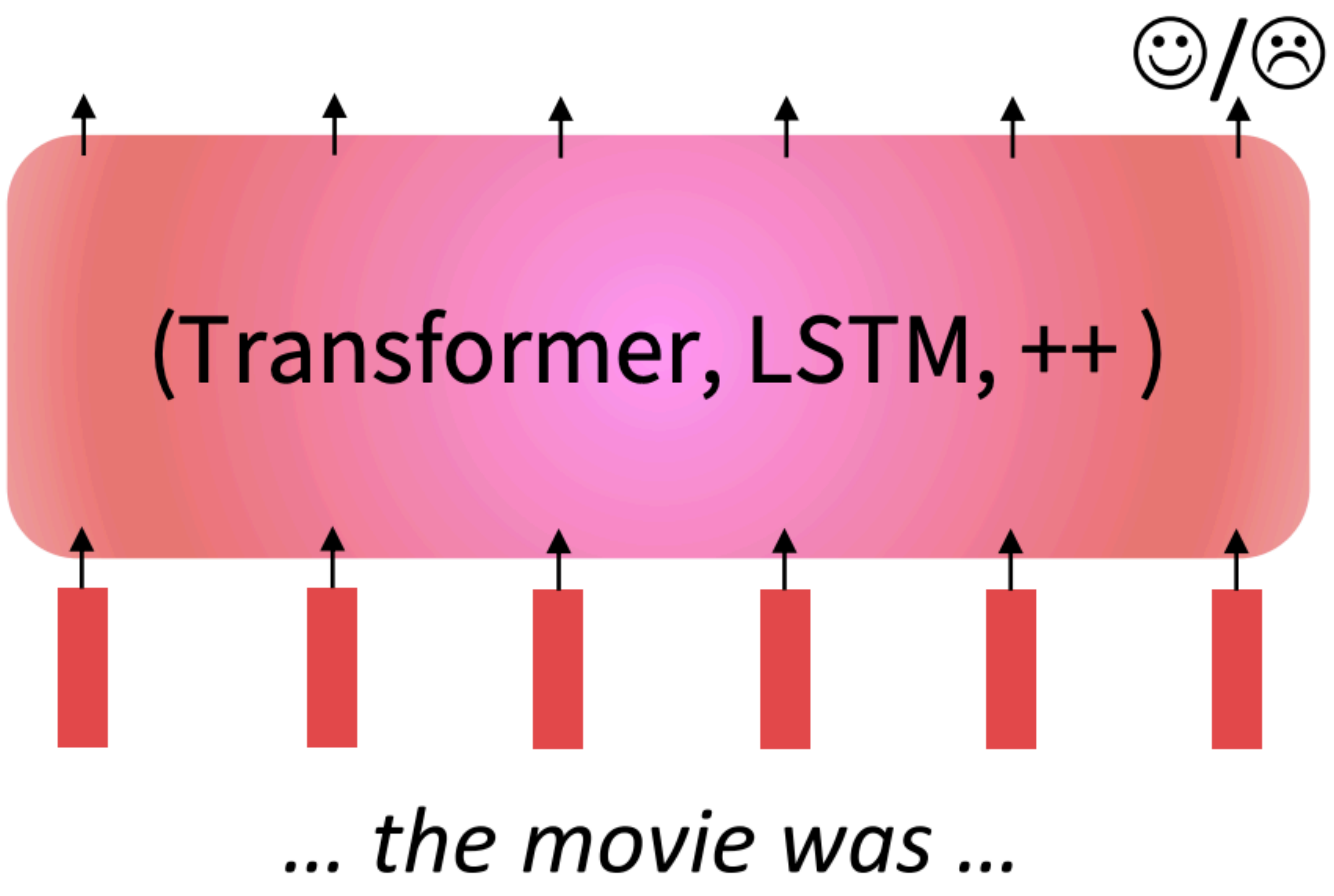
- Instruction tuning (fine-tune with instructions)
- Reinforcement learning with human feedback (train with modified objective that incorporates human preferences)

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**.

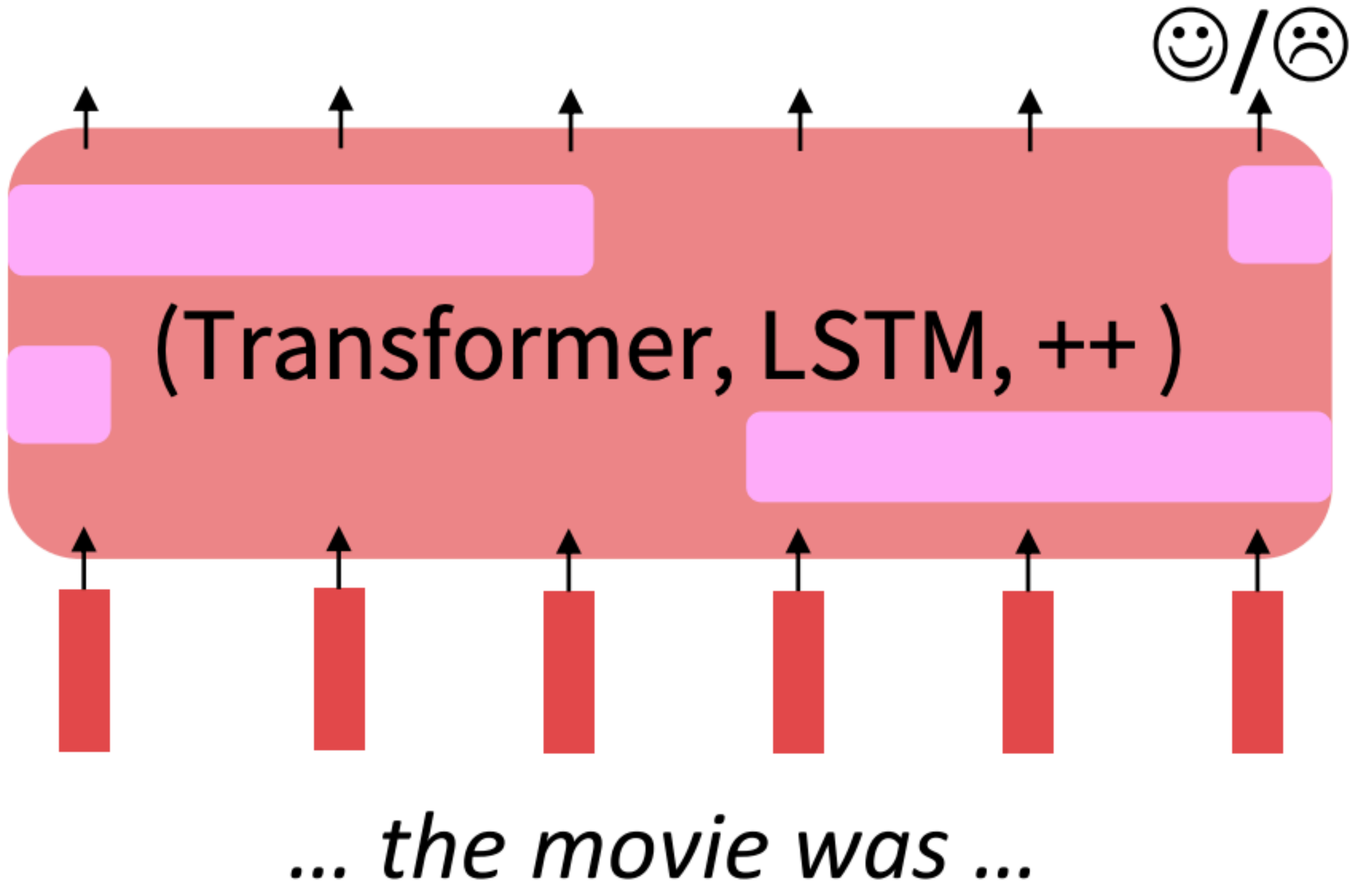
Full Finetuning

Adapt all parameters



Lightweight Finetuning

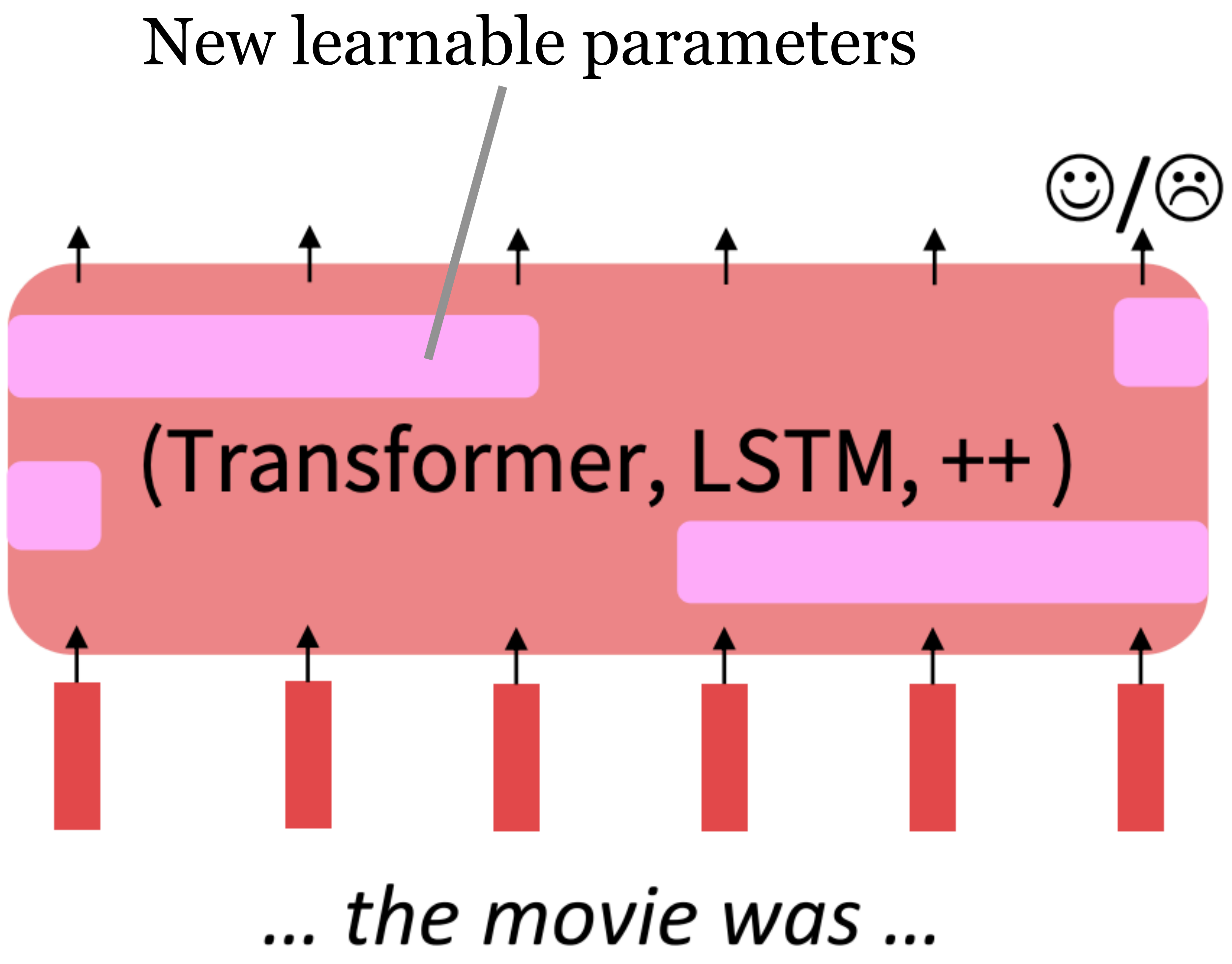
Train a few existing or new parameters

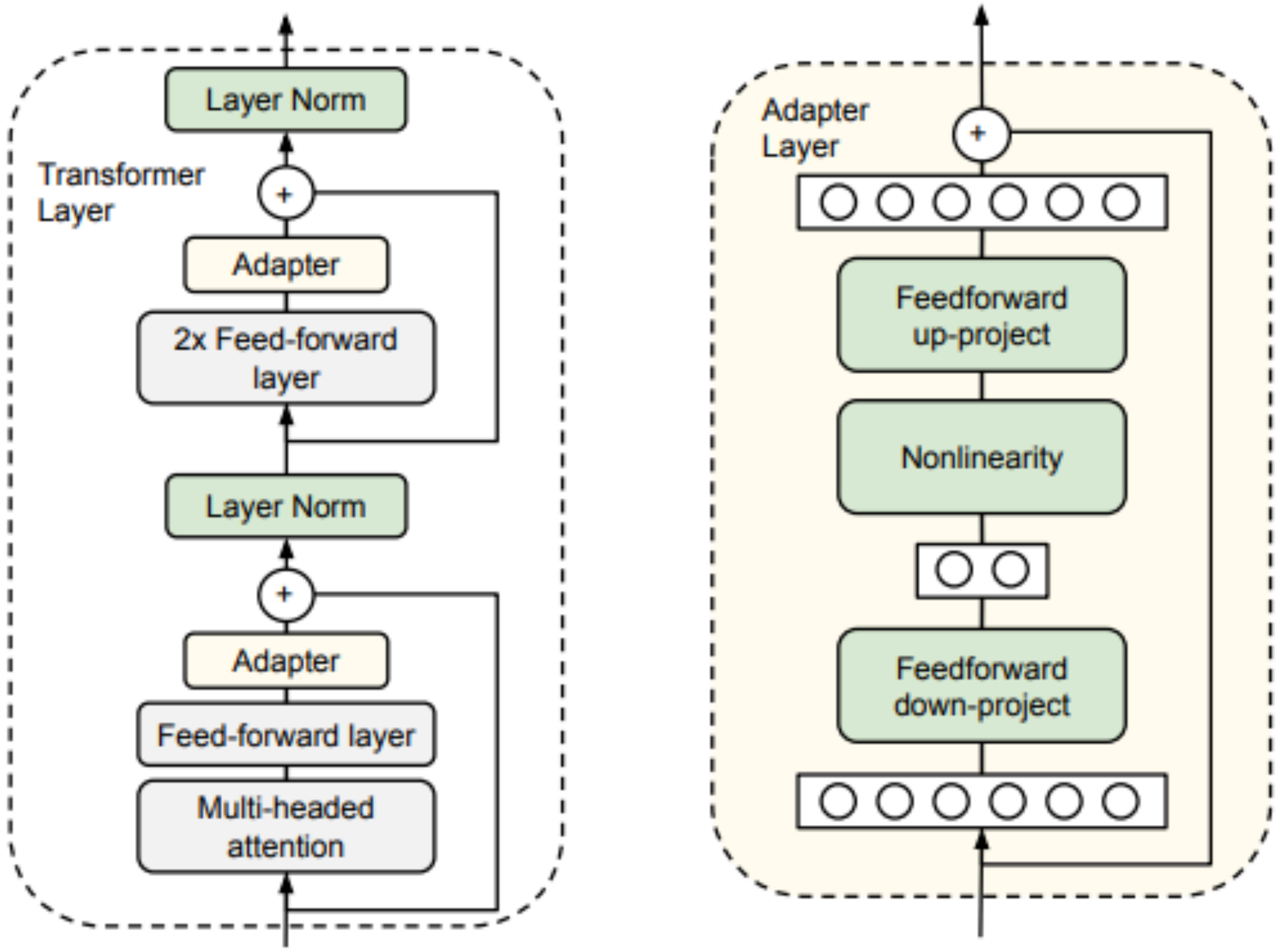


[Liu et al., 2019; Joshi et al., 2020]

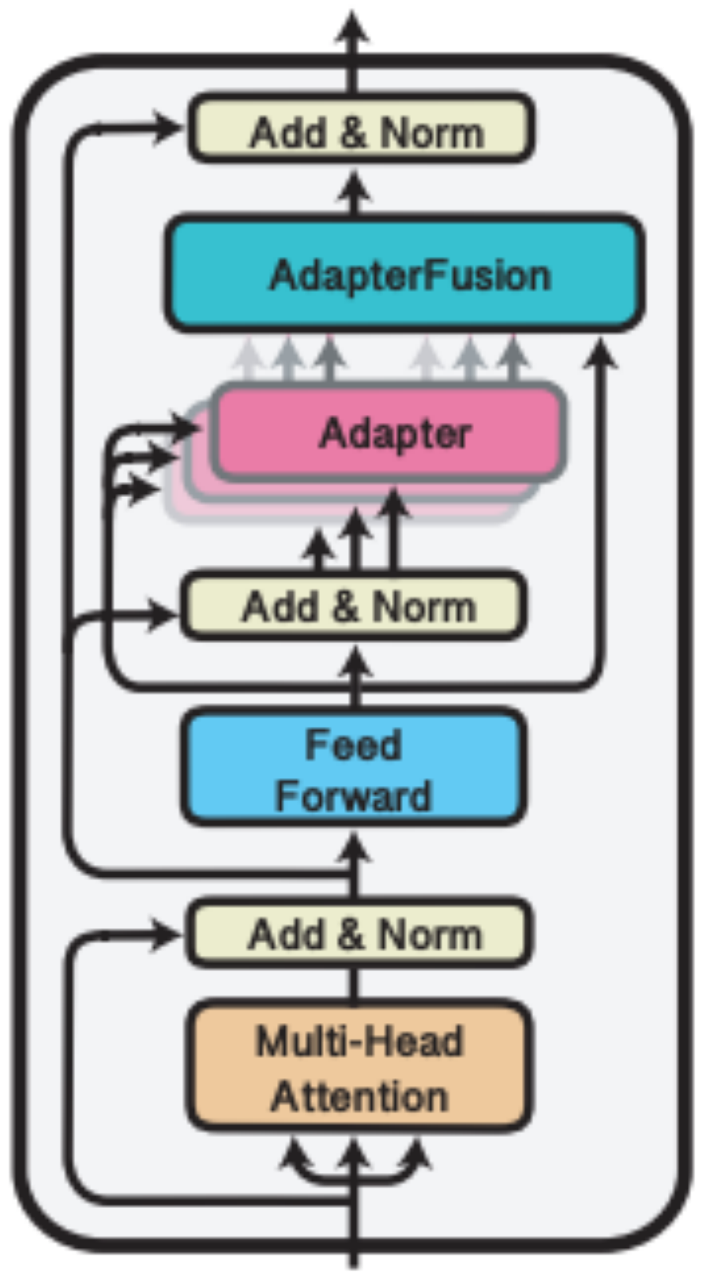
Parameter-Efficient Finetuning: Adapters

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





[Houlsby et al., 2019]

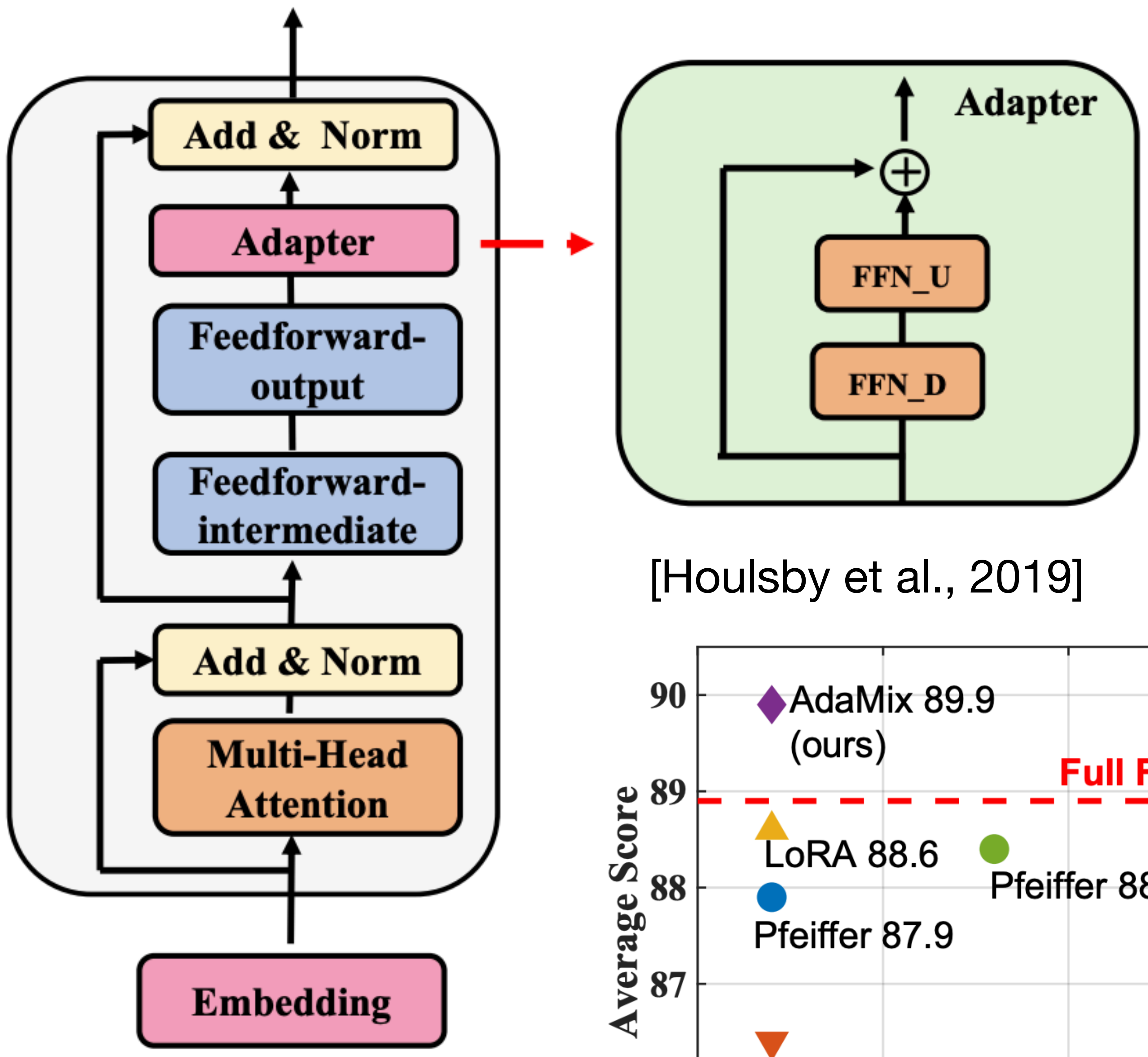


[Pfeiffer et al., 2021]

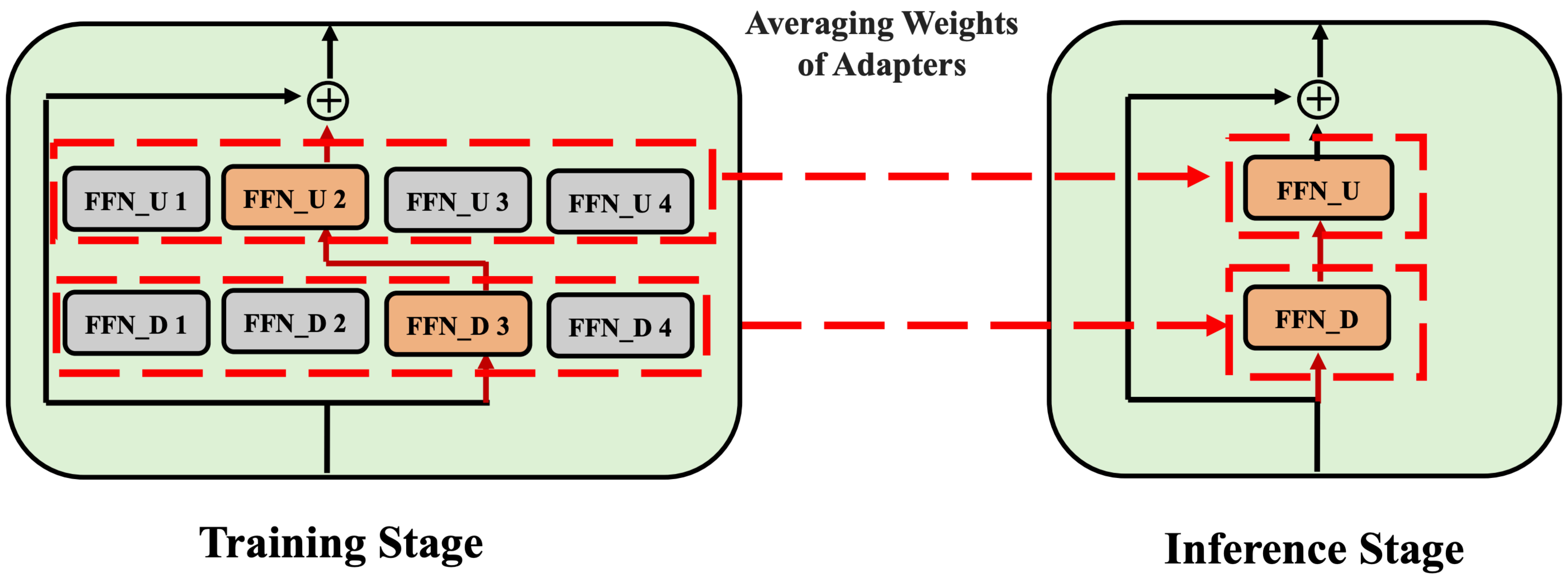
https://github.com/adapter-hub/adapter-transformers

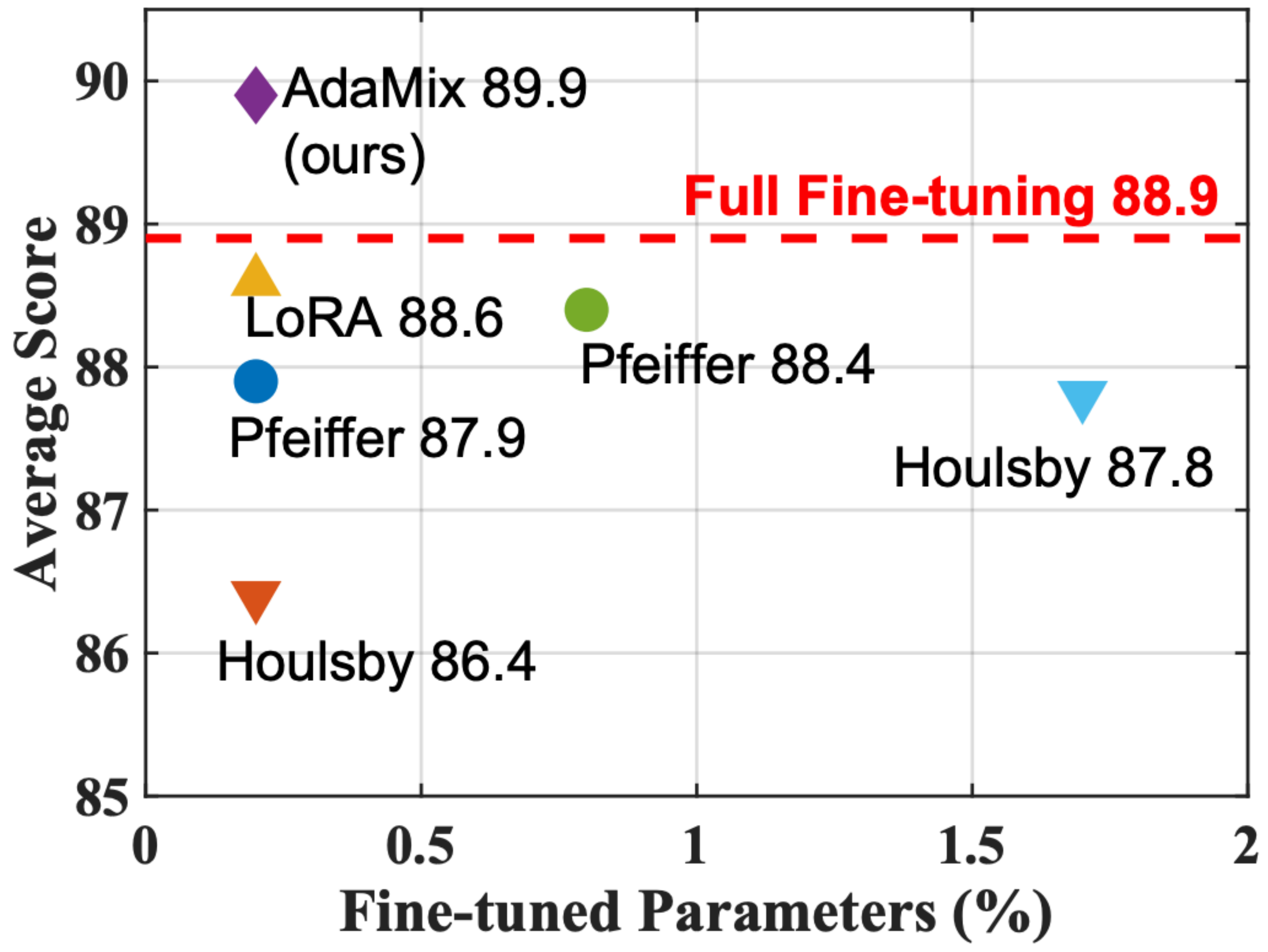
Parameter-Efficient Finetuning: Adapters

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



[Houlsby et al., 2019]



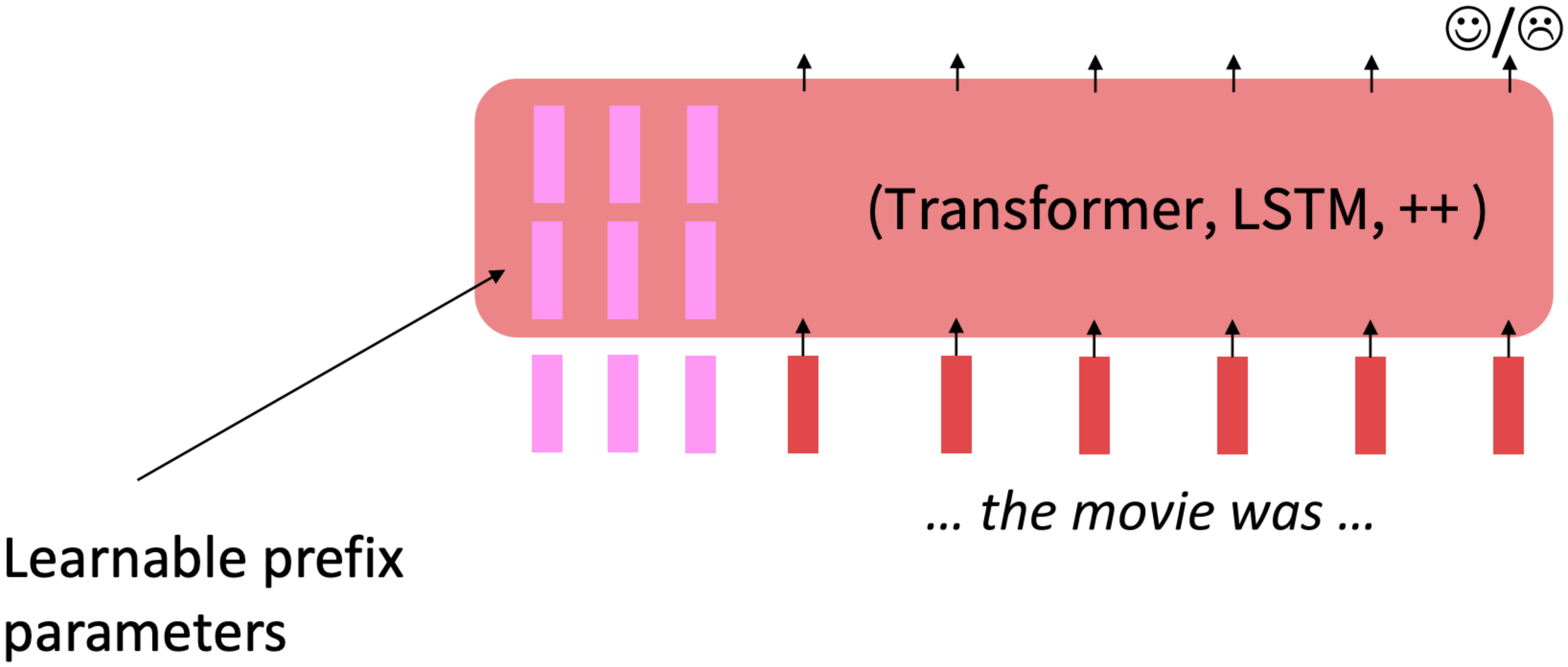


Performance on GLUE, fine-tuning of RoBERTa-large

[AdaMix, Wang et al., EMNLP 2022]

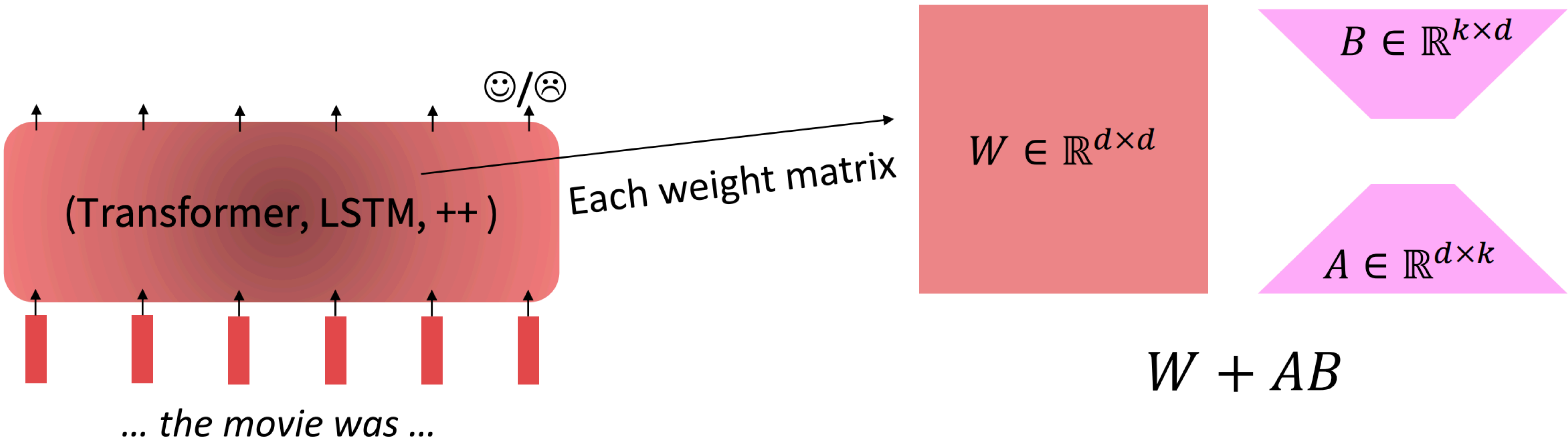
Parameter-Efficient Finetuning: Prefix-Tuning, Prompt tuning

- 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

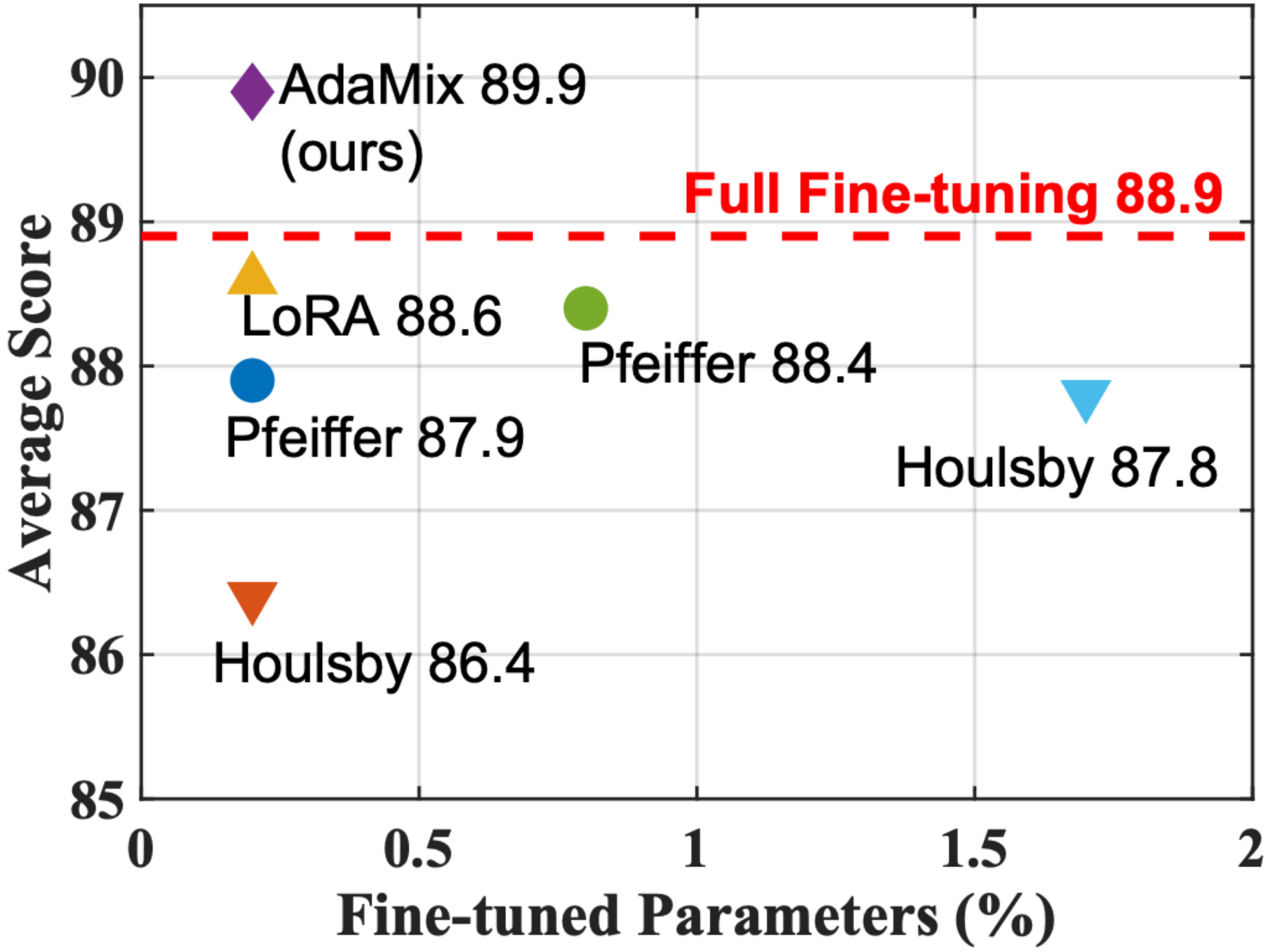
- Low-Rank Adaptation learns a low-rank "diff" between the pretrained and finetuned weight matrices.
- Easier to learn than prefix-tuning



[Hu et al., 2021]

Parameter-Efficient Finetuning: Low-Rank Adaptation

Model	#Param.	M A		S-B rson	Avg.
Full Fine-tuning†	355.0M	9		**2.4**	88.9
Pfeiffer Adapter†	3.0M	9		2.1	88.4
Pfeiffer Adapter†	0.8M	9		1.9	87.9
Houlsby Adapter†	6.0M	8		1.0	87.8
Houlsby Adapter†	0.8M	9		1.5	86.4
LoRA†	0.8M	9		2.3	88.6
AdaMix Adapter	0.8M	**9**		**2.4**	**89.9**



Good performance by tuning just a fraction of the weights

[AdaMix, Wang et al., EMNLP 2022]

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.