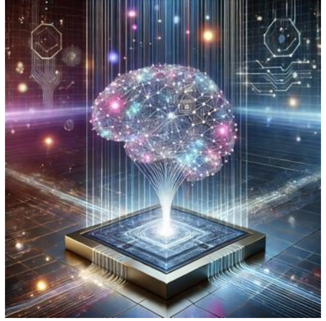
Multimodal Generative AI 2025
Fine-tuning Foundation Models

VISTA
École Polytechnique

Vicky Kalogeiton


created with ChatGPT, Oct 2024

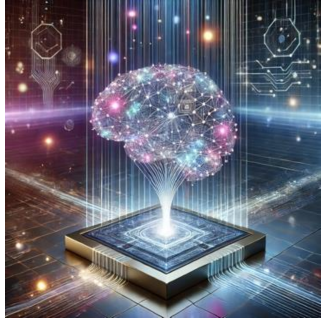
Today's lecture

Part I:
Definition:
Define usecase

Part II:
Select:
Foundation Model to use or pretrain

Part III:
Adapt:
Foundation Model

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]



created with ChatGPT, Oct 2024

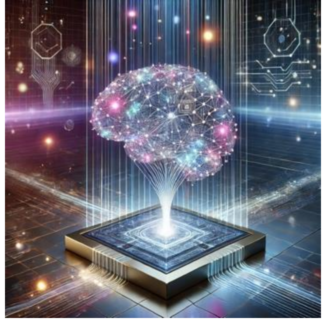
Today's lecture

Part I:
Definition:
Define usecase

Part II:
Select:
Foundation Model to use or pretrain

Part III:
Adapt:
Foundation Model

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]



created with ChatGPT, Oct 2024

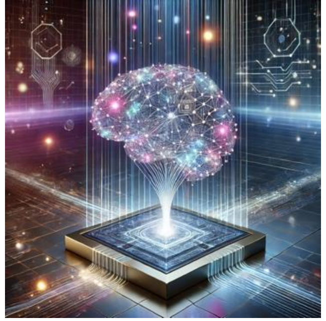
Today's lecture

Part I:
Definition:
Define
usecase

Part II:
Select:
Foundation
Model to use
or pretrain

Part III:
Adapt:
Foundation
Model

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]



created with ChatGPT, Oct 2024

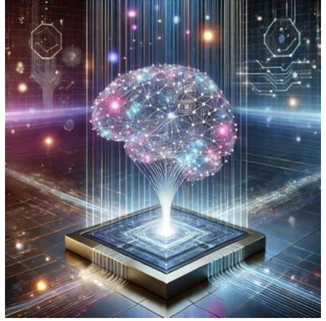
Today's lecture

Part I:
Definition:
Define usecase

Part II:
Select:
Foundation Model to use or pretrain

Part III:
Adapt:
Foundation Model

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]



created with ChatGPT, Oct 2024

Today's lecture

Part I:
Definition:
Define usecase

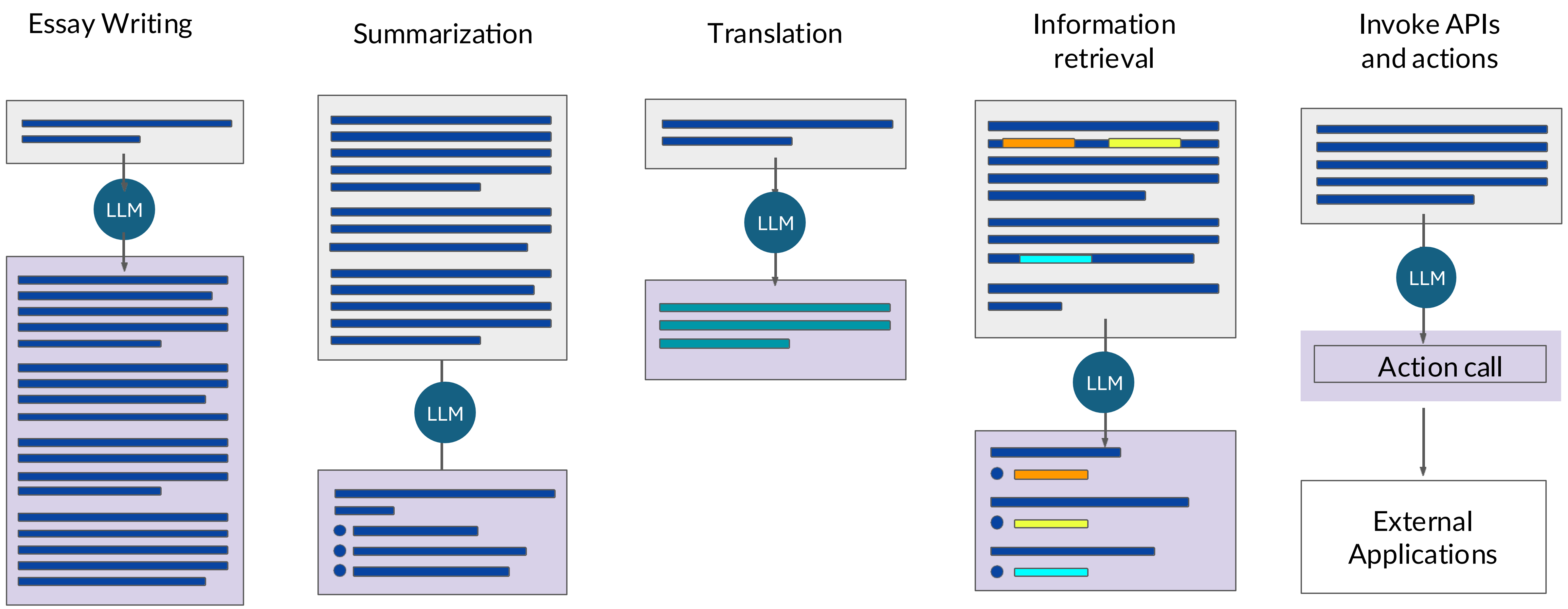
Part II:
Select:
Foundation Model to use or pretrain

Part III:
Adapt:
Foundation Model

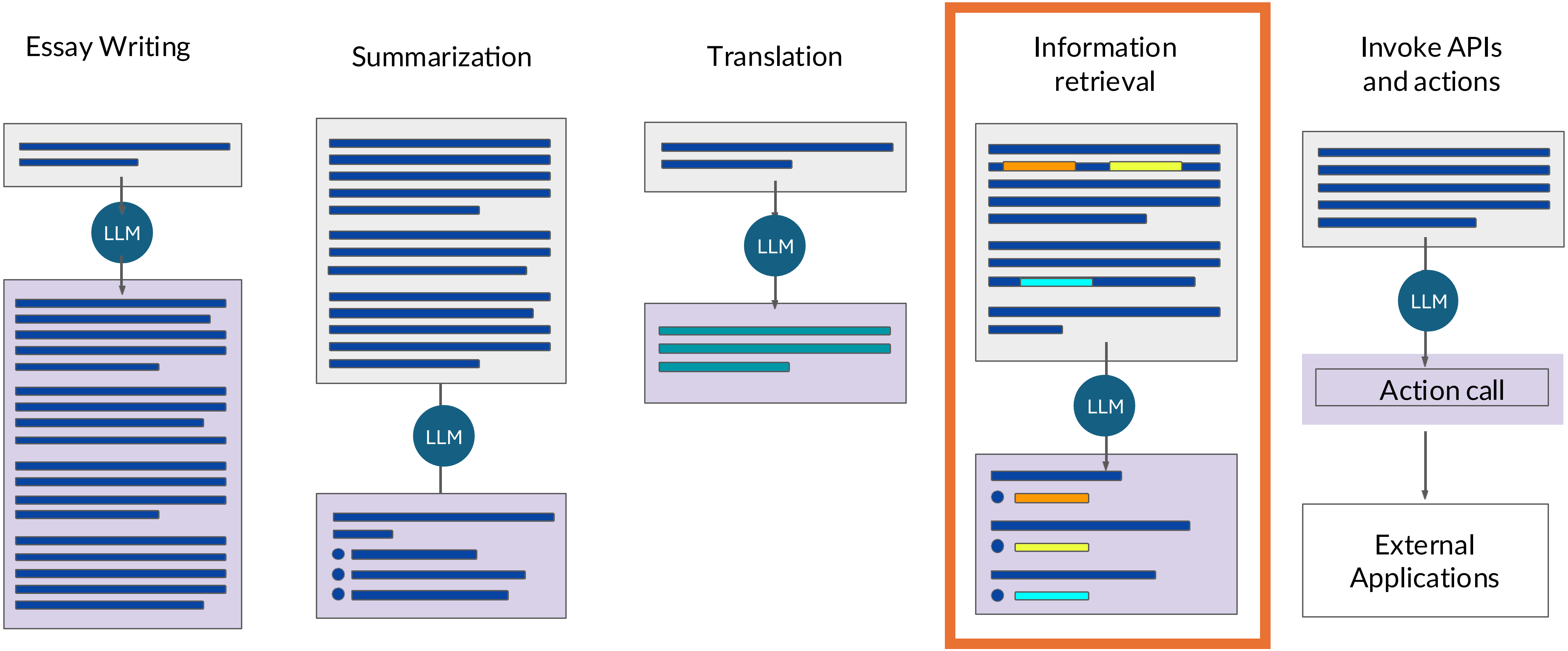
Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]

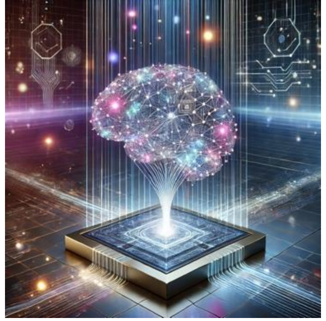
Generative AI project Scope

Good at many tasks?



Or a single task?





created with ChatGPT, Oct 2024

Today's lecture

Part I:
Definition:
Define usecase

Part II:
Select:
Foundation Model to use or pretrain

Part III:
Adapt:
Foundation Model

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]

Part II: Outline

Select

- Choose an existing model or pretrain your own

- Scaling
  - Challenges
  - Cost
  - Scaling laws

- Pre-training for domain adaptation

created with chatGPT

Part II: Outline

Select

- **Choose an existing model or pretrain your own**

- Scaling
  - Challenges
  - Cost
  - Scaling laws

- Pre-training for domain adaptation

created with chatGPT

Considerations for choosing a model

Foundation model

Pretrained
LLM

Train your own model

Custom
LLM

Model hubs

Model Card for T5 Large

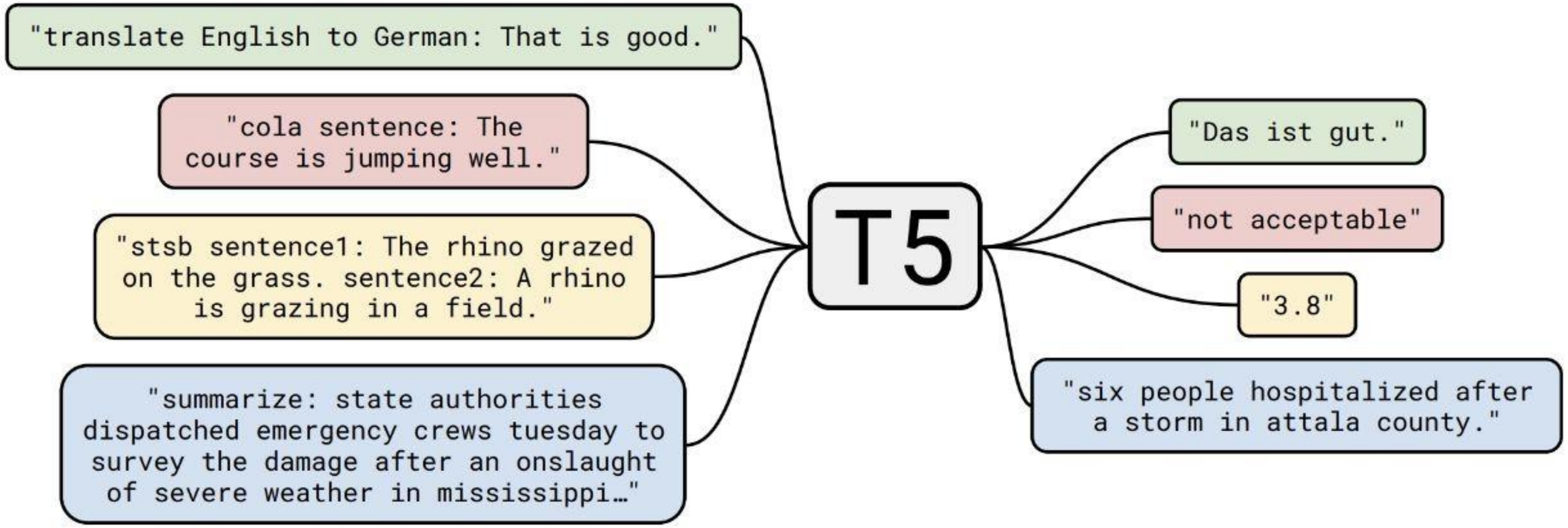


Table of Contents

Considerations for choosing a model

Foundation model

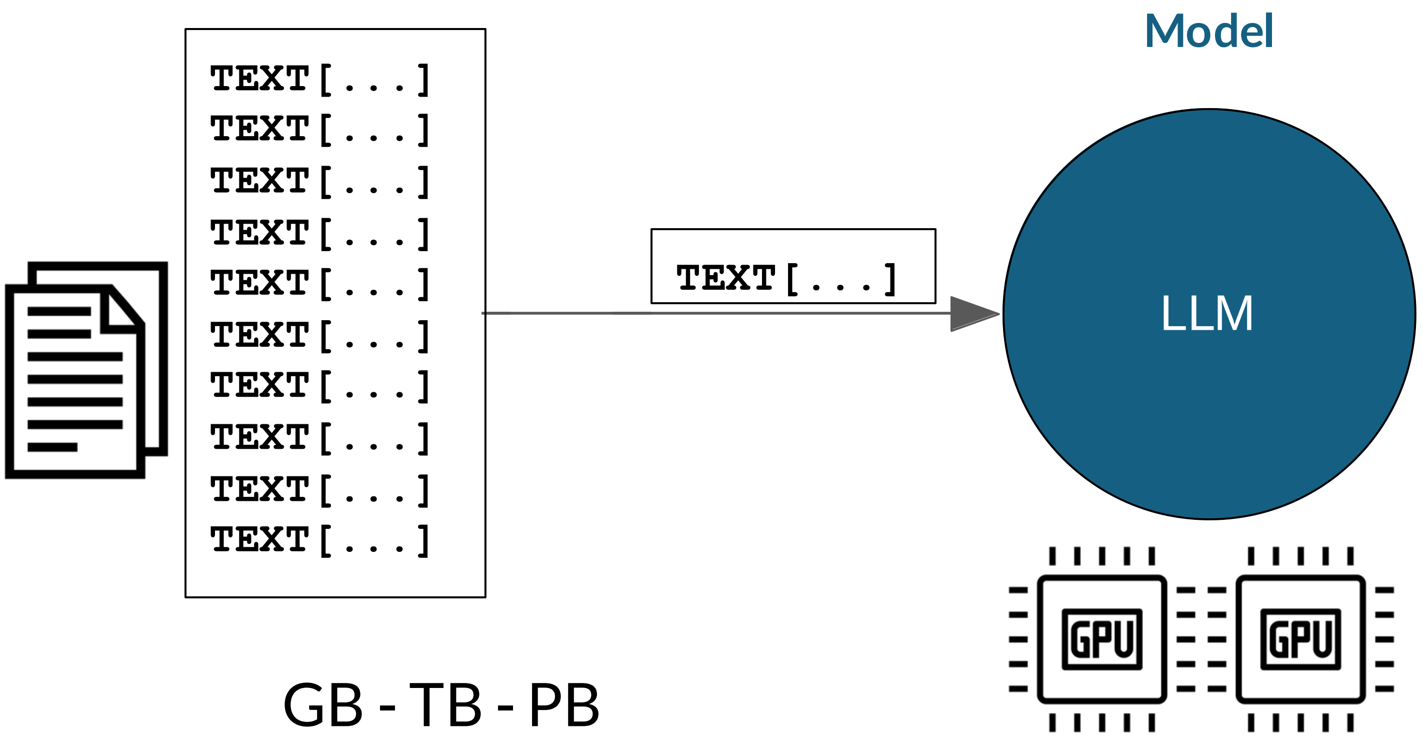
Pretrained LLM

Train your own model

Custom LLM

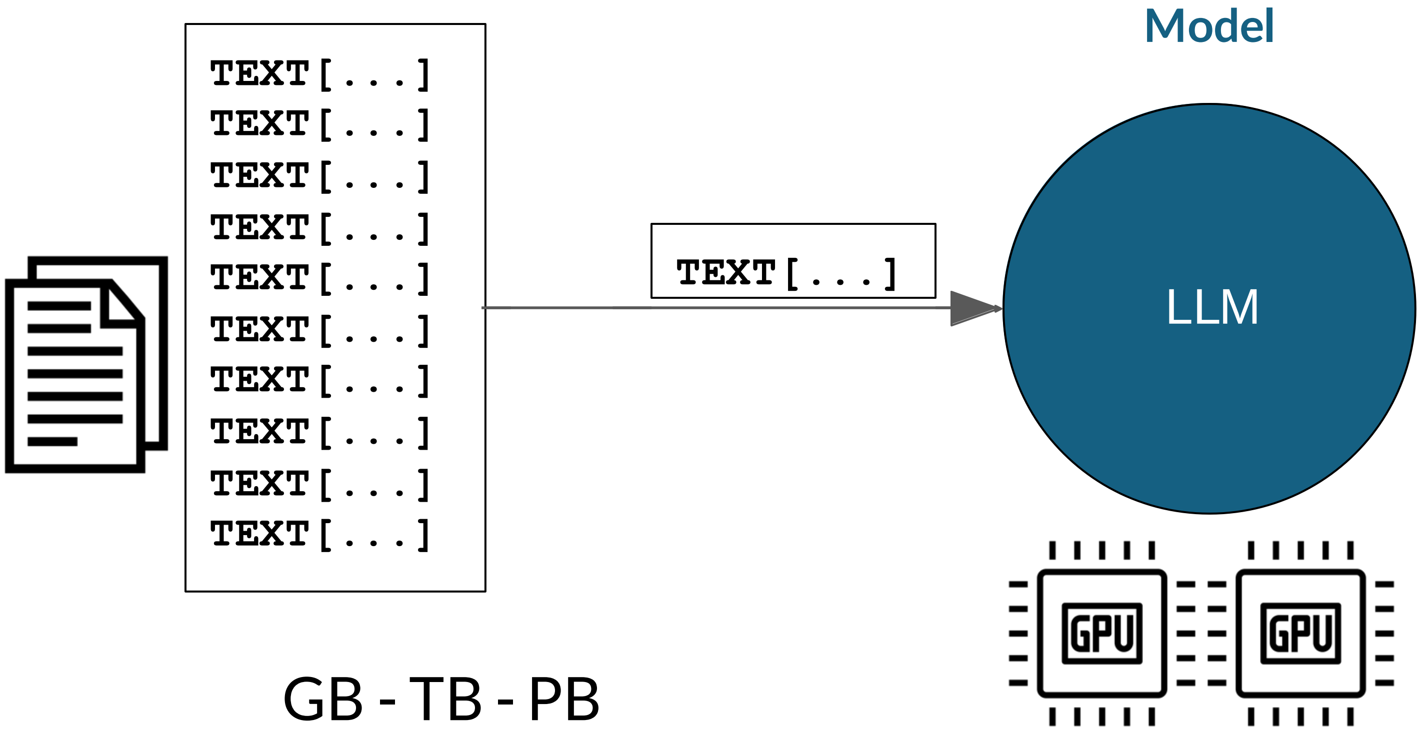
Pre-training objectives

LLM pre-training at a high level



GB - TB - PB
of unstructured data

LLM pre-training at a high level

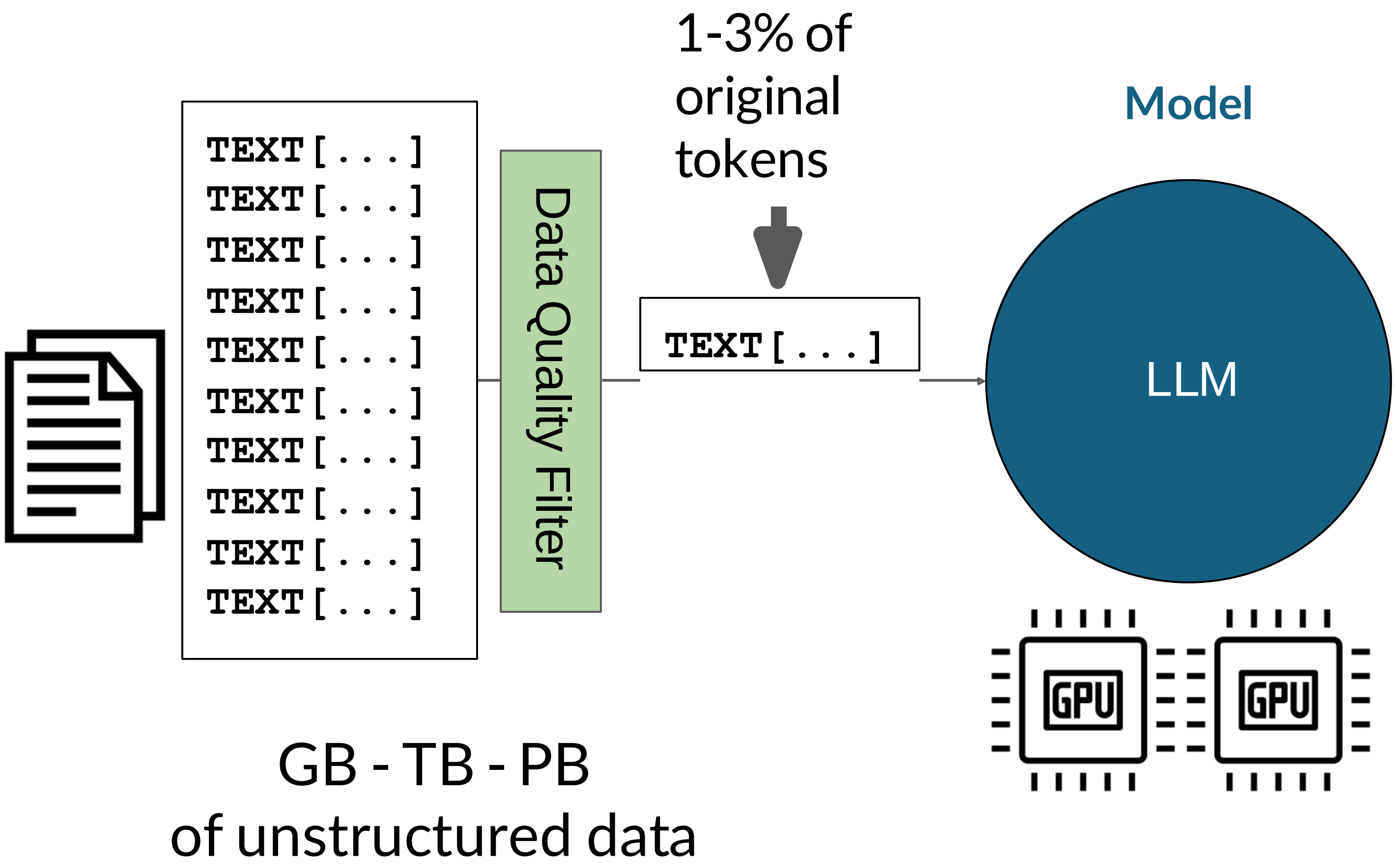


GB - TB - PB
of unstructured data

Token String	Token ID	Embedding / Vector Representation
'_The'	37	[-0.0513, -0.0584, 0.0230, ...]
'_teacher'	3145	[-0.0335, 0.0167, 0.0484, ...]
'_teaches'	11749	[-0.0151, -0.0516, 0.0309, ...]
'_the'	8	[-0.0498, -0.0428, 0.0275, ...]
'_student'	1236	[-0.0460, 0.0031, 0.0545, ...]
...	...	...

Vocabulary

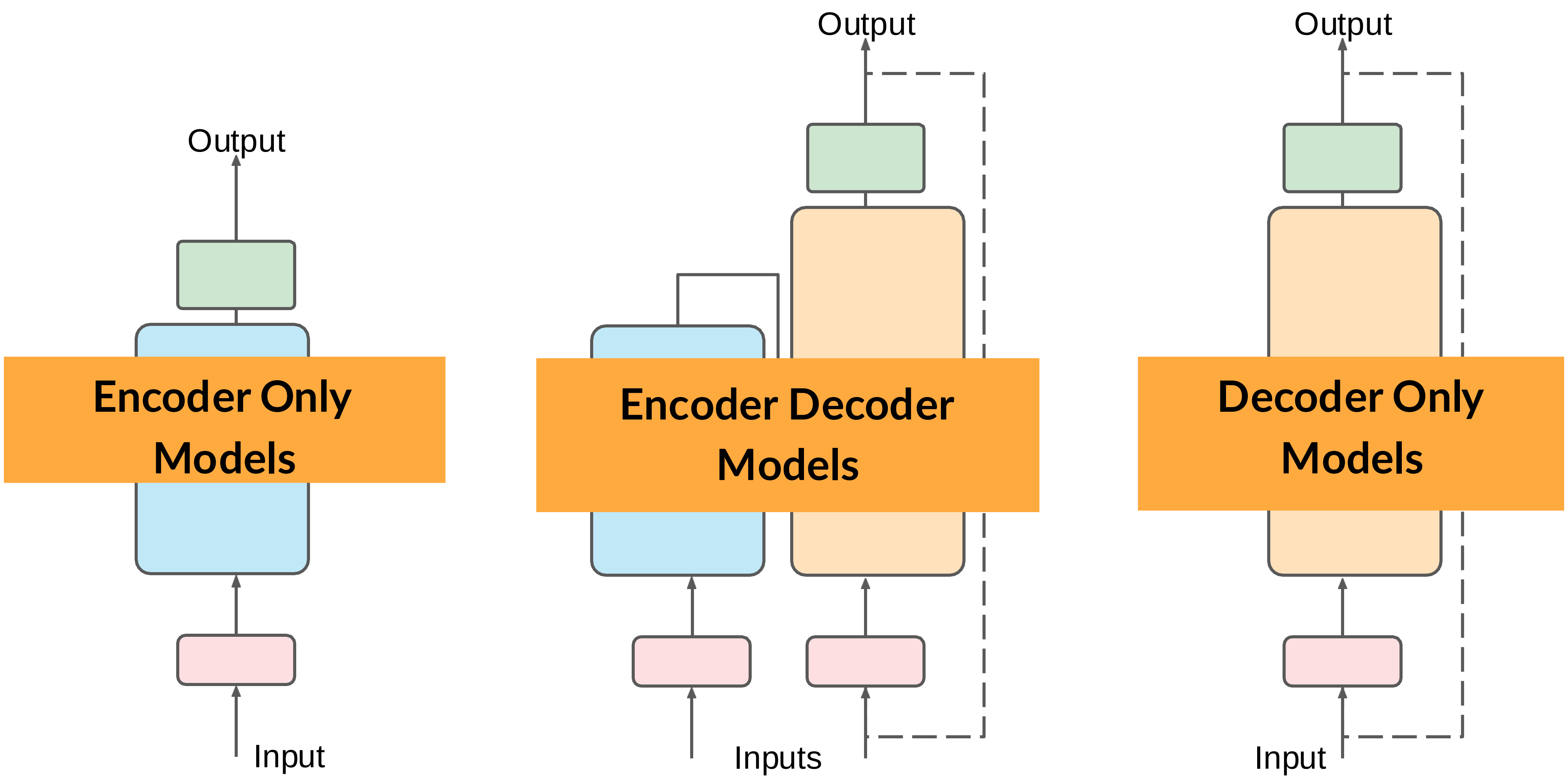
LLM pre-training at a high level



Token String	Token ID	Embedding / Vector Representation
'_The'	37	[-0.0513, -0.0584, 0.0230, ...]
'_teacher'	3145	[-0.0335, 0.0167, 0.0484, ...]
'_teaches'	11749	[-0.0151, -0.0516, 0.0309, ...]
'_the'	8	[-0.0498, -0.0428, 0.0275, ...]
'_student'	1236	[-0.0460, 0.0031, 0.0545, ...]
...	...	...

Vocabulary

Transformers



Autoencoding models

Original text

```
The teacher
teaches the
student.

[...]
```

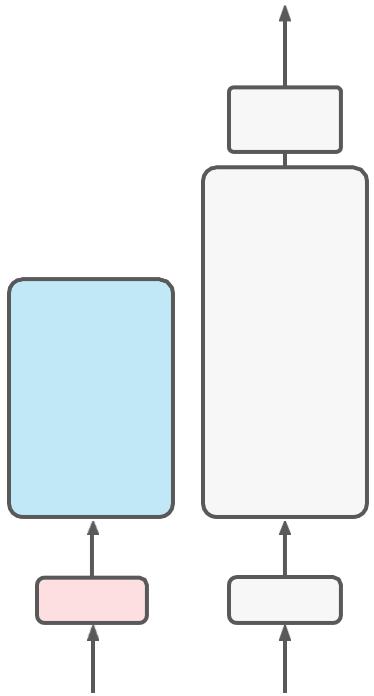
Masked Language Modeling (MLM)

The	teacher	**<MASK>**	the	student

Encoder-only LLM

Objective: Reconstruct text ("denoising")

The	teacher	**teaches**	the	student



Bidirectional context

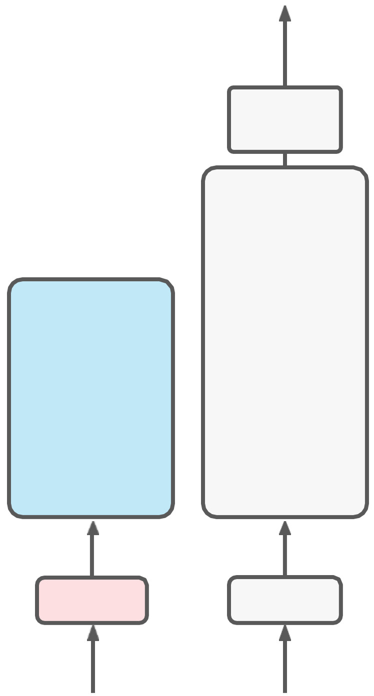
Autoencoding models

Good use cases:

- Sentiment analysis
- Named entity recognition
- Word classification

Example models:

- BERT
- ROBERTA

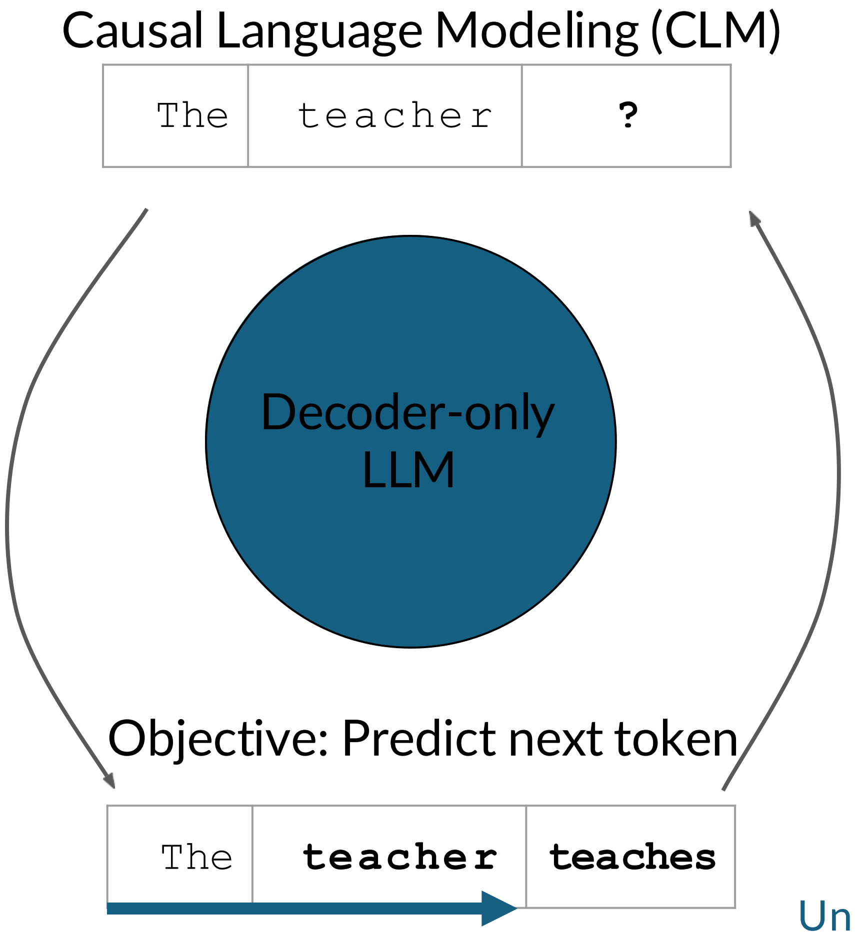


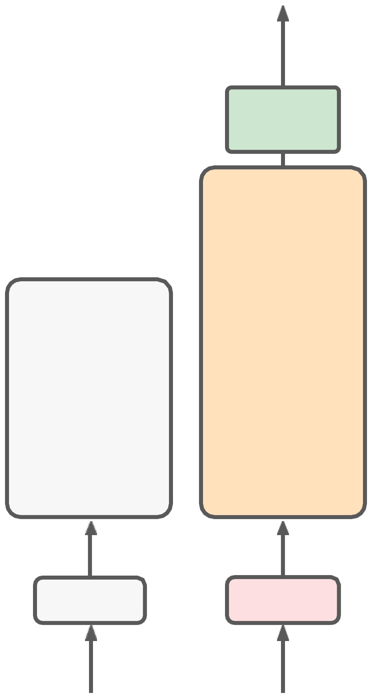
Autoregressive models

Original text

The teacher teaches the student.

[...]





Unidirectional context

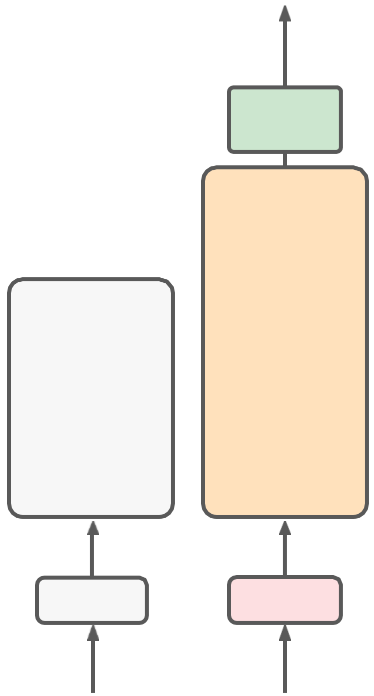
Autoregressive models

Good use cases:

- Text generation
- Other emergent behavior
  - Depends on model size

Example models:

- GPT
- BLOOM

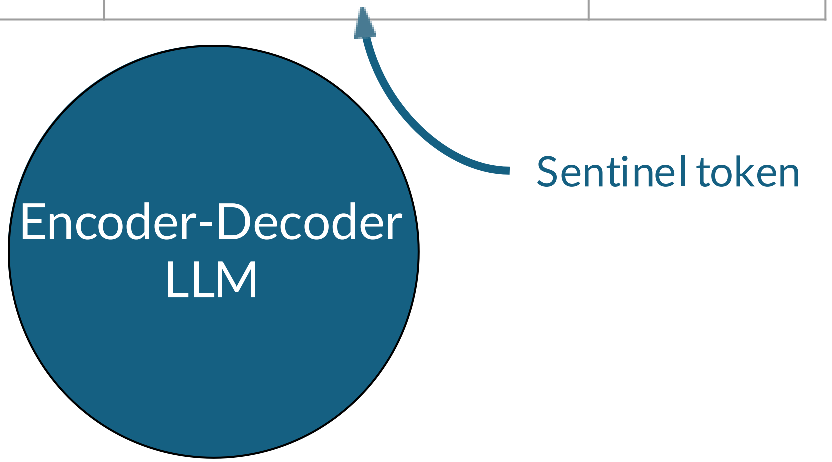


Sequence-to-sequence models

Span Corruption

The	teacher	**<MASK>**	**<MASK>**	student

The	teacher	**<X>**	student



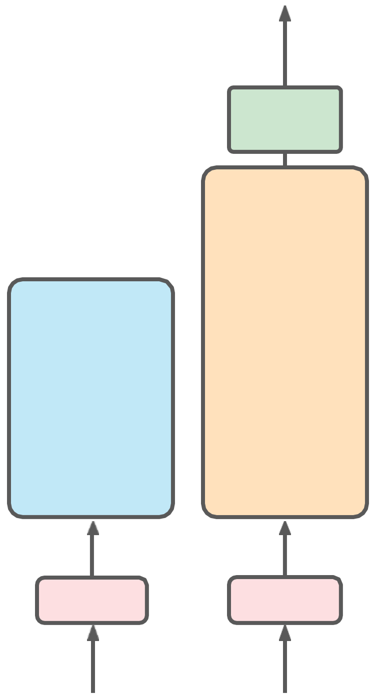
Original text

The teacher teaches the student.

[...]

Objective: Reconstruct span

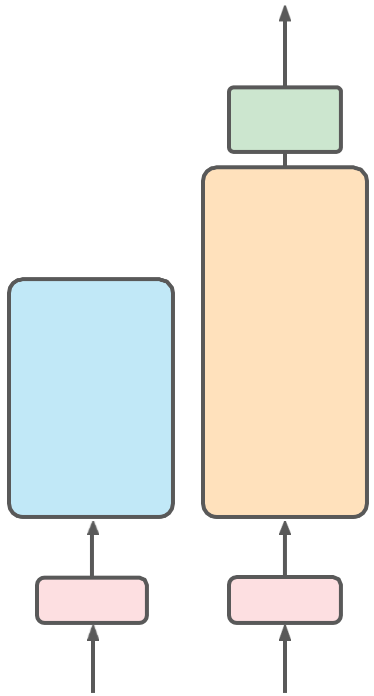
<x>	**teaches**	**the**



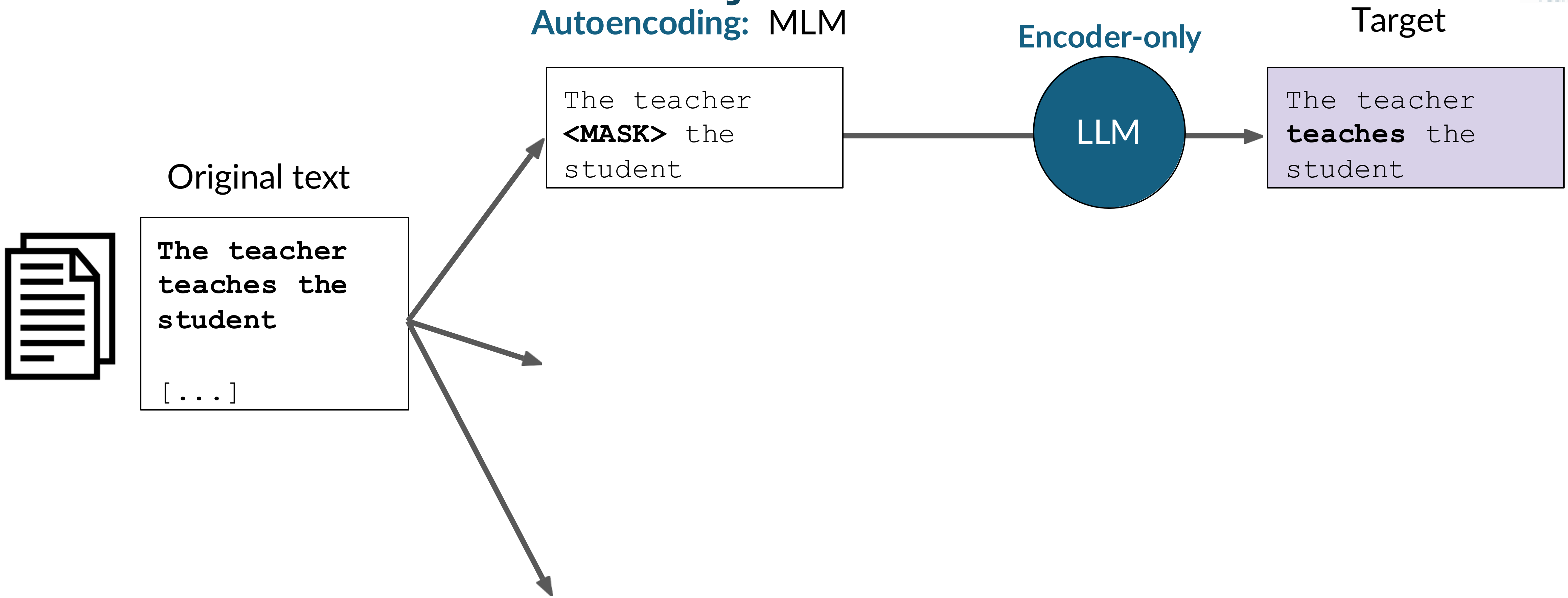
Sequence-to-sequence models

Good use cases:
- Translation
- Text summarization
- Question answering

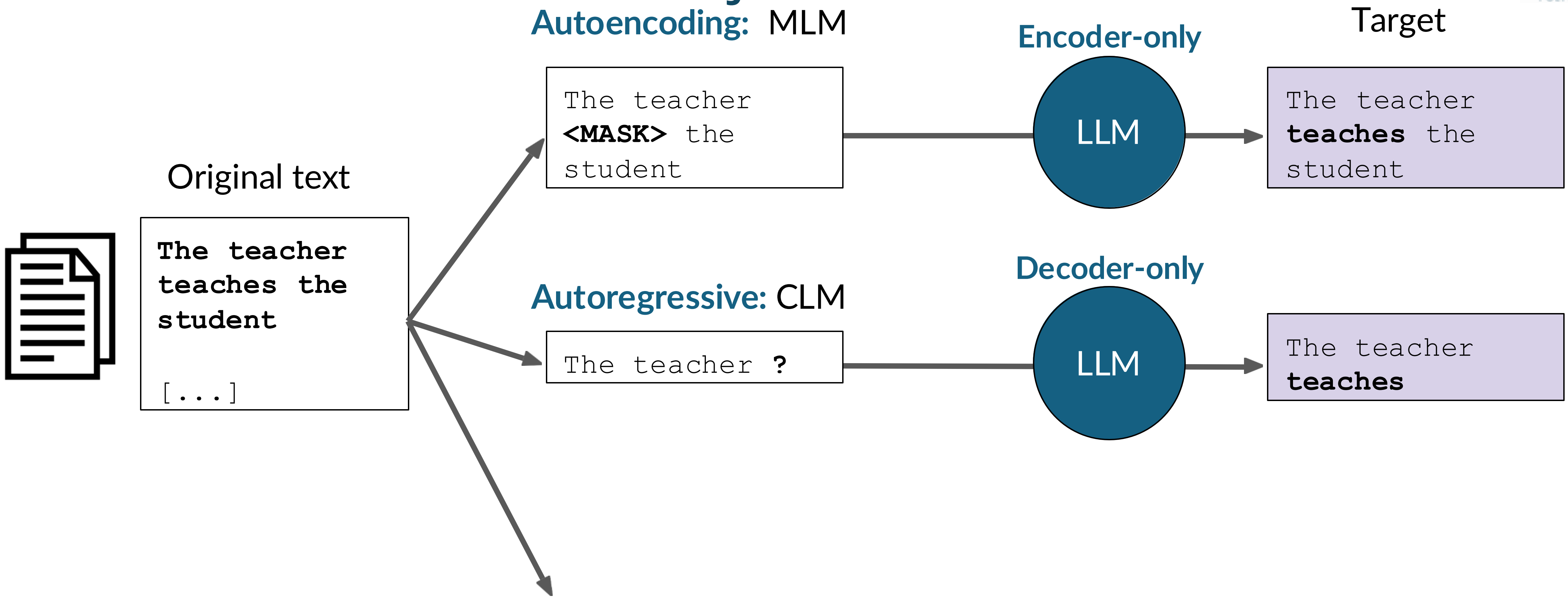
Example models:
- T5
- BART



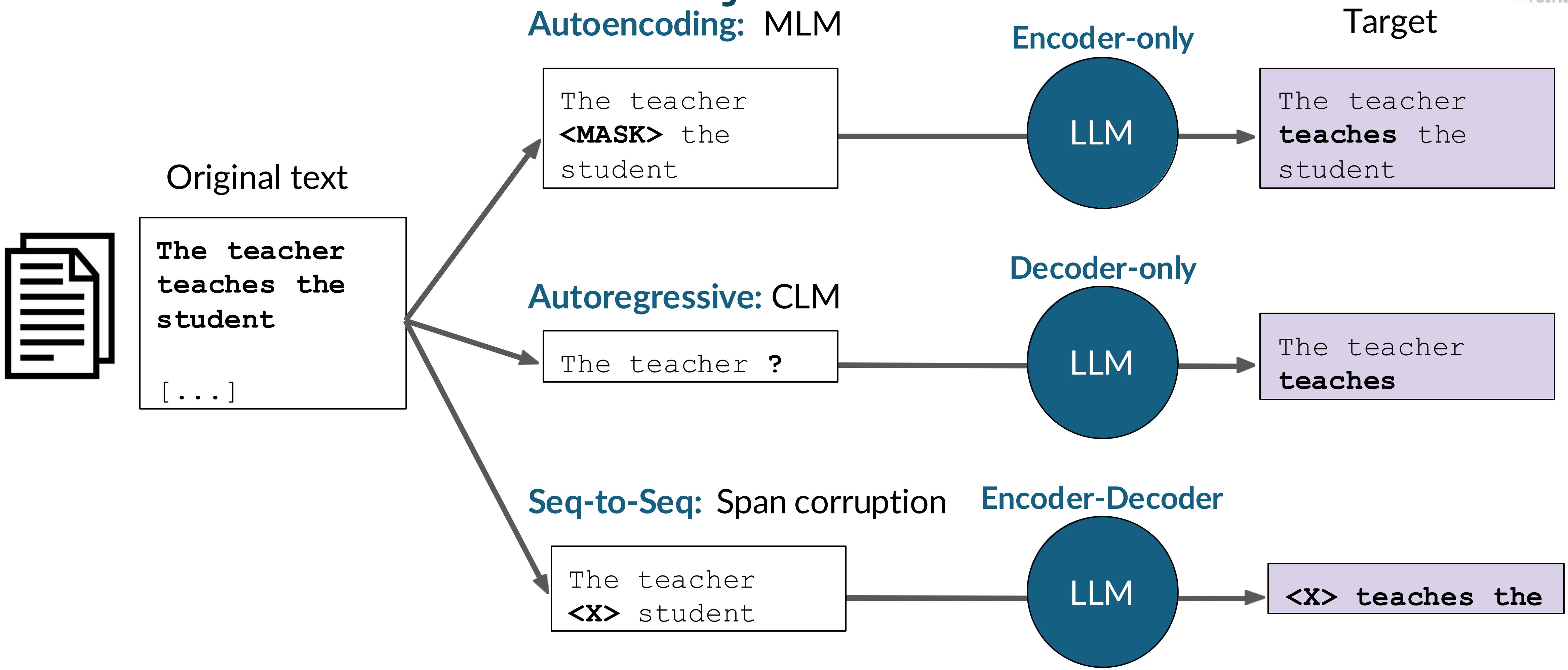
Summary: Model architectures and pre-training objectives



Summary: Model architectures and pre-training objectives



Summary: Model architectures and pre-training objectives



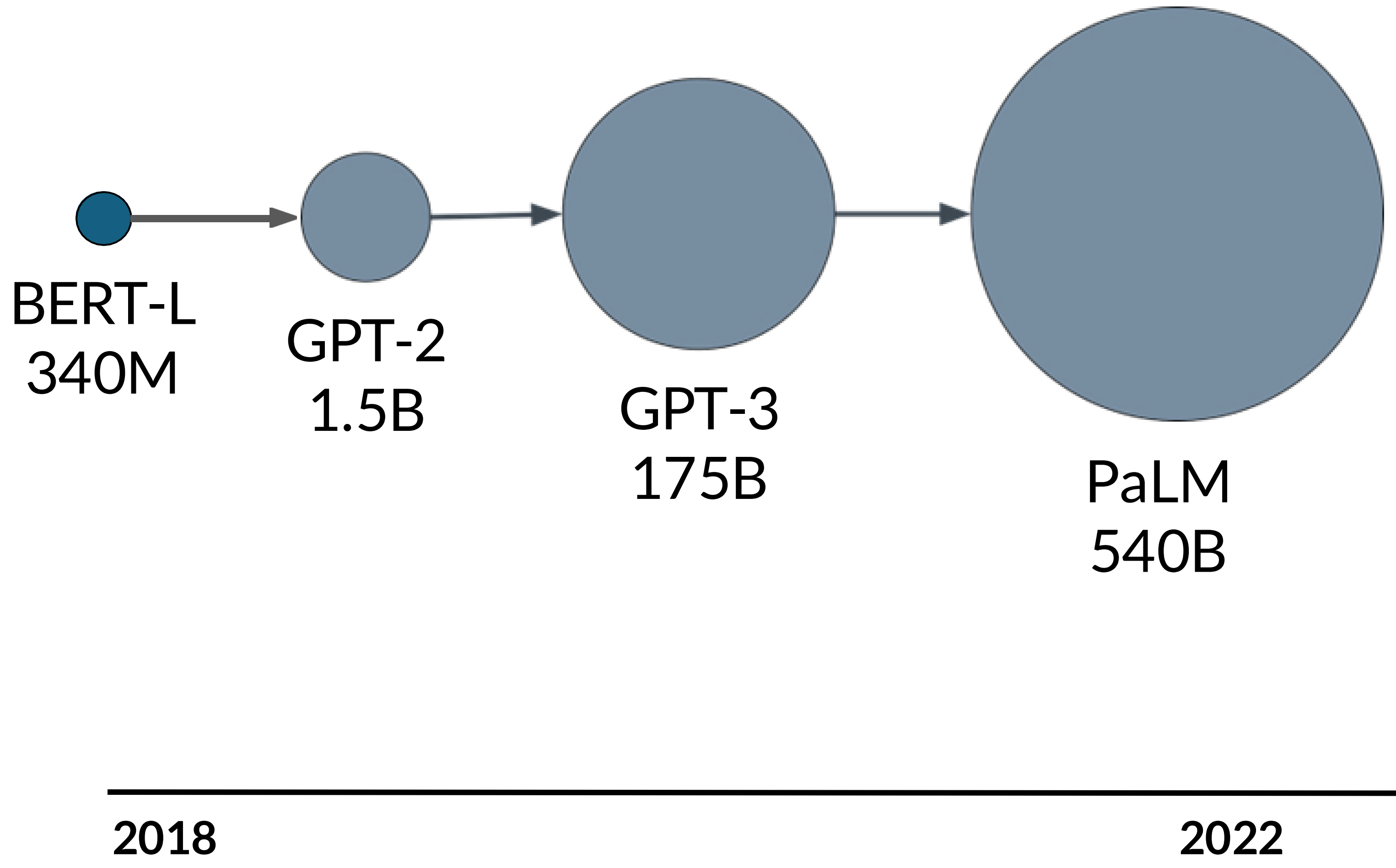
The significance of scale: task ability

BERT*
110M

BLOOM
176B

*Bert-base

Model size vs. time



Growth powered by:

- Introduction of transformer
- Access to massive datasets
- More powerful compute resources

Part II: Outline

Select

- Choose an existing model or pretrain your own

- **Scaling**
  - Challenges
  - Cost
  - Scaling laws

- Pre-training for domain adaptation

created with chatGPT

Part II: Outline

Select

- Choose an existing model or pretrain your own

- **Scaling**
  - **Challenges**
  - Cost
  - Scaling laws

- Pre-training for domain adaptation

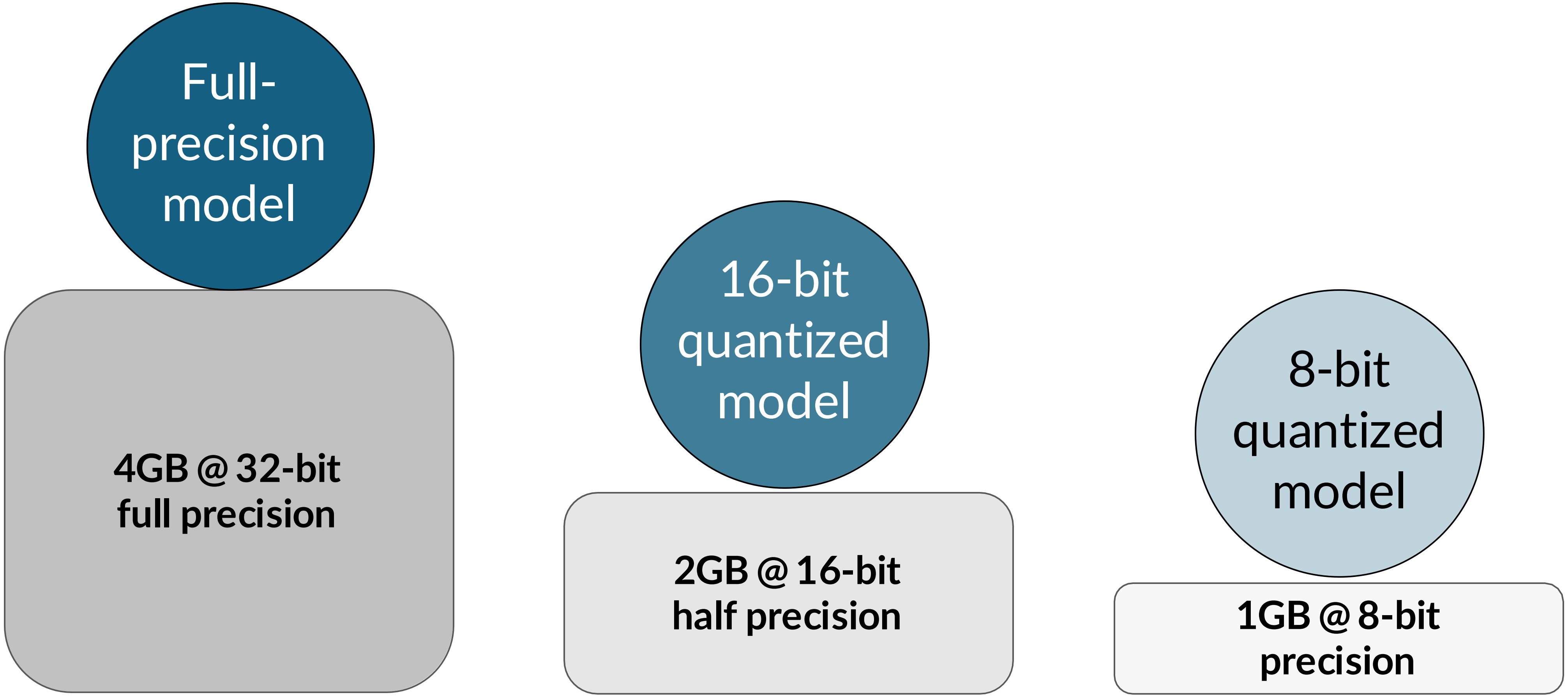
created with chatGPT

Challenges

Compute…

```
OutOfMemoryError: CUDA out of memory.
```

Approximate GPU RAM needed to store 1B parameters



Sources: https://huggingface.co/docs/transformers/v4.20.1/en/perf_train_gpu_one#anatomy-of-models-memory, https://github.com/facebookresearch/bitsandbytes

GPU RAM needed to train larger models

1B param model

175B param model

4,200 GB @ 32-bit full precision

500B param model

12,000 GB @ 32-bit full precision

GPU RAM needed to train larger models

As model sizes get larger, you will need to split your model across multiple GPUs for training

1B param model

4,200 GB @ 32-bit full precision

175B param model

500B param model

12,000 GB @ 32-bit full precision

Part II: Outline

Select

- Choose an existing model or pretrain your own

- **Scaling**
  - Challenges
  - **Cost**
  - Scaling laws

- Pre-training for domain adaptation

created with chatGPT

Scaling-up Transformers

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40GB	

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40GB	
Megatron-LM	72	2072	32	8.3B	174GB	512xV100 GPU (9 days)
Turing NLG	78	4256	28	17B	?	256xV100 GPU

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40GB	
Megatron-LM	72	2072	32	8.3B	174GB	512xV100 GPU (9 days)
Turing NLG	78	4256	28	17B	?	256xV100 GPU

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40GB	
Megatron-LM	72	2072	32	8.3B	174GB	512xV100 GPU (9 days)
Turing NLG			28	17B	?	256xV100 GPU

~350k euros!

Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40GB	
Megatron-LM	72	2072	32	8.3B	174GB	512xV100 GPU (9 days)
Turing NLG	78	4256	28	17B	?	256xV100 GPU
GPT-3	96	12288	96	175B	694GB	

Part II: Outline

Select

- Choose an existing model or pretrain your own

- **Scaling**
  - Challenges
  - Cost
  - **Scaling laws**

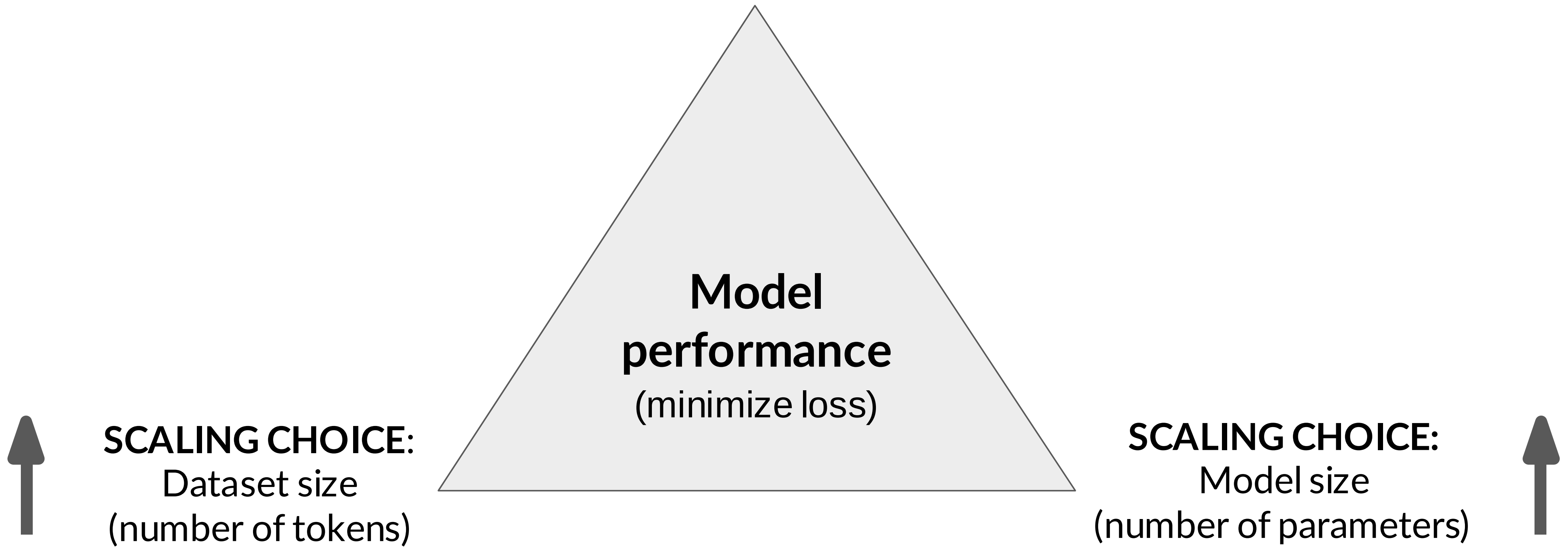
- Pre-training for domain adaptation

created with chatGPT

Scaling laws

Scaling choices for pre-training

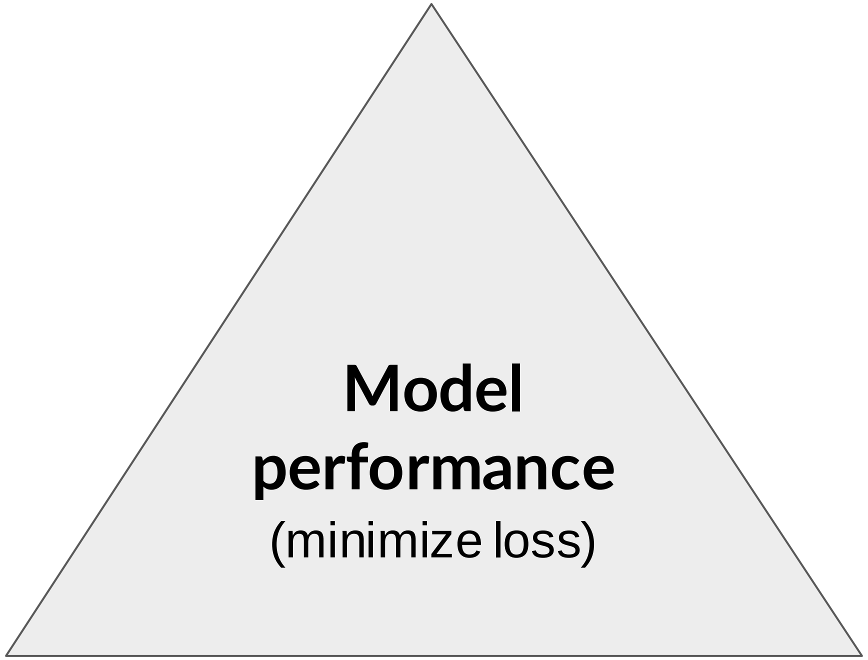
Goal: **maximize model performance**



Scaling choices for pre-training

Goal: **maximize model performance**

CONSTRAINT:
Compute budget
(GPUs, training time, cost)



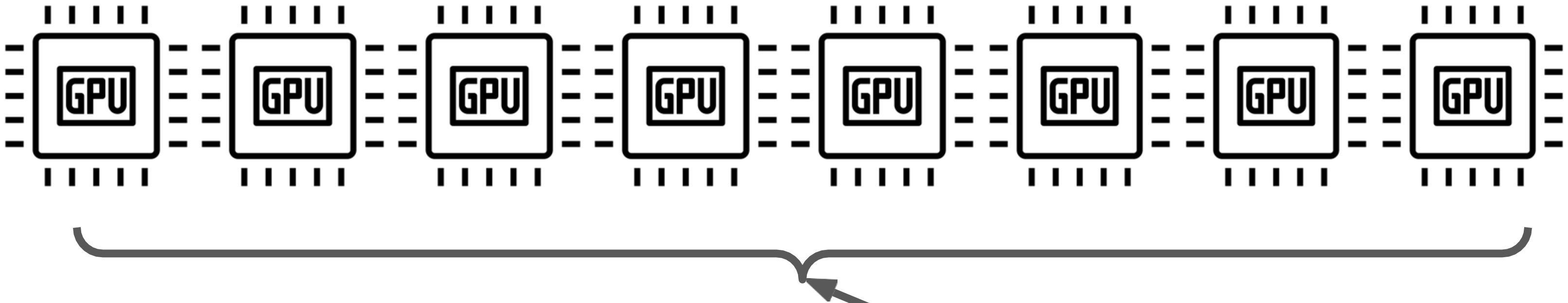
SCALING CHOICE:
Dataset size
(number of tokens)

SCALING CHOICE:
Model size
(number of parameters)

Compute budget for training LLMs

1 “petaflop/s-day” =
floating point operations performed at rate of 1 petaFLOP per second for one day

NVIDIA V100s



Note: 1 petaFLOP/s = 1,000,000,000,000,000 (one quadrillion) floating point operations per second

1 petaflop/s-day is these chips running at full efficiency for 24 hours

Compute budget for training LLMs

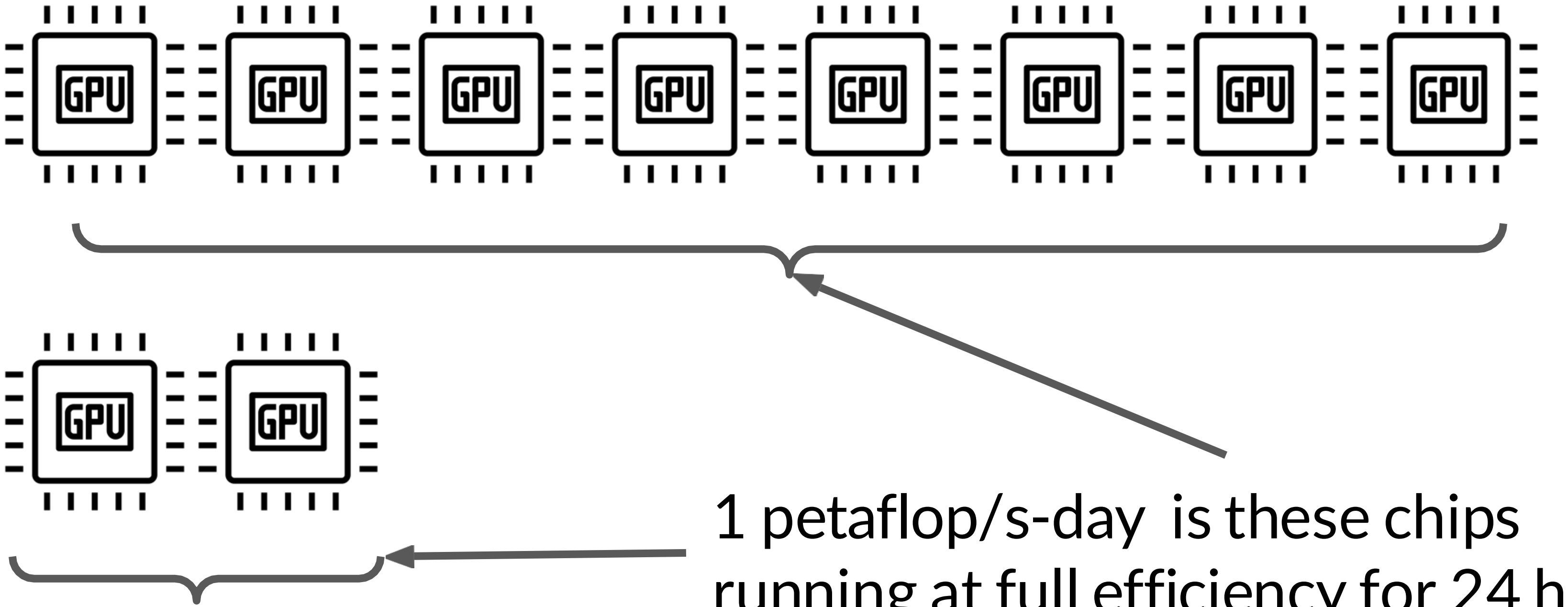
1 “petaflop/s-day” =

floating point operations performed at rate of 1 petaFLOP per second for one day

NVIDIA V100s

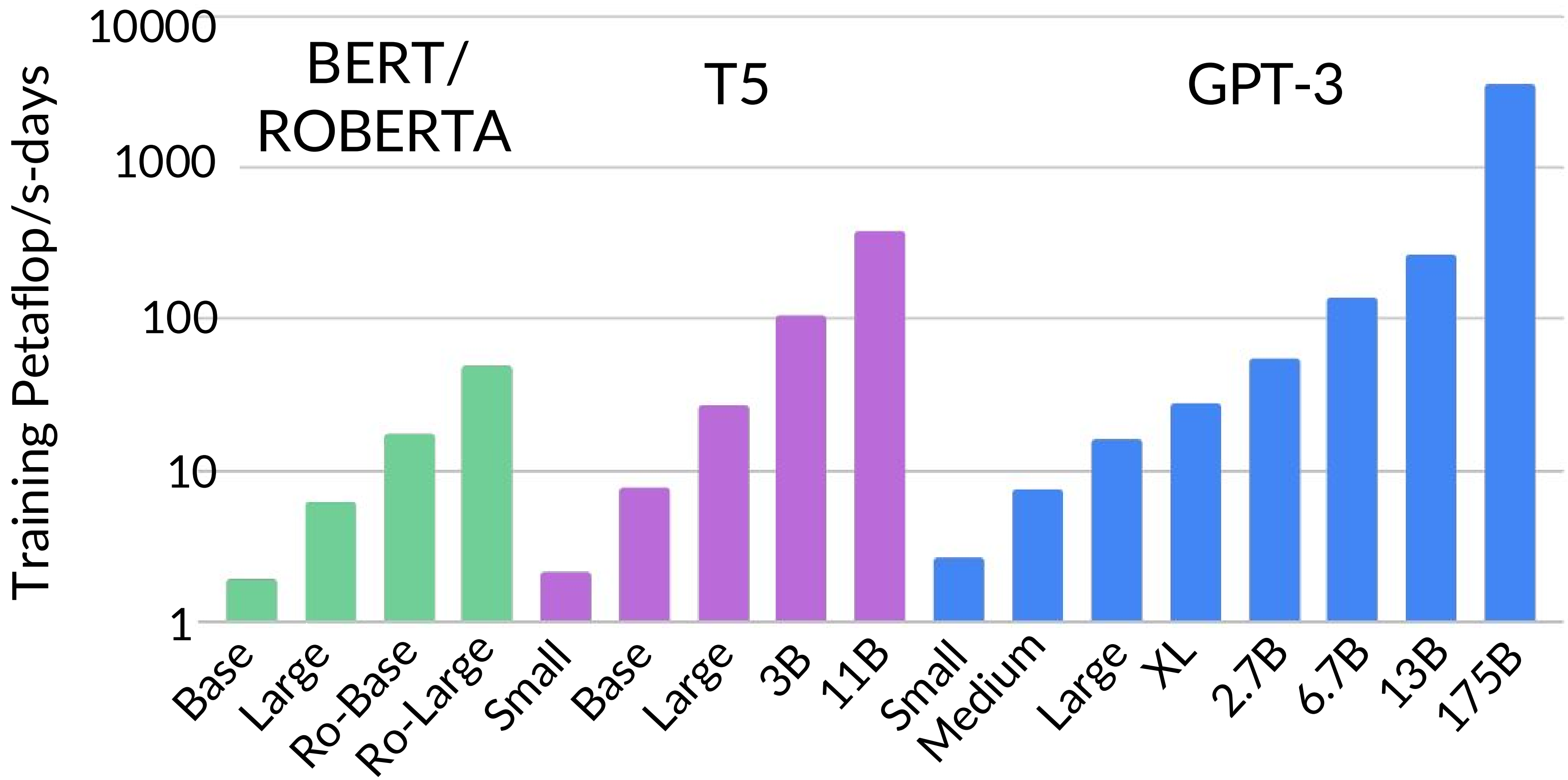
OR

NVIDIA A100s



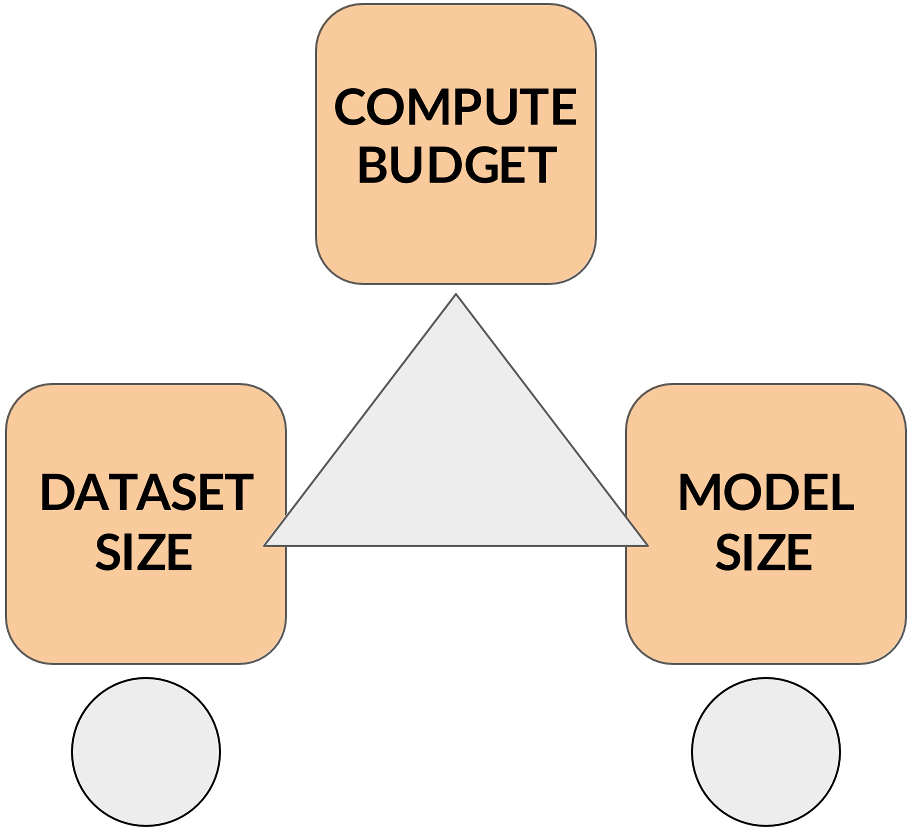
1 petaflop/s-day is these chips running at full efficiency for 24 hours

Number of petaflop/s-days to pre-train various LLMs



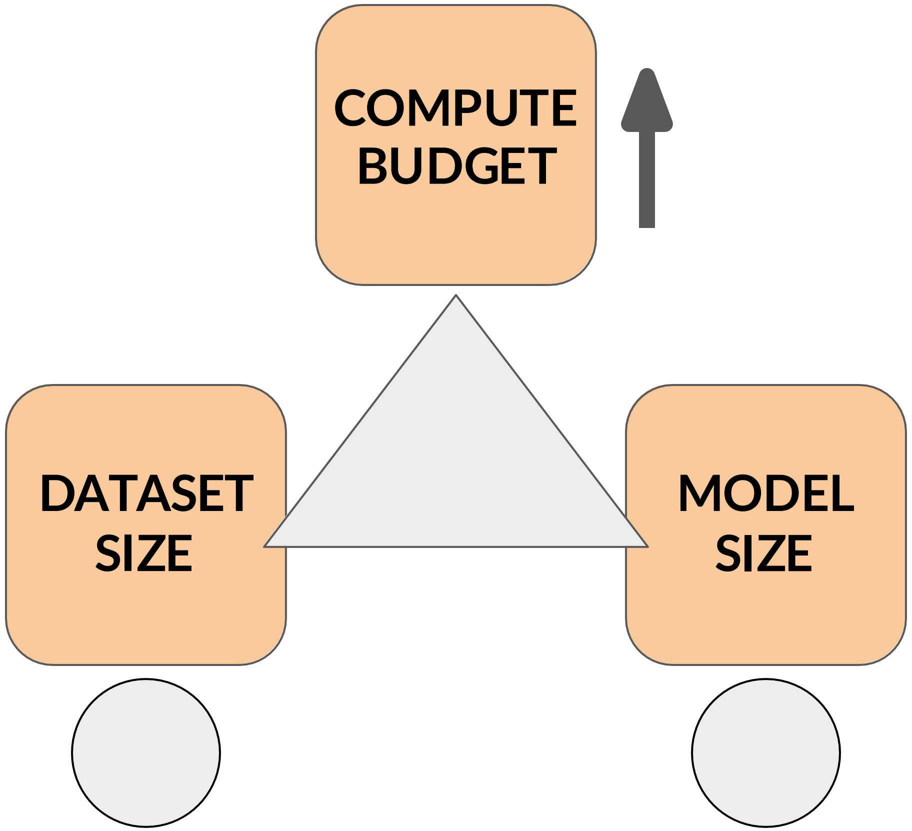
Source: Brown et al. 2020, "Language Models are Few-Shot Learners"

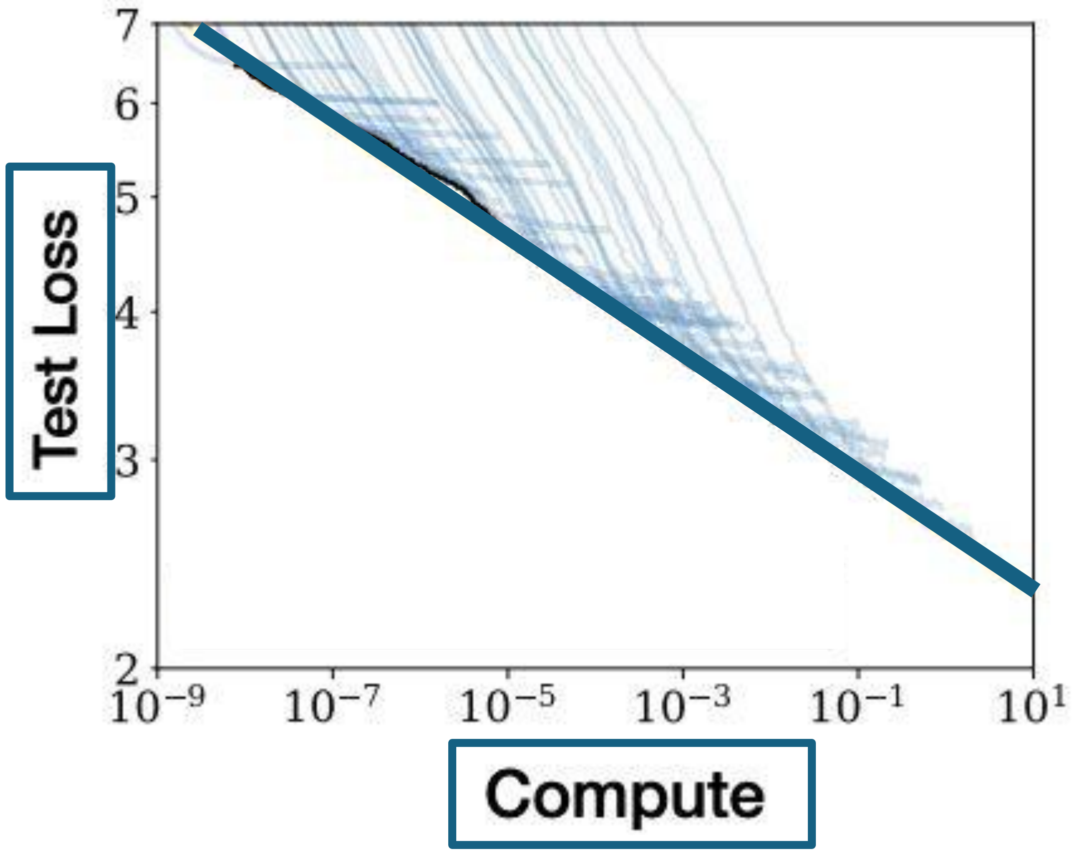
Compute budget vs. model performance



Source: Kaplan et al. 2020, “Scaling Laws for Neural Language Models”

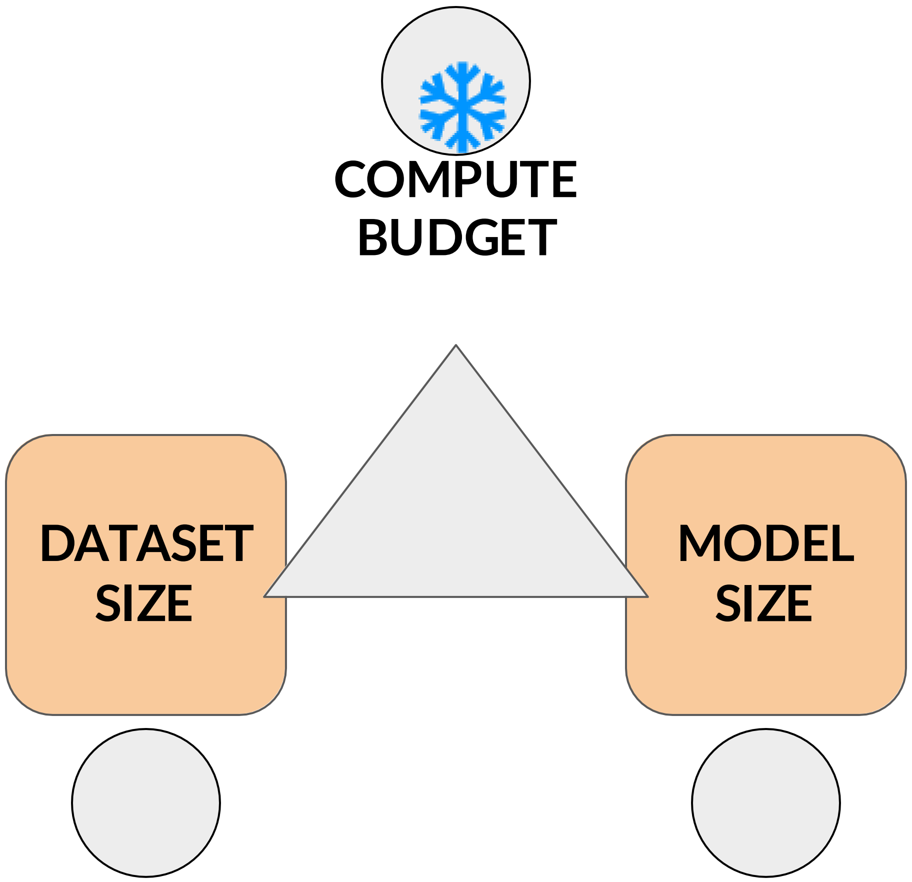
Compute budget vs. model performance





Source: Kaplan et al. 2020, "Scaling Laws for Neural Language Models"

Dataset size and model size vs. performance ❄️

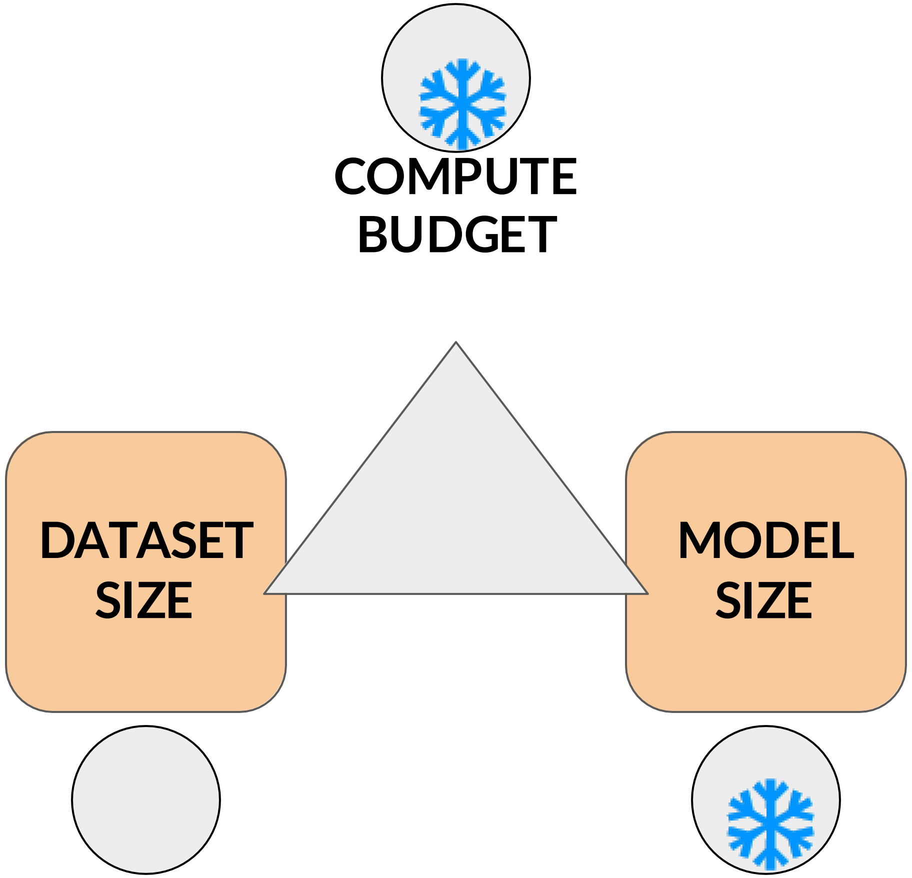


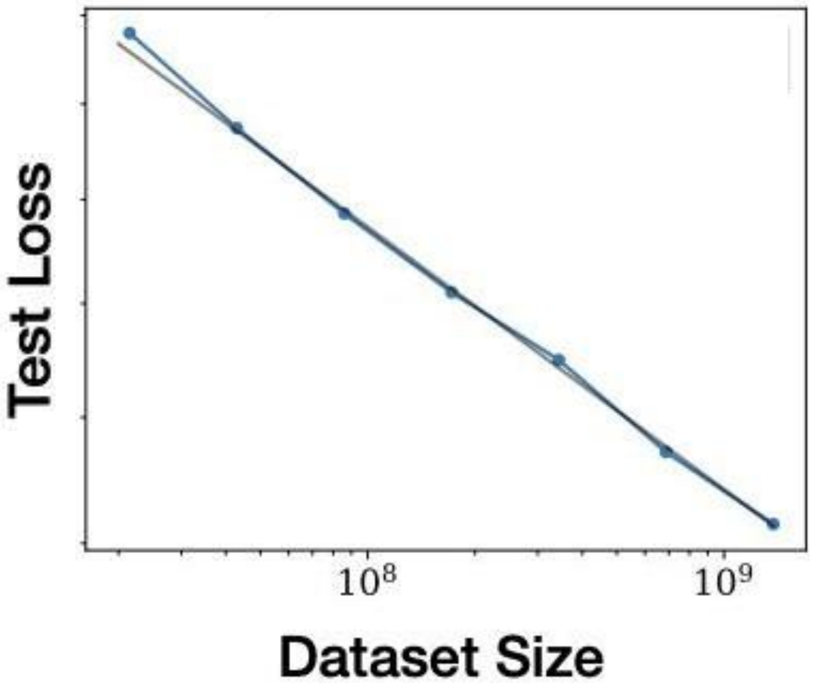
Compute resource constraints

- Hardware
- Project timeline
- Financial budget

Source: Kaplan et al. 2020, “Scaling Laws for Neural Language Models”

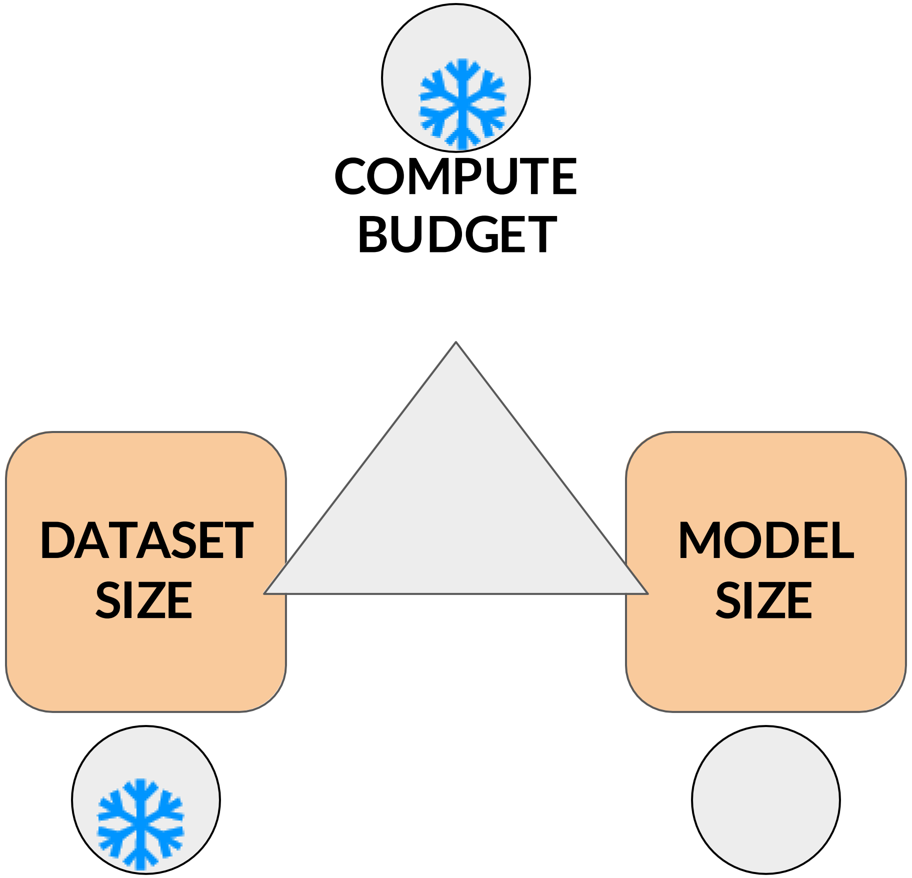
Dataset size and model size vs. performance ❄️

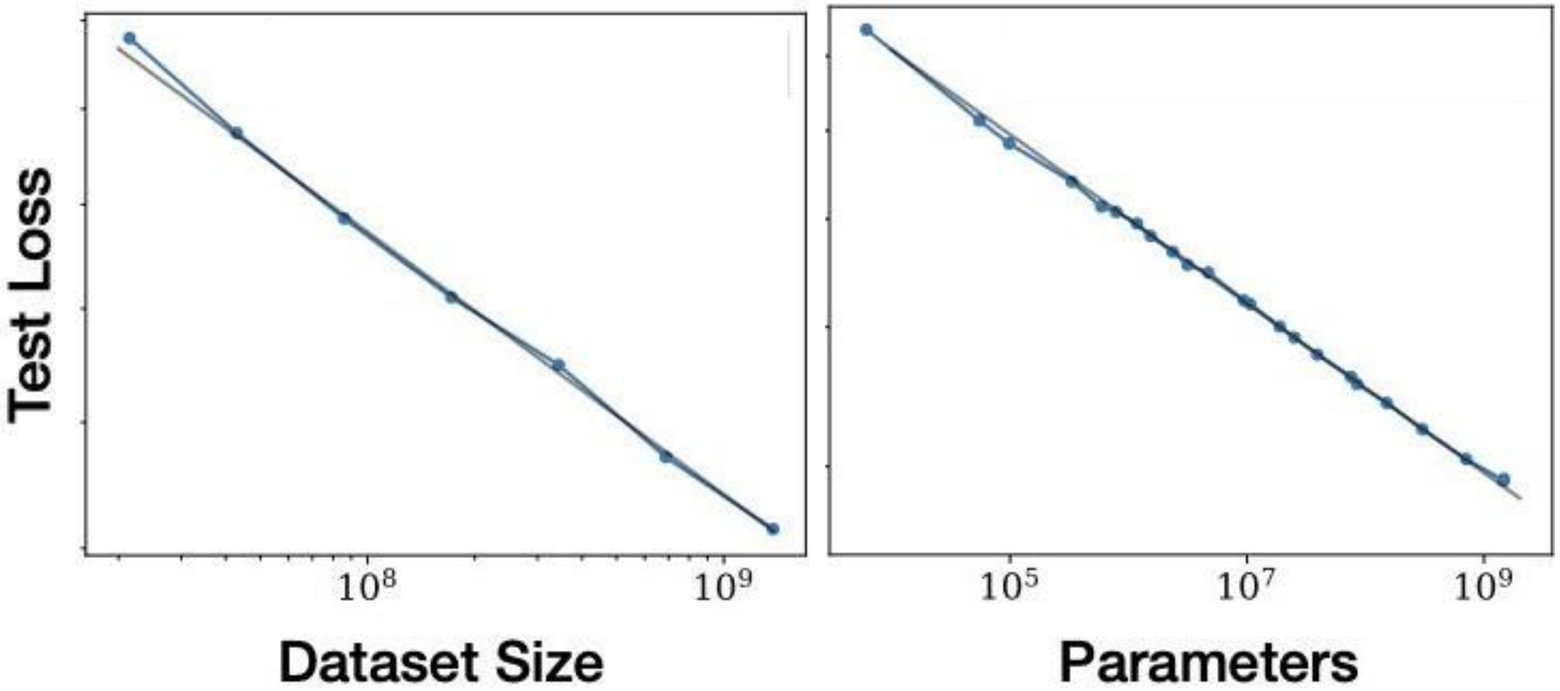




Source: Kaplan et al. 2020, "Scaling Laws for Neural Language Models"

Dataset size and model size vs. performance ❄





Source: Kaplan et al. 2020, "Scaling Laws for Neural Language Models"

Chinchilla paper

Training Compute-Optimal Large Language Models

Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre*

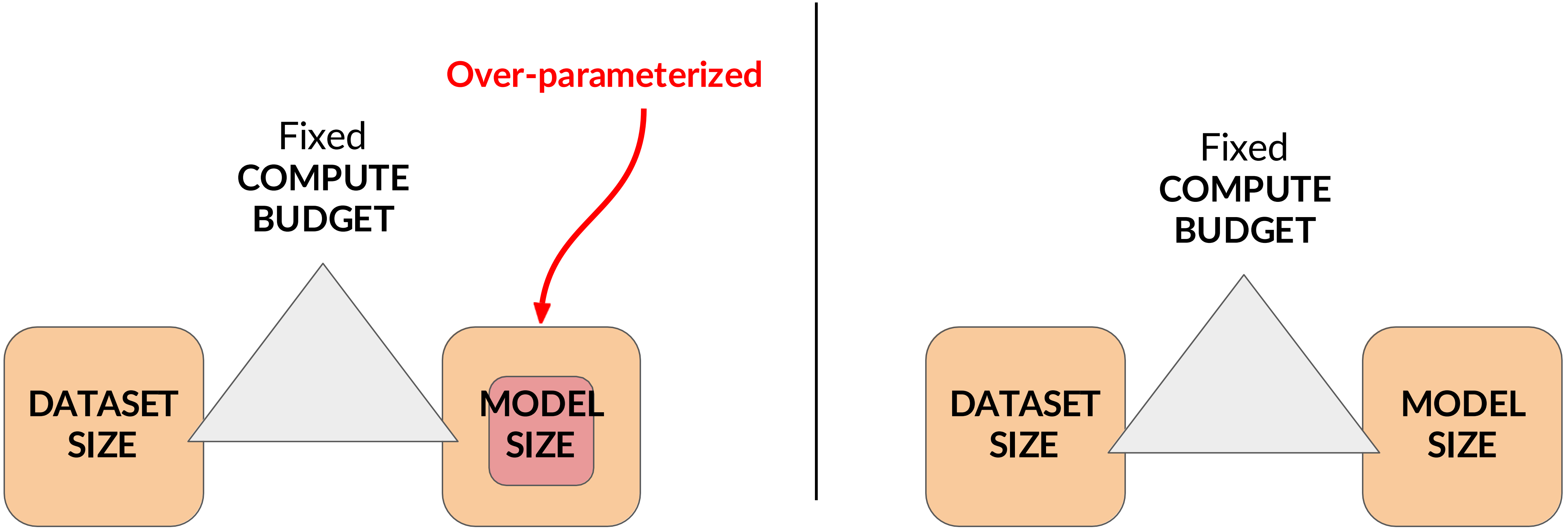
*Equal contributions

We investigate the optimal model size and number of tokens for training a transformer language model under a given compute budget. We find that current large language models are significantly under-trained, a consequence of the recent focus on scaling language models whilst keeping the amount of training data constant. By training over 400 language models ranging from 70 million to over 16 billion parameters on 5 to 500 billion tokens, we find that for compute-optimal training, the model size and the number of training tokens should be scaled equally: for every doubling of model size the number of training tokens should also be doubled. We test this hypothesis by training a predicted compute-optimal model, *Chinchilla*, that uses the same compute budget as *Gopher* but with 70B parameters and 4× more more data. *Chinchilla* uniformly and significantly outperforms *Gopher* (280B), GPT-3 (175B), Jurassic-1 (178B), and Megatron-Turing NLG (530B) on a large range of downstream evaluation tasks. This also means that *Chinchilla* uses substantially less compute for fine-tuning and inference, greatly facilitating downstream usage. As a highlight, *Chinchilla* reaches a state-of-the-art average accuracy of 67.5% on the MMLU benchmark, greater than a 7% improvement over *Gopher*.

Jordan et al. 2022

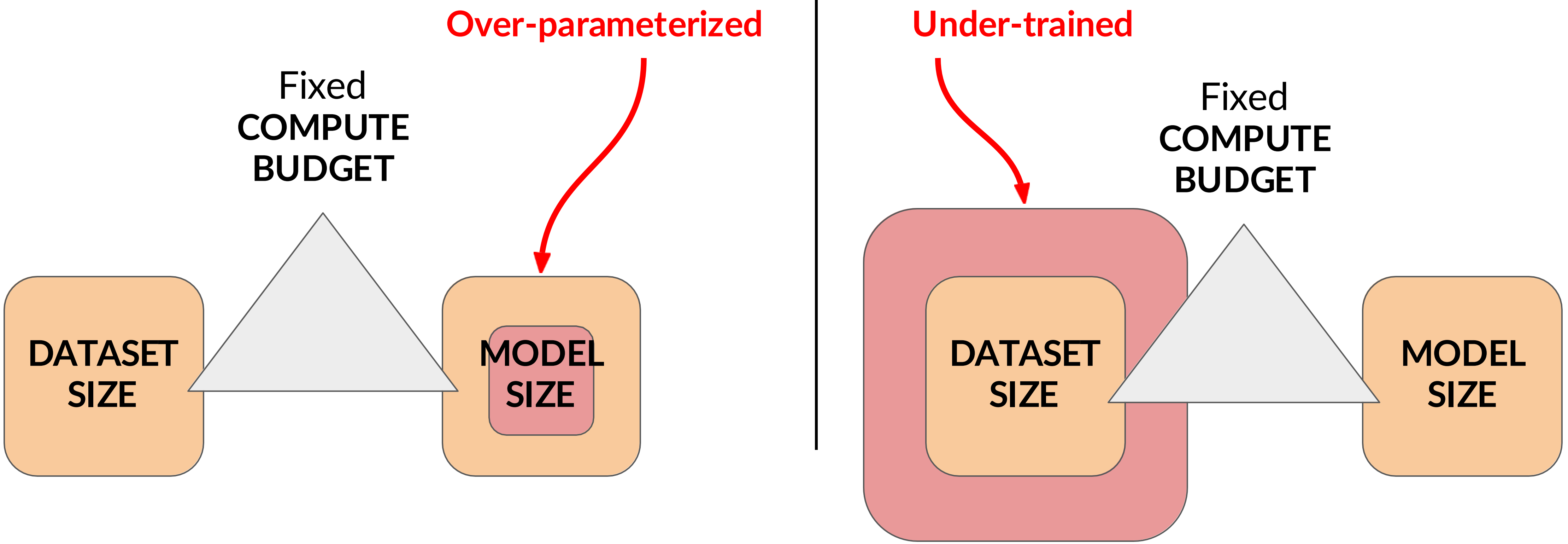
Compute optimal models

- Very large models may be **over-parameterized**



Compute optimal models

- Very large models may be **over-parameterized** and **under-trained**
- Smaller models trained on more data could perform as well as large models



Chinchilla scaling laws for model + dataset size

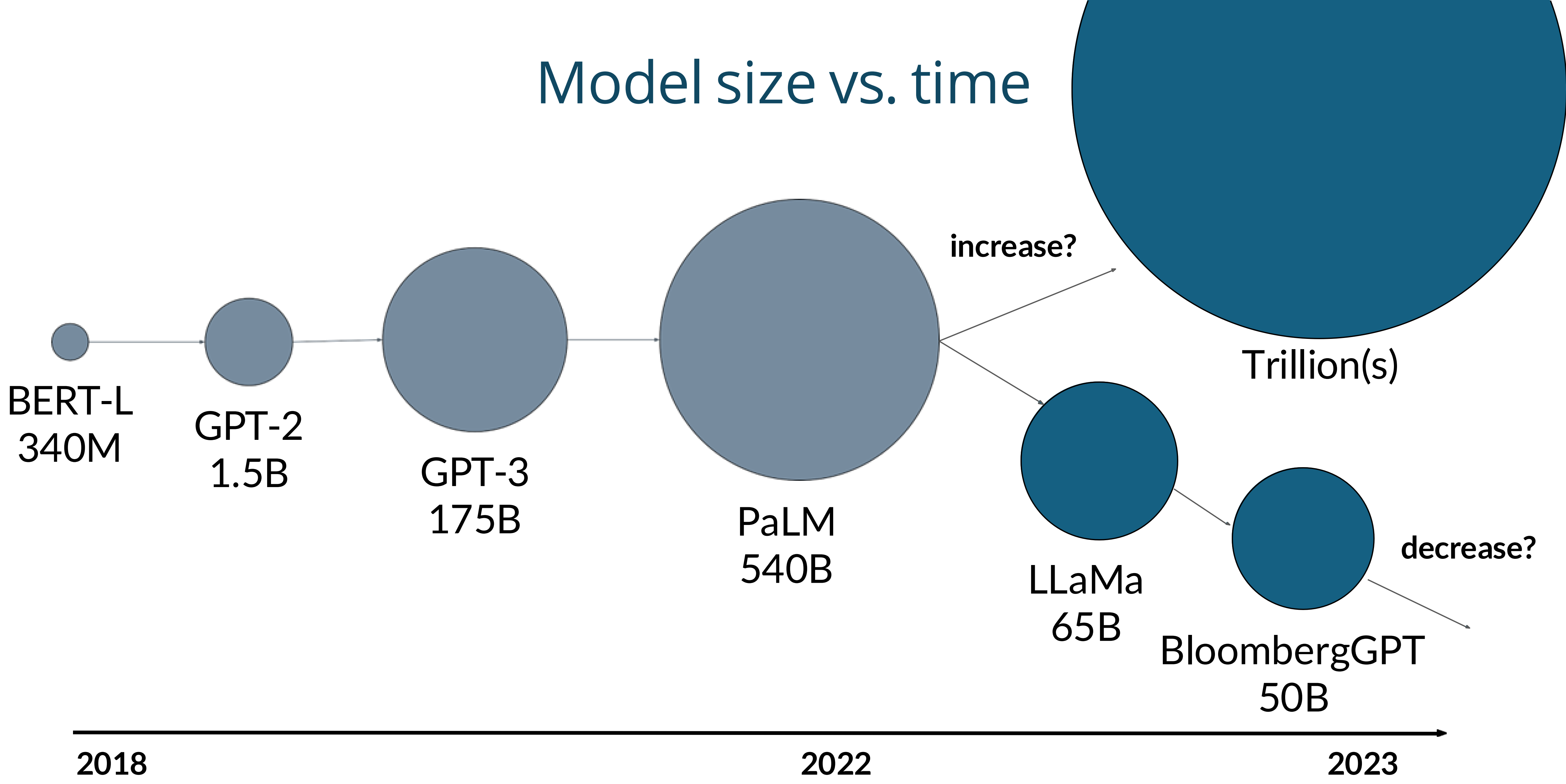
Model	# of parameters	Compute-optimal* # of tokens (~20x)	Actual # tokens
Chinchilla	70B	~1.4T	1.4T
LLaMA-65B	65B	~1.3T	1.4T
GPT-3	175B	~3.5T	300B
OPT-175B	175B	~3.5T	180B
BLOOM	176B	~3.5T	350B

Compute optimal training datasize
is ~**20x** number of parameters

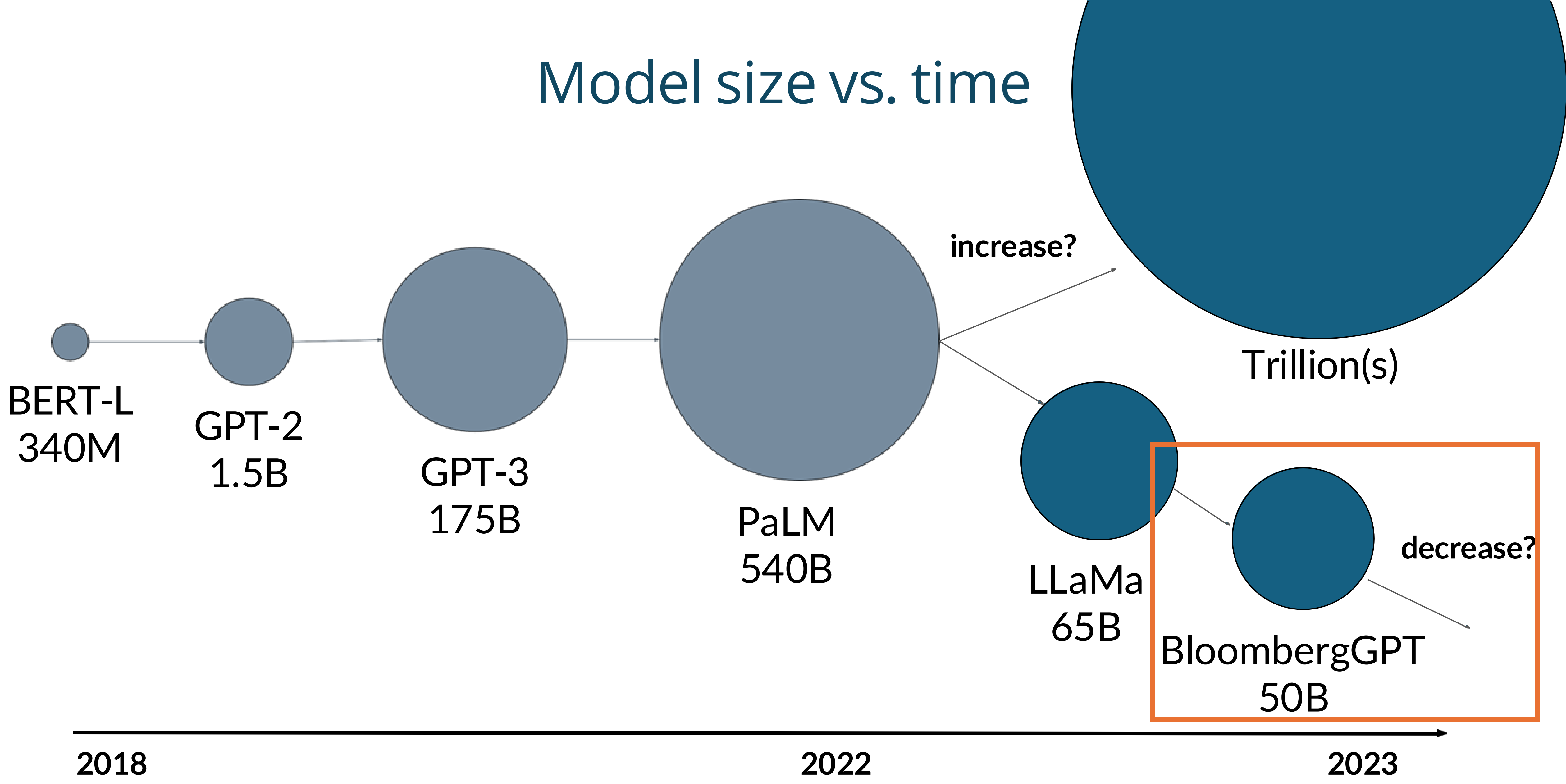
* assuming models are trained to be compute-optimal per Chinchilla paper

Sources: Hoffmann et al. 2022, "Training Compute-Optimal Large Language Models"
Touvron et al. 2023, "LLaMA: Open and Efficient Foundation Language Models"

Model size vs. time



Model size vs. time



Part II: Outline

Select

- Choose an existing model or pretrain your own

- Scaling
  - Challenges
  - Cost
  - Scaling laws

- **Pre-training for domain adaptation**

created with chatGPT

Pre-training for domain adaptation

Pre-training for domain adaptation

Legal language

The prosecutor had difficulty proving <u>mens rea</u>, as the defendant seemed unaware that his actions were illegal.

The judge dismissed the case, citing the principle of <u>res judicata</u> as the issue had already been decided in a previous trial.

Despite the signed agreement, the contract was invalid as there was no <u>consideration</u> exchanged between the parties.

Pre-training for domain adaptation

Legal language

The prosecutor had difficulty proving mens rea, as the defendant seemed unaware that his actions were illegal.

The judge dismissed the case, citing the principle of res judicata as the issue had already been decided in a previous trial.

Despite the signed agreement, the contract was invalid as there was no consideration exchanged between the parties.

Medical language

After a strenuous workout, the patient experienced severe myalgia that lasted for several days.

After the biopsy, the doctor confirmed that the tumor was malignant and recommended immediate treatment.

Sig: 1 tab po qid pc & hs

↓

Take one tablet by mouth four times a day, after meals, and at bedtime.

Solution →
Pre-training for domain adaptation

BloombergGPT: domain adaptation for finance

BloombergGPT: A Large Language Model for Finance

Shijie Wu[1,*], Ozan İrsoy[1,*], Steven Lu[1,*], Vadim Dabravolski[1], Mark Dredze[1,2], Sebastian Gehrmann[1], Prabhanjan Kambadur[1], David Rosenberg[1], Gideon Mann[1]
[1] **Bloomberg, New York, NY USA**
[2] **Computer Science, Johns Hopkins University, Baltimore, MD USA**
gmann16@bloomberg.net

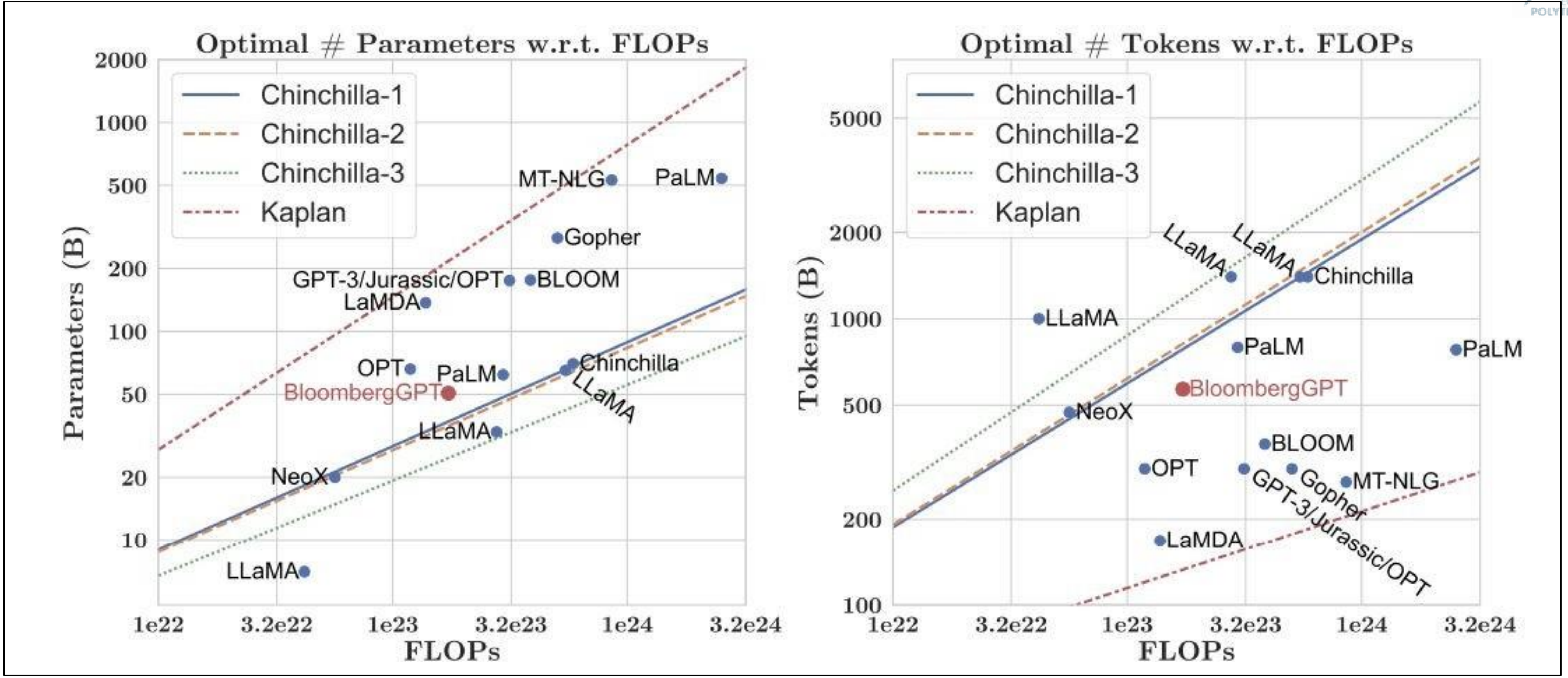
Abstract

The use of NLP in the realm of financial technology is broad and complex, with applications ranging from sentiment analysis and named entity recognition to question answering. Large Language Models (LLMs) have been shown to be effective on a variety of tasks; however, no LLM specialized for the financial domain has been reported in literature. In this work, we present BloombergGPT, a 50 billion parameter language model that is trained on a wide range of financial data. We construct a 363 billion token dataset based on Bloomberg's extensive data sources, perhaps the largest domain-specific dataset yet, augmented with 345 billion tokens from general purpose datasets. We validate BloombergGPT on standard LLM benchmarks, open financial benchmarks, and a suite of internal benchmarks that most accurately reflect our intended usage. Our mixed dataset training leads to a model that outperforms existing models on financial tasks by significant margins without sacrificing performance on general LLM benchmarks. Additionally, we explain our modeling choices, training process, and evaluation methodology. As a next step, we plan to release training logs (Chronicles) detailing our experience in training BloombergGPT.

~51% **Financial (Public & Private)**

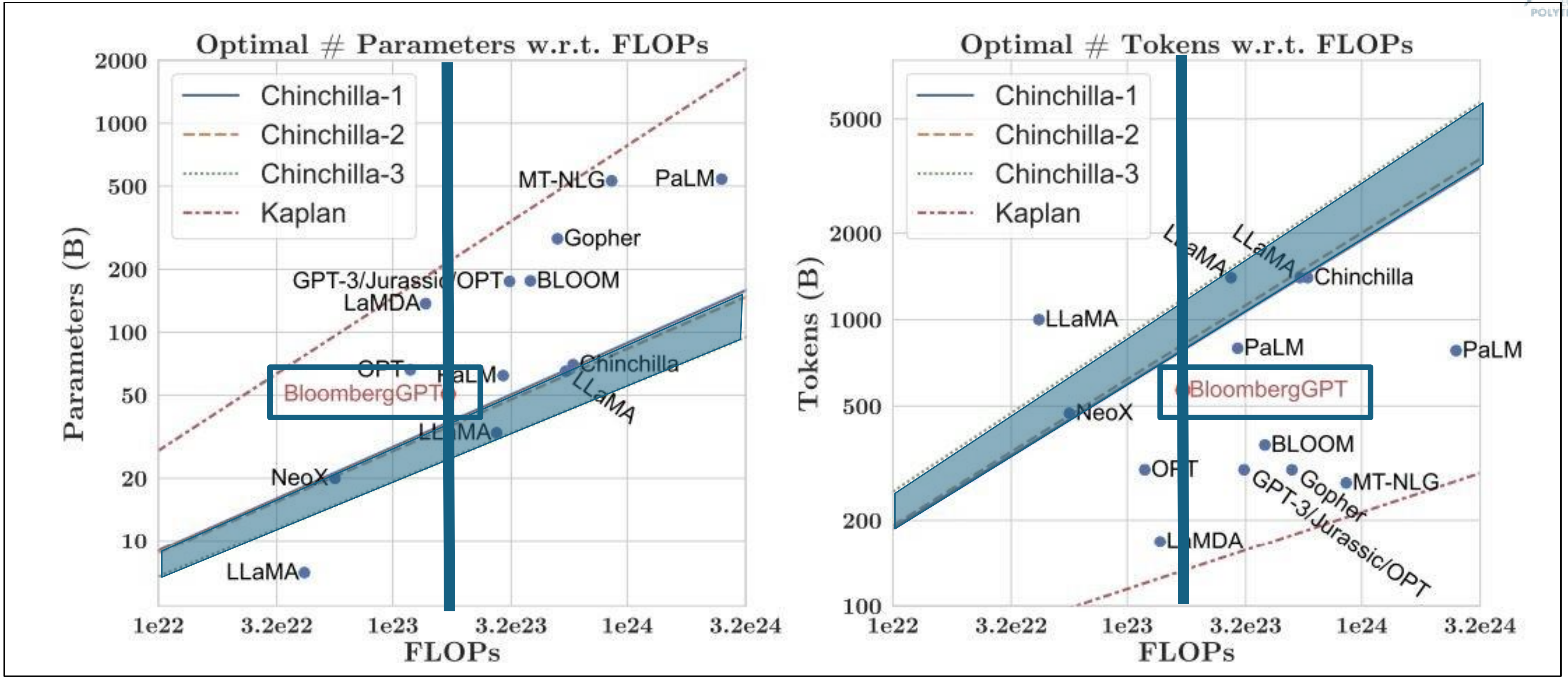
~49% **Other (Public)**

BloombergGPT relative to other LLMs



Source: Wu et al. 2023, "BloombergGPT: A Large Language Model for Finance"

BloombergGPT relative to other LLMs



Source: Wu et al. 2023, "BloombergGPT: A Large Language Model for Finance"

Part II: Summary

Select

- Choose an existing model or pretrain your own

- Scaling
  - Challenges
  - Cost
  - Scaling laws

- Pre-training for domain adaptation

created with chatGPT

Considerations for choosing a model

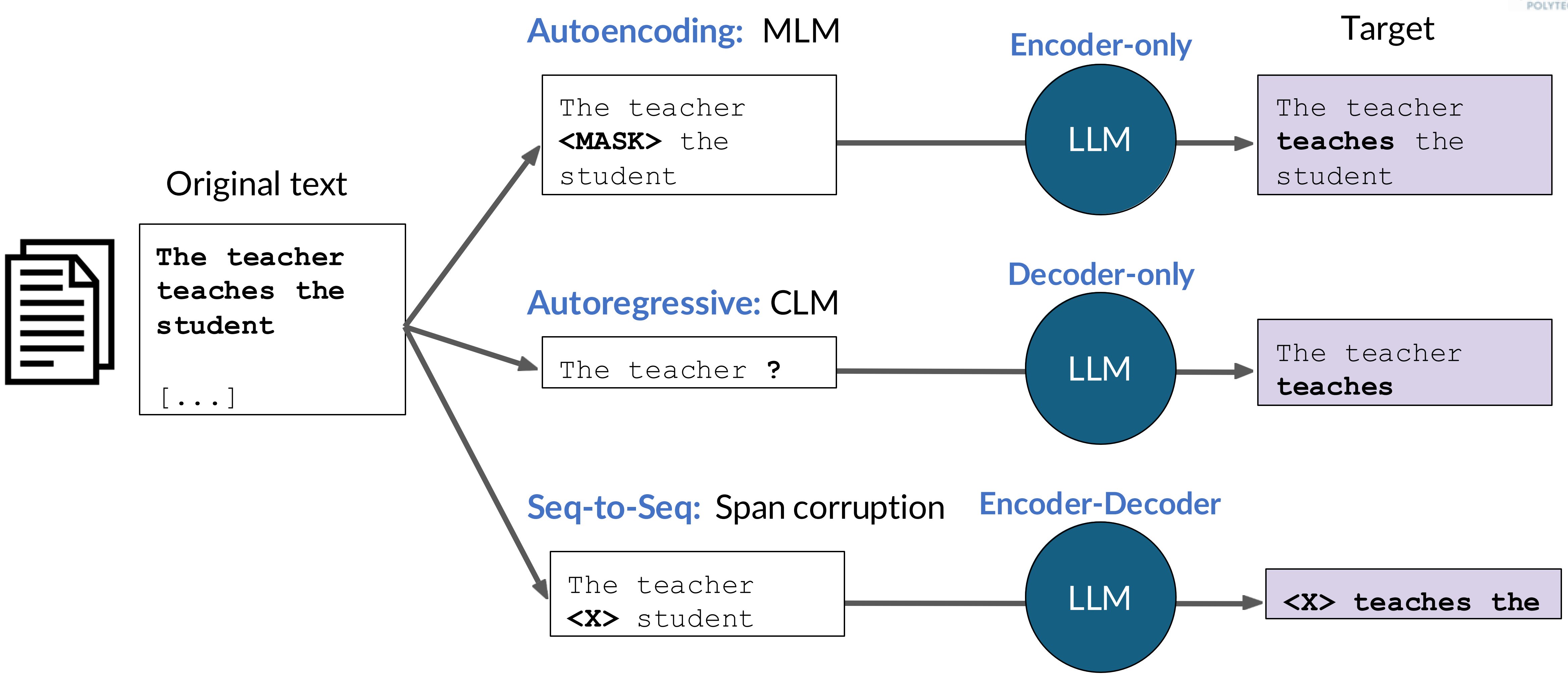
Foundation model

Pretrained LLM

Train your own model

Custom LLM

Model architectures and pre-training objectives



Compute…

```
OutOfMemoryError: CUDA out of memory.
```

GPU RAM needed to train larger models

As model sizes get larger, you will need to split your model across multiple GPUs for training

1B param model

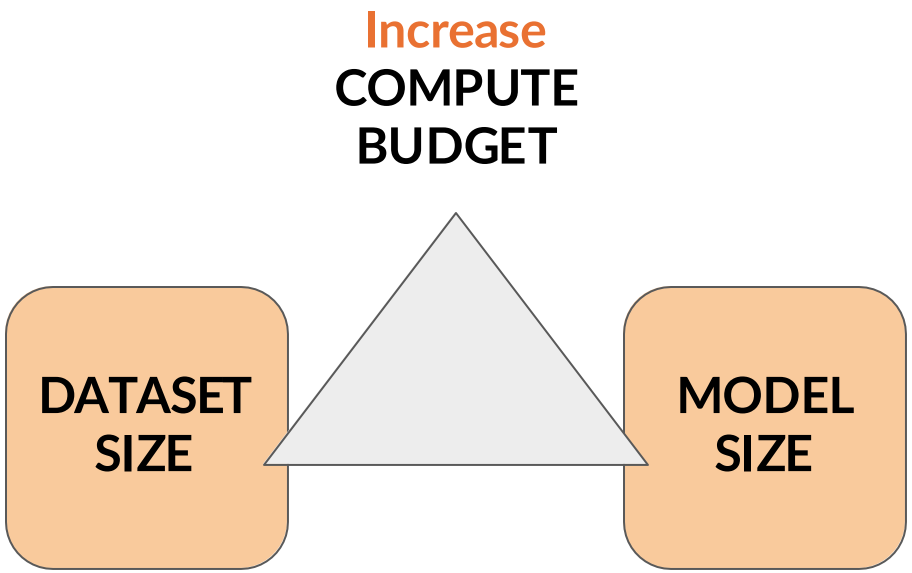
4,200 GB @ 32-bit full precision

175B param model

500B param model

12,000 GB @ 32-bit full precision

Increase compute budget → increase performance?

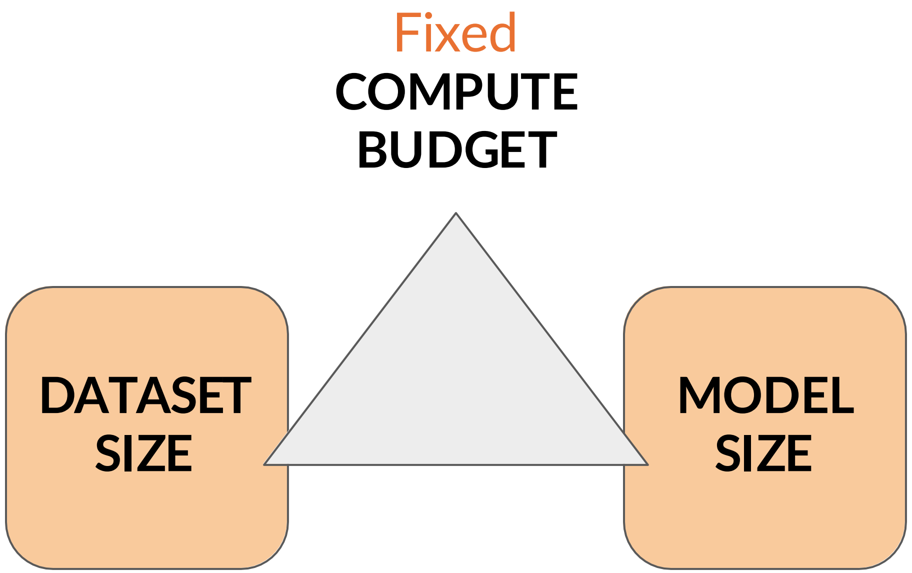


Scaling up Transformers

Model	Layers	Width	Heads	Params	Data	Training
Transformer Base	12	512	8	65M		8xP100 (12h)
Transformer Large	12	1024	16	213M		8xP100 (12h)
Bert Base	12	768	12	110M	13GB	
Bert Large	24	1024	16	340M	13GB	
XLNet Large	24	1024	16	~340M	126GB	512xTPUv3 (2.5 days)
RoBERTa	24	1024	16	355M	160GB	1024xV100 GPU (1 day)
GPT-2	48	1600	?	1.5B	40GB	
Megatron-LM	72	2072	32	8.3B	174GB	512xV100 GPU (9 days)
Turing NLG			28	17B	?	256xV100 GPU

~350k euros!

Chinchilla scaling laws for model + dataset size



Model	# params	Compute-optimal* # of tokens (~20x)	Actual tokens
Chinchilla	70B	~1.4T	1.4T
LLaMA-65B	65B	~1.3T	1.4T
GPT-3	175B	~3.5T	300B
OPT-175B	175B	~3.5T	180B
BLOOM	176B	~3.5T	350B

Compute optimal training datasize
is ~**20x** number of parameters

* assuming models are trained to be compute-optimal per Chinchilla paper

Sources: Hoffmann et al. 2022, "Training Compute-Optimal Large Language Models"
Touvron et al. 2023, "LLaMA: Open and Efficient Foundation Language Models"

Pre-training for domain adaptation

Legal language

The prosecutor had difficulty proving <u>mens rea</u>, as the defendant seemed unaware that his actions were illegal.

The judge dismissed the case, citing the principle of <u>res judicata</u> as the issue had already been decided in a previous trial.

Despite the signed agreement, the contract was invalid as there was no <u>consideration</u> exchanged between the parties.

Medical language

After a strenuous workout, the patient experienced severe <u>myalgia</u> that lasted for several days.

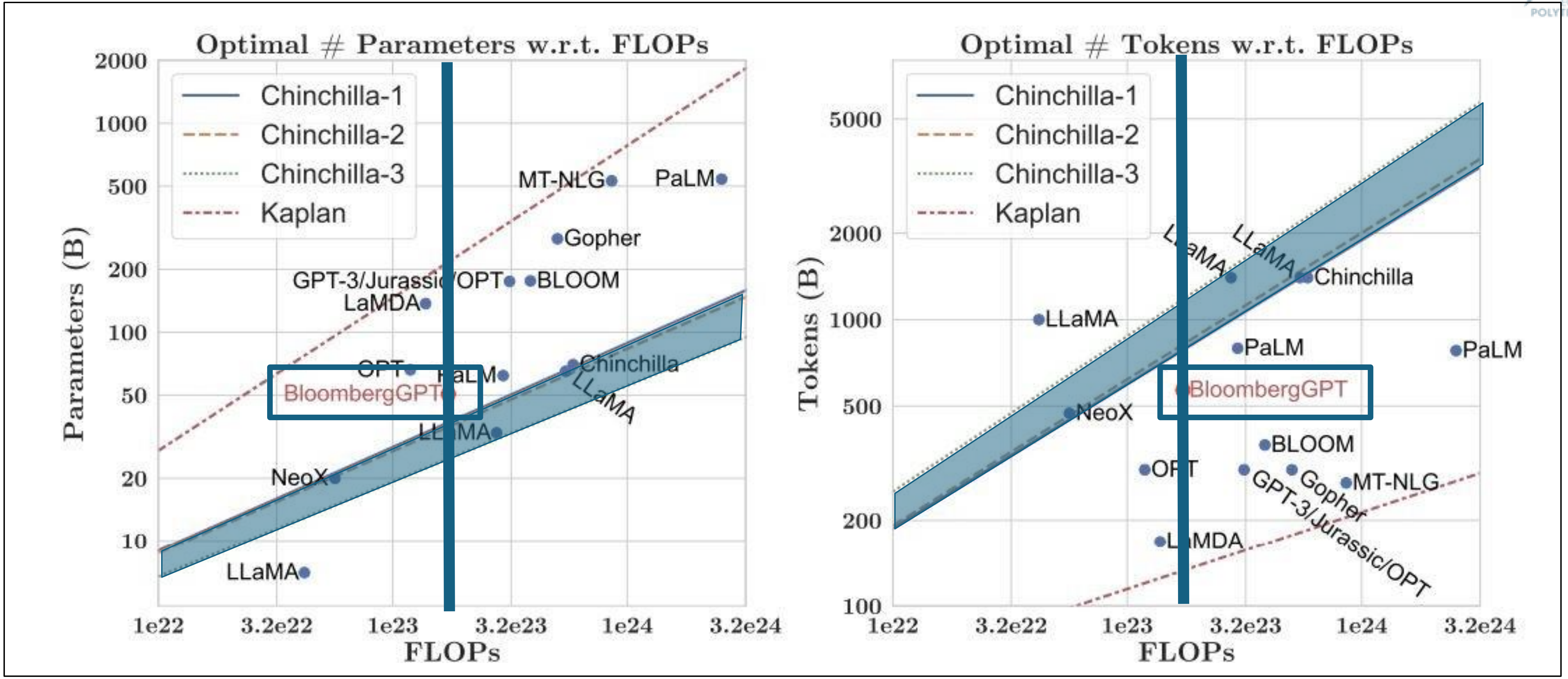
After the <u>biopsy</u>, the doctor confirmed that the tumor was <u>malignant</u> and recommended immediate treatment.

Sig: 1 tab po qid pc & hs

↓

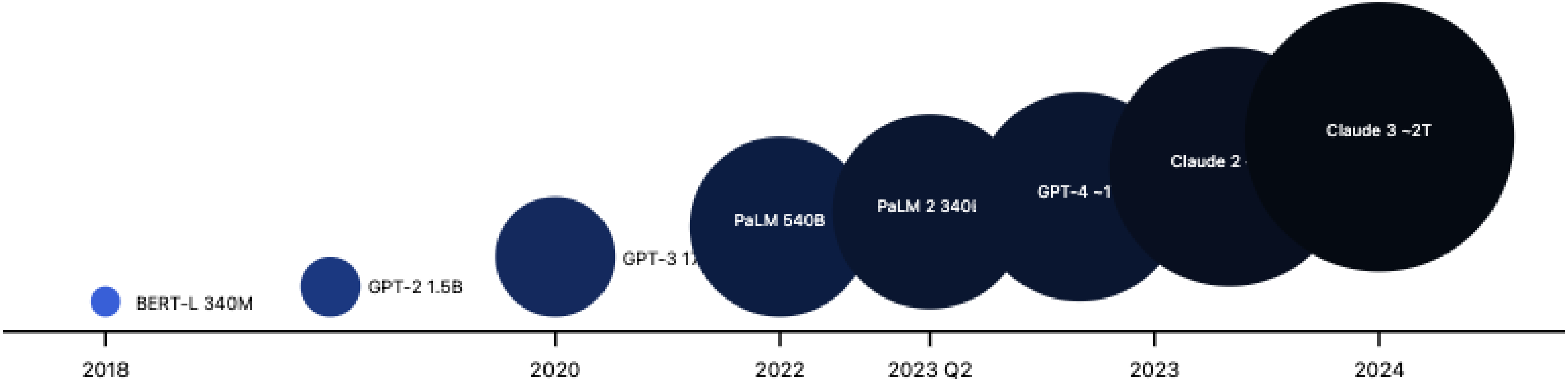
Take one tablet by mouth four times a day, after meals, and at bedtime.

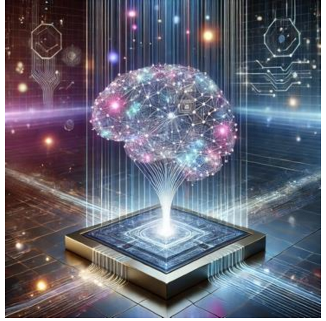
BloombergGPT relative to other LLMs



Source: Wu et al. 2023, "BloombergGPT: A Large Language Model for Finance"

Model size vs. time





created with ChatGPT, Oct 2024

Today's lecture

Part I:
Definition:
Define usecase

Part II:
Select:
Foundation Model to use or pretrain

Part III:
Adapt:
Foundation Model

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]

Part III: Outline

Adapt Foundation Models

- Prompting & Prompt Engineering
- Fine-tuning
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

created with chatGPT

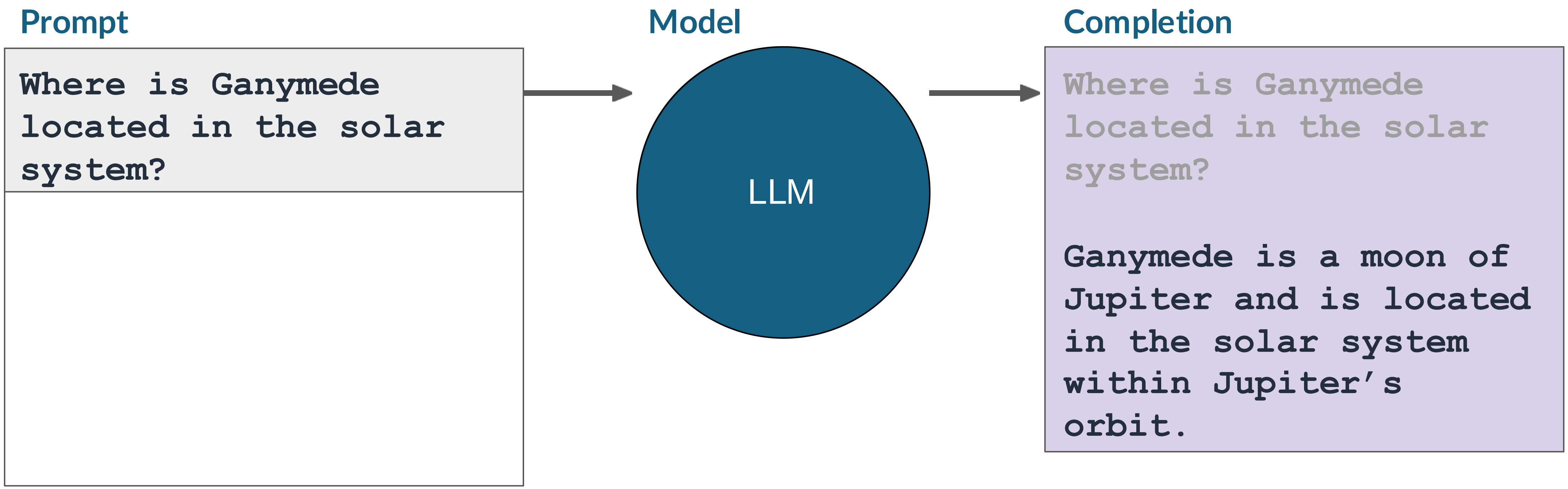
Part III: Outline

Adapt Foundation Models

- **Prompting & Prompt Engineering**
- Fine-tuning
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

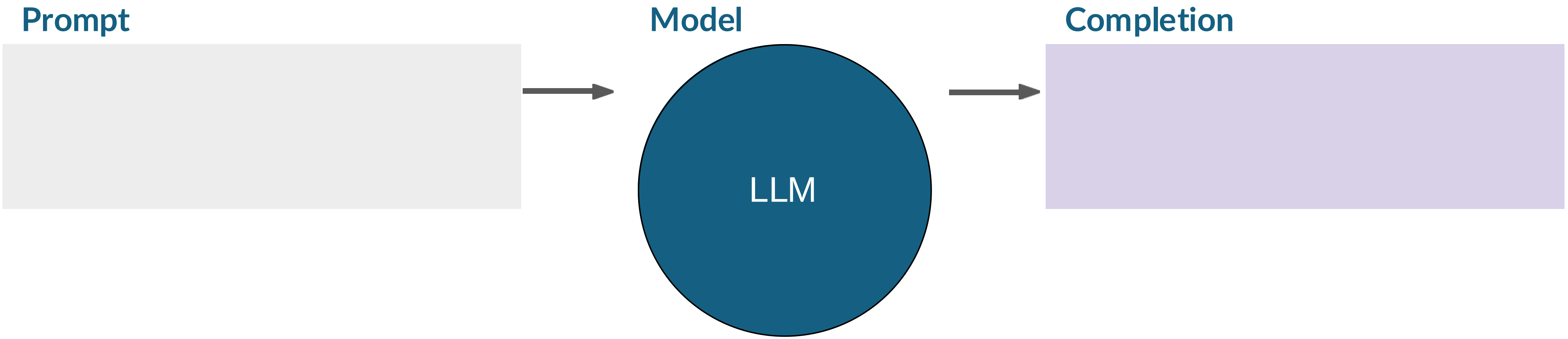
created with chatGPT

Prompting and prompt engineering

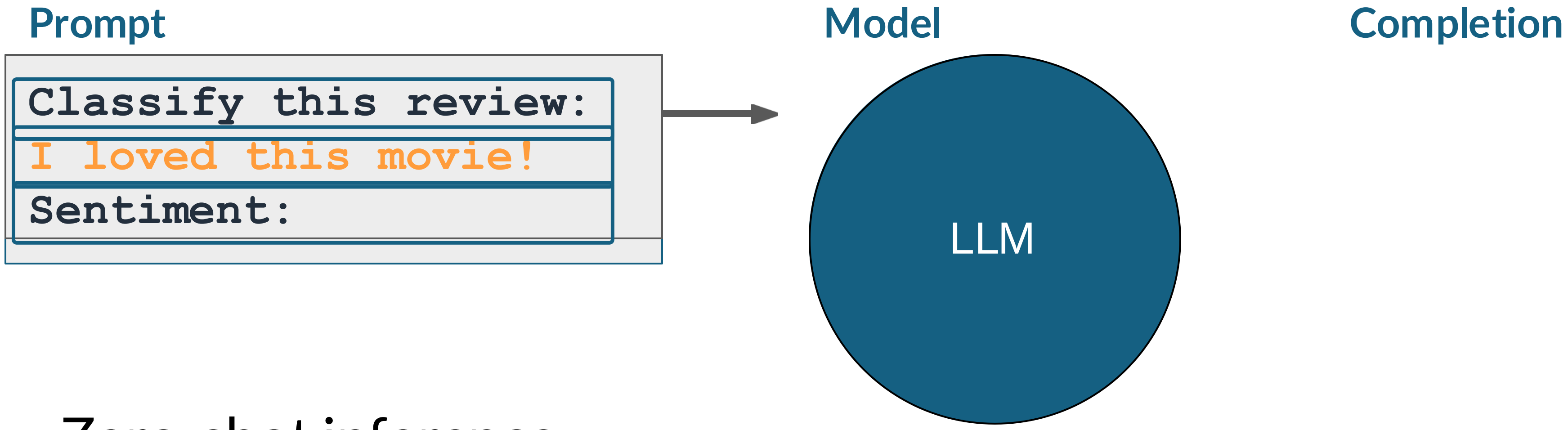


Context window: typically a few thousand words

In-context learning (ICL) - zero shot inference

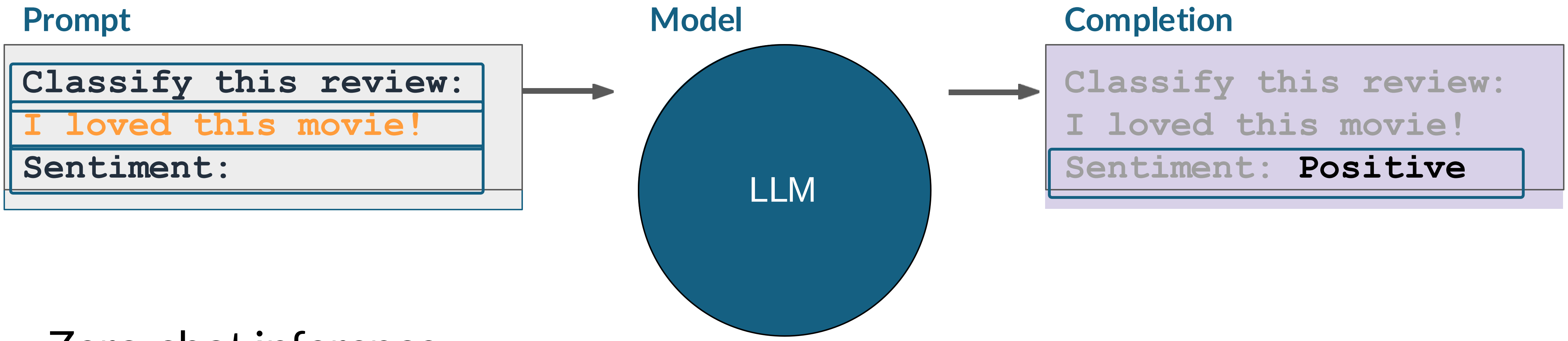


In-context learning (ICL) - zero shot inference



Zero-shot inference

In-context learning (ICL) - zero shot inference



Zero-shot inference

In-context learning (ICL) - zero shot inference

Prompt

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

Model

LLM

Completion

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

In-context learning (ICL) - one shot inference

Prompt

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

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

Model

LLM

Completion

In-context learning (ICL) - one shot inference

Prompt

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

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

Model

LLM

Completion

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

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

One-shot inference

In-context learning (ICL) - few shot inference

Prompt

```
Classify this review:

Classify this review:

Classify this review:
```

Model

LLM

Completion

In-context learning (ICL) - few shot inference

Prompt

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

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

Classify this review:
This is not great.
Sentiment:
```

Model

LLM

Completion

In-context learning (ICL) - few shot inference

Prompt

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

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

Classify this review:
This is not great.
Sentiment:
```

Model

LLM

Completion

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

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

Classify this review:
This is not great.
Sentiment: Negative
```

Summary: In-context learning (ICL)

Prompt // Zero Shot

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

Context Window
(few thousand words)

Prompt // One Shot

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

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

Prompt // Few Shot

>5 or 6 examples

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

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

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

The significance of scale: task ability

*Bert-base

Part III: Outline

Adapt Foundation Models

- Prompting & Prompt Engineering
- **Fine-tuning**
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

created with chatGPT

Fine-tuning

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: Positive
```

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

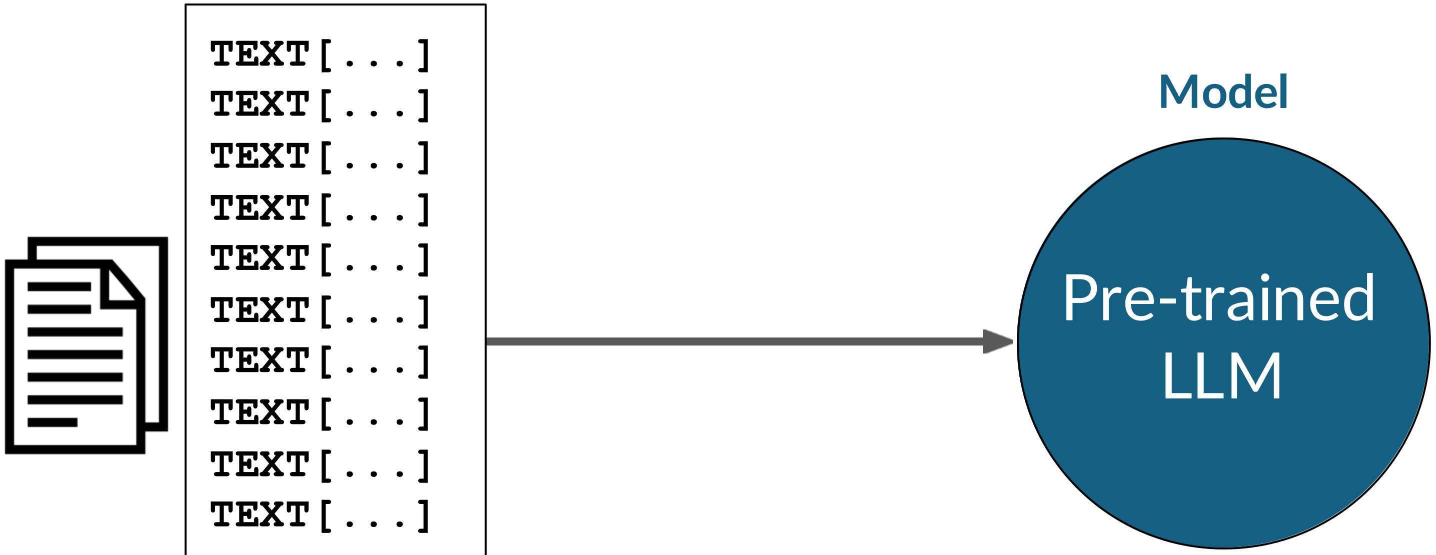
```
Classify this review:
I loved this movie!
Sentiment: Positive

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:
```

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

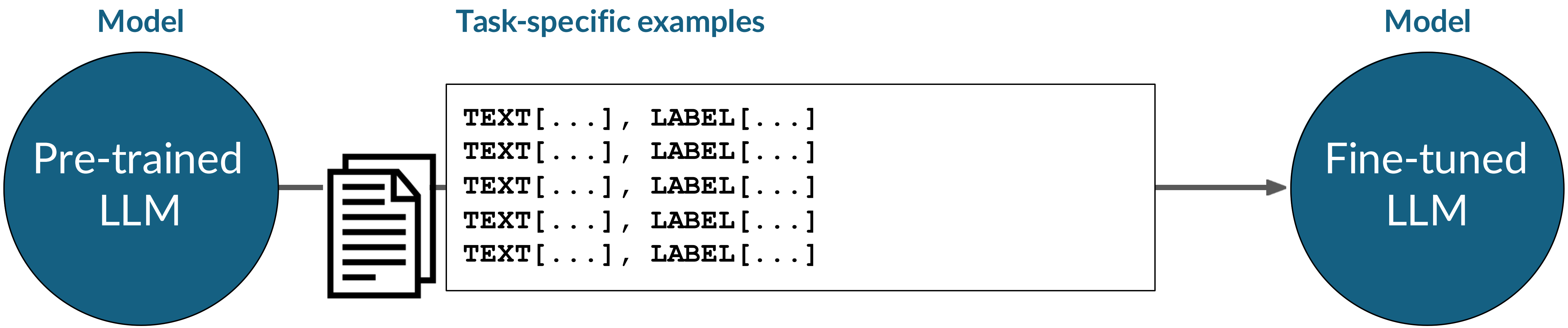
LLM fine-tuning at a high level



GB - TB - PB
of unstructured textual data

LLM pre-training

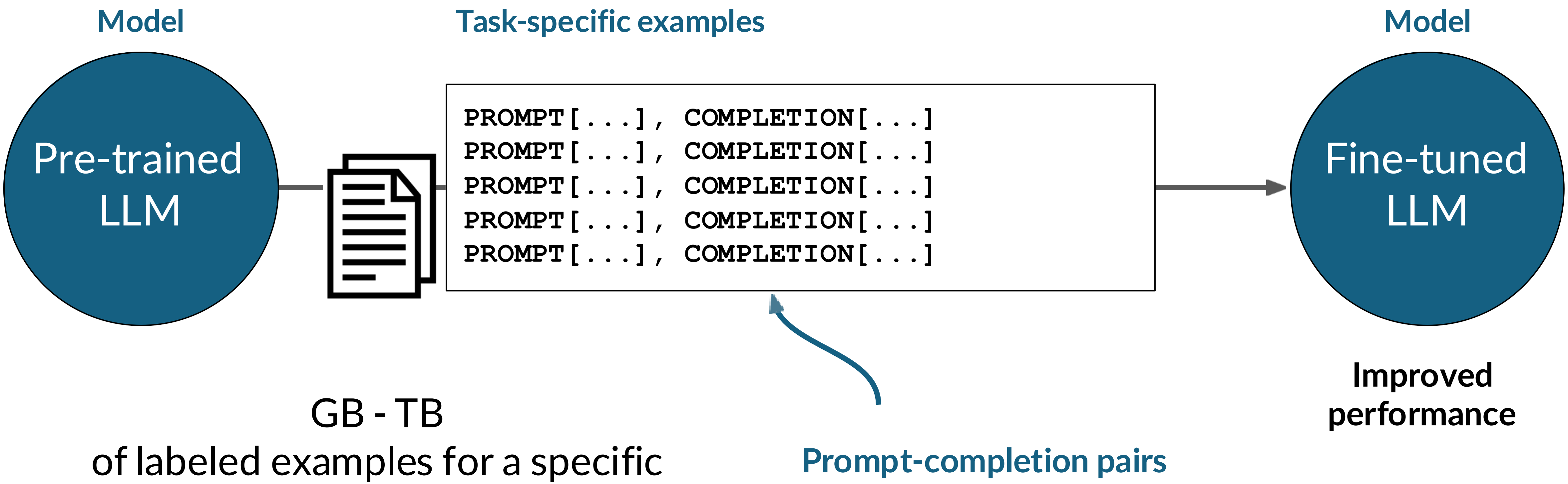
LLM fine-tuning at a high level



GB - TB
of labeled examples for a specific task or set of tasks

LLM fine-tuning

LLM fine-tuning at a high level



GB - TB
of labeled examples for a specific task or set of tasks

LLM fine-tuning

Part III: Outline

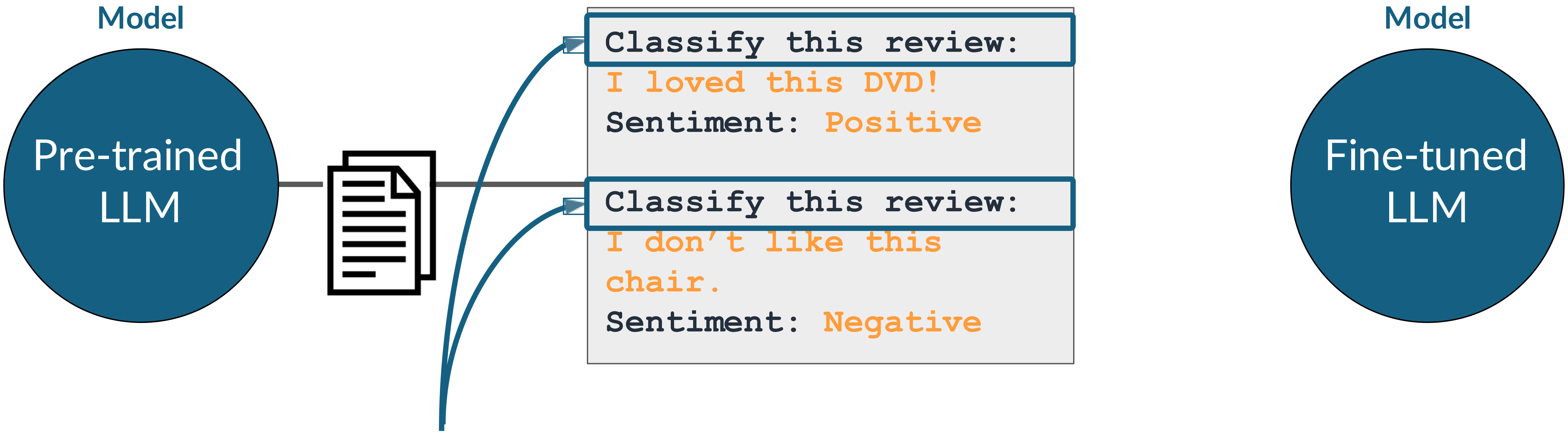
Adapt Foundation Models

- Prompting & Prompt Engineering
- **Fine-tuning**
  - **Instruction fine-tuning**
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

created with chatGPT

Instruction fine-tuning

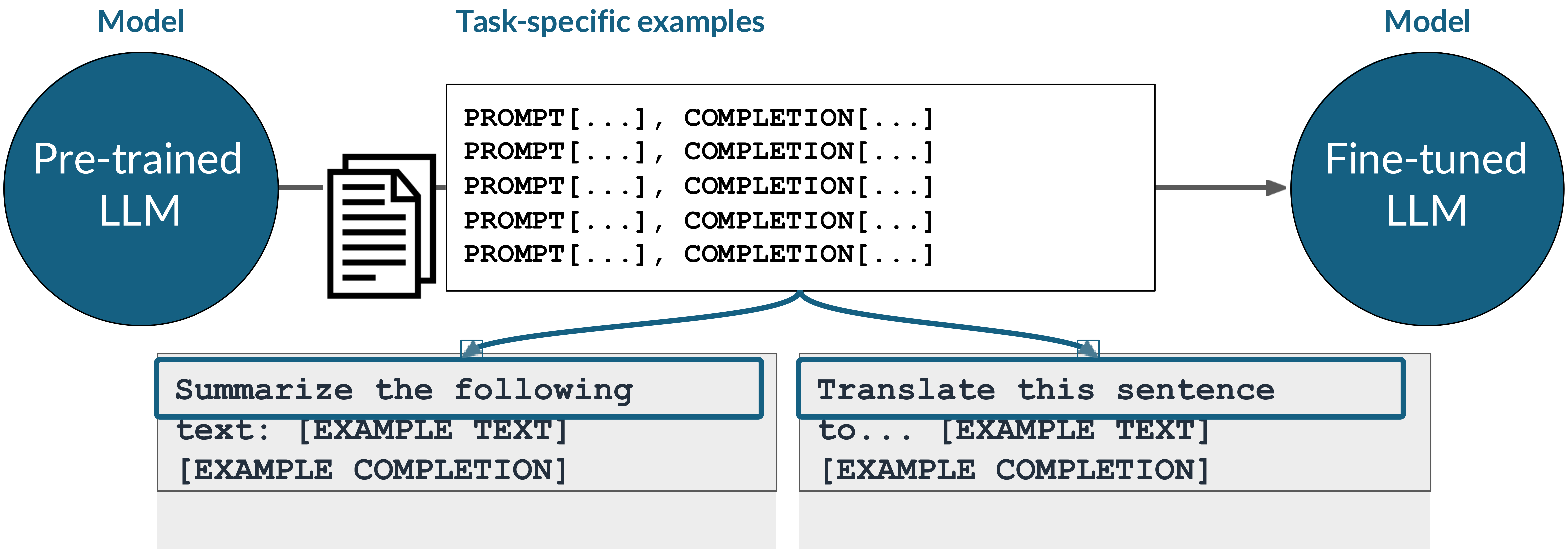
Using prompts to fine-tune LLMs with instruction



Each prompt/completion pair includes a specific "instruction" to the LLM

LLM fine-tuning

Using prompts to fine-tune LLMs with instruction



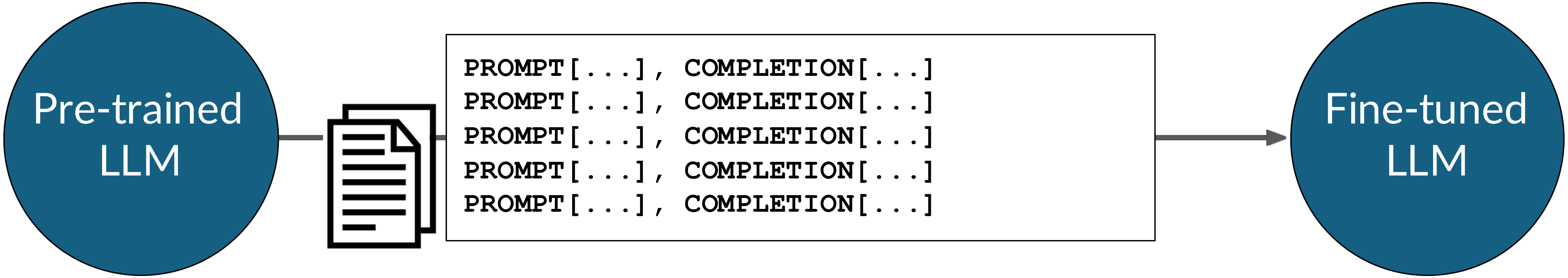
LLM fine-tuning

Using prompts to fine-tune LLMs with instruction

Model

Task-specific examples

Model



Full fine-tuning updates all parameters

Improved performance

LLM fine-tuning

Sample prompt instruction templates

- To store and process all the gradients, optimizers, …, full fine-tuning / Instruction fine-tuning requires:
  - memory
  - compute budget

→ updates all parameters

- How to instruction tune?
  - Start from datasets!

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

Prepared instruction dataset

Training splits

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

Training

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

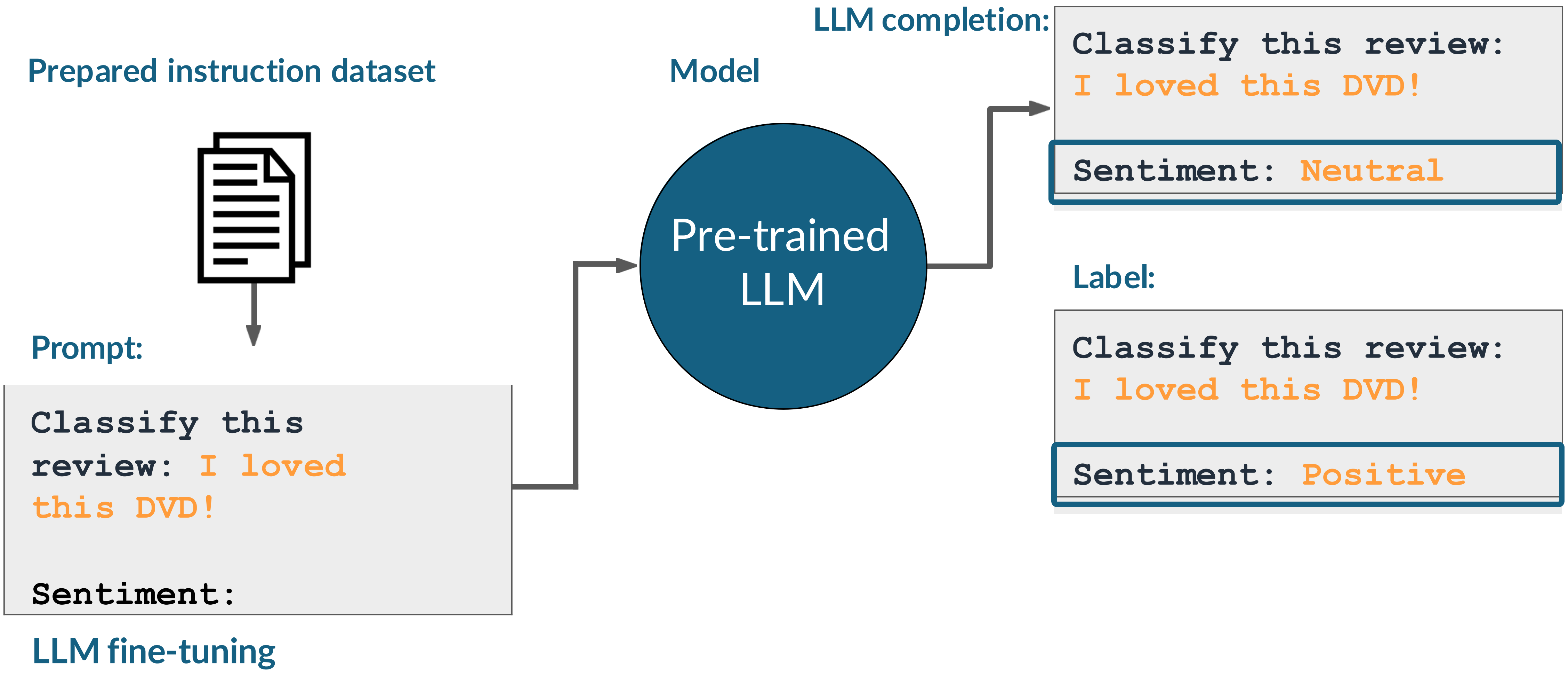
Validation

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

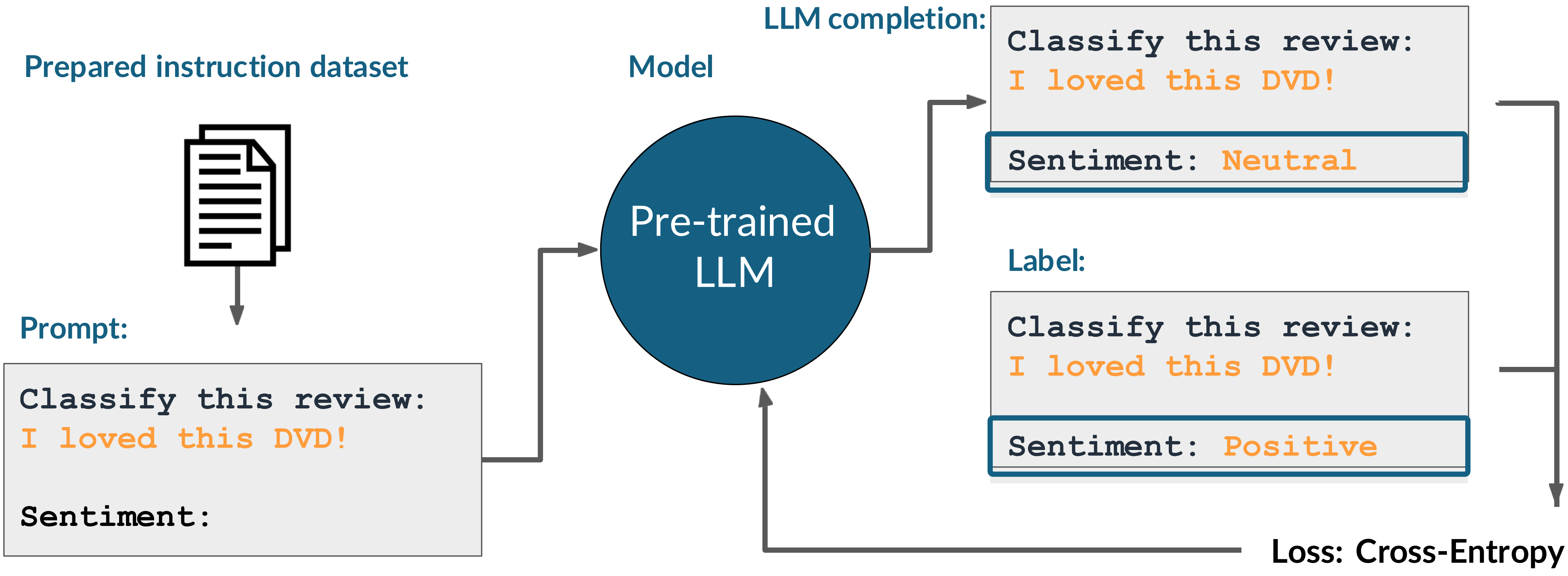
Test

LLM fine-tuning

LLM fine-tuning process



LLM fine-tuning process



LLM fine-tuning

LLM fine-tuning process

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

LLM fine-tuning process

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

LLM fine-tuning process

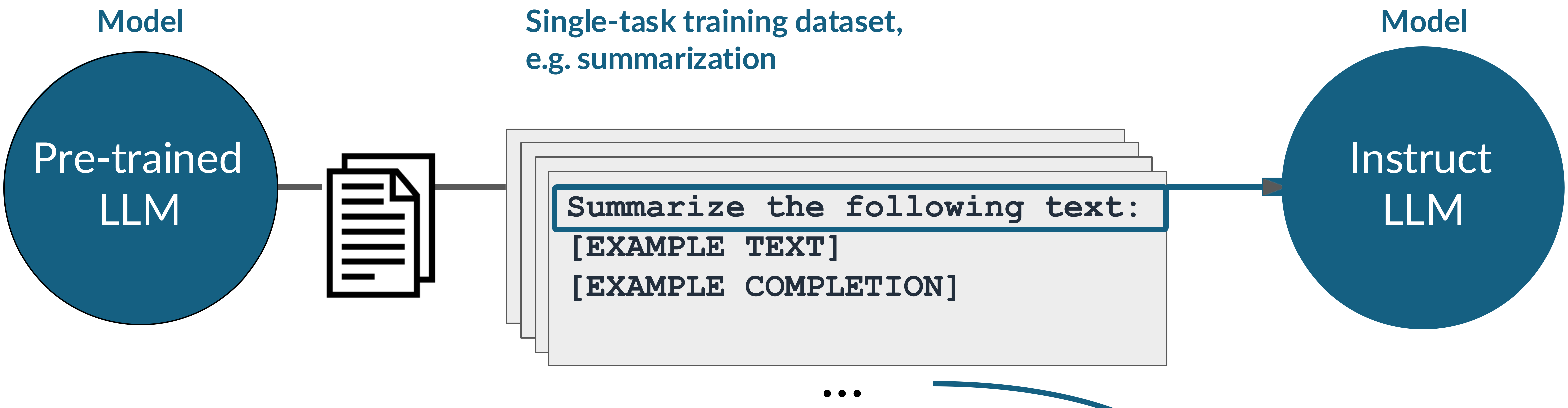
Part III: Outline

Adapt Foundation Models

- Prompting & Prompt Engineering
- **Fine-tuning**
  - Instruction fine-tuning
  - **Fine-tuning on a single task**
  - Fine-tuning on multiple tasks
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

created with chatGPT

Fine-tuning on a single task



Often, only 500-1000 examples needed to fine-tune a single task

Catastrophic forgetting

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

Before fine-tuning

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
```

Catastrophic forgetting

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

After fine-tuning

Prompt

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

Model

LLM

Completion

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

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

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.
The garden was positive.

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)

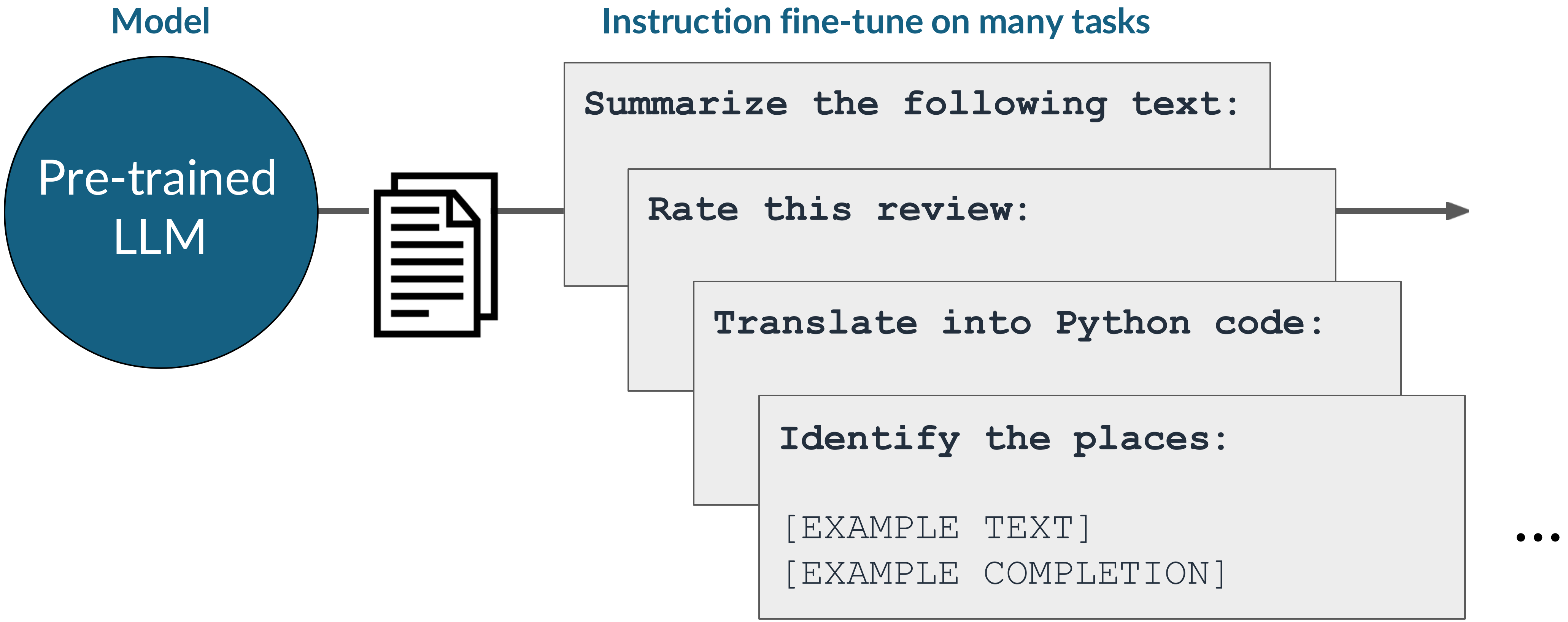
Part III: Outline

Adapt Foundation Models

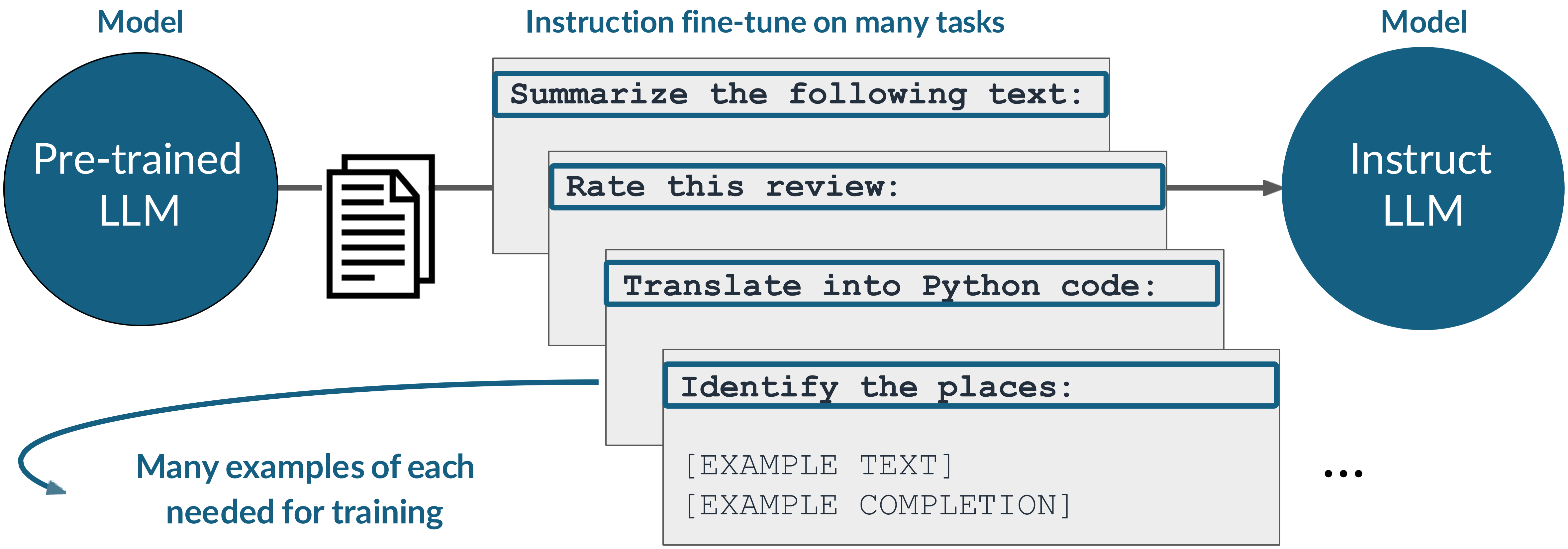
- Prompting & Prompt Engineering
- **Fine-tuning**
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - **Fine-tuning on multiple tasks**
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

created with chatGPT

Multi-task, instruction fine-tuning



Multi-task, instruction fine-tuning



Part III: Outline

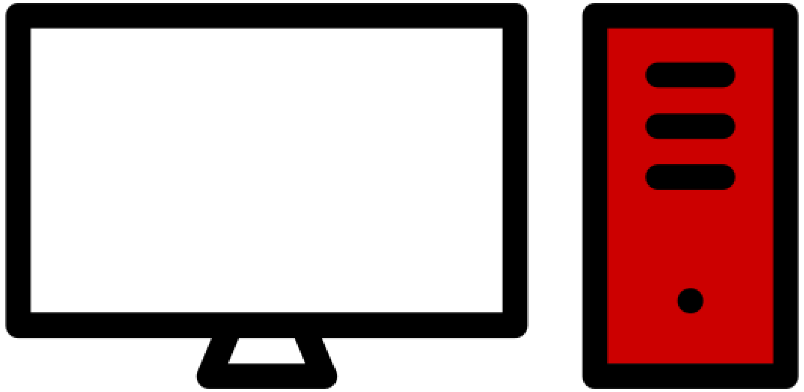
Adapt Foundation Models

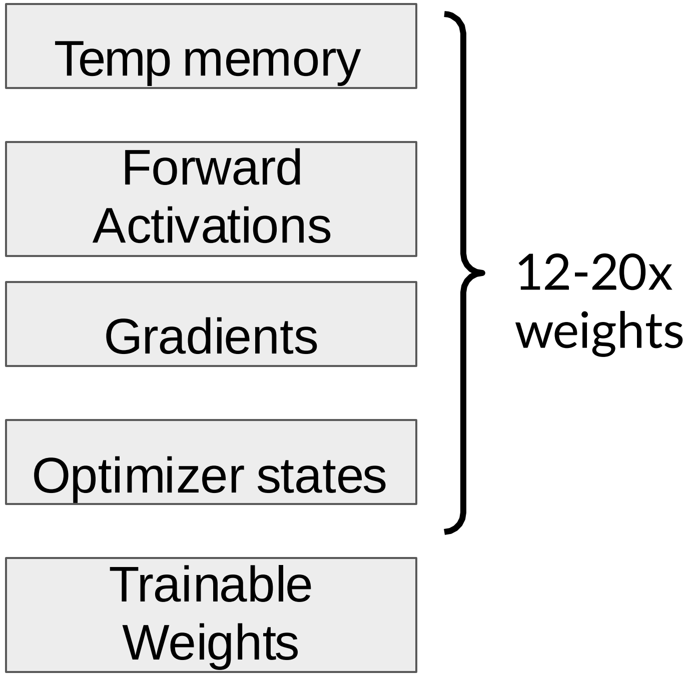
- Prompting & Prompt Engineering
- **Fine-tuning**
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - **Parameter efficient fine-tuning (PEFT)**
    - LoRA
    - Prompt tuning

created with chatGPT

Full fine-tuning of LLMs is challenging

- Computationally intensive
- Store additional items (hundreds of GBs)
- Memory allocation

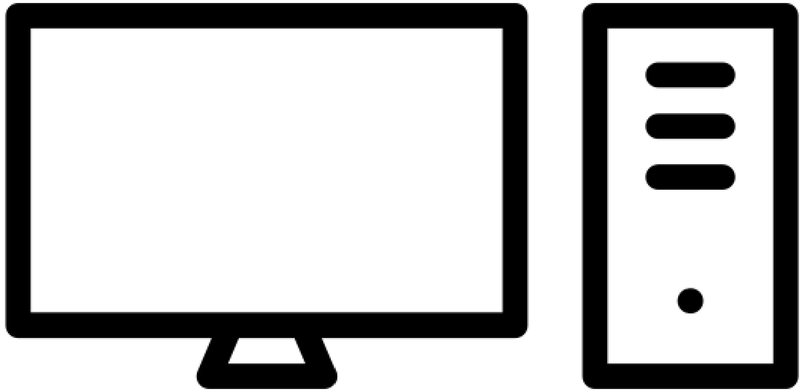




Parameter efficient fine-tuning (PEFT)

Small number of trainable layers

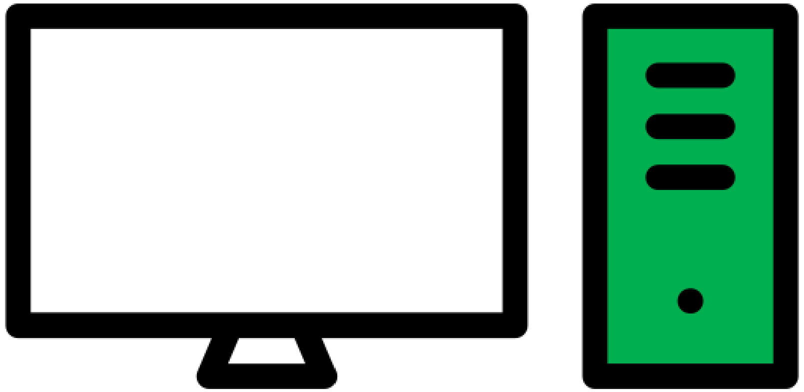
LLM with most layers frozen



Parameter efficient fine-tuning (PEFT)

New trainable layers

LLM with additional layers for PEFT

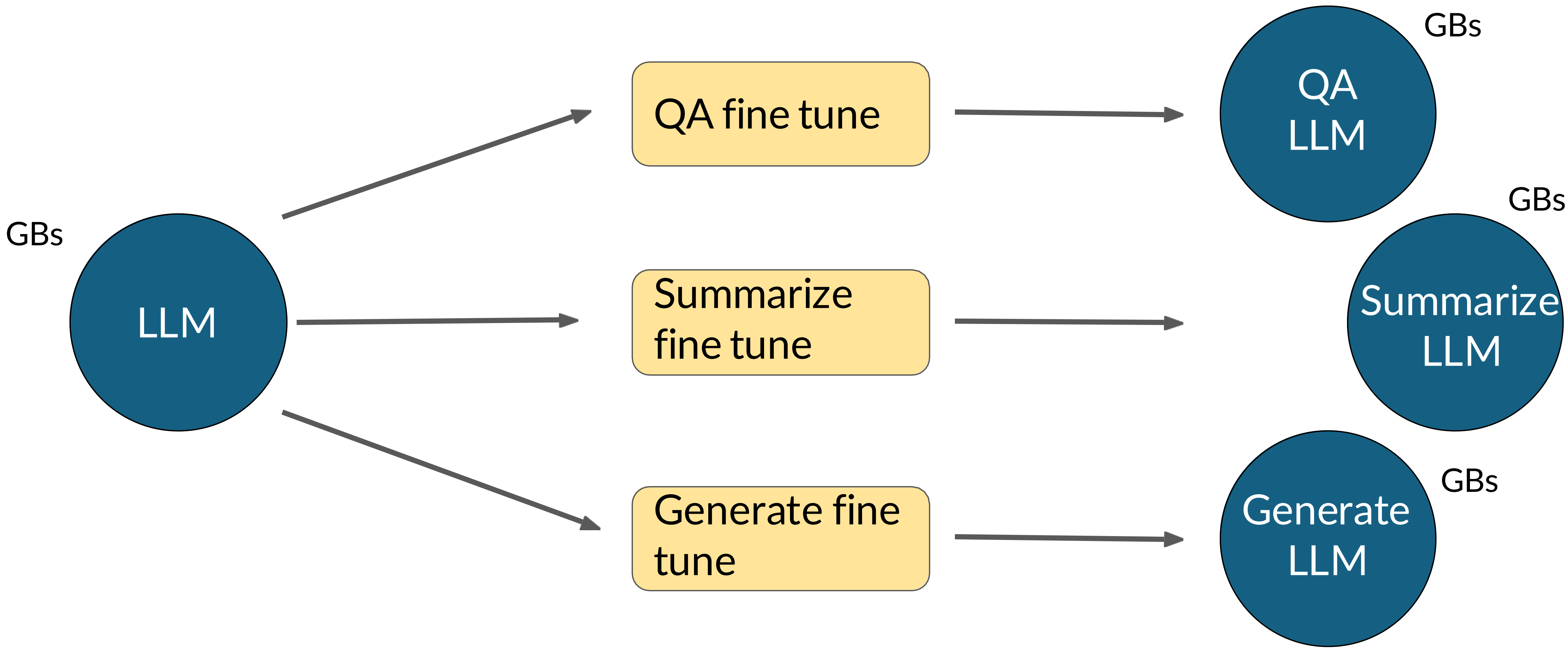


Less prone to catastrophic forgetting

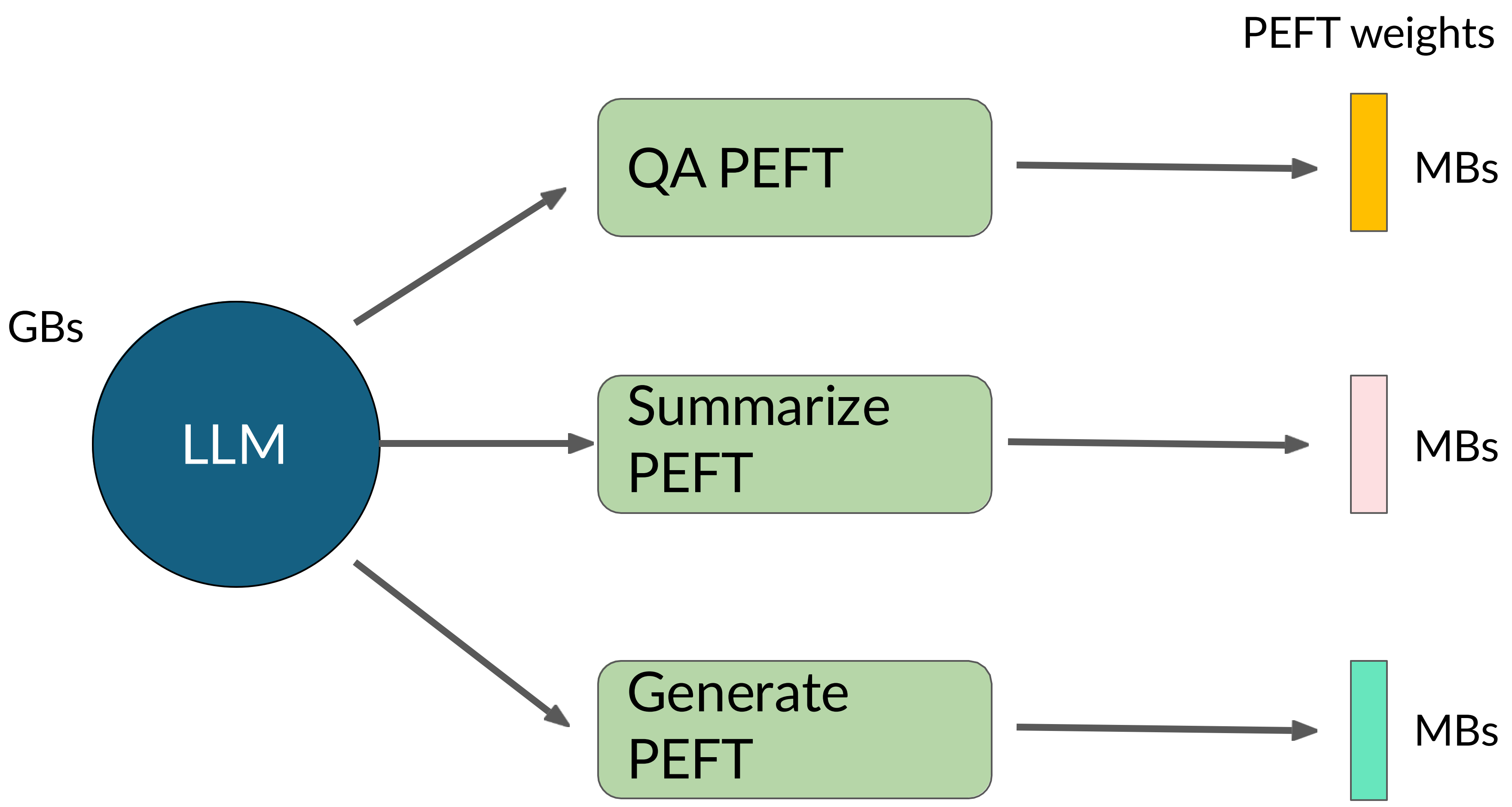
Other components

Trainable weights

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



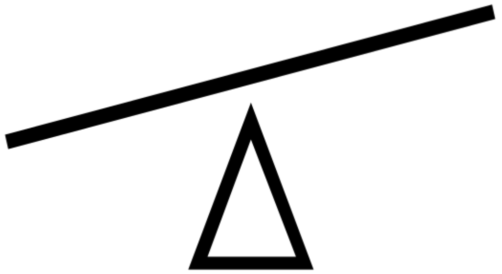
PEFT fine-tuning saves space and is flexible



PEFT Trade-offs

Parameter Efficiency

Memory Efficiency



Training Speed

Model Performance

Inference Costs

Categories of PEFT methods

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

PEFT methods

Selective

Select subset of initial LLM parameters to fine-tune

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

PEFT methods

Selective

Select subset of initial LLM parameters to fine-tune

Reparameterization

Reparameterize model weights using a low-rank representation

LoRA

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

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

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",

Part III: Outline

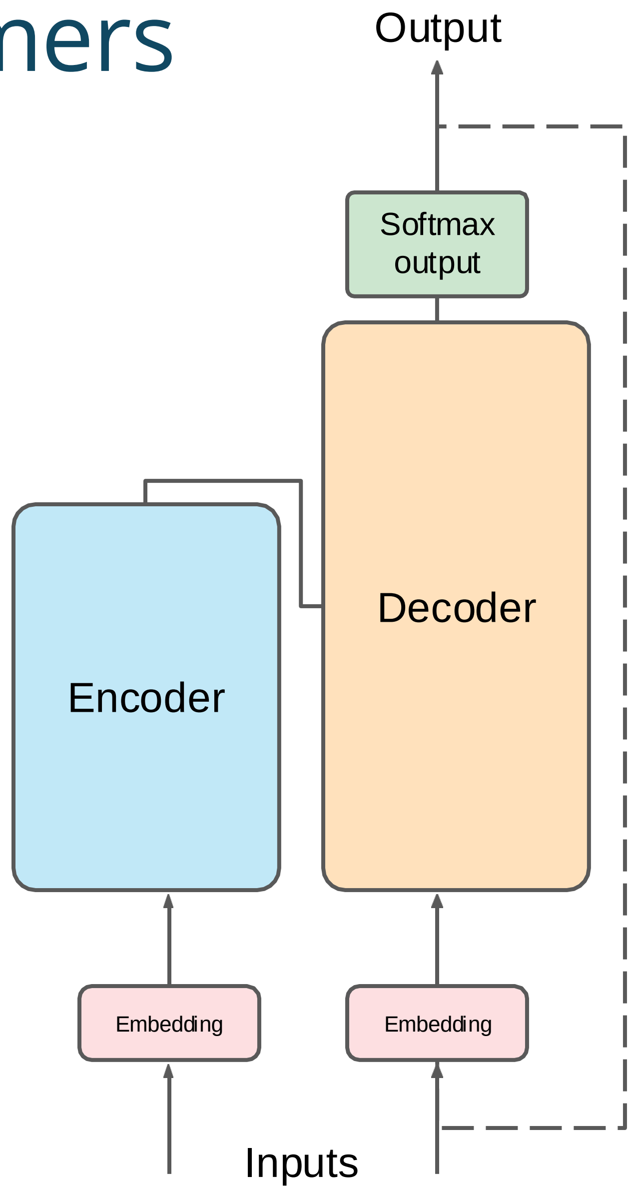
Adapt Foundation Models

- Prompting & Prompt Engineering
- **Fine-tuning**
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - **Parameter efficient fine-tuning (PEFT)**
    - **LoRA**
    - Prompt tuning

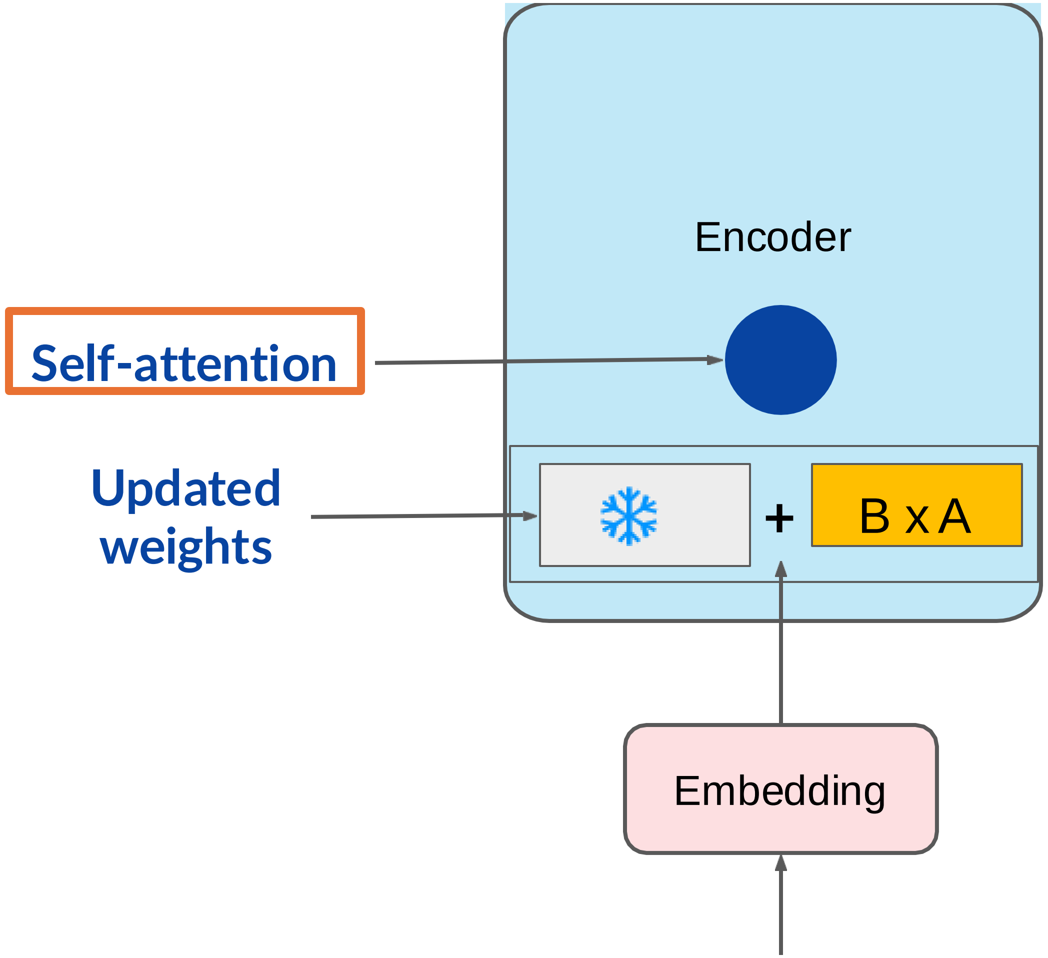
created with chatGPT

Low-Rank Adaptation of Large Language Models (LoRA)

Reminder: Transformers



LoRA: Low Rank Adaption of LLMs

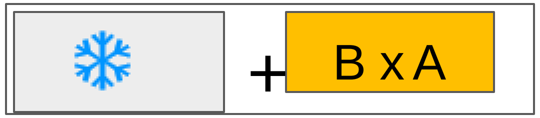


1. Freeze most of the original LLM weights.
2. Inject 2 **rank decomposition matrices**
3. Train the weights of the smaller matrices

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 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!**

Part III: Outline

Adapt Foundation Models

- Prompting & Prompt Engineering
- **Fine-tuning**
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - **Parameter efficient fine-tuning (PEFT)**
    - LoRA
    - **Prompt tuning**

created with chatGPT

Prompt Tuning with soft prompts (not prompt engineering)

Prompt tuning is **not** prompt engineering!

Prompt

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

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

Model

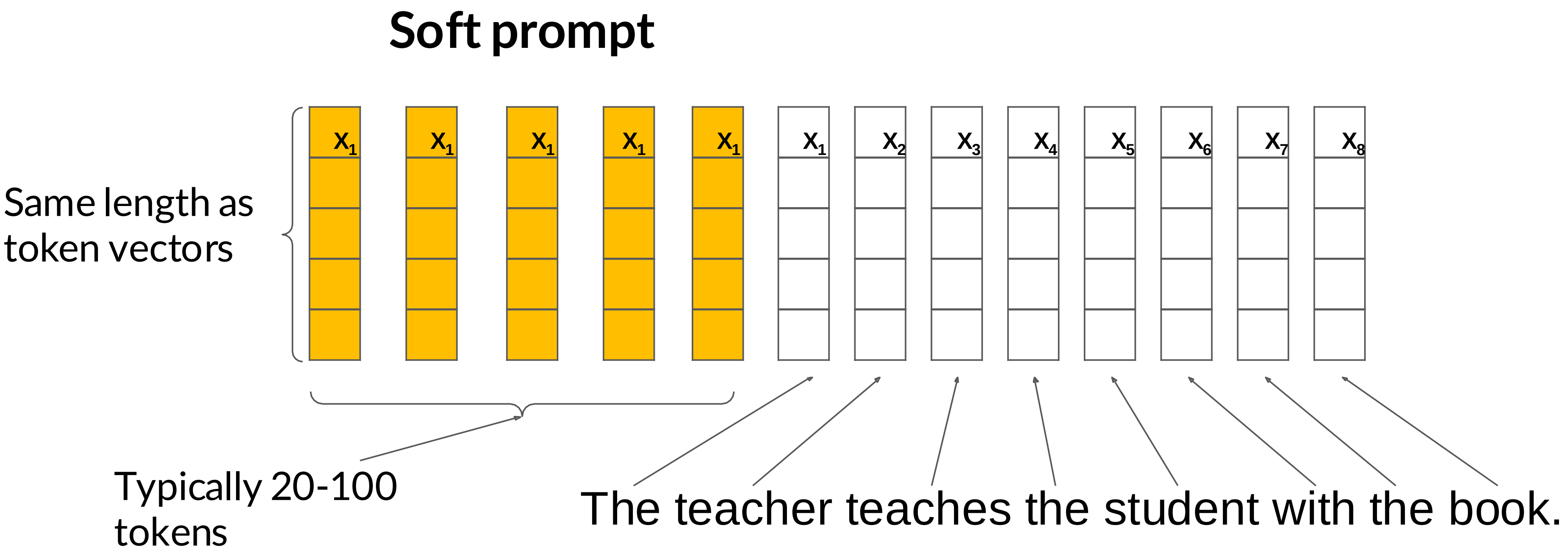
LLM

Completion

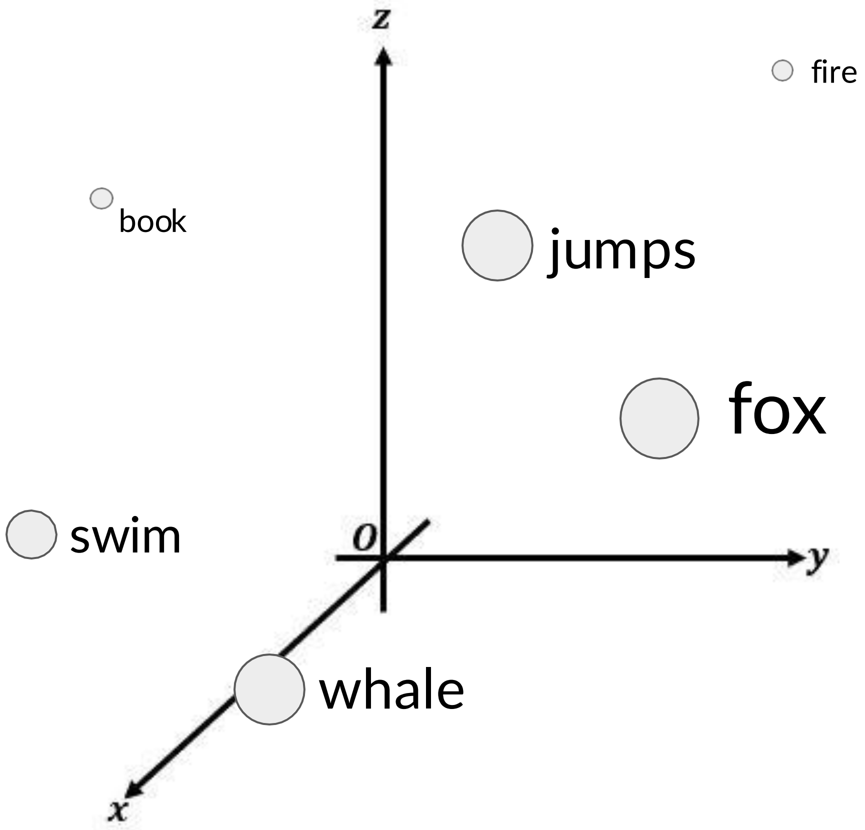
```
Classify this review:
I loved this DVD!
Sentiment: Positive

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

One-shot or Few-shot Inference

Prompt tuning adds trainable “soft prompt” to inputs



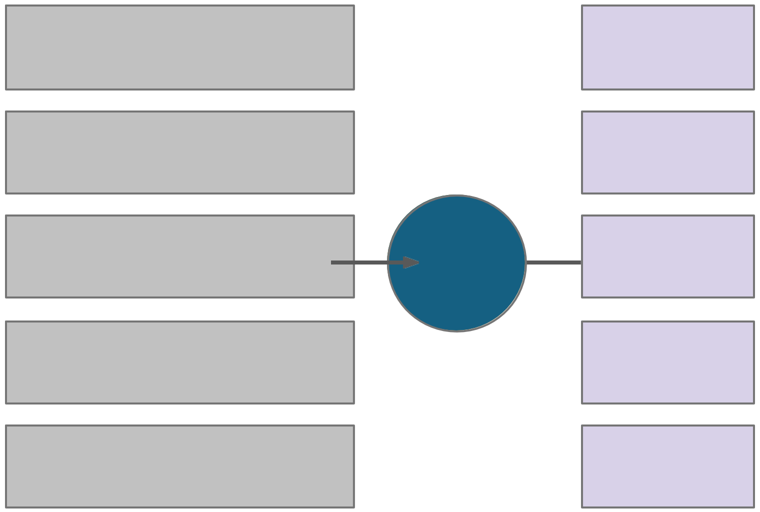
Soft prompts



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

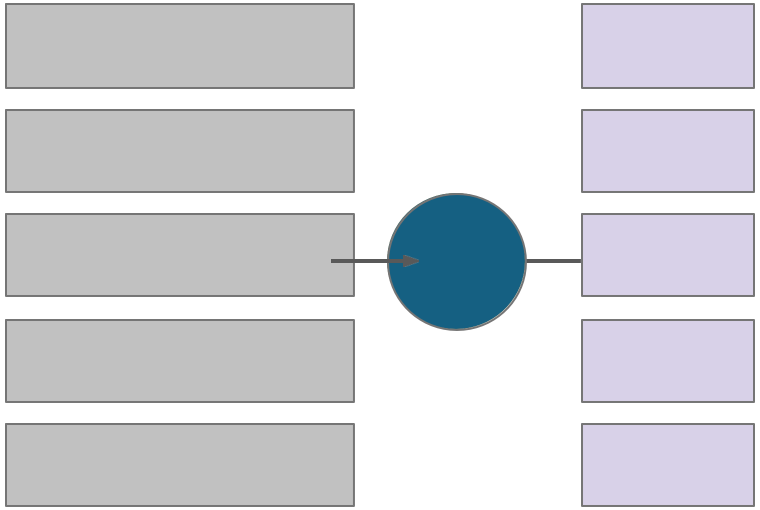
Full Fine-tuning vs prompt tuning

Weights of model updated during training



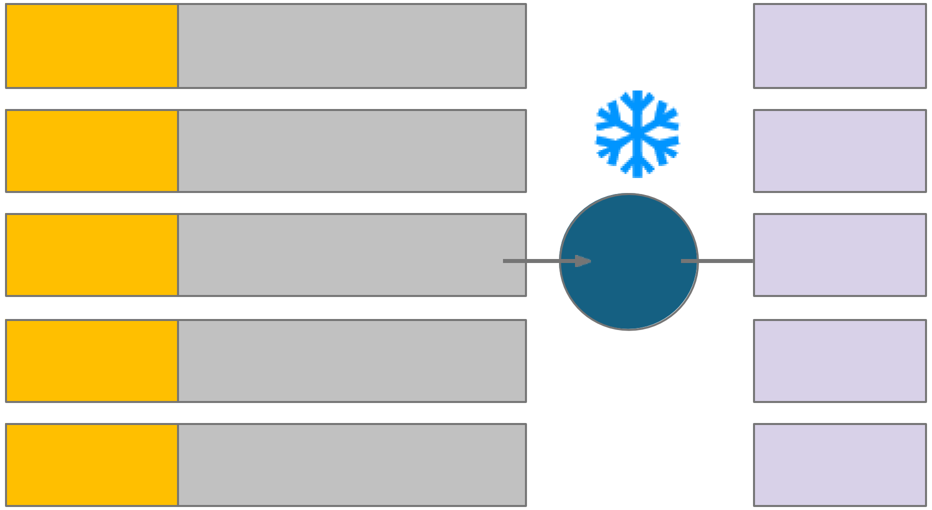
Full Fine-tuning vs prompt tuning

Weights of model updated during training



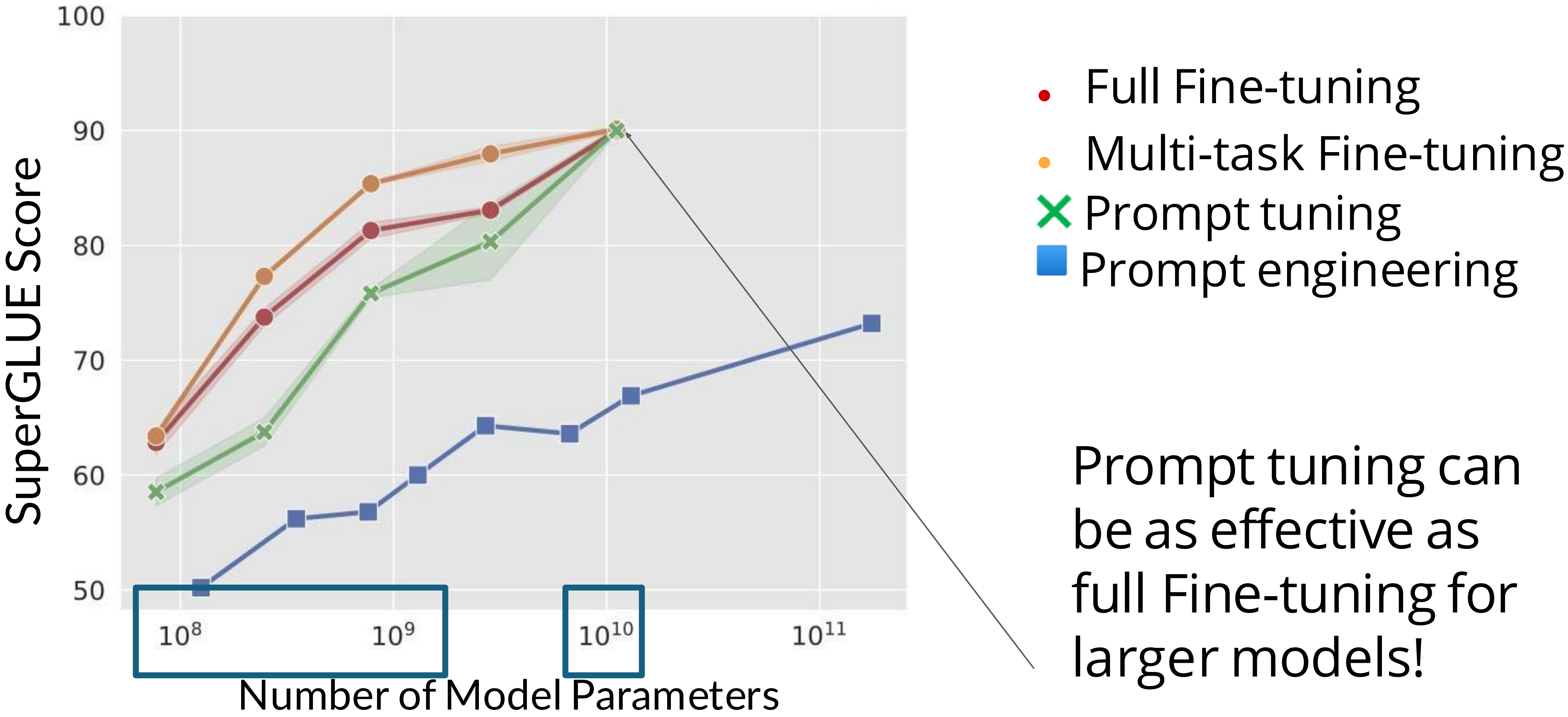
Millions to Billions of parameter updated

Weights of model frozen and soft prompt trained



10K - 100K of parameters updated

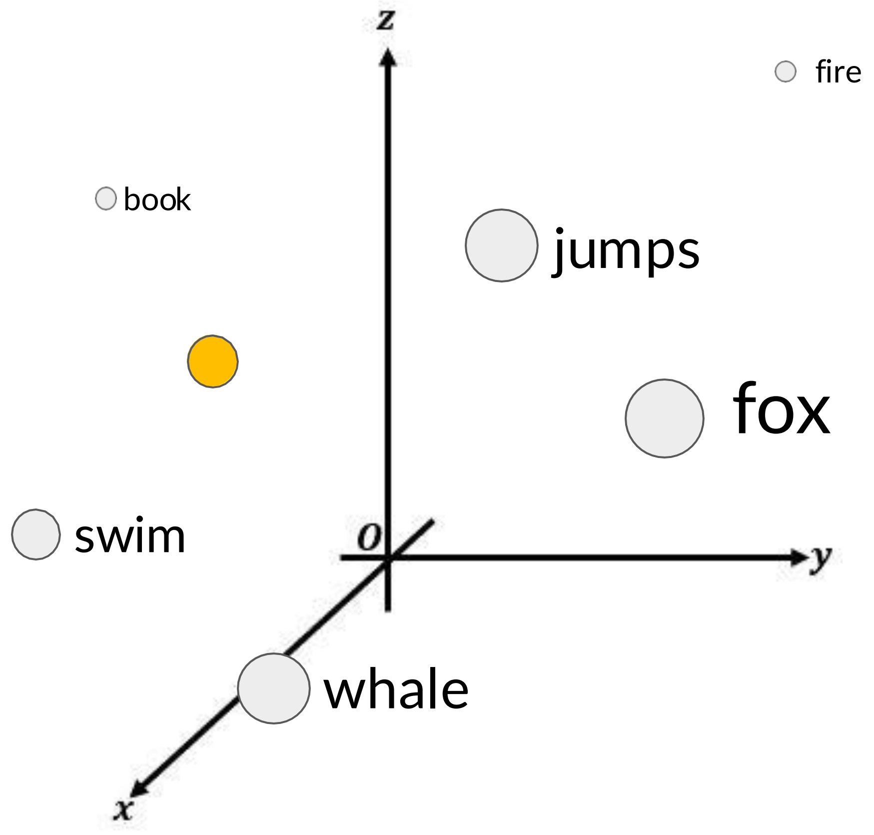
Performance of prompt tuning



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

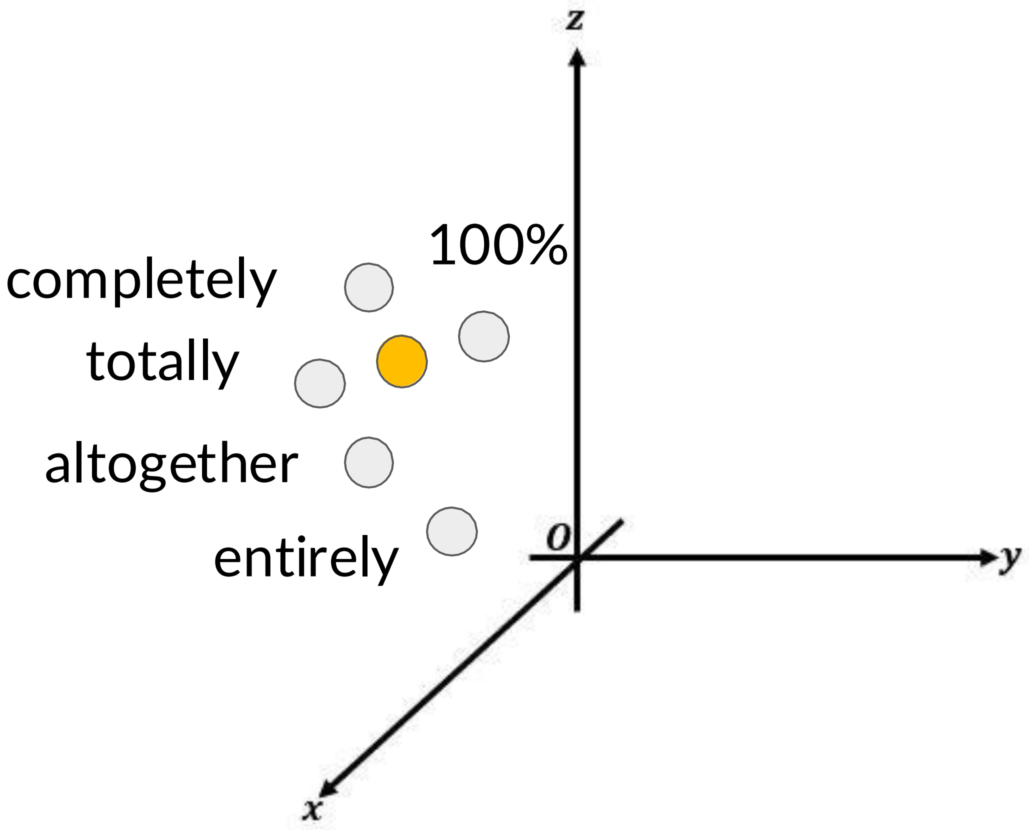
Source: Lester et al. 2021, "The Power of Scale for Parameter-Efficient Prompt Tuning"

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.

Part III: Summary

Adapt Foundation Models

- Prompting & Prompt Engineering
- Fine-tuning
  - Instruction fine-tuning
  - Fine-tuning on a single task
  - Fine-tuning on multiple tasks
  - Parameter efficient fine-tuning (PEFT)
    - LoRA
    - Prompt tuning

created with chatGPT

Summary: Prompt engineering with In-context learning (ICL)

Prompt // Zero Shot

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

Prompt // One Shot

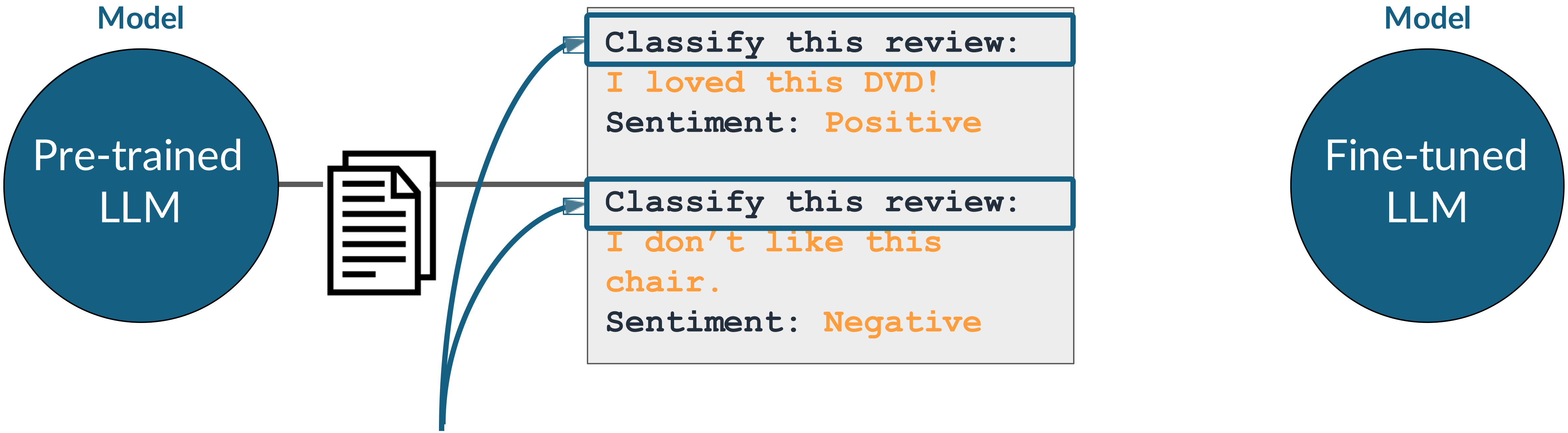
```
Classify this review:
I loved this movie!
Sentiment: Positive

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

Prompt // Few Shot >5 or 6 examples

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

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

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

Context Window
(few thousand words)

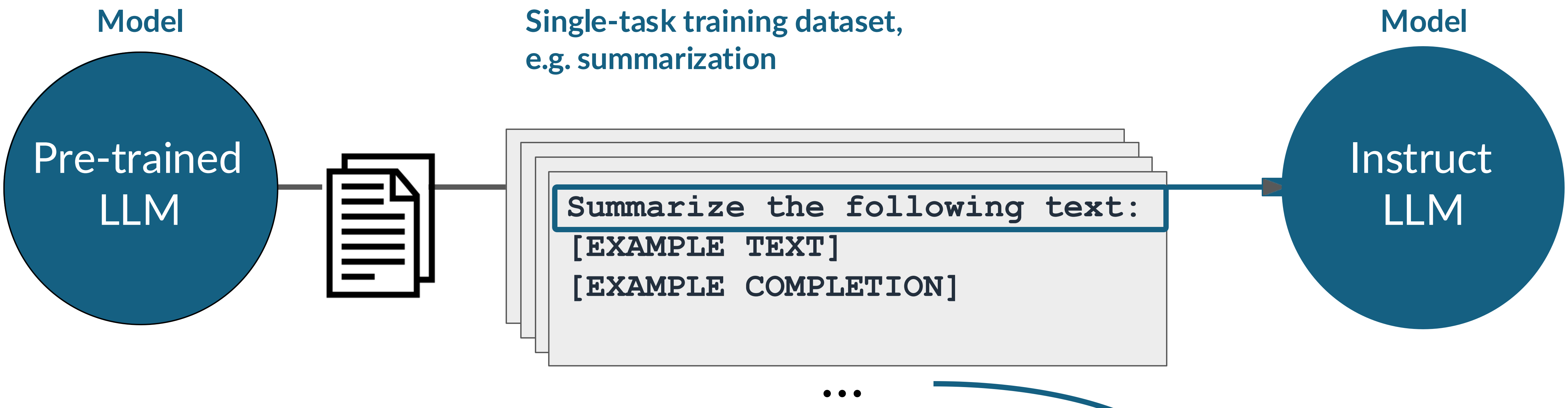
Summary: fine-tune LLMs w/ instructions



Each prompt/completion pair includes a specific "instruction" to the LLM

LLM fine-tuning

Summary: Fine-tuning on a single task

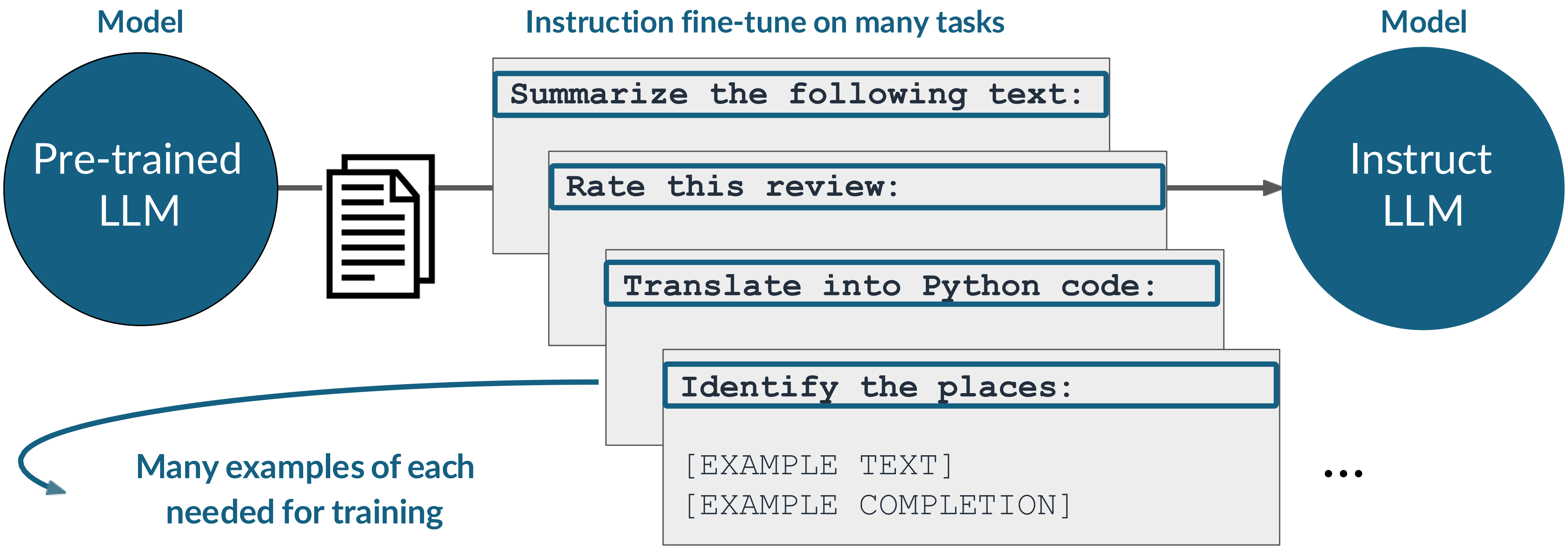


Often, only 500-1000 examples needed to fine-tune a single task

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)

Summary: Multi-task, instruction fine-tuning



Summary: 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”,

Thank you

Slides adapted from various sources: [Intro to Large Language Models, Andrej Karpathy, Executive Education Polytechnique, Udemy, Deeplearning.ai, Stanford University CS231n, Financial Times, New York Times, Hi!Paris summer school 2023]