DeepLearning.AI

aws

Generative AI and large-language models (LLMs)

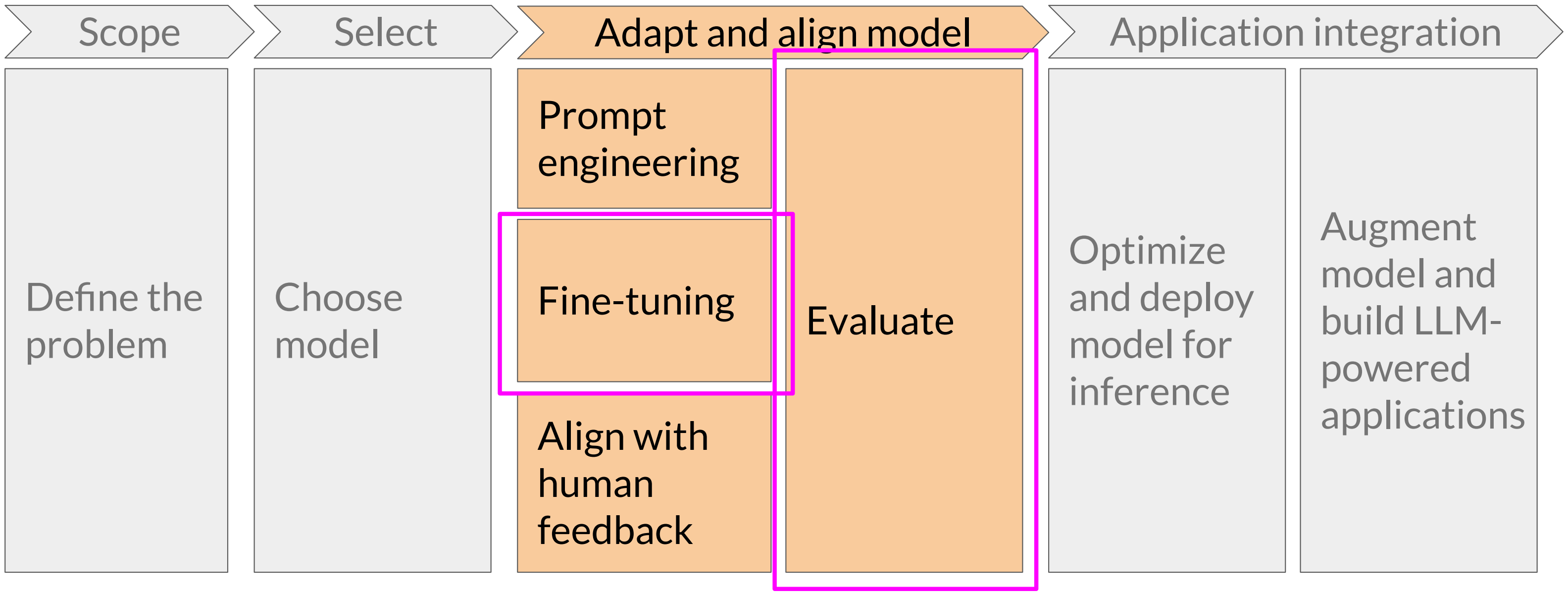
FINE-TUNING, INSTRUCTION PROMPTS, AND PARAMETER EFFICIENT FINE-TUNING

Fine-tuning with instruction prompts

GenAI project lifecycle

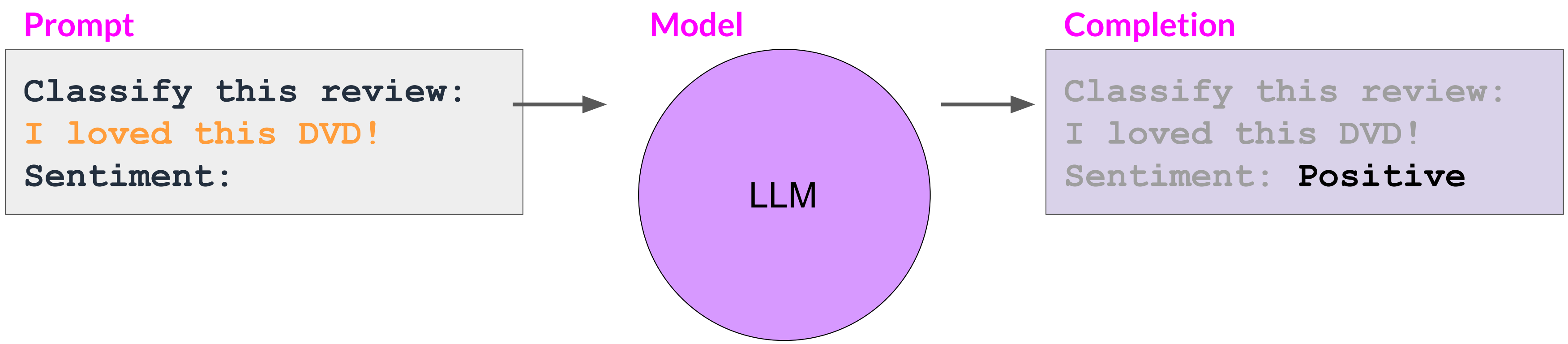
Scope	Select	Adapt and align model		Application integration	
Define the problem	Choose model	Prompt engineering Fine-tuning Align with human feedback	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications

GenAI project lifecycle



Fine-tuning an LLM with instruction prompts

In-context learning (ICL) - zero shot inference



In-context learning (ICL) - zero shot inference

Prompt

```
Classify this review:
I loved this DVD!
Sentiment:
```

Model

LLM

Completion

```
Classify this review:
I loved this DVD!
Sentiment: eived a
very nice book review
```

In-context learning (ICL) - one/few shot inference

Prompt

```
Classify this review:
I loved this DVD!
Sentiment: Positive

Classify this review:
I don't like this
chair.
Sentiment:
```

Model

LLM

Completion

```
Classify this review:
I loved this DVD!
Sentiment: Positive

Classify this review:
I don't like this
chair.
Sentiment: Negative
```

One-shot or Few-shot Inference

Limitations of in-context learning

```
Classify this review:
I loved this movie!
Sentiment: Positive

Classify this review:
I don't like this chair.
Sentiment: Negative

Classify this review:
This sofa is so ugly.
Sentiment: Negative

Classify this review:
Who would use this product?
Sentiment:
```

Context Window

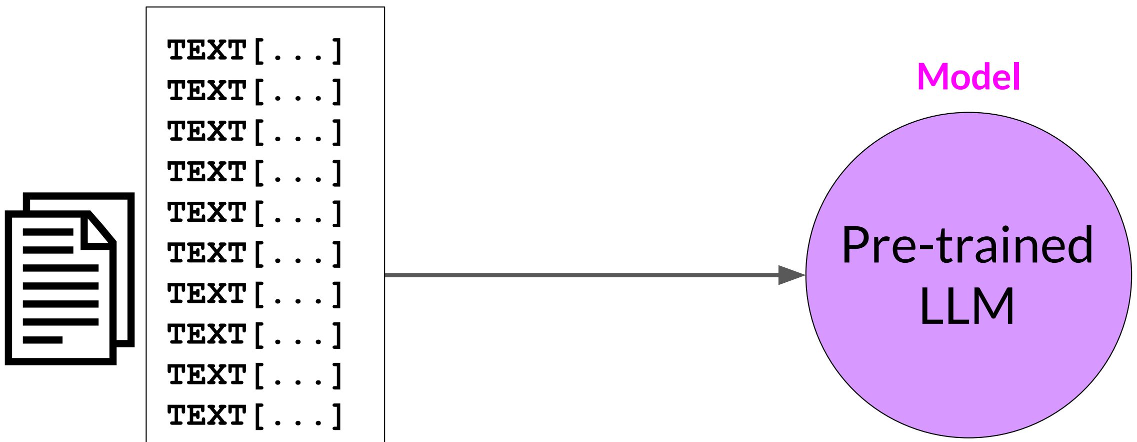
Even with multiple examples

- In-context learning may not work for smaller models LLM
- Examples take up space in the context window

Instead, try **fine-tuning** the model

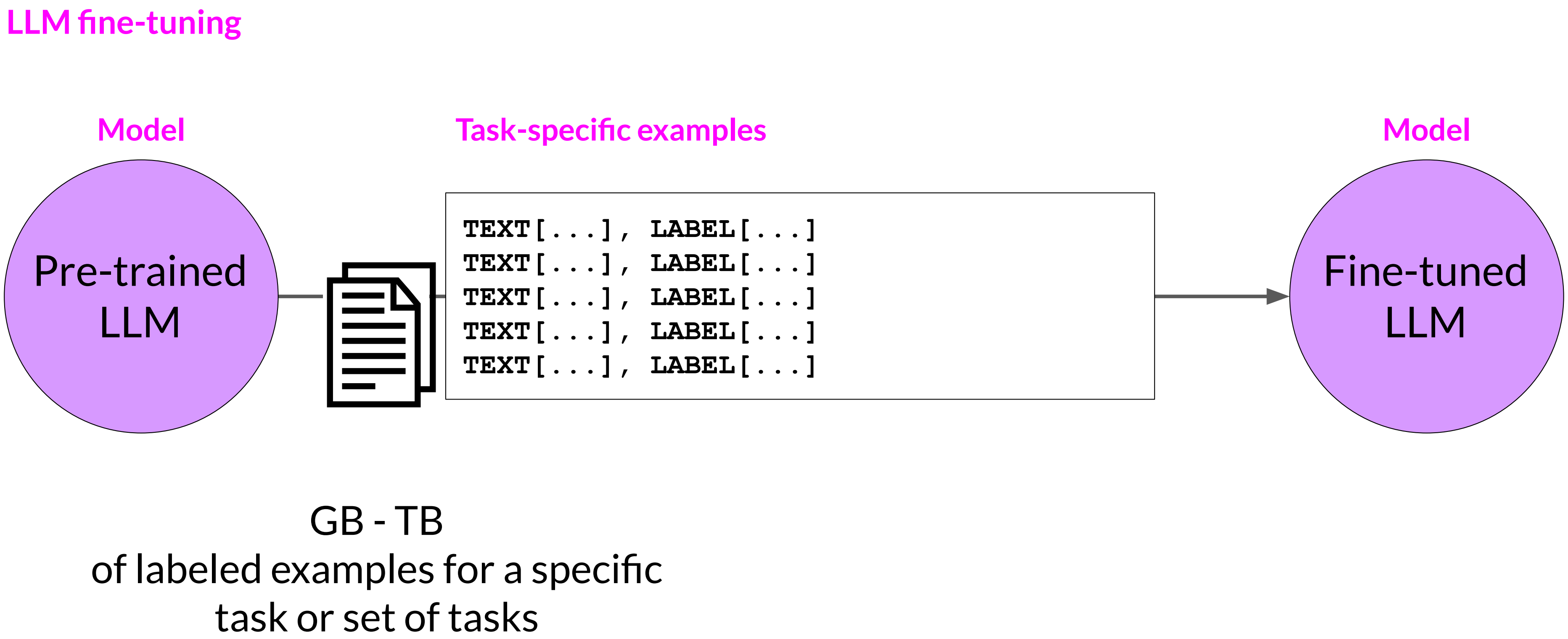
LLM fine-tuning at a high level

LLM pre-training



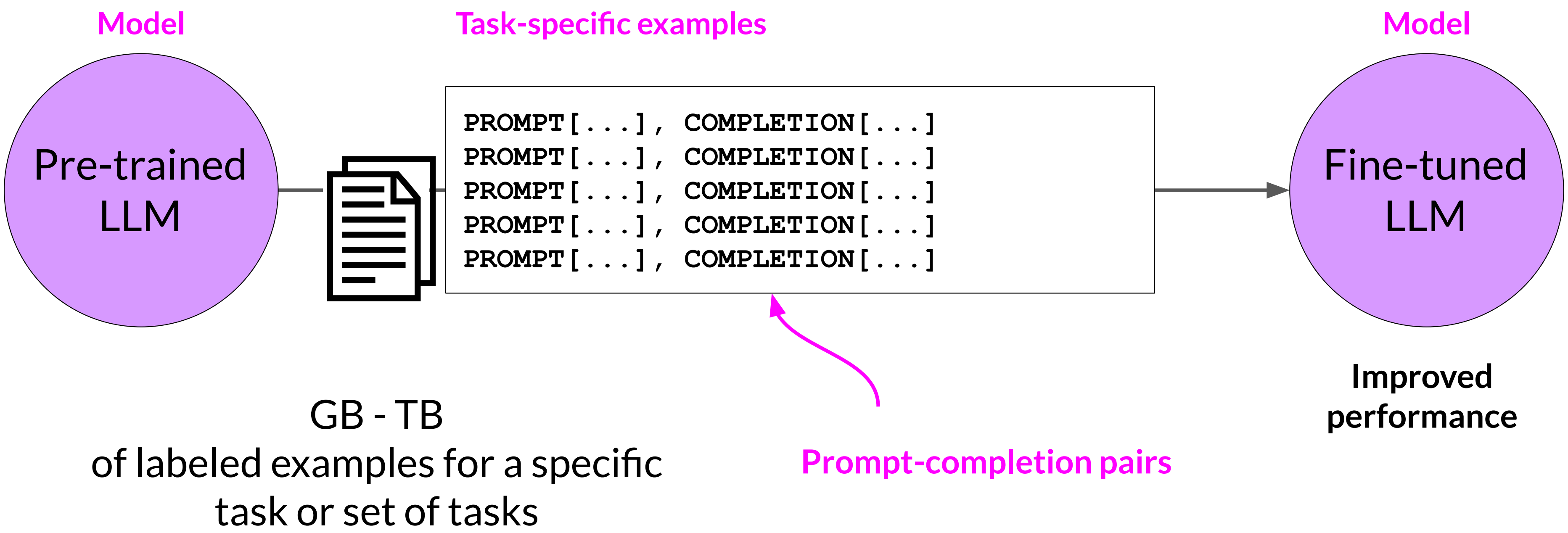
GB - TB - PB
of unstructured textual data

LLM fine-tuning at a high level



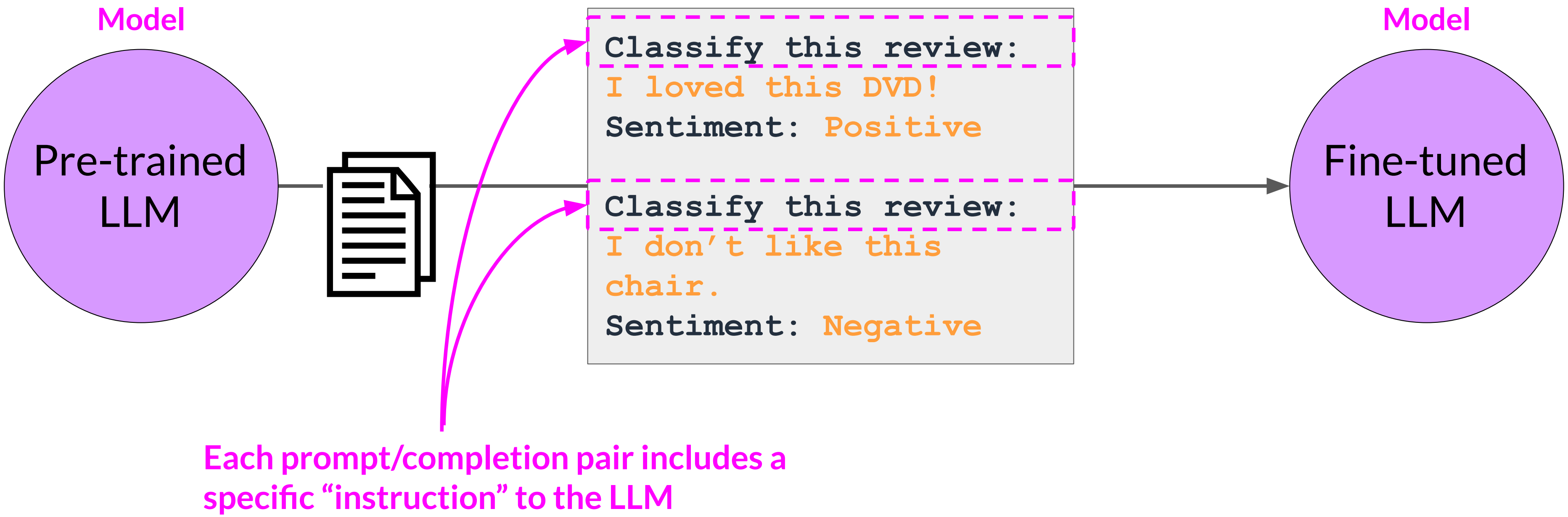
LLM fine-tuning at a high level

LLM fine-tuning



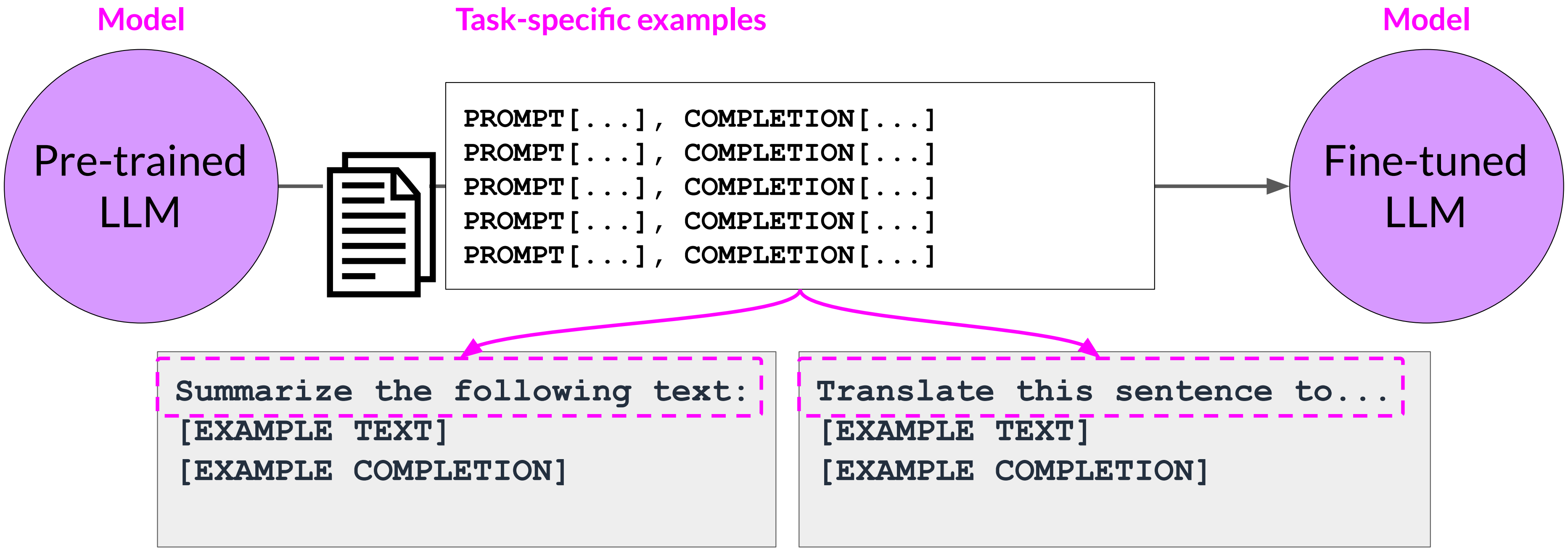
Using prompts to fine-tune LLMs with instruction

LLM fine-tuning



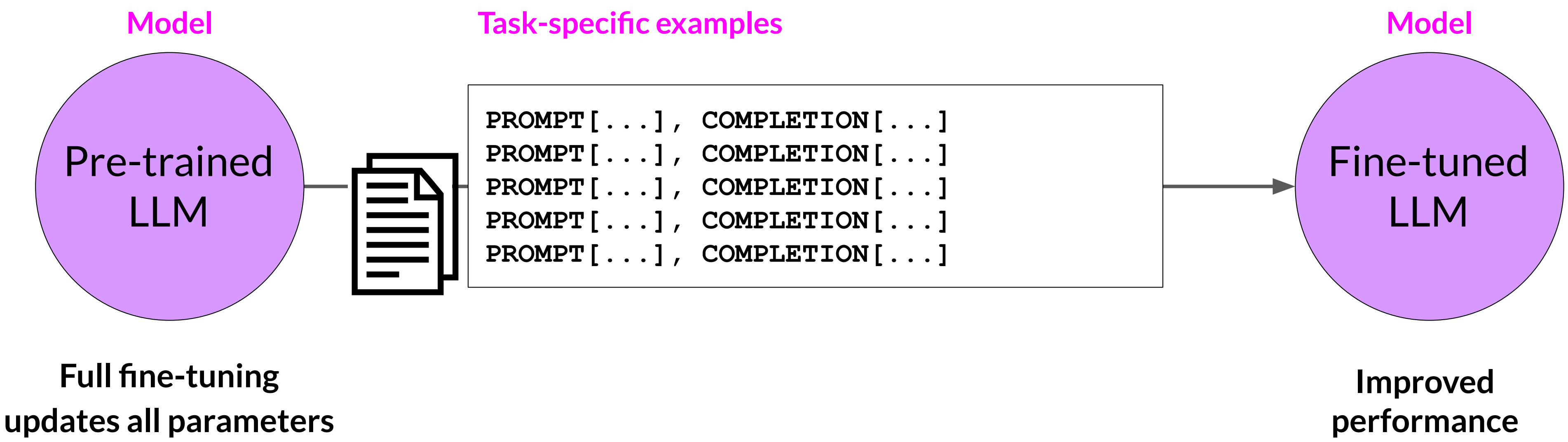
Using prompts to fine-tune LLMs with instruction

LLM fine-tuning



Using prompts to fine-tune LLMs with instruction

LLM fine-tuning



Sample prompt instruction templates

Classification / sentiment analysis

```
jinja: "Given the following review:\n{{review_body}}\npredict the associated rating\
  \ from the following choices (1 being lowest and 5 being highest)\n- {{ answer_choices\
  \ | join('\\n- ') }} \n|||\n{{answer_choices[star_rating-1]}}"
```

Text generation

```
jinja: Generate a {{star_rating}}-star review (1 being lowest and 5 being highest)
  about this product {{product_title}}.          |||        {{review_body}}
```

Text summarization

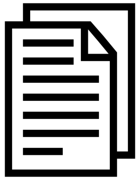
```
jinja: "Give a short sentence describing the following product review:\n{{review_body}}\
  \ \n|||\n{{review_headline}}"
```

Source: https://github.com/bigscience-workshop/promptsource/blob/main/promptsource/templates/amazon_polarity/templates.yaml

LLM fine-tuning process

LLM fine-tuning

Prepared instruction dataset



Training splits

```
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
```

Training

```
PROMPT[...], COMPLETION[...]
...
```

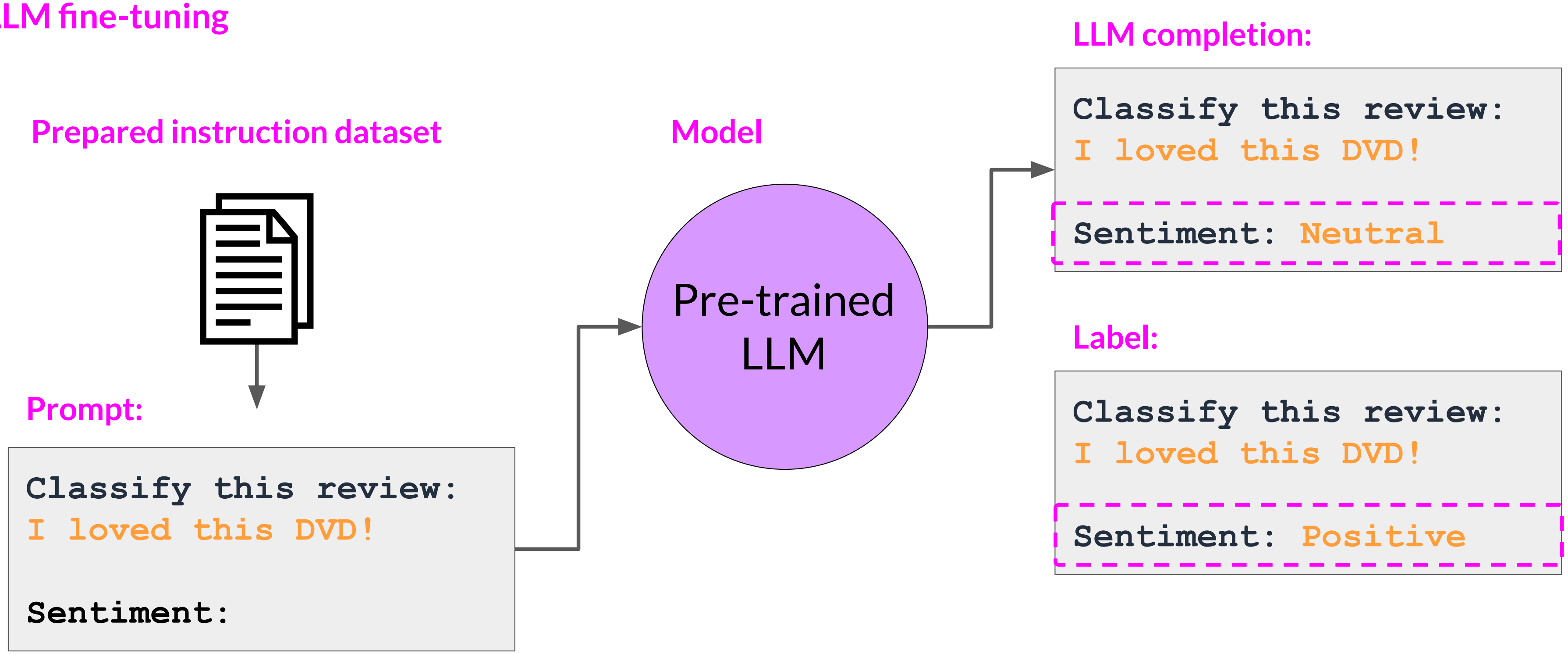
Validation

```
PROMPT[...], COMPLETION[...]
...
```

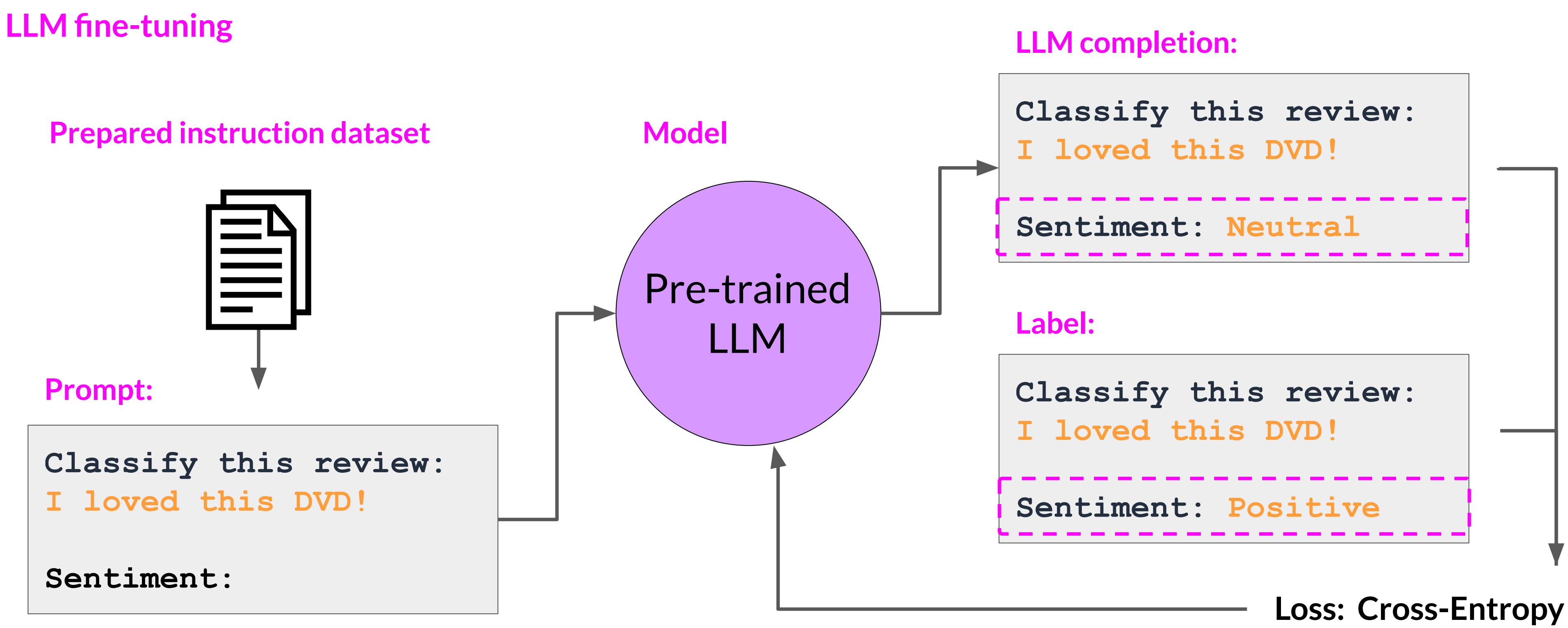
Test

LLM fine-tuning process

LLM fine-tuning



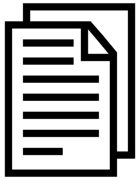
LLM fine-tuning process



LLM fine-tuning process

LLM fine-tuning

Prepared instruction dataset



Training splits

```
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
```
Training

```
PROMPT[...], COMPLETION[...]
...
```
Validation

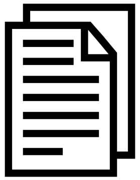
validation_accuracy

```
PROMPT[...], COMPLETION[...]
...
```
Test

LLM fine-tuning process

LLM fine-tuning

Prepared instruction dataset



Training splits

```
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
```

Training

```
PROMPT[...], COMPLETION[...]
...
```

Validation

```
PROMPT[...], COMPLETION[...]
...
```

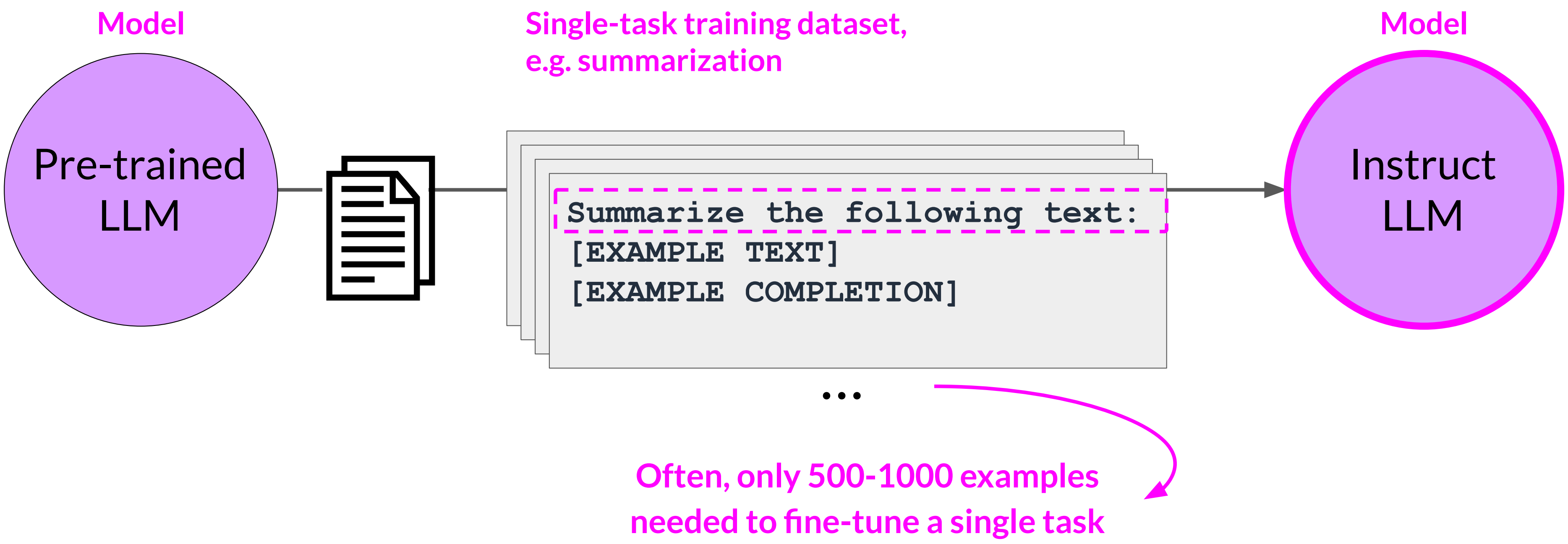
Test

test_accuracy

LLM fine-tuning process

Fine-tuning on a single task

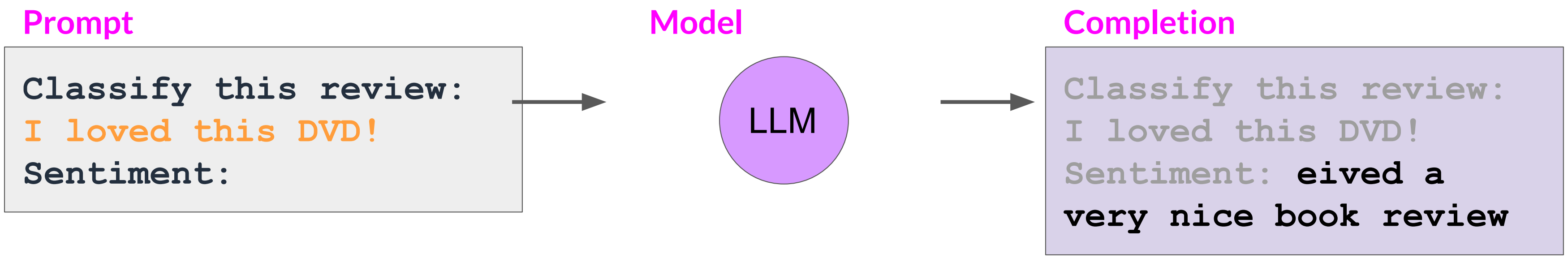
Fine-tuning on a single task



Catastrophic forgetting

- Fine-tuning can significantly increase the performance of a model on a specific task…

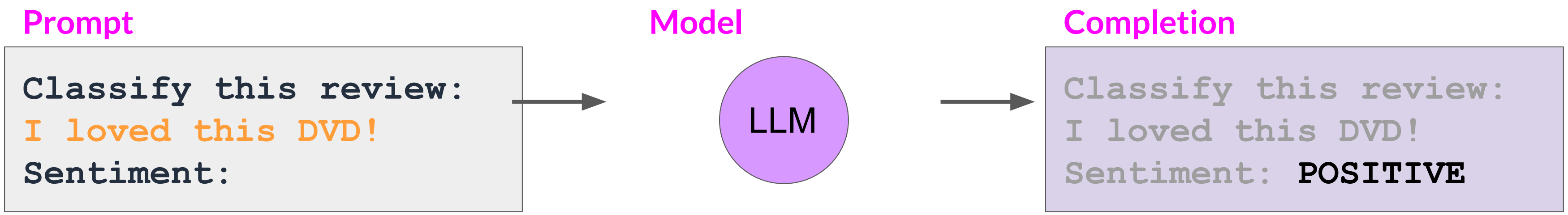
Before fine-tuning



Catastrophic forgetting

- Fine-tuning can significantly increase the performance of a model on a specific task...

After fine-tuning



Catastrophic forgetting

- ...but can lead to reduction in ability on other tasks

Before fine-tuning

Prompt

```
What is the name of
the cat?
Charlie the cat roamed
the garden at night.
```

Model

LLM

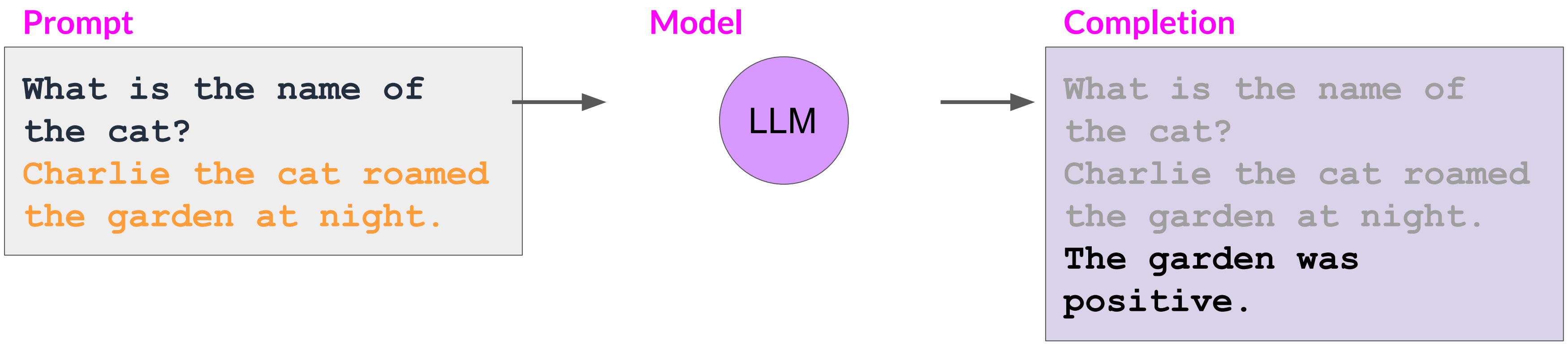
Completion

```
What is the name of
the cat?
Charlie the cat roamed
the garden at night.
Charlie
```

Catastrophic forgetting

- ...but can lead to reduction in ability on other tasks

After fine-tuning

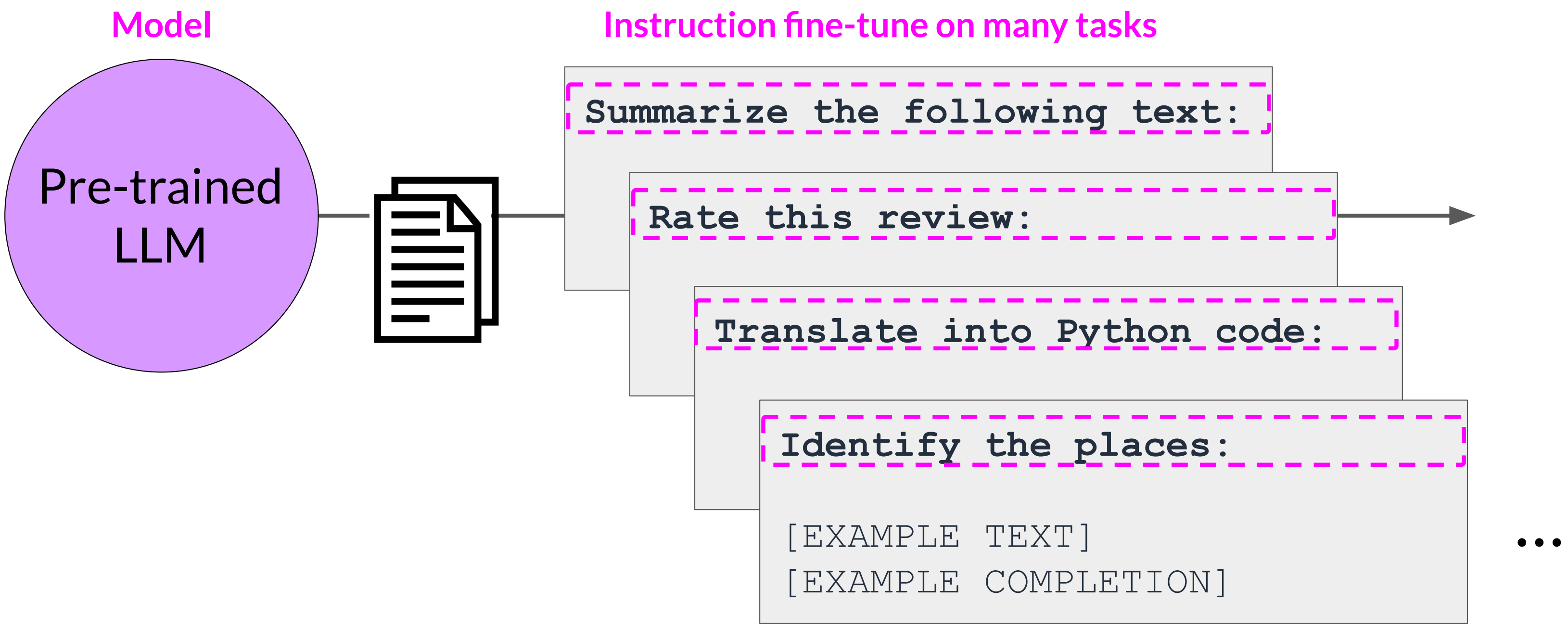


How to avoid catastrophic forgetting

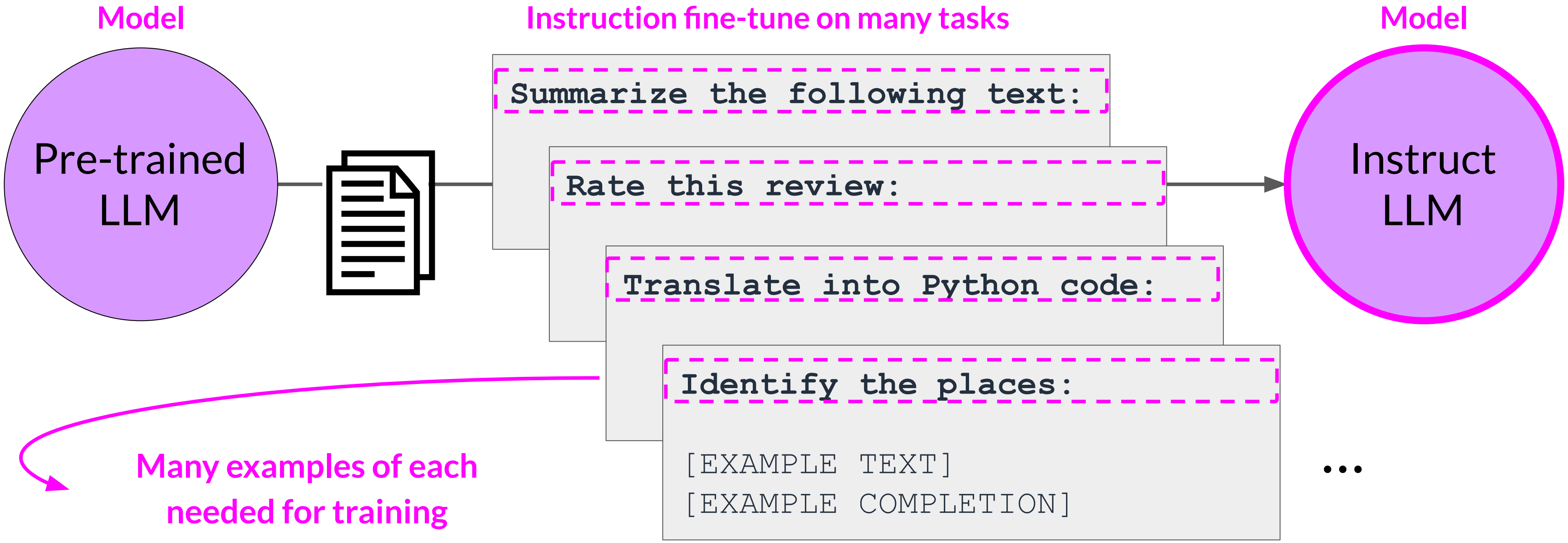
- First note that you might not have to!
- Fine-tune on **multiple tasks** at the same time
- Consider **Parameter Efficient Fine-tuning** (PEFT)

Multi-task, instruction fine-tuning

Multi-task, instruction fine-tuning

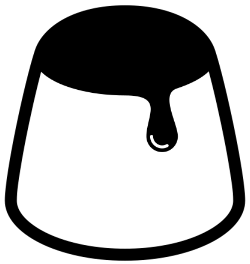


Multi-task, instruction fine-tuning



Instruction fine-tuning with FLAN

- FLAN models refer to a specific set of instructions used to perform instruction fine-tuning

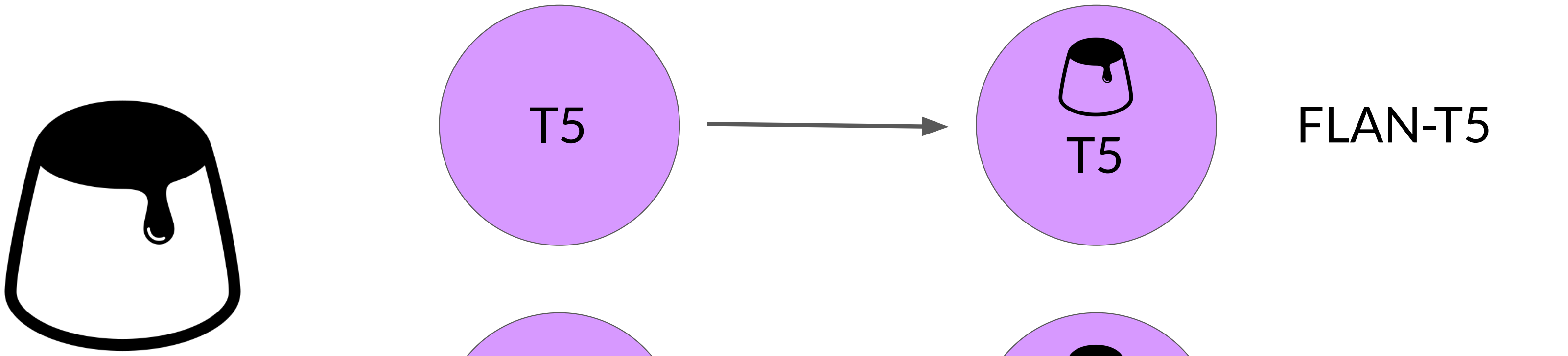


FLAN

"The metaphorical dessert to the main course of pretraining"

Instruction fine-tuning with FLAN

- FLAN models refer to a specific set of instructions used to perform instruction fine-tuning



FLAN
(**F**ine-tuned
LAnguage **N**et)

FLAN-T5: Fine-tuned version of pre-trained T5 model

- FLAN-T5 is a great, general purpose, instruct model

T0-SF	Muffin	CoT (reasoning)	Natural Instructions
- Commonsense Reasoning, - Question Generation, - Closed-book QA, - Adversarial QA, - Extractive QA …	- Natural language inference, - Code instruction gen, - Code repair - Dialog context generation, - Summarization (SAMSum) …	- Arithmetic reasoning, - Commonsense reasoning - Explanation generation, - Sentence composition, - Implicit reasoning, …	- Cause effect classification, - Commonsense reasoning, - Named Entity Recognition, - Toxic Language Detection, - Question answering …
55 Datasets **14 Categories** **193 Tasks**	**69 Datasets** **27 Categories** **80 Tasks**	**9 Datasets** **1 Category** **9 Tasks**	**372 Datasets** **108 Categories** **1554 Tasks**

Source: Chung et al. 2022, “Scaling Instruction-Finetuned Language Models”

FLAN-T5: Fine-tuned version of pre-trained T5 model

- FLAN-T5 is a great, general purpose, instruct model

T0-SF

- Commonsense Reasoning,
- Question Generation,
- Closed-book QA,
- Adversarial QA,
- Extractive QA

…

55 Datasets
14 Categories
193 Tasks

Muffin

- Natural language inference,
- Code instruction gen,
- Code repair
- Dialog context generation,
- **Summarization (SAMSum)**

…

69 Datasets
27 Categories
80 Tasks

CoT (reasoning)

- Arithmetic reasoning,
- Commonsense reasoning
- Explanation generation,
- Sentence composition,
- Implicit reasoning,

…

9 Datasets
1 Category
9 Tasks

Natural Instructions

- Cause effect classification,
- Commonsense reasoning,
- Named Entity Recognition,
- Toxic Language Detection,
- Question answering

…

372 Datasets
108 Categories
1554 Tasks

Source: Chung et al. 2022, “Scaling Instruction-Finetuned Language Models”

SAMSum: A dialogue dataset

Sample prompt training dataset (**samsum**) to fine-tune FLAN-T5 from pretrained T5

Datasets: **samsum** Tasks: Summarization Languages: English

dialogue (string)	summary (string)
"Amanda: I baked cookies. Do you want some? Jerry: Sure! Amanda: I'll bring you tomorrow :-)"	"Amanda baked cookies and will bring Jerry some tomorrow."
"Olivia: Who are you voting for in this election? Oliver: Liberals as always. Olivia: Me too!! Oliver: Great"	"Olivia and Olivier are voting for liberals in this election. "
"Tim: Hi, what's up? Kim: Bad mood tbh, I was going to do lots of stuff but ended up procrastinating Tim: What did…	"Kim may try the pomodoro technique recommended by Tim to get more stuff done."

Source: https://huggingface.co/datasets/samsum, https://github.com/google-research/FLAN/blob/2c79a31/flan/v2/templates.py#L3285

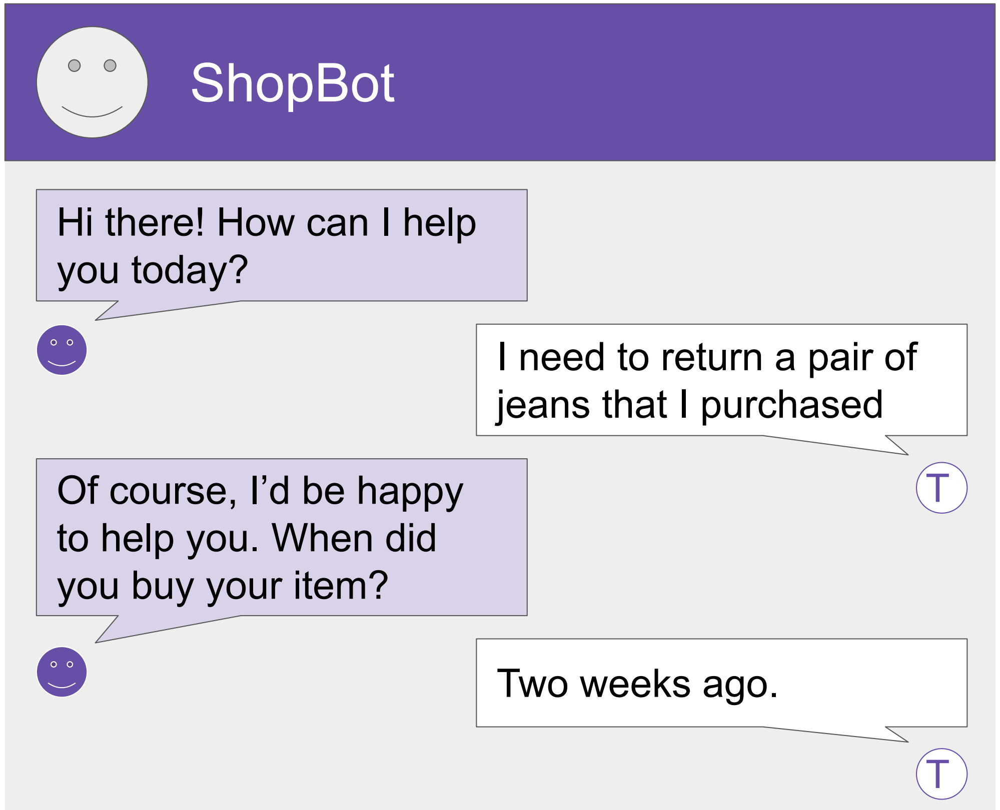
Sample FLAN-T5 prompt templates

```
"samsum": [
       ("{dialogue}\n\Briefly summarize that dialogue.", "{summary}"),
       ("Here is a dialogue:\n{dialogue}\n\nWrite a short summary!",
        "{summary}"),
       ("Dialogue:\n{dialogue}\n\nWhat is a summary of this dialogue?",
        "{summary}"),
       ("{dialogue}\n\nWhat was that dialogue about, in two sentences or less?",
        "{summary}"),
       ("Here is a dialogue:\n{dialogue}\n\nWhat were they talking about?",
        "{summary}"),
       ("Dialogue:\n{dialogue}\nWhat were the main points in that "
        "conversation?", "{summary}"),
       ("Dialogue:\n{dialogue}\nWhat was going on in that conversation?",
        "{summary}"),
]
```

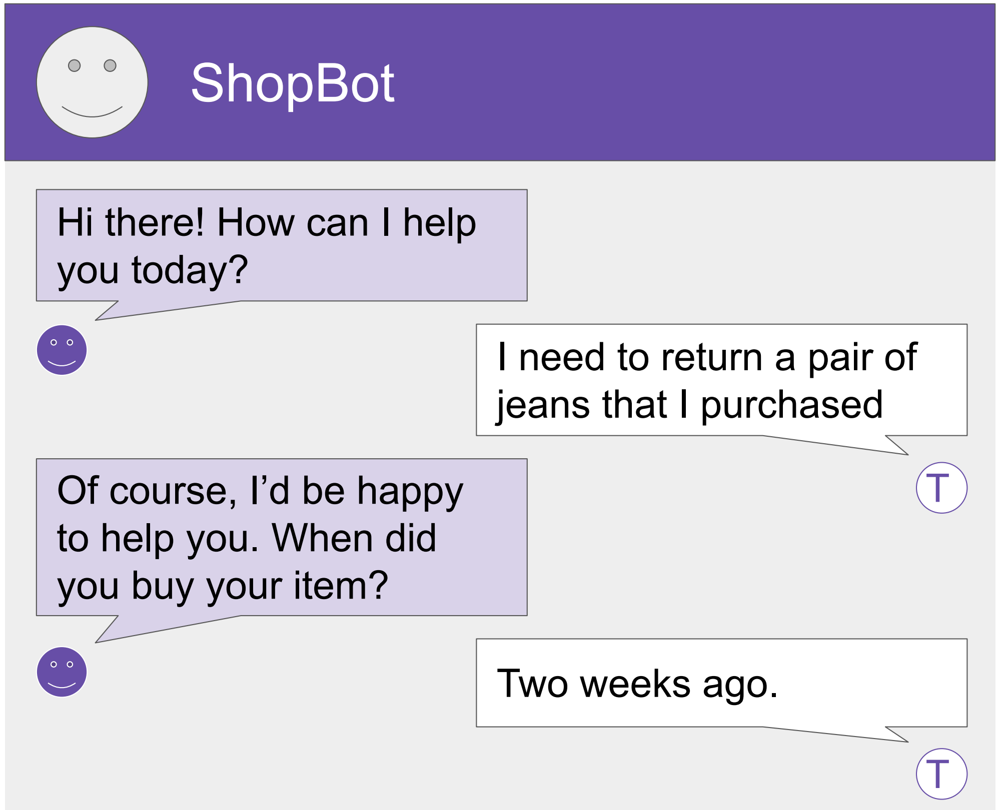
Sample FLAN-T5 prompt templates

```
"samsum": [
       ("{dialogue}\n\Briefly summarize that dialogue.", "{summary}"),
       ("Here is a dialogue:\n{dialogue}\n\nWrite a short summary!",
        "{summary}"),
       ("Dialogue:\n{dialogue}\n\nWhat is a summary of this dialogue?",
        "{summary}"),
       ("{dialogue}\n\nWhat was that dialogue about, in two sentences or less?",
        "{summary}"),
       ("Here is a dialogue:\n{dialogue}\n\nWhat were they talking about?",
        "{summary}"),
       ("Dialogue:\n{dialogue}\nWhat were the main points in that "
        "conversation?", "{summary}"),
       ("Dialogue:\n{dialogue}\nWhat was going on in that conversation?",
        "{summary}"),
]
```

Improving FLAN-T5's summarization capabilities



Improving FLAN-T5's summarization capabilities



Goal: Summarize conversations to identify actions to take

Improving FLAN-T5's summarization capabilities

Further fine-tune FLAN-T5 with a domain-specific instruction dataset (**dialogsum**)

Datasets: knkarthick/**dialogsum** like 13

Tasks: Summarization Text2Text Generation Text Generation Languages: English Multilinguality: monolingual Size Categorie

Language Creators: expert-generated Annotations Creators: expert-generated Source Datasets: original License: mit

Dataset card Files and versions Community 4

Dataset Preview

Split

train (12.5k rows)

id (string)	dialogue (string)	summary (string)
"train_0"	"#Person1#: Hi, Mr. Smith. I'm Doctor Hawkins. Why are you here today? #Person2#: I found it would be a good…	"Mr. Smith's getting a check-up, and Doctor Hawkins advises him to have one every year. Hawkins'll give some…
"train_1"	"#Person1#: Hello Mrs. Parker, how have you been? #Person2#: Hello Dr. Peters. Just fine thank you. Ricky…	"Mrs Parker takes Ricky for his vaccines. Dr. Peters checks the record and then gives Ricky a vaccine."
"train_2"	"#Person1#: Excuse me, did you see a set of keys? #Person2#: What kind of keys? #Person1#: Five keys and a small foot ornament. #Person2#: What a shame! I didn't see them. #Person1#: Well, can you help me look for it? That's my first time here. #Person2#: Sure. It's my pleasure. I'd like to help you look for the missing keys. #Person1#: It's very kind of you. #Person2#: It's not a big deal.Hey, I found them. #Person1#: Oh, thank God! I don't know how to thank you, guys. #Person2#: You're welcome."	"#Person1#'s looking for a set of keys and asks for #Person2#'s help to find them."

Example support-dialog summarization

Prompt (created from template)

```
Summarize the following conversation.
Tommy: Hello. My name is Tommy Sandals, I have a reservation.
Mike: May I see some identification, sir, please?
Tommy: Sure. Here you go.
Mike: Thank you so much. Have you got a credit card, Mr.
Sandals?
Tommy: I sure do.
Mike: Thank you, sir. You'll be in room 507, nonsmoking,
queen bed.
Tommy: That's great, thank you!
Mike: Enjoy your stay!
```

Source: https://huggingface.co/datasets/knkarthick/dialogsum/viewer/knkarthick--dialogsum/

Summary **before** fine-tuning FLAN-T5 with our dataset

Prompt (created from template)

```
Summarize the following
conversation.
Tommy: Hello. My name is
Tommy Sandals, I have a
reservation.
Mike: May I see some
...
...
...
Tommy: That's great, thank
you!
Mike: Enjoy your stay!
```

Model

FLAN-T5

Completion (Summary)

```
Tommy Sandals has a reservation
for a room at the Venetian
Hotel in Las Vegas.
```

Adequate completion, but does not match human baseline.

```
Human baseline summary:
Tommy Sandals has got a
reservation. Mike asks for his
identification and credit card
and helps his check-in.
```

Summary **before** fine-tuning FLAN-T5 with our dataset

Prompt (created from template)

```
Summarize the following
conversation.
Tommy: Hello. My name is
Tommy Sandals, I have a
reservation.
Mike: May I see some
...
...
...
Tommy: That's great, thank
you!
Mike: Enjoy your stay!
```

Model

FLAN-T5

Completion (Summary)

```
Tommy Sandals has a reservation
for a room at the Venetian
Hotel in Las Vegas.
```

Adequate completion, but does not match human baseline.

```
Human baseline summary:
Tommy Sandals has got a
reservation. Mike asks for his
identification and credit card
and helps his check-in.
```

Summary **before** fine-tuning FLAN-T5 with our dataset

Prompt (created from template)

```
Summarize the following
conversation.
Tommy: Hello. My name is
Tommy Sandals, I have a
reservation.
Mike: May I see some
...
...
...
Tommy: That's great, thank
you!
Mike: Enjoy your stay!
```

Model

FLAN-T5

Completion (Summary)

```
Tommy Sandals has a reservation
for a room at the Venetian
Hotel in Las Vegas.
```

Adequate completion, but does not match human baseline.

```
Human baseline summary:
Tommy Sandals has got a
reservation. Mike asks for his
identification and credit card
and helps his check-in.
```

Summary **after** fine-tuning FLAN-T5 with our dataset

Prompt (created from template)

```
Summarize the following
conversation.
Tommy: Hello. My name is
Tommy Sandals, I have a
reservation.
Mike: May I see some
...
...
...
Tommy: That's great, thank
you!
Mike: Enjoy your stay!
```

Model

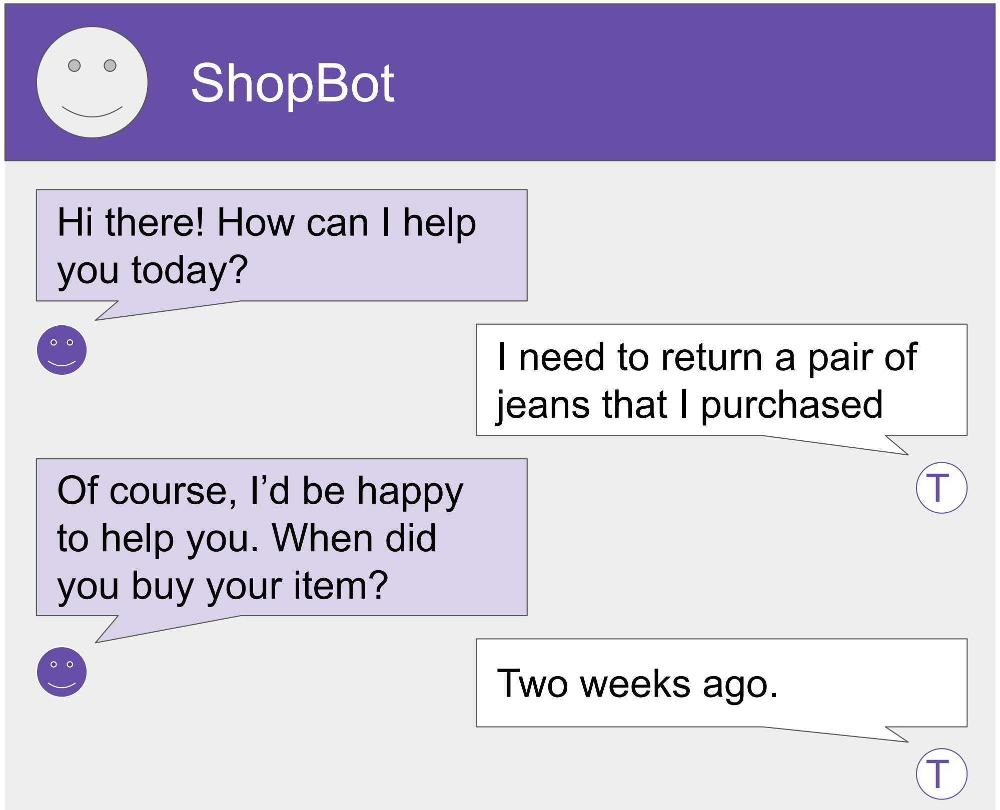
FLAN-T5
(Fine-tuned)

Completion (Summary)

```
Tommy Sandals has a
reservation and checks in
showing his ID and credit
card. Mike helps him to
check in and approves his
reservation.
```

Better summary,
more-closely matches
human baseline.

Fine-tuning with your own data



Model evaluation metrics

LLM Evaluation - Challenges

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}}$$

LLM Evaluation - Challenges

"Mike really loves drinking tea." = "Mike adores sipping tea."

❤️ ☕ = ❤️ ☕

"Mike does not drink coffee." ≠ "Mike does drink coffee."

🤢 ☕ ≠ 🧐 ☕

LLM Evaluation - Metrics

ROUGE

- Used for text summarization
- Compares a summary to one or more reference summaries

BLEU SCORE

- Used for text translation
- Compares to human-generated translations

LLM Evaluation - Metrics - Terminology

n-gram

The dog lay on the rug as I sipped a cup of tea.

bigram

unigram

LLM Evaluation - Metrics - ROUGE-1

Reference (human):

It is cold outside.

Generated output:

It is very cold outside.

$$\text{ROUGE-1 Recall} = \frac{\text{unigram matches}}{\text{unigrams in reference}} = \frac{4}{4} = 1.0$$

$$\text{ROUGE-1 Precision:} = \frac{\text{unigram matches}}{\text{unigrams in output}} = \frac{4}{5} = 0.8$$

$$\text{ROUGE-1 F1:} = 2\,\frac{\text{precision x recall}}{\text{precision + recall}} = 2\,\frac{0.8}{1.8} = 0.89$$

LLM Evaluation - Metrics - ROUGE-1

Reference (human):

It is cold outside.

Generated output:

It is not cold outside.

$$\text{ROUGE-1 Recall} = \frac{\text{unigram matches}}{\text{unigrams in reference}} = \frac{4}{4} = 1.0$$

$$\text{ROUGE-1 Precision:} = \frac{\text{unigram matches}}{\text{unigrams in output}} = \frac{4}{5} = 0.8$$

$$\text{ROUGE-1 F1:} = 2\,\frac{\text{precision x recall}}{\text{precision + recall}} = 2\,\frac{0.8}{1.8} = 0.89$$

LLM Evaluation - Metrics - ROUGE-2

Reference (human):

It is cold outside.

It is	is cold	cold outside

Generated output:

It is very cold outside.

It is	is very	very cold	cold outside

LLM Evaluation - Metrics - ROUGE-2

Reference (human):

It is cold outside.

| It is | is cold |
| cold outside | |

Generated output:

It is very cold outside.

| It is | is very |
| very cold | cold outside |

$$\text{ROUGE-2 Recall:} = \frac{\text{bigram matches}}{\text{bigrams in reference}} = \frac{2}{3} = 0.67$$

$$\text{ROUGE-2 Precision:} = \frac{\text{bigram matches}}{\text{bigrams in output}} = \frac{2}{4} = 0.5$$

$$\text{ROUGE-2 F1:} = 2\,\frac{\text{precision x recall}}{\text{precision + recall}} = 2\,\frac{0.335}{1.17} = 0.57$$

LLM Evaluation - Metrics - ROUGE-L

Reference (human):

It is cold outside.

Generated output:

It is very cold outside.

Longest common subsequence (LCS):

It is

cold outside

2

LLM Evaluation - Metrics - ROUGE-L

Reference (human):

It is cold outside.

Generated output:

It is very cold outside.

ROUGE-L Recall: $= \dfrac{\text{LCS(Gen, Ref)}}{\text{unigrams in reference}} = \dfrac{2}{4} = 0.5$

ROUGE-L Precision: $= \dfrac{\text{LCS(Gen, Ref)}}{\text{unigrams in output}} = \dfrac{2}{5} = 0.4$

ROUGE-L F1: $= 2 \dfrac{\text{precision x recall}}{\text{precision + recall}} = 2 \dfrac{0.2}{0.9} = 0.44$

LLM Evaluation - Metrics - ROUGE-L

Reference (human):

It is cold outside.

Generated output:

It is very cold outside.

LCS:
Longest common subsequence

ROUGE-L Recall: $= \dfrac{\text{LCS(Gen, Ref)}}{\text{unigrams in reference}} = \dfrac{2}{4} = 0.5$

ROUGE-L Precision: $= \dfrac{\text{LCS(Gen, Ref)}}{\text{unigrams in output}} = \dfrac{2}{5} = 0.4$

ROUGE-L F1: $= 2 \dfrac{\text{precision x recall}}{\text{precision + recall}} = 2 \dfrac{0.2}{0.9} = 0.44$

LLM Evaluation - Metrics - ROUGE hacking

Reference (human):

It is cold outside.

Generated output:

cold cold cold cold

LLM Evaluation - Metrics - ROUGE clipping

Reference (human):

It is cold outside.

Generated output:

cold cold cold cold

$$\text{ROUGE-1 Precision} = \frac{\text{unigram matches}}{\text{unigrams in output}} = \frac{4}{4} = 1.0$$

$$\text{Modified precision} = \frac{\text{clip(unigram matches)}}{\text{unigrams in output}} = \frac{1}{4} = 0.25$$

Generated output:

outside cold it is

$$\text{Modified precision} = \frac{\text{clip(unigram matches)}}{\text{unigrams in output}} = \frac{4}{4} = 1.0$$

LLM Evaluation - Metrics

ROUGE

- Used for text summarization
- Compares a summary to one or more reference summaries

BLEU SCORE

- Used for text translation
- Compares to human-generated translations

LLM Evaluation - Metrics - BLEU

BLEU metric = Avg(precision across range of n-gram sizes)

Reference (human):

I am very happy to say that I am drinking a warm cup of tea.

Generated output:

I am very happy that I am drinking a cup of tea. - BLEU 0.495

I am very happy that I am drinking a warm cup of tea. - BLEU 0.730

I am very happy to say that I am drinking a warm tea. - BLEU 0.798

I am very happy to say that I am drinking a warm cup of tea. - BLEU 1.000

LLM Evaluation - Metrics

ROUGE

- Used for text summarization
- Compares a summary to one or more reference summaries

BLEU SCORE

- Used for text translation
- Compares to human-generated translations

Benchmarks

Evaluation benchmarks

GLUE

SuperGLUE

HELM

MMLU (Massive Multitask Language Understanding)

BIG-bench

GLUE

GLUE

The tasks included in SuperGLUE benchmark:

Corpus	|Train|	|Test|	Task	Metrics	Domain
			Single-Sentence Tasks		
CoLA	8.5k	**1k**	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
			Similarity and Paraphrase Tasks		
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	**391k**	paraphrase	acc./F1	social QA questions
			Inference Tasks		
MNLI	393k	**20k**	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	**146**	coreference/NLI	acc.	fiction books

Source: Wang et al. 2018, "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding"

SuperGLUE

SuperGLUE

The tasks included in SuperGLUE benchmark:

Corpus	|Train|	|Dev|	|Test|	Task	Metrics	Text Sources
BoolQ	9427	3270	3245	QA	acc.	Google queries, Wikipedia
CB	250	57	250	NLI	acc./F1	various
COPA	400	100	500	QA	acc.	blogs, photography encyclopedia
MultiRC	5100	953	1800	QA	$F1_a$/EM	various
ReCoRD	101k	10k	10k	QA	F1/EM	news (CNN, Daily Mail)
RTE	2500	278	300	NLI	acc.	news, Wikipedia
WiC	6000	638	1400	WSD	acc.	WordNet, VerbNet, Wiktionary
WSC	554	104	146	coref.	acc.	fiction books

Source: Wang et al. 2019, "SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems"

GLUE and SuperGLUE leaderboards

GLUE SuperGLUE Paper Code Tasks Leaderboard FAQ Diagnostics Submit Login

Rank	Name
1	Microsoft Alexande
2	JDExplore d-team
3	Microsoft Alexande
4	DIRL Team
5	ERNIE Team - Bai
6	AliceMind & DIRL
7	DeBERTa Team - I
8	HFL iFLYTEK
9	PING-AN Omni-Si
10	T5 Team - Google

SuperGLUE GLUE

Leaderboard Version: **2.0**

	Rank	Name	Model	URL	Score	BoolQ	CB	COPA	MultiRC	ReCoRD	RTE	WiC	WSC	AX-b	AX-g
	1	JDExplore d-team	Vega v2		91.3	90.5	98.6/99.2	99.4	88.2/62.4	94.4/93.9	96.0	77.4	98.6	-0.4	100.0/50.0
+	2	Liam Fedus	ST-MoE-32B		91.2	92.4	96.9/98.0	99.2	89.6/65.8	95.1/94.4	93.5	77.7	96.6	72.3	96.1/94.1
	3	Microsoft Alexander v-team	Turing NLR v5		90.9	92.0	95.9/97.6	98.2	88.4/63.0	96.4/95.9	94.1	77.1	97.3	67.8	93.3/95.5
	4	ERNIE Team - Baidu	ERNIE 3.0		90.6	91.0	98.6/99.2	97.4	88.6/63.2	94.7/94.2	92.6	77.4	97.3	68.6	92.7/94.7
	5	Yi Tay	PaLM 540B		90.4	91.9	94.4/96.0	99.0	88.7/63.6	94.2/93.3	94.1	77.4	95.9	72.9	95.5/90.4
+	6	Zirui Wang	T5 + UDG, Single Model (Google Brain)		90.4	91.4	95.8/97.6	98.0	88.3/63.0	94.2/93.5	93.0	77.9	96.6	69.1	92.7/91.9
+	7	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4		90.3	90.4	95.7/97.6	98.4	88.2/63.7	94.5/94.1	93.2	77.5	95.9	66.7	93.3/93.8

Disclaimer: metrics may not be up-to-date. Check https://super.gluebenchmark.com and https://gluebenchmark.com/leaderboard for the latest.

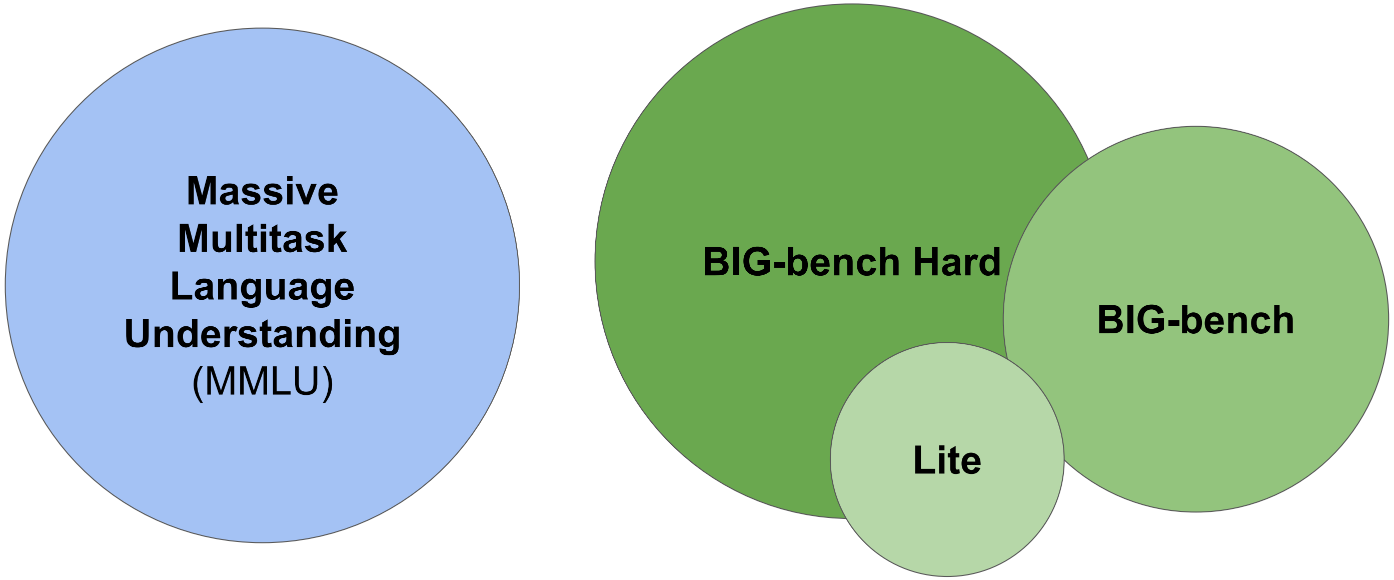
Benchmarks for massive models

Massive Multitask Language Understanding (MMLU)

2021

Source: Hendrycks, 2021. "Measuring Massive Multitask Language Understanding"

Benchmarks for massive models



2021

2022

Source: Hendrycks, 2021. "Measuring Massive Multitask Language Understanding"

Source: Suzgun et al. 2022. "Challenging BIG-Bench tasks and whether chain-of-thought can solve them"

Holistic Evaluation of Language Models (HELM)

HELM

Metrics:

1. Accuracy
2. Calibration
3. Robustness
4. Fairness
5. Bias
6. Toxicity
7. Efficiency

Models

Scenarios

	J1-Jumbo	J1-Grande	J1-Large	Anthropic-LM	BLOOM	T0pp	Cohere XL	Cohere Large	Cohere Medium	Cohere Small	GPT-NeoX
NaturalQuestions (open)			✔	✔	✔	✔	✔	✔	✔	✔	✔
NaturalQuestions (closed)	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
BoolQ	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
NarrativeQA	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
QuAC	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
HellaSwag	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
OpenBookQA	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
TruthfulQA	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
MMLU	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
MS MARCO				✔	✔		✔	✔	✔	✔	✔
TREC				✔	✔		✔	✔	✔	✔	✔
XSUM	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
CNN/DM	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
IMDB	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
CivilComments	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
RAFT	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

Holistic Evaluation of Language Models (HELM)

HELM Models Scenarios Results Raw runs v0.2.2 (last updated 2023-03-19)

Core scenarios

The scenarios where we evaluate all the models.

[Accuracy | Calibration | Robustness | Fairness | Efficiency | General information | Bias | Toxicity | Summarization metrics | JSON]

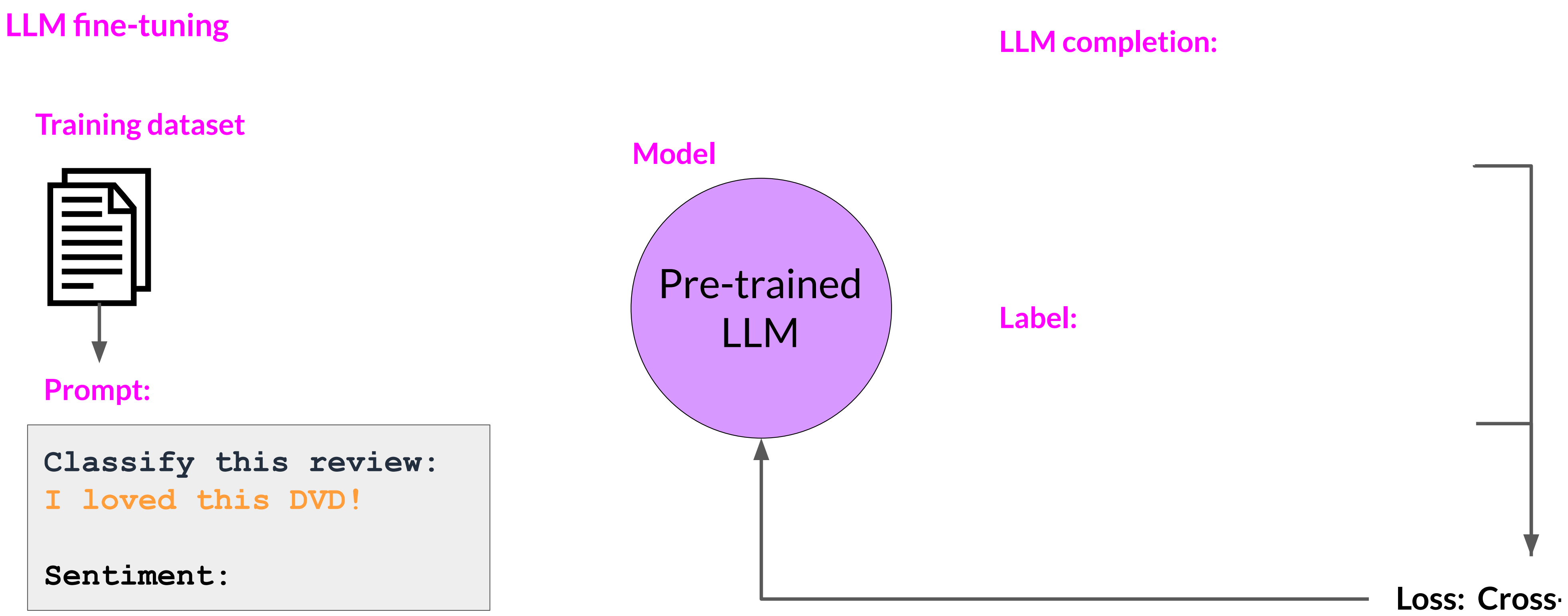
Accuracy

Model/adapter	Mean win rate ↑ [sort]	MMLU - EM ↑ [sort]	BoolQ - EM ↑ [sort]	NarrativeQA - F1 ↑ [sort]	NaturalQuestions (closed-book) - F1 ↑ [sort]	NaturalQuestions (open-book) - F1 ↑ [sort]	QuAC - F1 ↑ [sort]	HellaSwag - EM ↑ [sort]	OpenbookQA - EM ↑ [sort]	TruthfulQA - EM ↑ [sort]
Cohere Command beta (52.4B)	**0.93**	0.452	0.856	**0.752**	0.372	0.76	0.432	0.811	0.582	0.269
text-davinci-002	0.93	0.568	0.877	0.727	0.383	0.713	0.445	0.815	0.594	**0.61**
text-davinci-	0.898	**0.569**	0.881	0.727	0.406	0.77	0.525	0.822	0.646	0.593

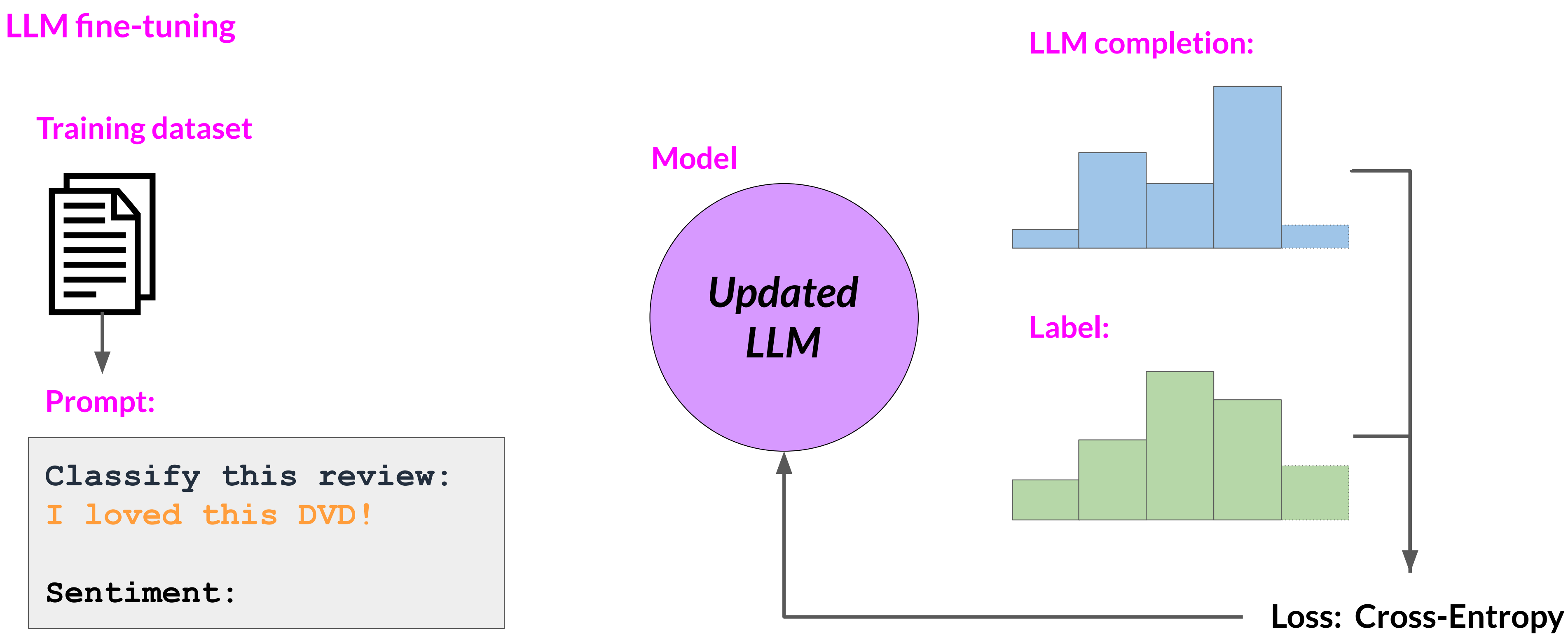
Disclaimer: metrics may not be up-to-date. Check https://crfm.stanford.edu/helm/latest for the latest.

Key takeaways

LLM fine-tuning process

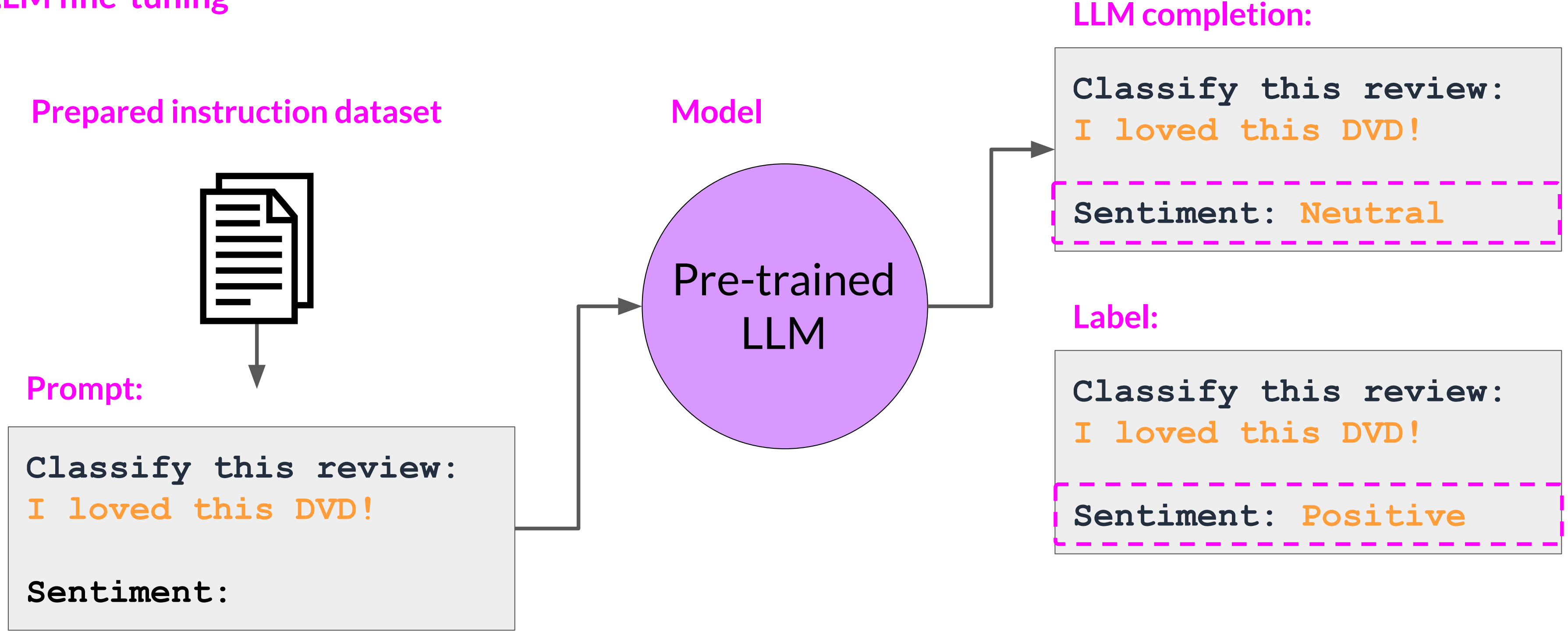


LLM fine-tuning process



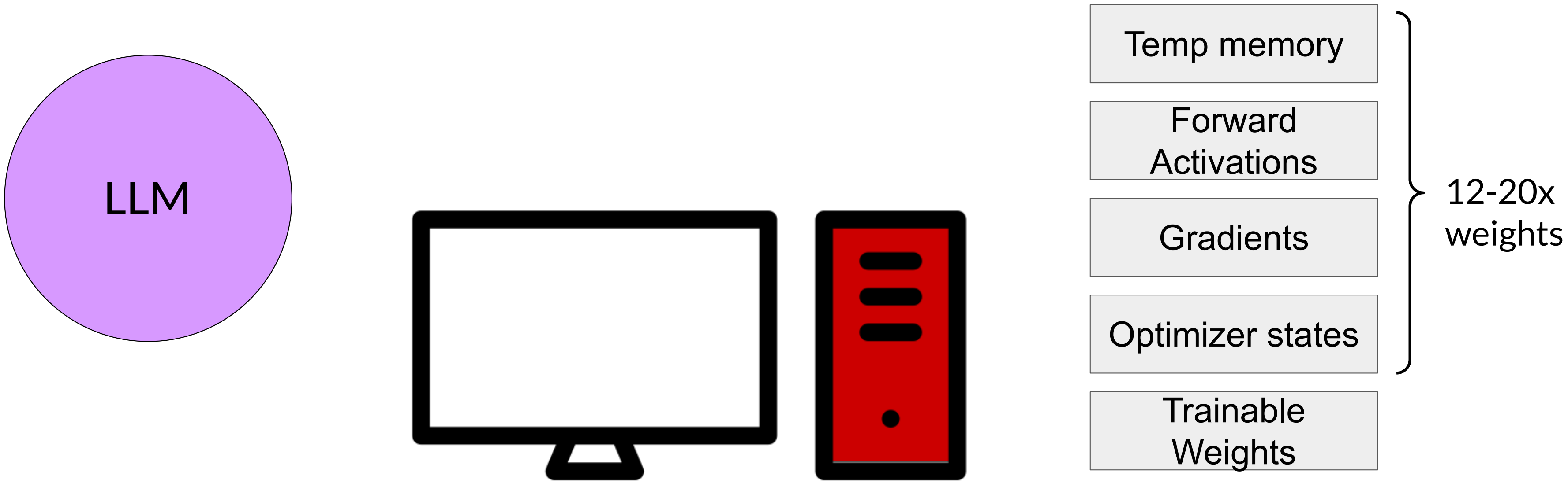
LLM fine-tuning process

LLM fine-tuning



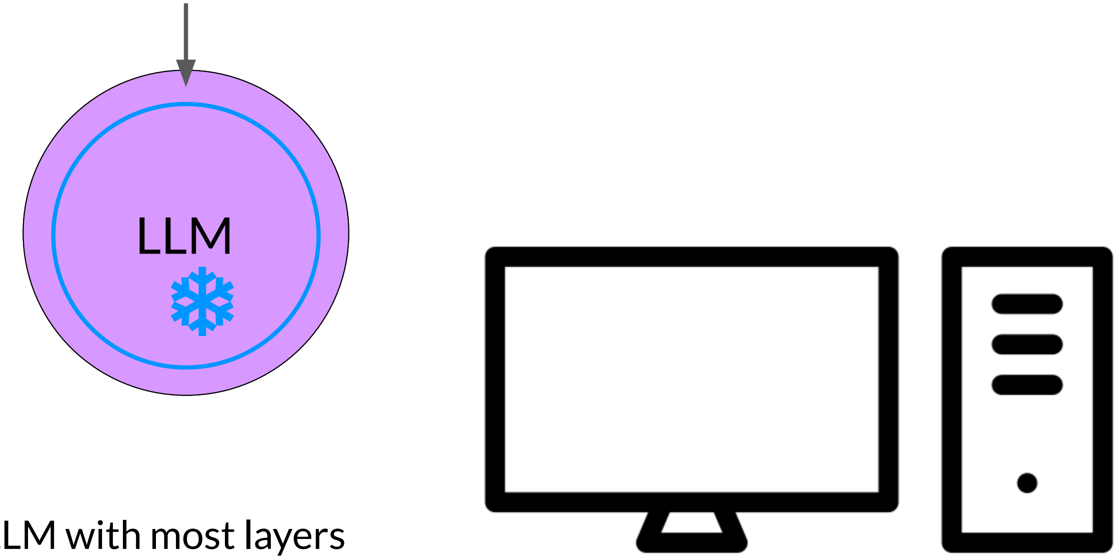
Parameter-efficient Fine-tuning (PEFT)

Full fine-tuning of large LLMs is challenging



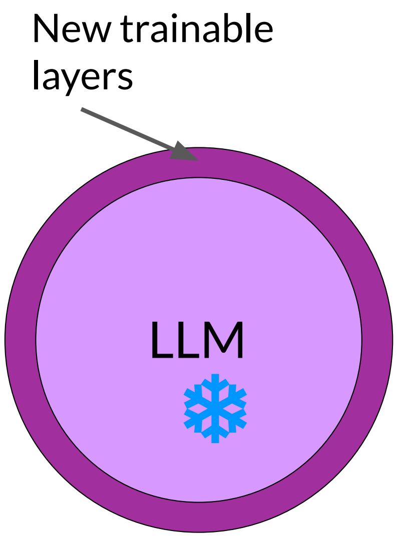
Parameter efficient fine-tuning (PEFT)

Small number of trainable layers

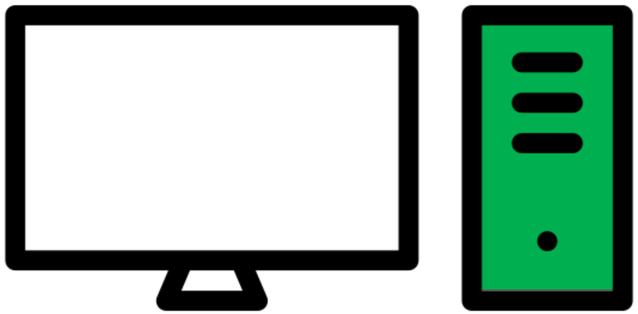


LLM with most layers frozen

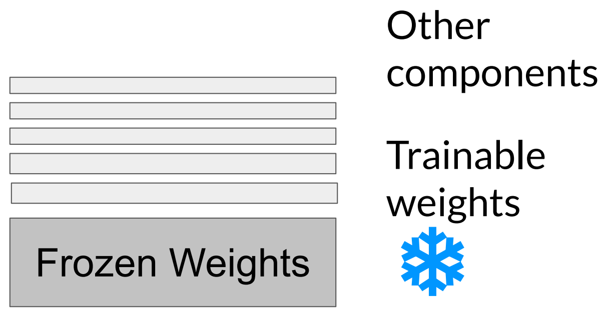
Parameter efficient fine-tuning (PEFT)



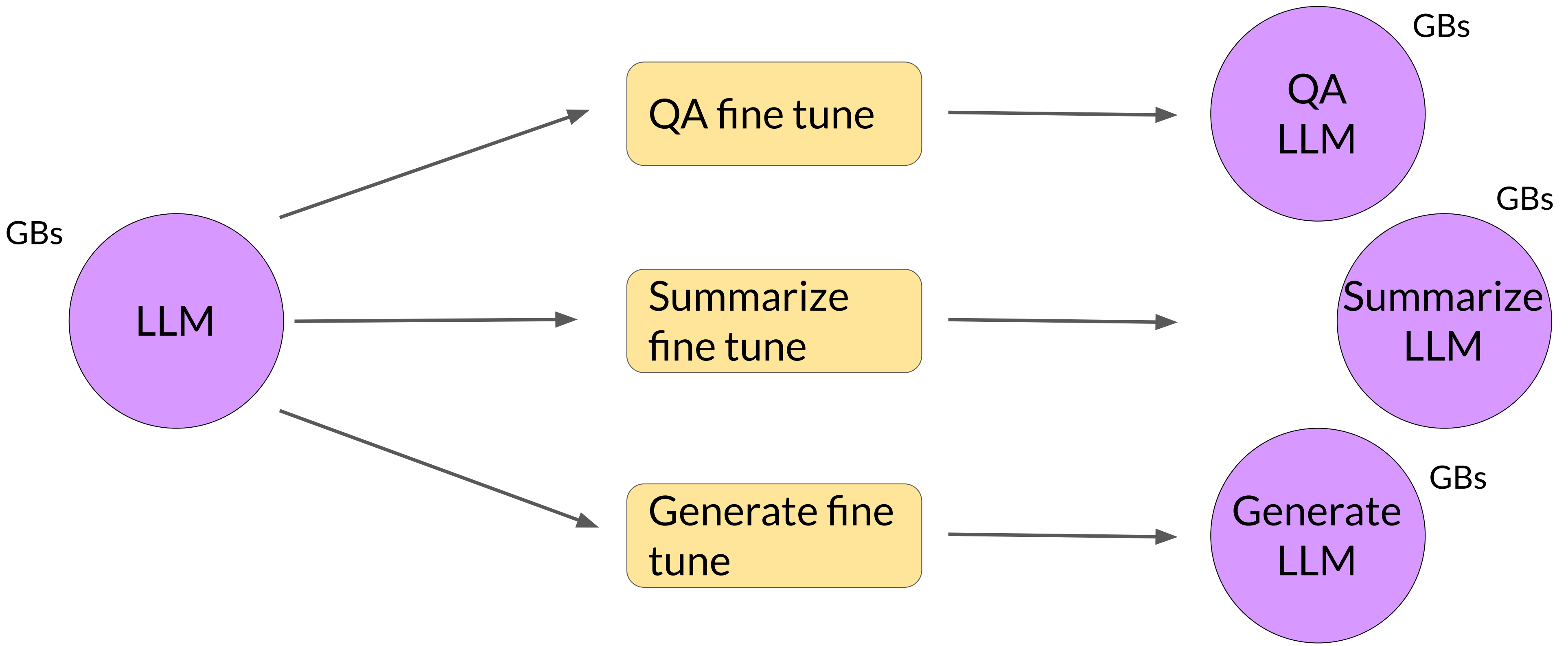
LLM with additional
layers for PEFT



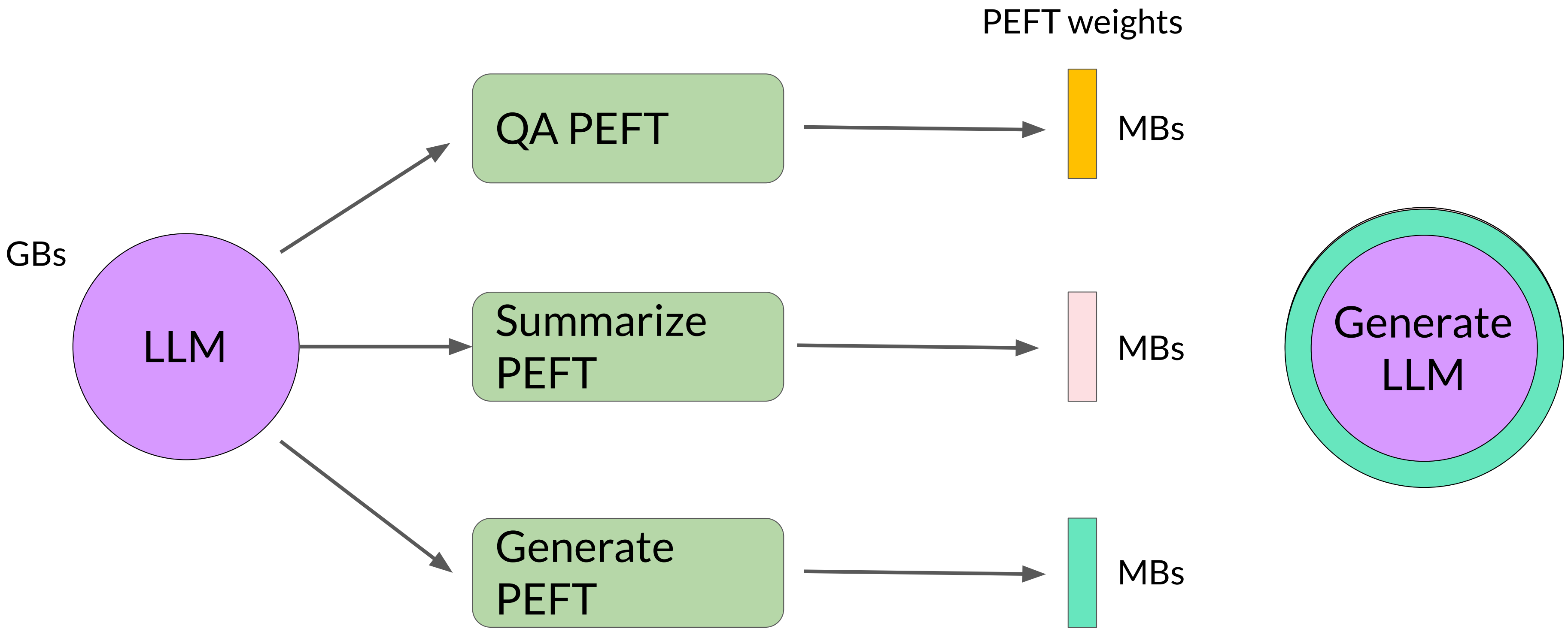
Less prone to
catastrophic forgetting



Full fine-tuning creates full copy of original LLM per task



PEFT fine-tuning saves space and is flexible

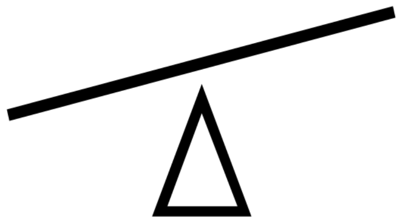


PEFT Trade-offs

Parameter Efficiency

Memory Efficiency

Training Speed



Model Performance

Inference Costs

PEFT methods

Selective

Select subset of initial LLM parameters to fine-tune

Reparameterization

Reparameterize model weights using a low-rank representation

Source: Lialin et al. 2023, “Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning”,

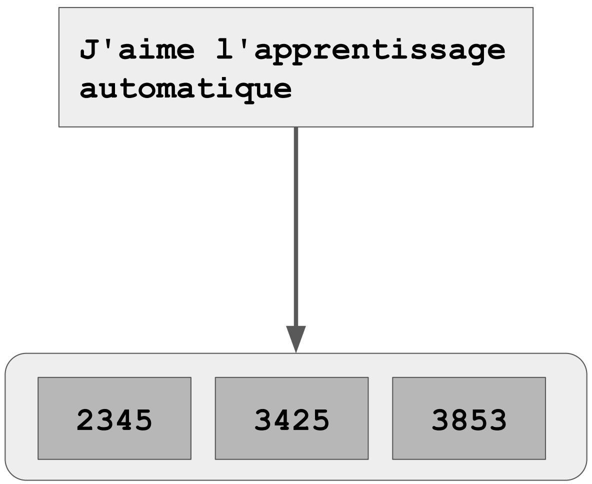
PEFT methods

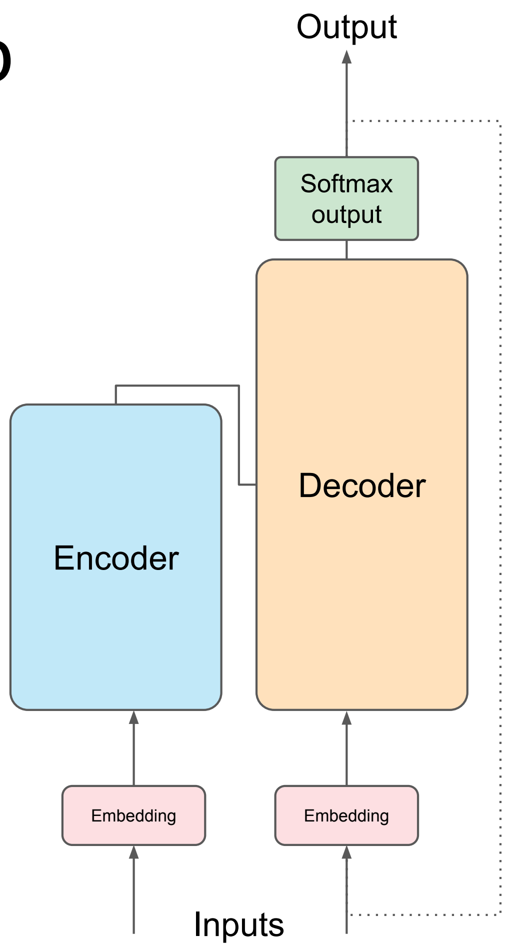
Selective	Reparameterization	Additive
Select subset of initial LLM parameters to fine-tune	**Reparameterize** model weights using a low-rank representation **LoRA**	**Add** trainable layers or parameters to model Adapters Soft Prompts **Prompt Tuning**

Source: Lialin et al. 2023, "Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning",

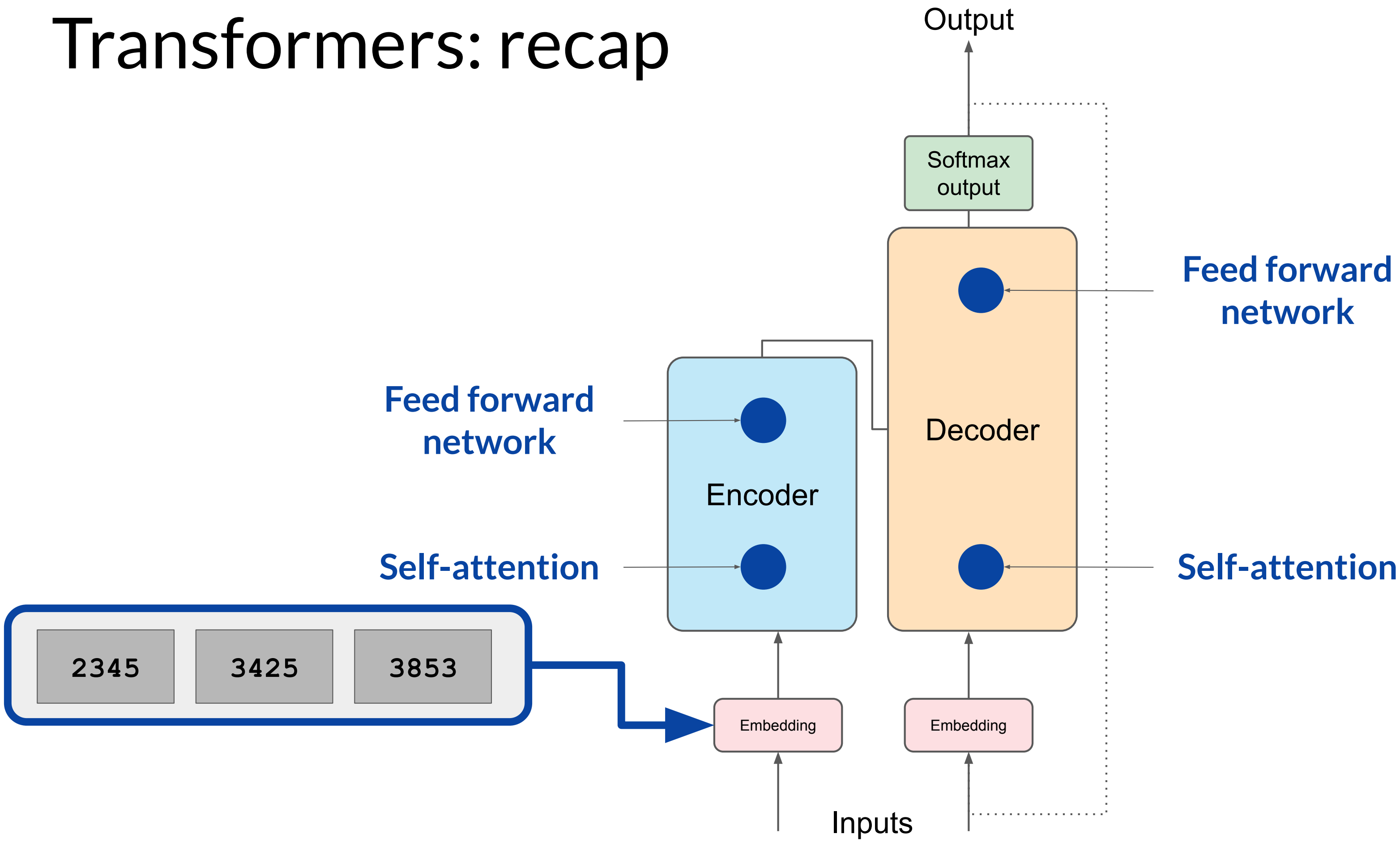
Low-Rank Adaptation of Large Language Models (LoRA)

Transformers: recap

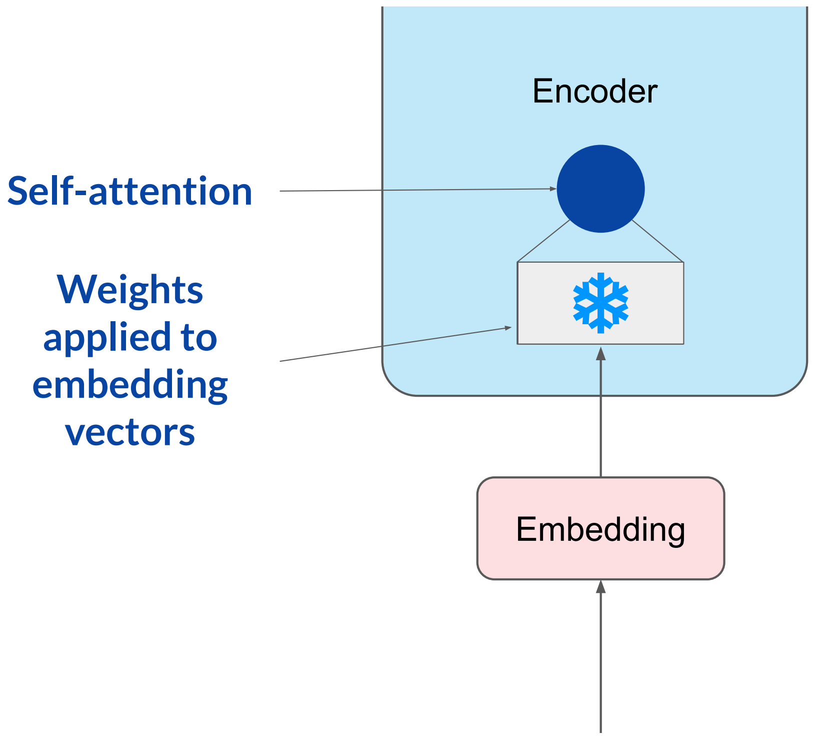




Transformers: recap

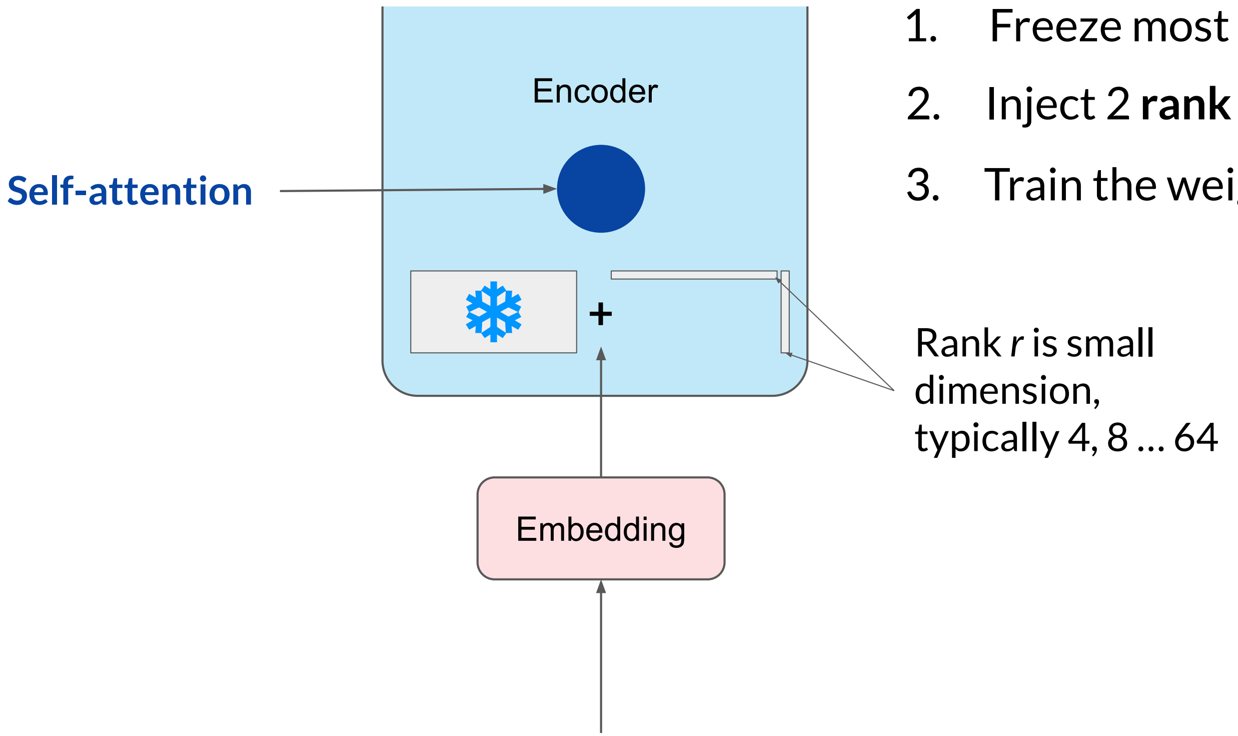


LoRA: Low Rank Adaption of LLMs



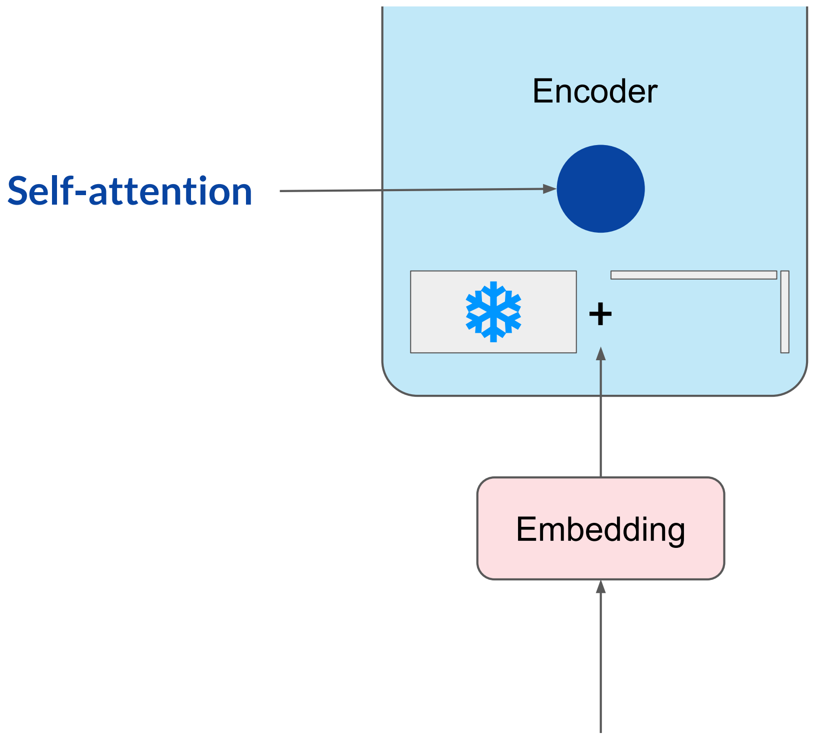
1. Freeze most of the original LLM weights.

LoRA: Low Rank Adaption of LLMs



1. Freeze most of the original LLM weights.
2. Inject 2 **rank decomposition matrices**
3. Train the weights of the smaller matrices

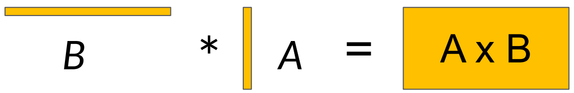
LoRA: Low Rank Adaption of LLMs



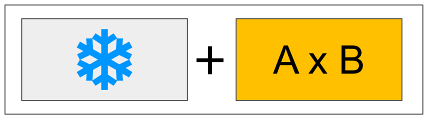
1. Freeze most of the original LLM weights.
2. Inject 2 **rank decomposition matrices**
3. Train the weights of the smaller matrices

Steps to update model for inference

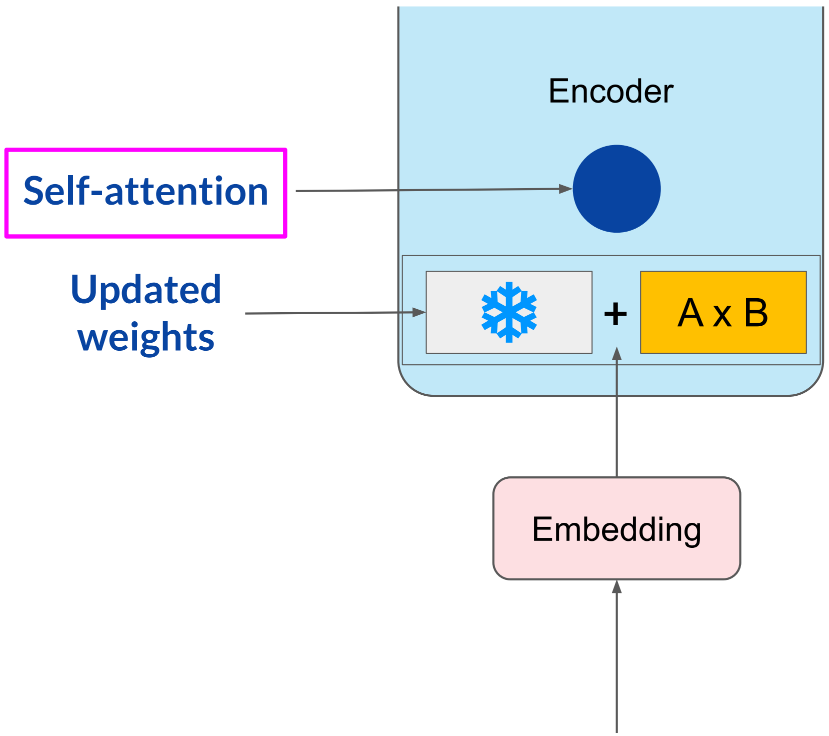
1. Matrix multiply the low rank matrices



2. Add to original weights



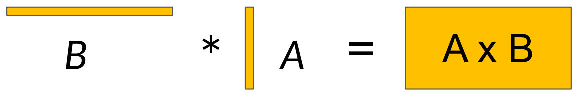
LoRA: Low Rank Adaption of LLMs



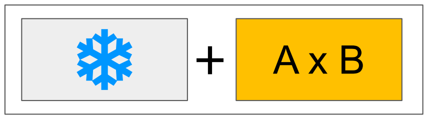
1. Freeze most of the original LLM weights.
2. Inject 2 **rank decomposition matrices**
3. Train the weights of the smaller matrices

Steps to update model for inference:

1. Matrix multiply the low rank matrices



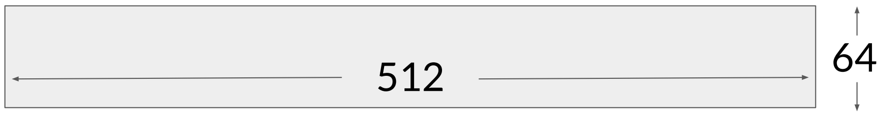
2. Add to original weights



Concrete example using base Transformer as reference

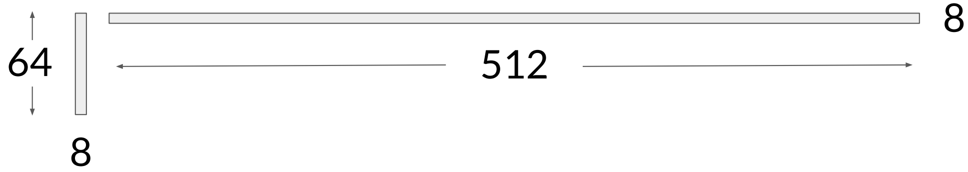
Use the base Transformer model presented by Vaswani et al. 2017:

- Transformer weights have dimensions $d \times k = 512 \times 64$
- So 512 x 64 = 32,768 trainable parameters



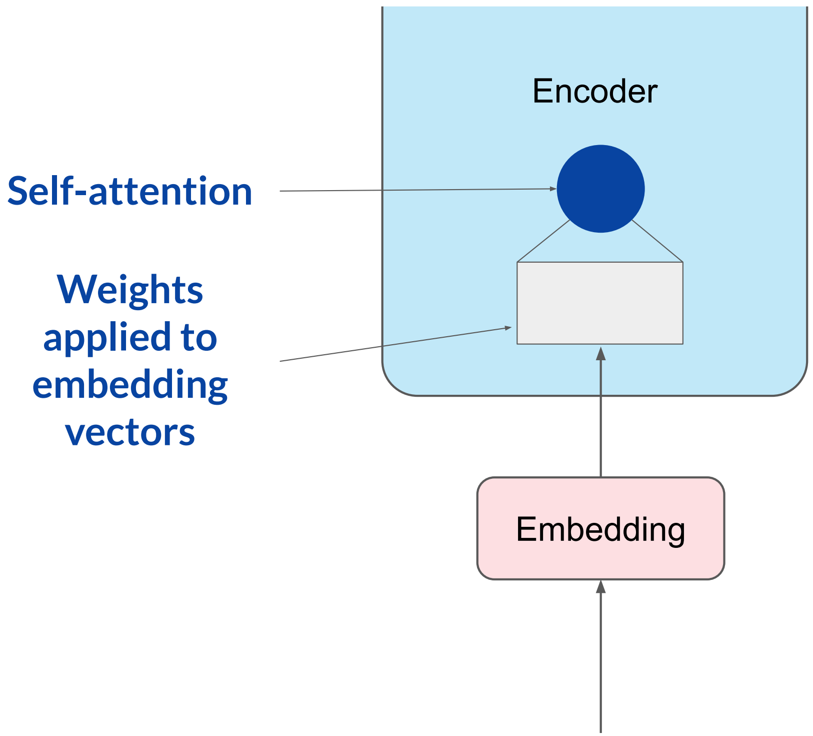
In LoRA with rank $r = 8$:

- A has dimensions $r \times k = 8 \times 64 = 512$ parameters
- B has dimension $d \times r = 512 \times 8 = 4{,}096$ trainable parameters



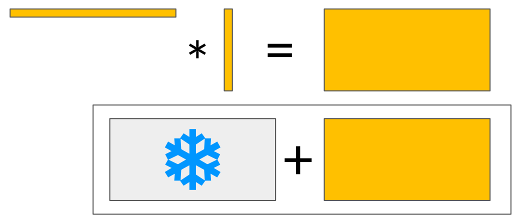
86% reduction in parameters to train!

LoRA: Low Rank Adaption of LLMs

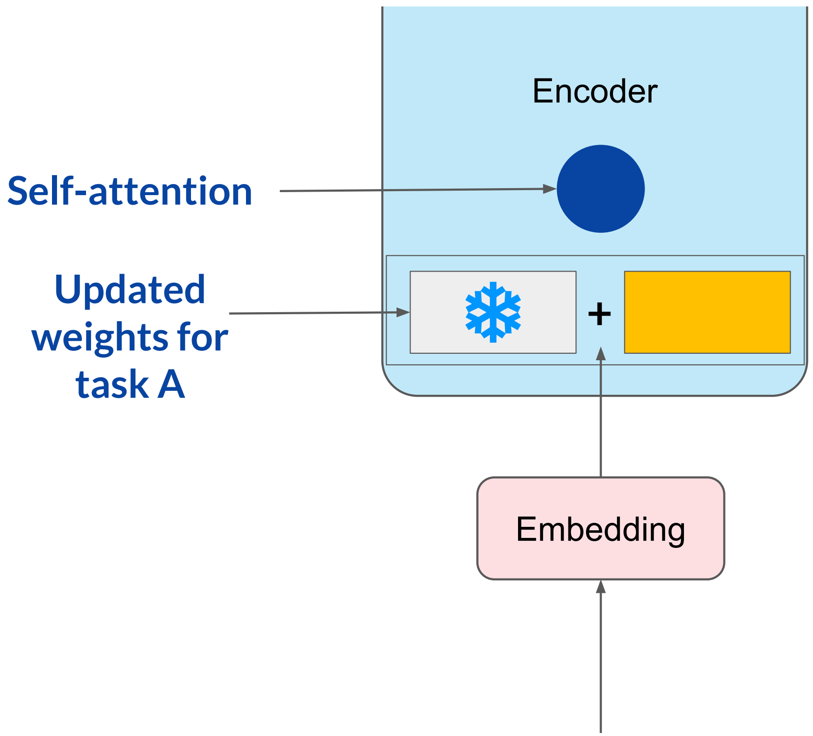


1. Train different rank decomposition matrices for different tasks
2. Update weights before inference

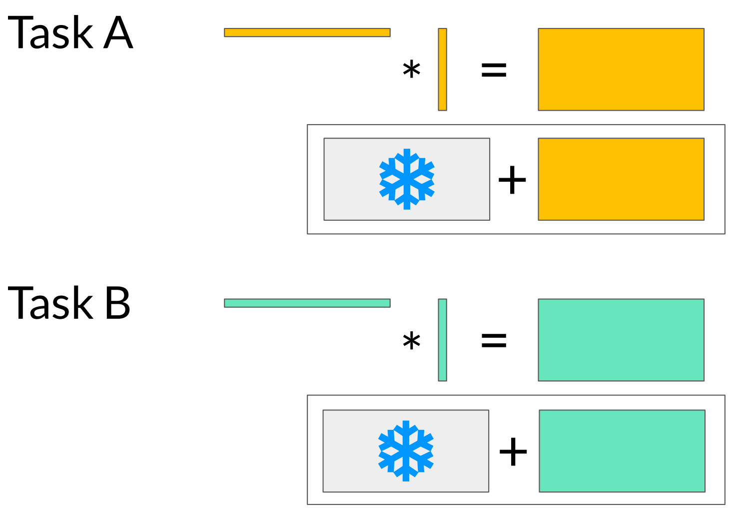
Task A



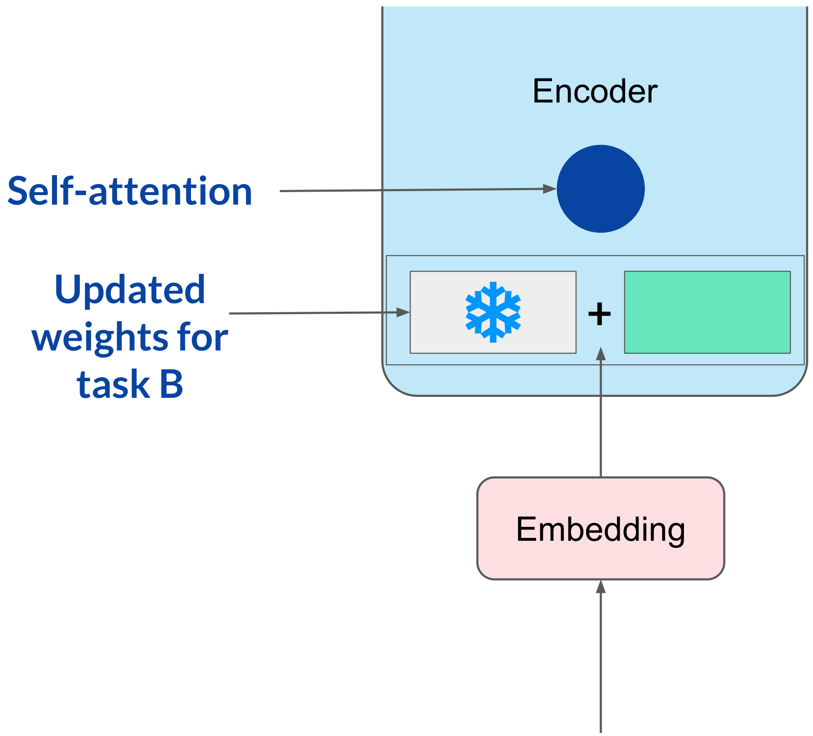
LoRA: Low Rank Adaption of LLMs



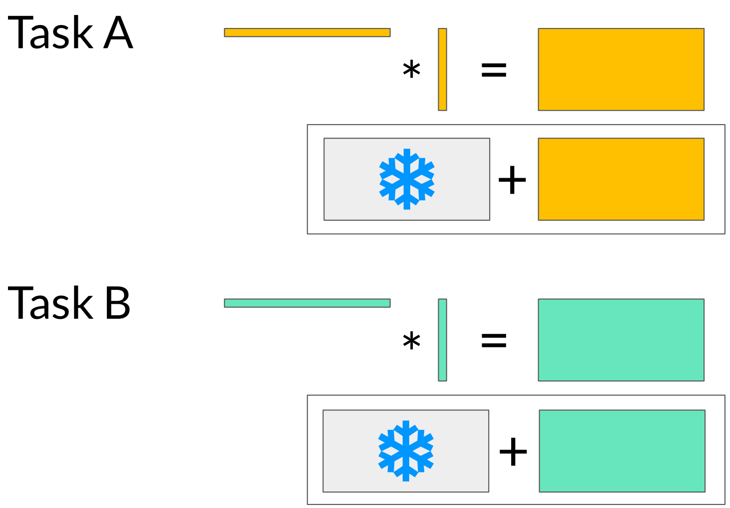
1. Train different rank decomposition matrices for different tasks
2. Update weights before inference



LoRA: Low Rank Adaption of LLMs



1. Train different rank decomposition matrices for different tasks
2. Update weights before inference



Sample ROUGE metrics for full vs. LoRA fine-tuning

Base model ROUGE

Full fine-tune ROUGE

FLAN-T5

Dialog summarization

```
flan_t5_base
{'rouge1': 0.2334,
 'rouge2': 0.0760,
 'rougeL': 0.2014,
 'rougeLsum': 0.2015}
```

Baseline score

Sample ROUGE metrics for full vs. LoRA fine-tuning

FLAN-T5

Dialog summarization

Base model ROUGE

```
flan_t5_base
{'rouge1': 0.2334,
 'rouge2': 0.0760,
 'rougeL': 0.2014,
 'rougeLsum': 0.2015}
```

Baseline score

+80.63%

Full fine-tune ROUGE

```
flan_t5_base_instruct_full
{'rouge1': 0.4216,
 'rouge2': 0.1804,
 'rougeL': 0.3384,
 'rougeLsum': 0.3384}
```

-3.20%

LoRA fine tune ROUGE

```
flan_t5_base_instruct_lora
{'rouge1': 0.4081,
 'rouge2': 0.1633,
 'rougeL': 0.3251,
 'rougeLsum': 0.3249}
```

Choosing the LoRA rank

Rank r	val_loss	BLEU	NIST	METEOR	ROUGE_L	CIDEr
1	1.23	68.72	8.7215	0.4565	0.7052	2.4329
2	1.21	69.17	8.7413	0.4590	0.7052	2.4639
4	1.18	**70.38**	**8.8439**	**0.4689**	0.7186	**2.5349**
8	1.17	69.57	8.7457	0.4636	**0.7196**	2.5196
16	**1.16**	69.61	8.7483	0.4629	0.7177	2.4985
32	**1.16**	69.33	8.7736	0.4642	0.7105	2.5255
64	**1.16**	69.24	8.7174	0.4651	0.7180	2.5070
128	**1.16**	68.73	8.6718	0.4628	0.7127	2.5030
256	**1.16**	68.92	8.6982	0.4629	0.7128	2.5012
512	**1.16**	68.78	8.6857	0.4637	0.7128	2.5025
1024	1.17	69.37	8.7495	0.4659	0.7149	2.5090

- Effectiveness of higher rank appears to plateau
- Relationship between rank and dataset size needs more empirical data

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

QLoRA: Quantized LoRA

- Introduces 4-bit NormalFloat (nf4) data type for 4-bit quantization
- Supports double-quantization to reduce memory ~0.4 bits per parameter (~3 GB for a 65B model)
- Unified GPU-CPU memory management reduces GPU memory usage
- LoRA adapters at every layer - not just attention layers
- Minimizes accuracy trade-off

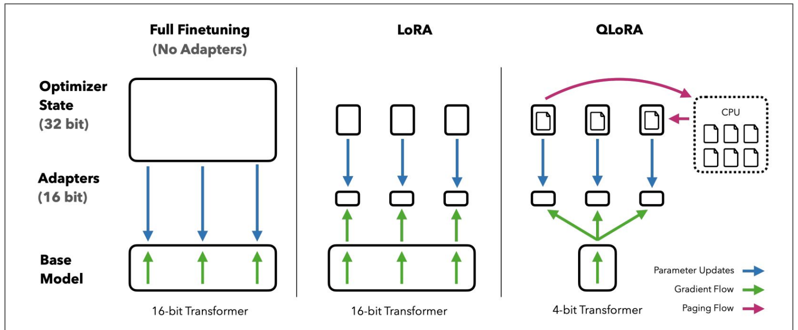
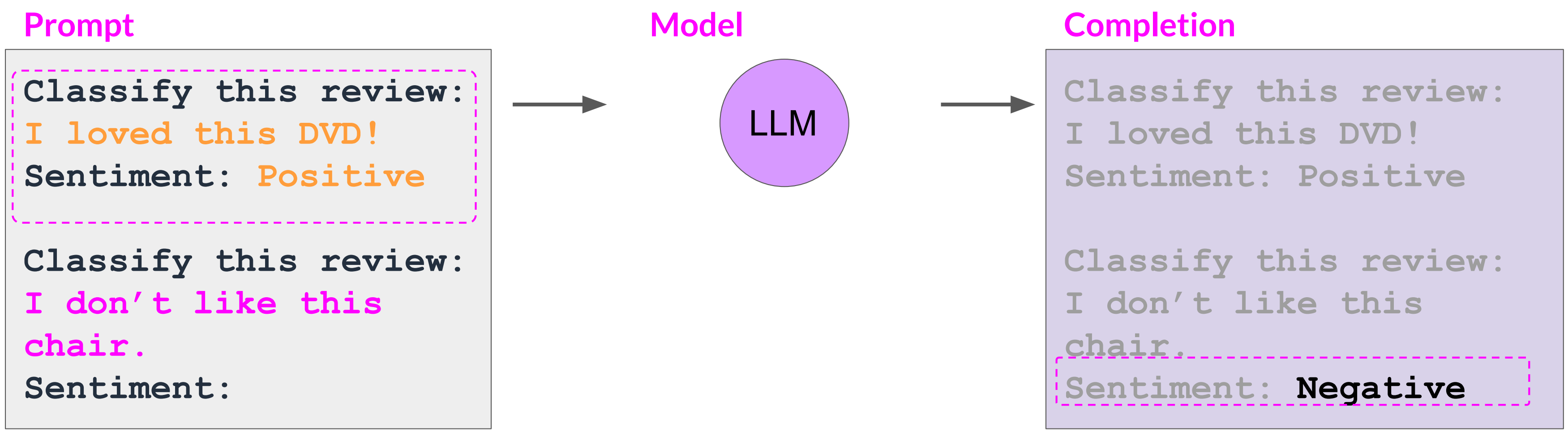


Figure 1: Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.

Source: Dettmers et al. 2023, "QLoRA: Efficient Finetuning of Quantized LLMs"

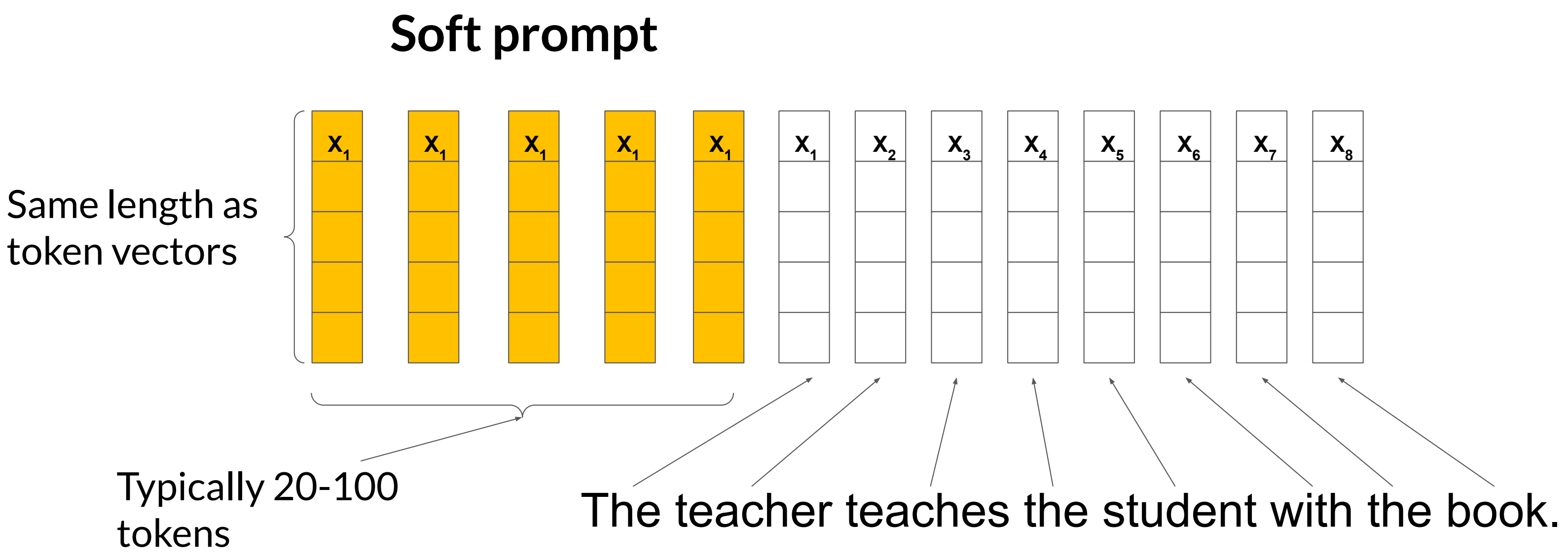
Prompt tuning with soft prompts

Prompt tuning is **not** prompt engineering!

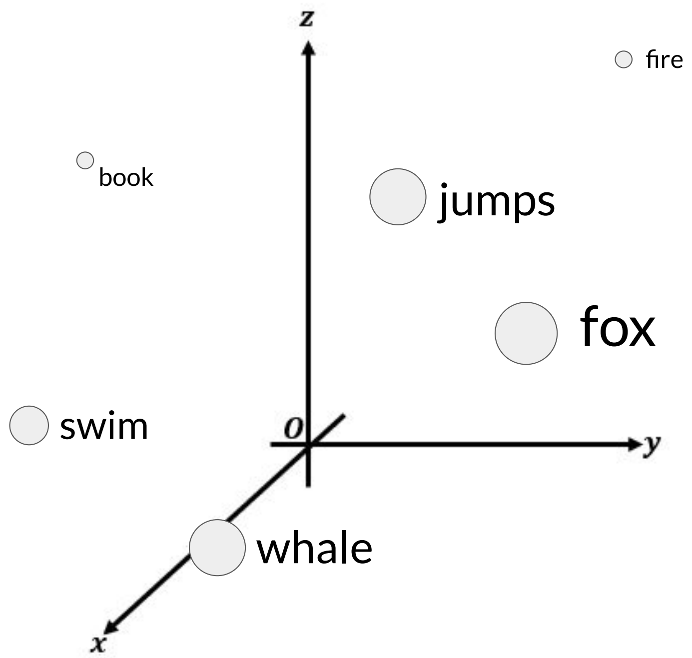


One-shot or Few-shot Inference

Prompt tuning adds trainable "soft prompt" to inputs

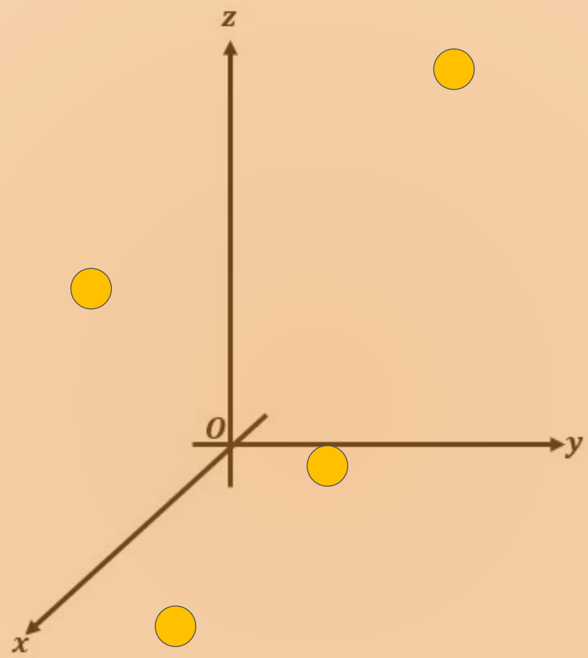


Soft prompts



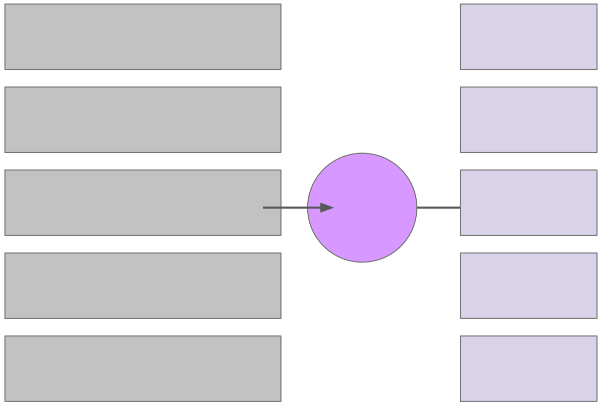
Embeddings of each token exist at unique point in multi-dimensional space

Soft prompts



Full Fine-tuning vs prompt tuning

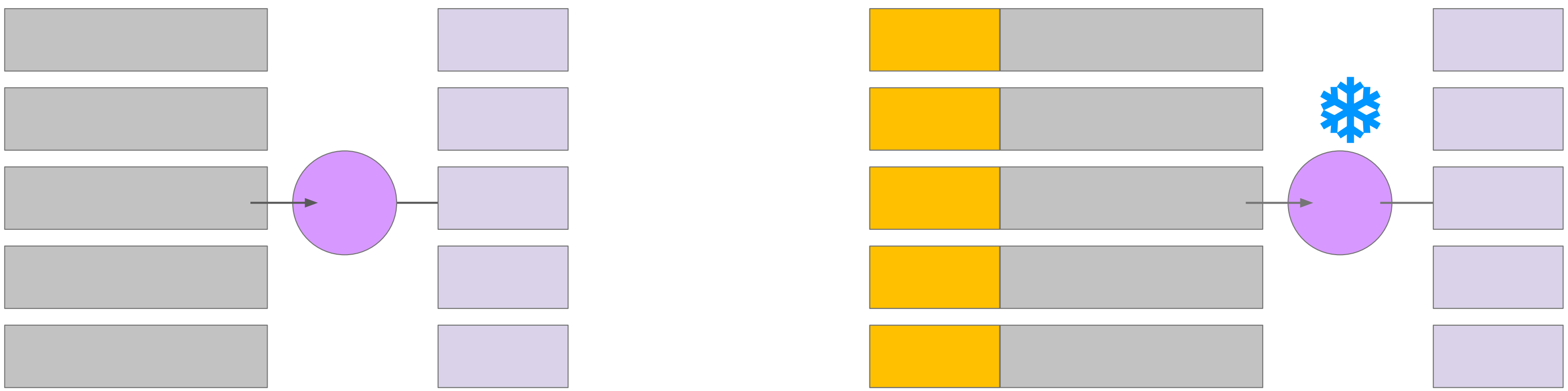
Weights of model updated
during training



Full Fine-tuning vs prompt tuning

Weights of model updated
during training

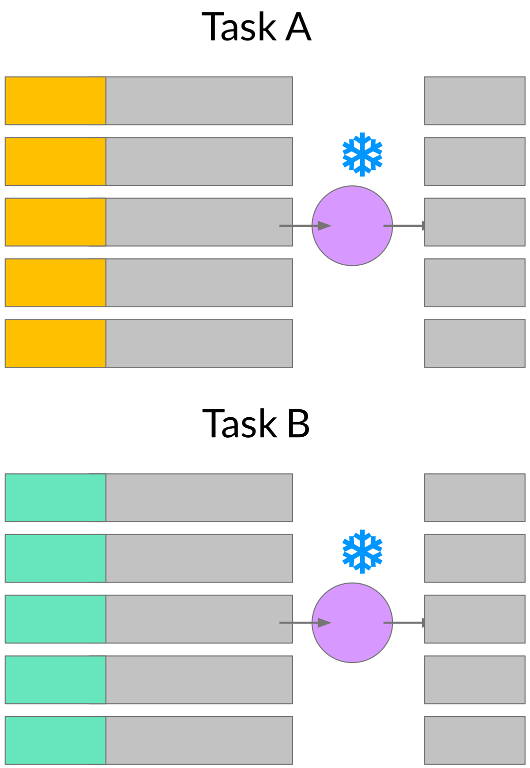
Weights of model frozen and
soft prompt trained



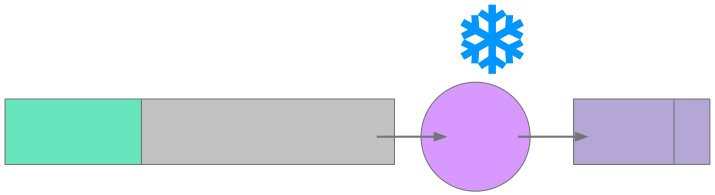
Millions to Billions of
parameter updated

10K - 100K of parameters
updated

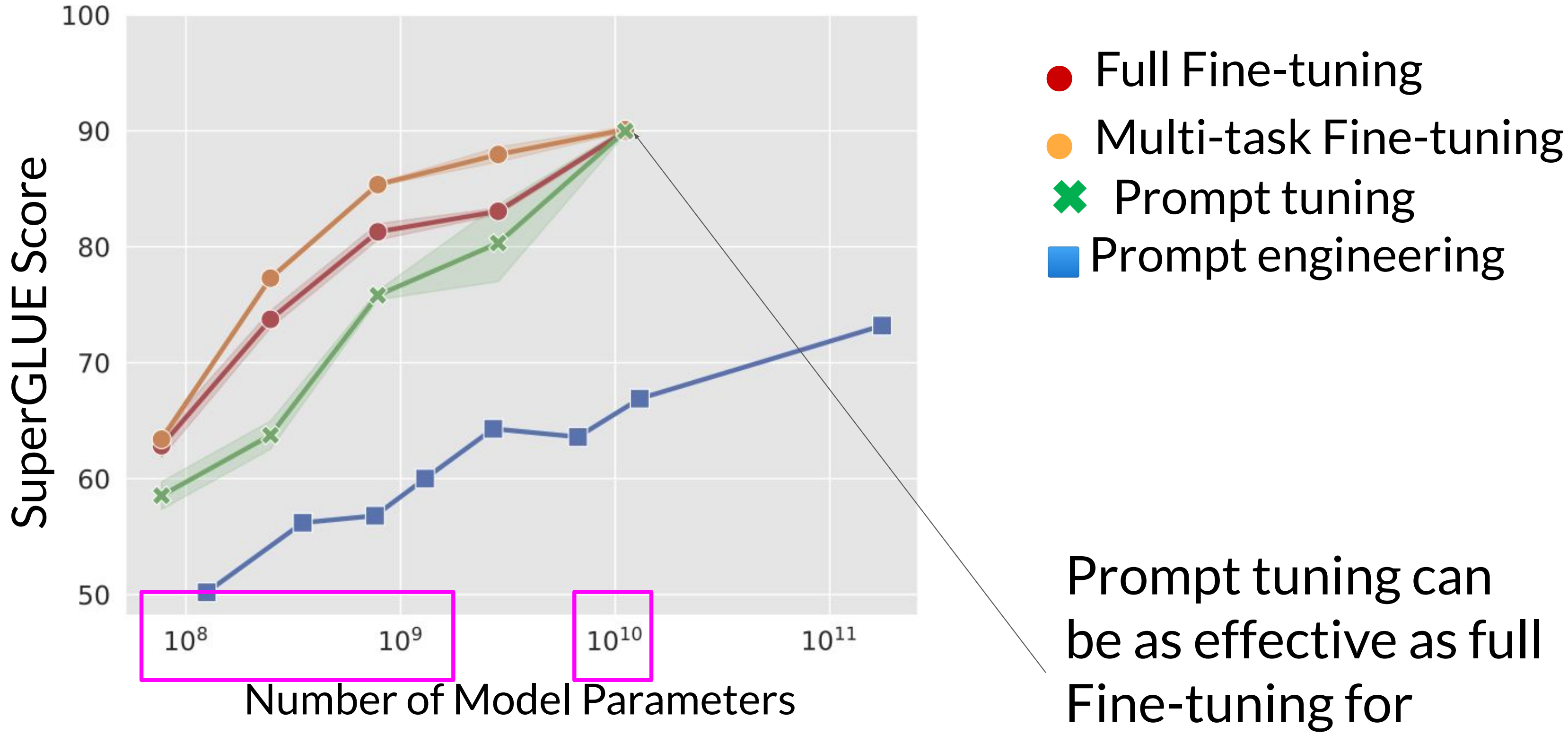
Prompt tuning for multiple tasks



Switch out soft prompt at inference time to change task!



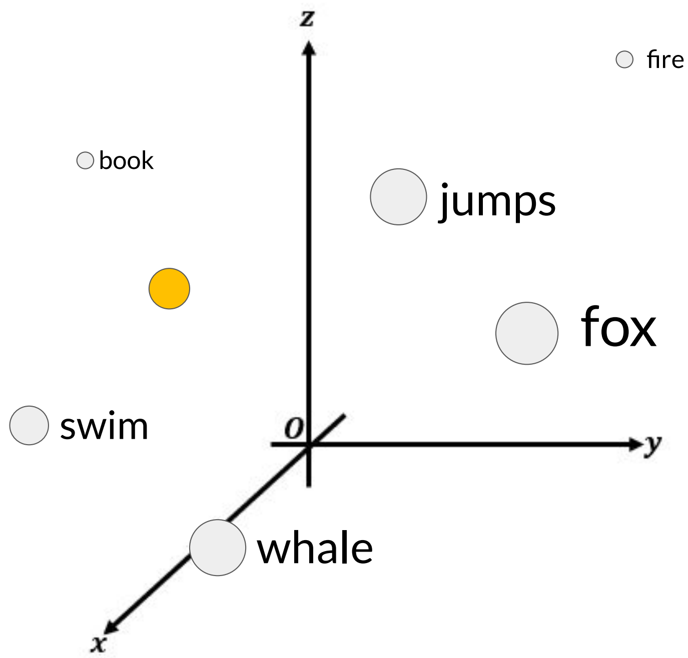
Performance of prompt tuning



Prompt tuning can be as effective as full Fine-tuning for larger models!

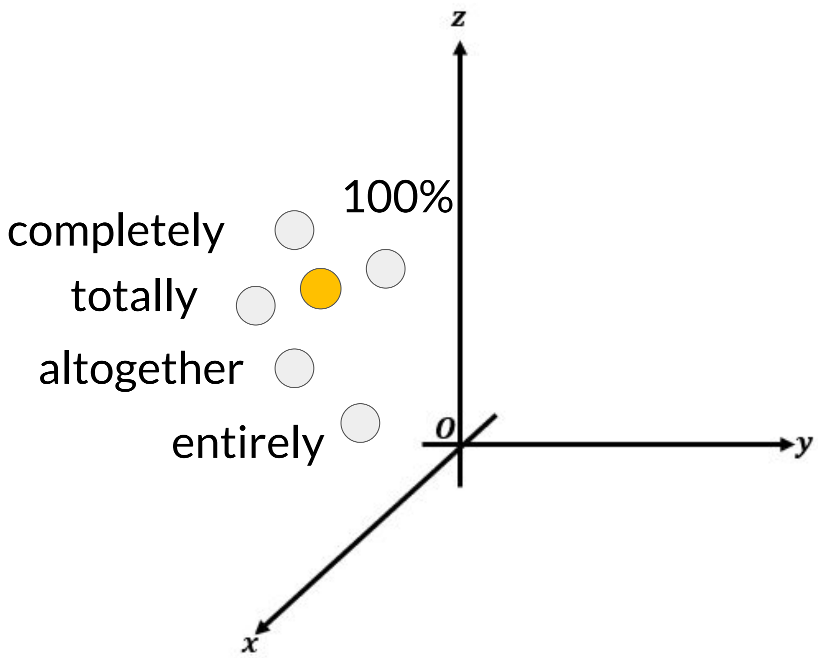
Source: Lester et al. 2021, "The Power of Scale for Parameter-Efficient Prompt Tuning"

Interpretability of soft prompts



Trained soft-prompt embedding does not correspond to a known token…

Interpretability of soft prompts



...but nearest neighbors form a semantic group with similar meanings.

PEFT methods summary

Selective

Select subset of initial LLM parameters to fine-tune

Reparameterization

Reparameterize model weights using a low-rank representation

LoRA

Additive

Add trainable layers or parameters to model

Adapters

Soft Prompts

Prompt Tuning