



Instruction Finetuning

Abhijeet Kumar,
Director, Data Scientist
Fidelity Investments

Agenda



01. State of LLMs

Available options with us.
Why need ?

02. Memory Math & Scaling Laws

Challenge - How much memory ?
Quantization – Train or Infer
Scaling Laws

03. Train models to follow Instructions

FLAN-T5
ALPACA
DOLLY etc.

04. Finetuning for Use-Case

1. Full Finetuning
2. Parameter Efficient Finetuning
 - a) Prompt Tuning
 - b) LORA

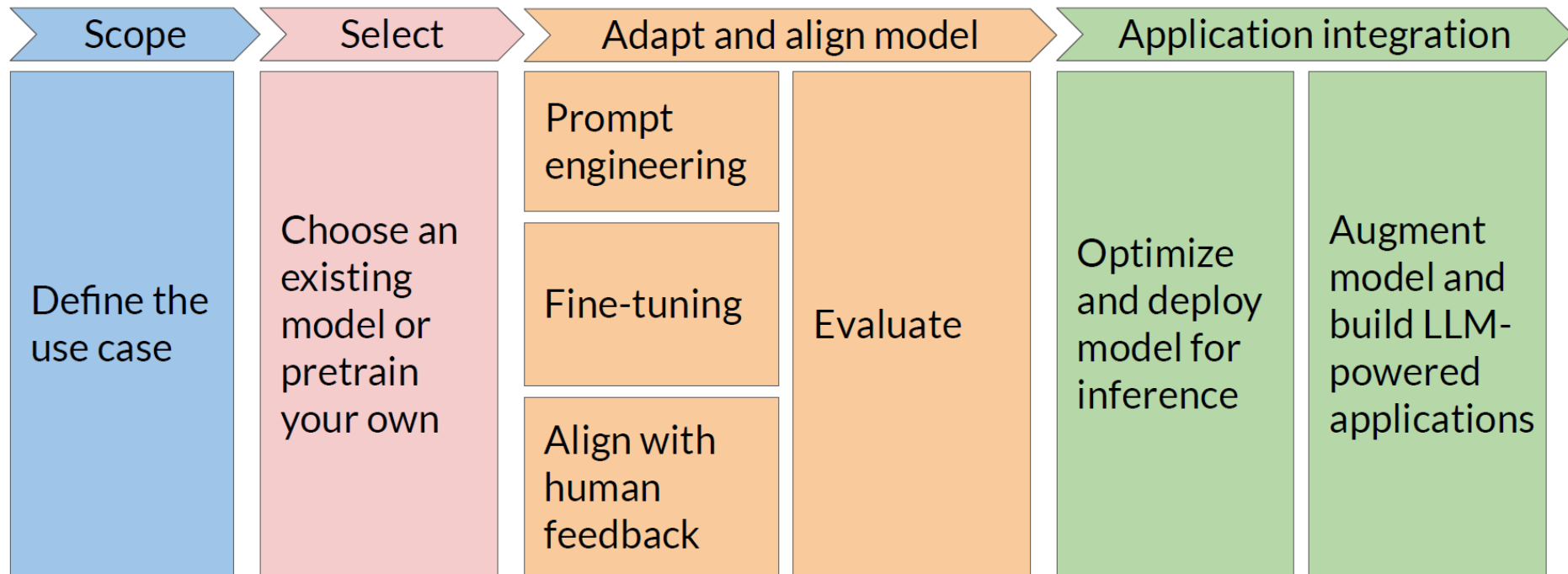
05. Implementation

1. Sentiment Analysis: text Classification
2. Dialog Summarization.
3. Training Alpaca/Dolly.

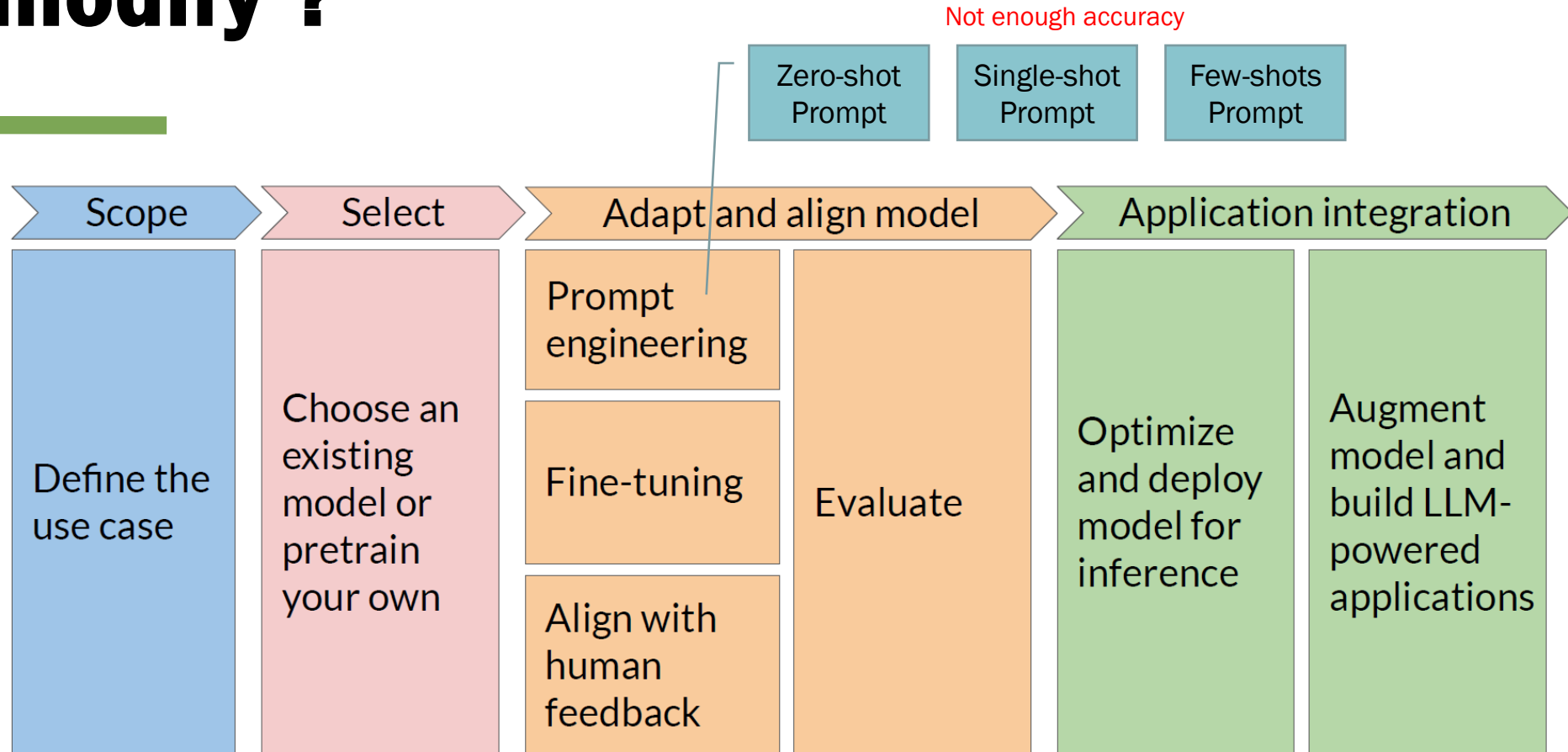
State of LLMs

Complexity Increases					
	Open-source LLMs (Llama, Falcon-40B)	3 rd Party API based LLMs (GPT-4, Anthropic)	Few-Shot Tuning LLMs (PEFT over Falcon-40B, Fine-tune GPT-3.5)	Instruction-Tuned LLMs (Supervised, RLHF, Finetuning Falcon-40B, T5-XXL)	(Pre)Training Fidelity LLM (Bloomberg GPT)
<u>Time to Market</u>	Fastest	Fastest	Fast	Time consuming	Long Term Effort
<u>Customization</u>	Least	Least	Medium, Refine tasks.	High	Very High
<u>Accuracy</u>	Low (In-domain task)	High	High (In-Domain Tasks)	Very High (In-Domain Tasks)	Very High (In-domain tasks)
<u>Privacy</u>	✓	✗	✓	✓	✓
<u>Cost</u>	Low	High	Low	High	Very high
<u>Data Req.</u>	Zero-shot, Few shots	Zero-Shot, Few shots	Less Labelled Training Data Required (~100-1K)	More Labelled Training Data Required (~50K)	Lots of Unsupervised & Supervised Data Required.
<u>Skill Req.</u>	Prompt (AI) Engineers	Prompt (AI) Engineers	Data Scientists	Highly Skilled Data Scientists	Highly skilled Data Scientist
<u>Use-cases</u>	Simple Task (General English)	Complex Task (General English)	Complex task, nuances of domain learnt	Better at complex tasks, learn domain nuances well	Domain Foundational Model
<u>Model Updates</u>	Becomes Stale	3 rd party Responsibility	Owned	Owned	Owned (Costly)

Why modify ?

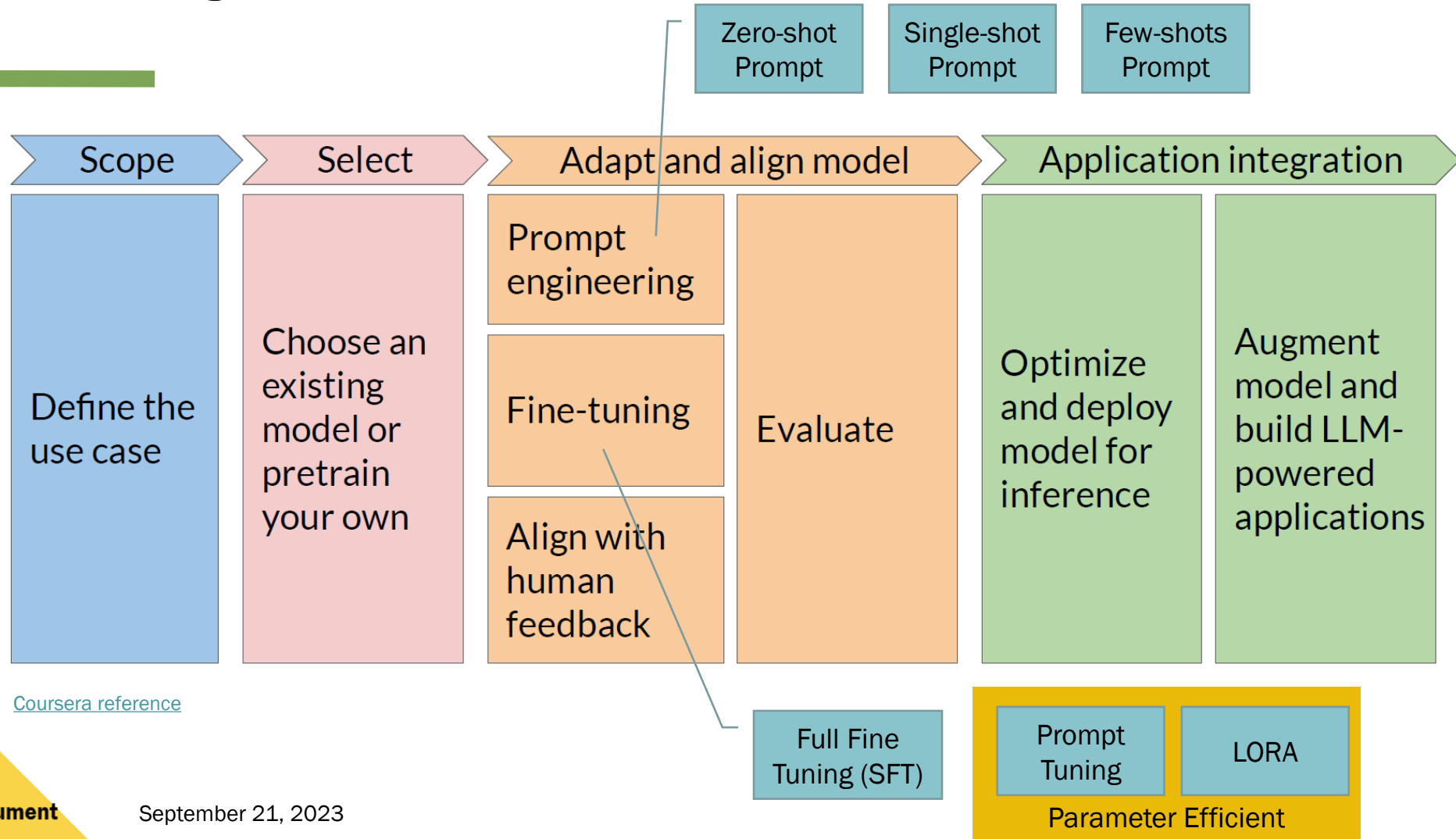


Why modify ?

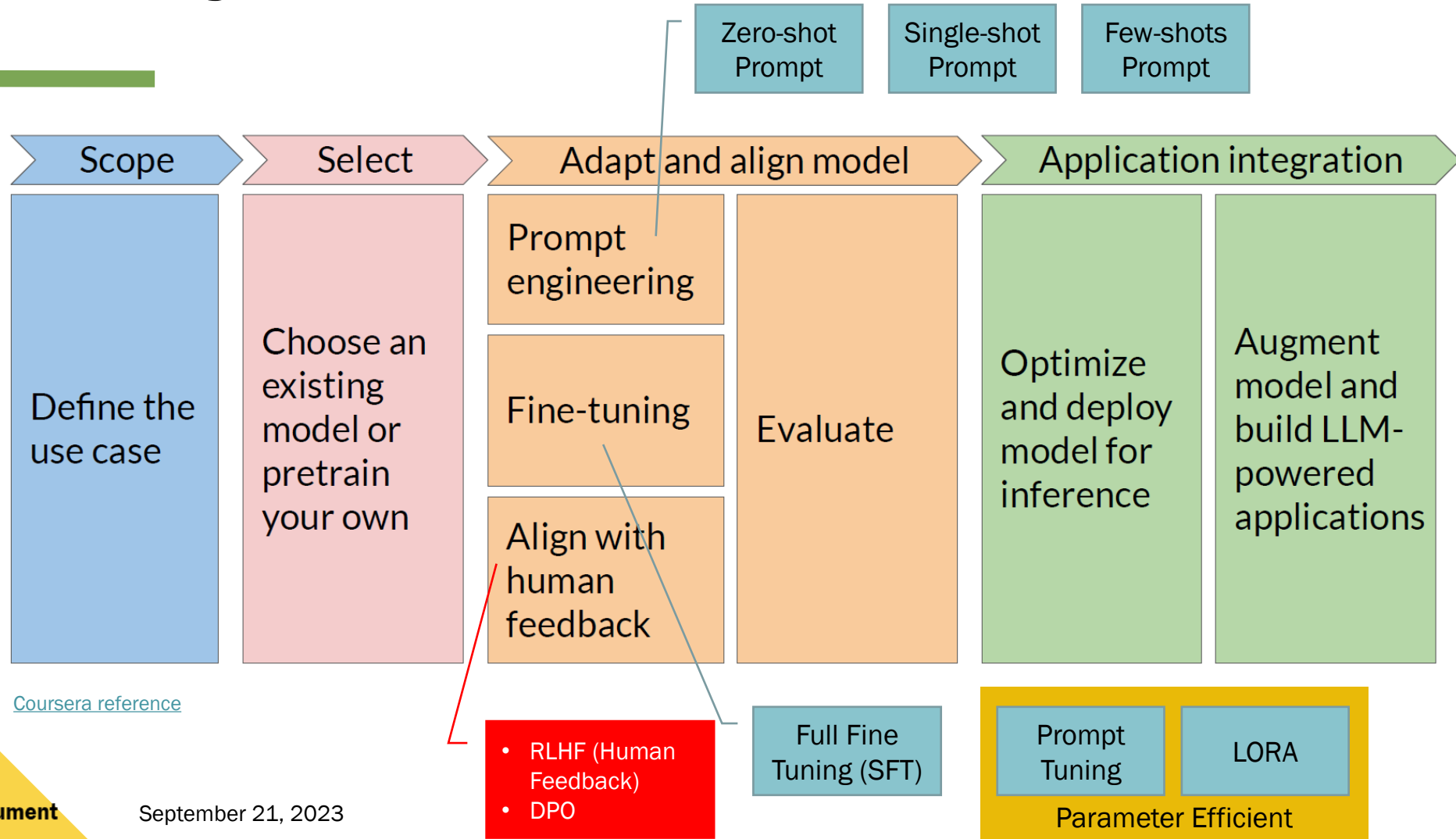


[Coursera reference](#)

Why modify ?



Why modify ?





Memory Math, Quantization & Scaling Laws

Memory

`OutOfMemoryError`: CUDA out of memory.

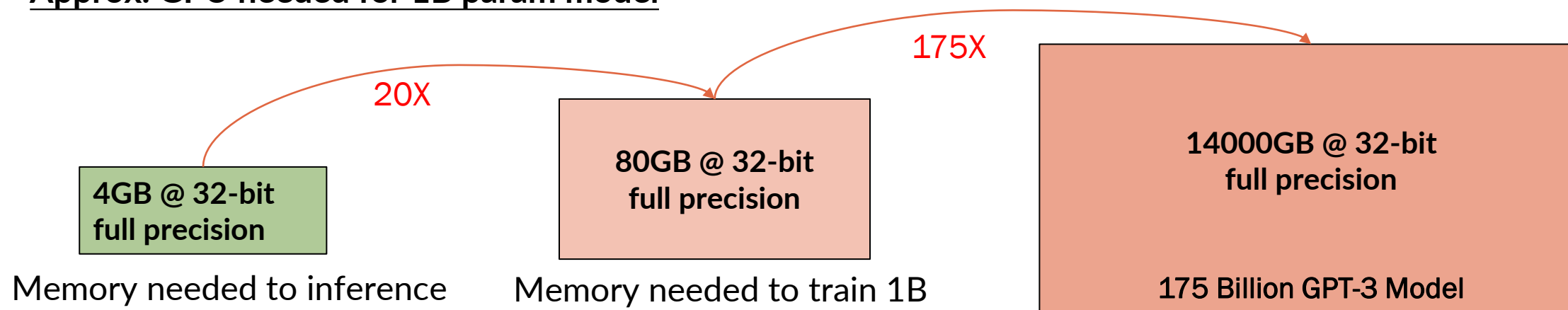
Let's do a quick math....

1 parameter = 4 bytes (32-bit float)

1B parameters = 4×10^9 bytes = **4GB**

Param name	Bytes per param
Weights	4 bytes
Adam states	8 bytes
Gradients	4 bytes
Activations & temp memory	8 bytes

Approx. GPU needed for 1B param model



Memory

`OutOfMemoryError`: CUDA out of memory.

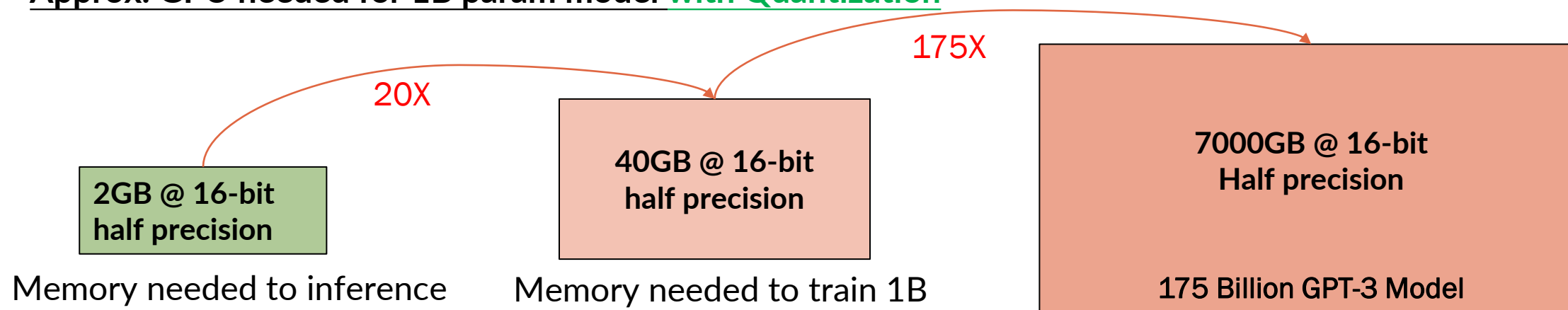
Let's do a quick math....

1 parameter = 4 bytes (32-bit float)

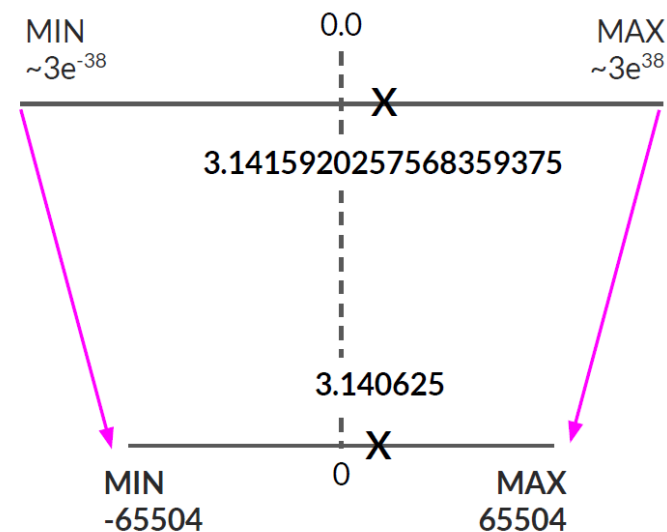
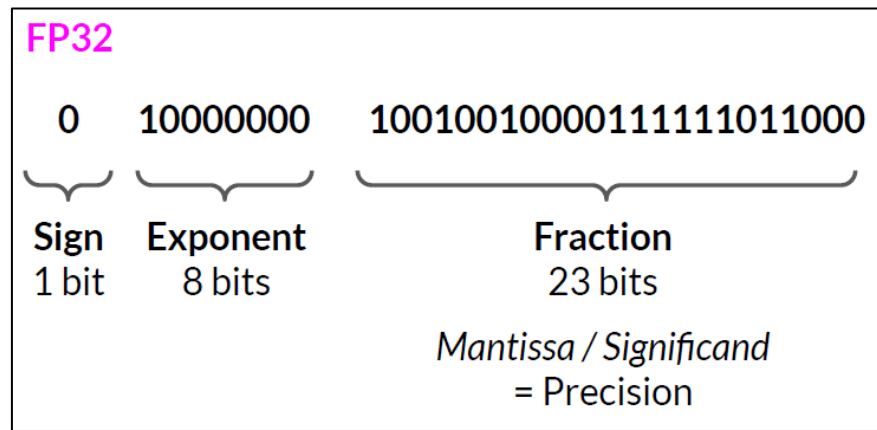
1B parameters = 4×10^9 bytes = **4GB**

Param name	Bytes per param
Weights	4 bytes
Adam states	8 bytes
Gradients	4 bytes
Activations & temp memory	8 bytes

Approx. GPU needed for 1B param model **with Quantization**



Quantization



Data Type	PI value	Exponent	Fraction
FP32	3.1415920257568359375	8	23
FP16	3.140625	5	10
BF16	3.140625	8	7
INT8**	3	0	7

**INT8 & INT4 can only be used for inference (not training)



Training Models to follow instructions

Instruction Following Models



Multi-task Instruction Finetuning from base models

<https://ai.googleblog.com/2021/10/introducing-flan-more-generalizable.html>

Instruction Following Models



Base Model	Instruction Models	Finetuning ?	Dataset (Task)
T5-XXL	FLAN T5-XXL	SFT	FLAN – 1836 tasks, 15M pairs
Falcon-40B	Falcon-40B-Instruct	SFT	Baize – 100K pairs
Llama2-70B	Llama2-70B-Chat	RLHF	27.5K pairs, 2.9M pairs
GPT-3.5	Chat-GPT	RLHF	-----
PALM	Flan-PALM	SFT	FLAN – 1836 tasks, 15M pairs
Pythia	Dolly	SFT	Dolly – 15K pairs
Llama-13B	Alpaca-13B	SFT	GPT3 – 52K pairs
Llama-13B	Vicuna-13B	SFT	GPT-4 – 70K pairs
Llama-13B	Koala-13B	SFT	Public Dialogues - 500K pairs

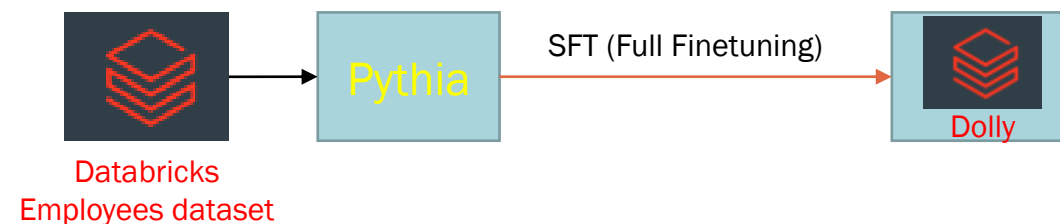
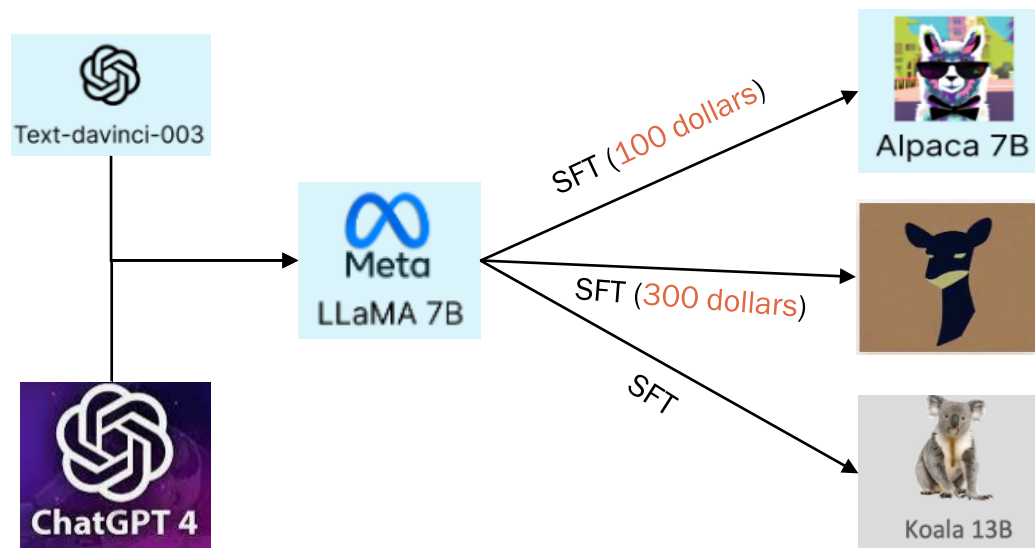
Instruction Following Models



Base Model	Instruction Models	Finetuning ?	Dataset (Task)
T5-XXL	FLAN T5-XXL	SFT	FLAN – 1836 tasks, 15M pairs
Falcon-40B	Falcon-40B-Instruct	SFT	Baize – 100K pairs
Llama2-70B	Llama2-70B-Chat	RLHF	27.5K pairs, 2.9M pairs
GPT-3.5	Chat-GPT	RLHF	-----
PALM	Flan-PALM	SFT	FLAN – 1836 tasks, 15M pairs
Pythia	Dolly	SFT	Dolly – 15K pairs
Llama-13B	Alpaca-13B	SFT	GPT3 – 52K pairs
Llama-13B	Vicuna-13B	SFT	GPT-4 – 70K pairs
Llama-13B	Koala-13B	SFT	Public Dialogues - 500K pairs

Imitating Models

Imitating Models



The False Promise of Imitating Proprietary LLMs



[Alpaca reference](#)
[Vicuna reference](#)
[Koala reference](#)
[Dolly reference](#)

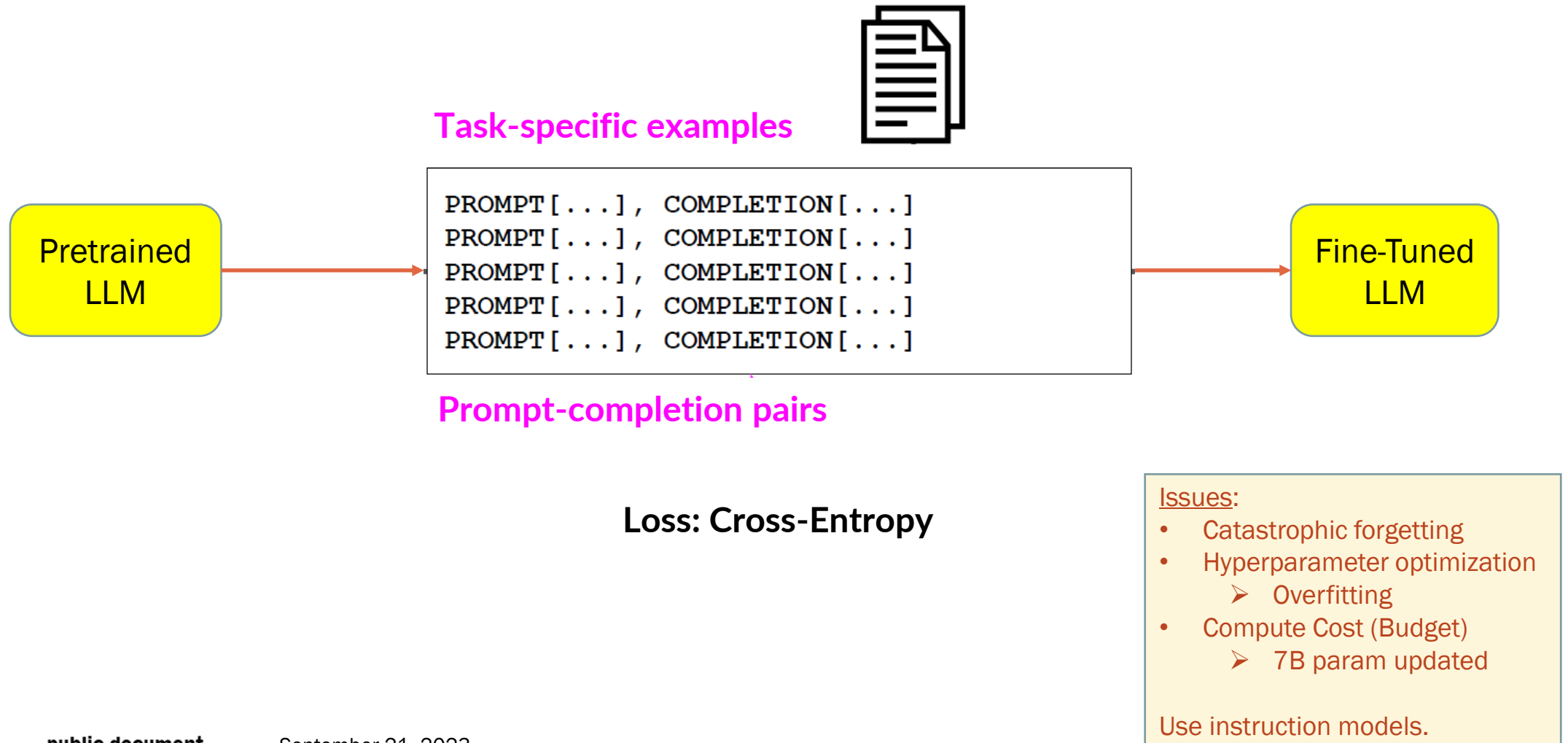


Finetuning for Single Use-case

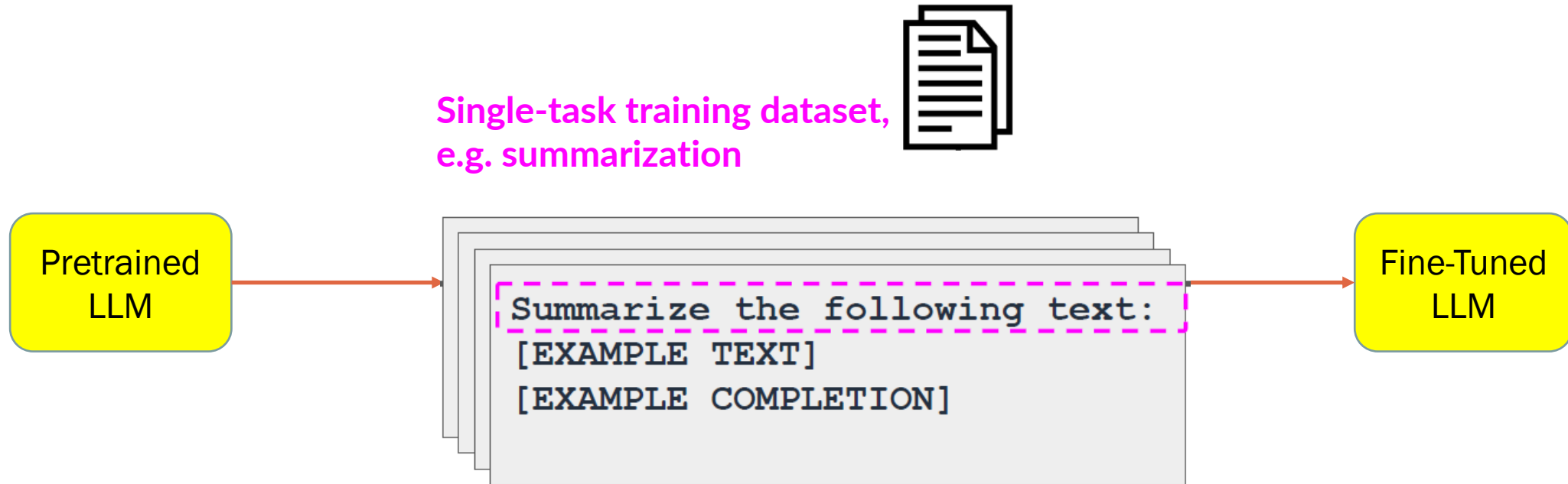
Why ? Need ?

1. Fine-tuning can significantly increase the performance of a model on a specific task.
 - but can lead to reduction in ability on other tasks. Phenomenon called catastrophic forgetting
Who cares ? use-case solved. (Generalist vs Specialist)
2. Examples take up space in the context window.
 - Less space for new tokens generated.
 - Larger prompts leads to slow inferencing. Show Llama-2-70B-Chat model as example
3. In-context learning may not work for smaller models.

Full Finetuning aka SFT



Full Finetuning aka SFT



500-1000 examples needed to fine-tune a single task

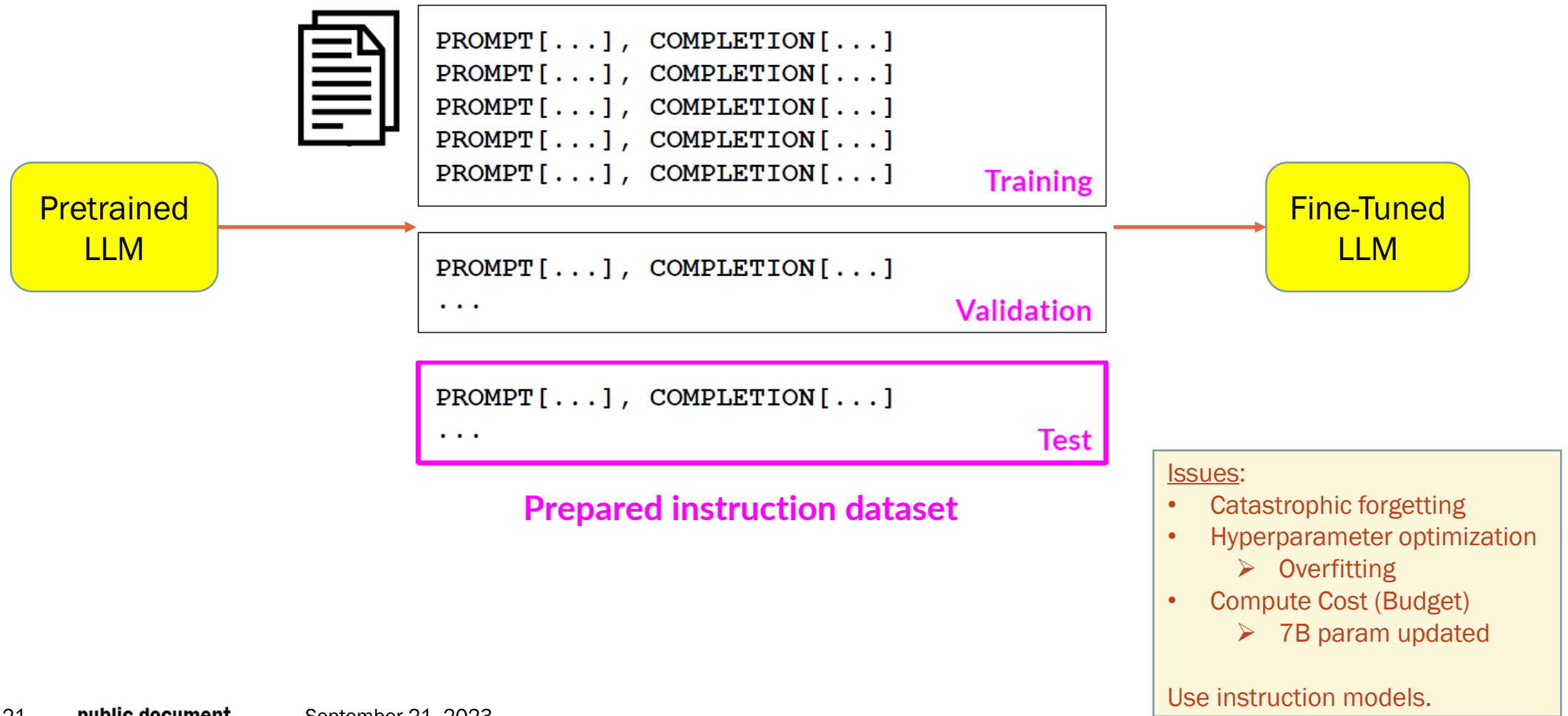
Loss: Cross-Entropy

Issues:

- Catastrophic forgetting
- Hyperparameter optimization
 - Overfitting
- Compute Cost (Budget)
 - 7B param updated

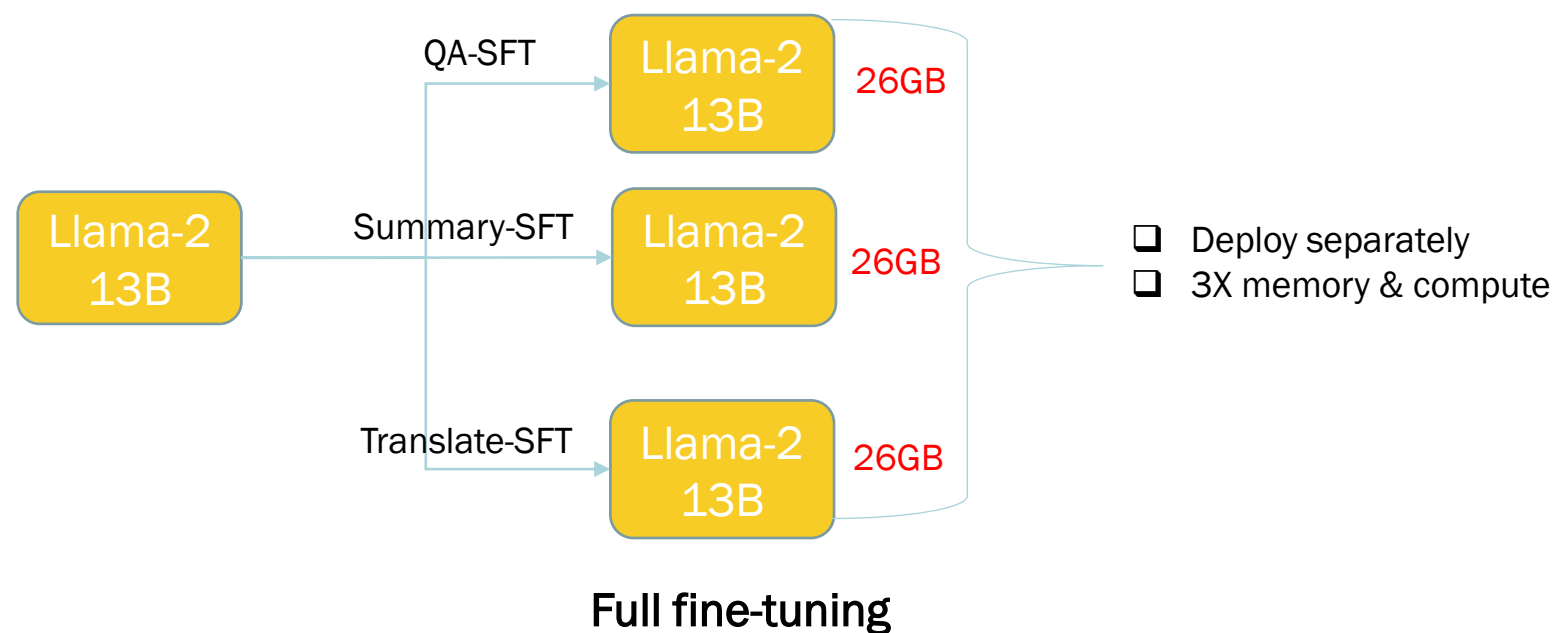
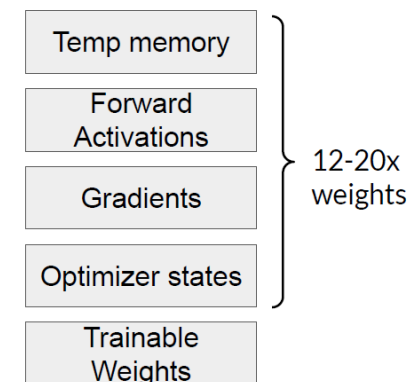
Use instruction models.

Full Finetuning aka SFT



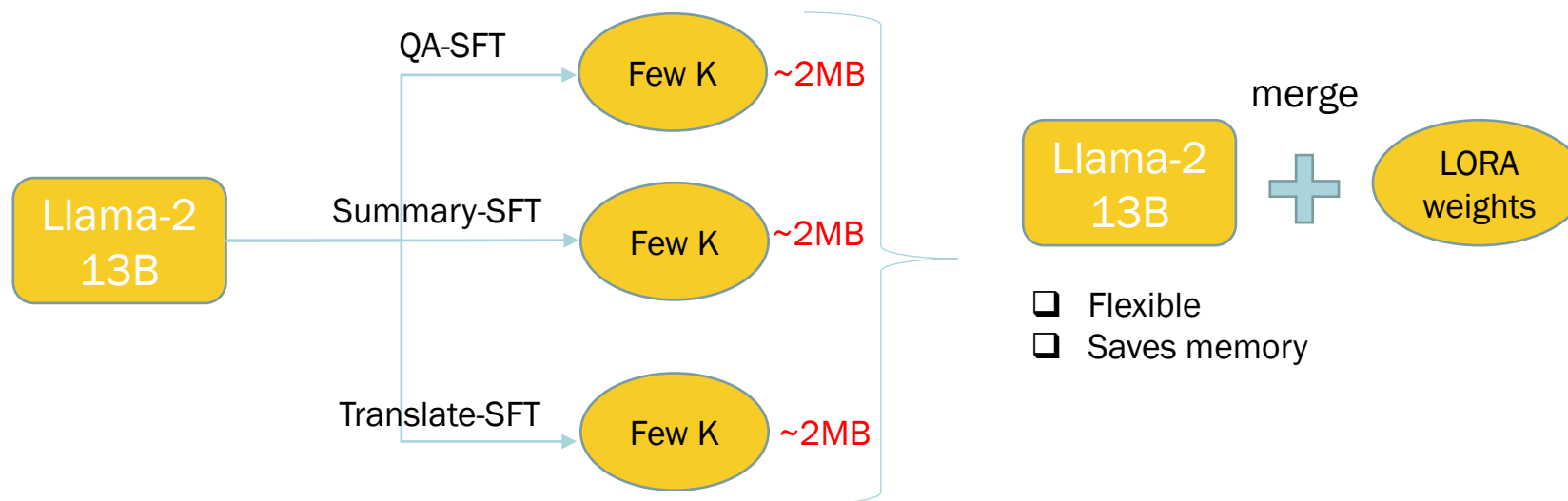
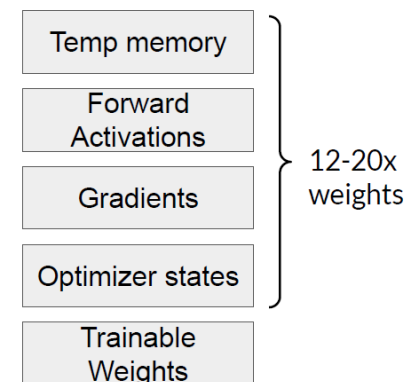
Parameter Efficient Finetuning (PEFT)

- Less prone to catastrophic forgetting – **Frozen Weights**
- Full fine-tuning of large LLMs is challenging – **Compute budget**
- Full fine-tuning creates full copy of original LLM per task - **Inefficient**
- PEFT fine-tuning saves space and is flexible



Parameter Efficient Finetuning (PEFT)

- Less prone to catastrophic forgetting – **Frozen Weights**
- Full fine-tuning of large LLMs is challenging – **Compute budget**
- Full fine-tuning creates full copy of original LLM per task - **Inefficient**
- PEFT fine-tuning saves space and is flexible



Parameter Efficient Finetuning

Low Rank Adaption (LORA)

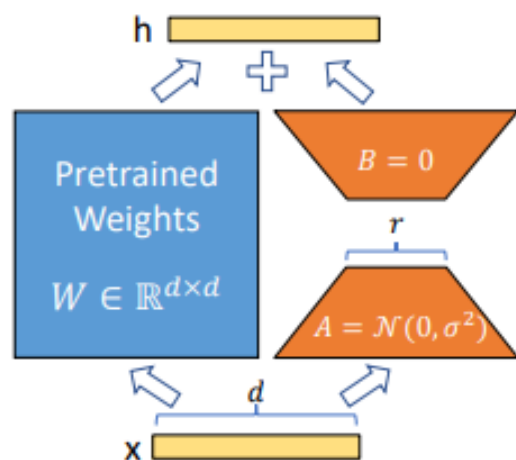


Figure 1: Our reparametrization. We only train A and B .

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

$$B * A = A \times B$$

2. Add to original weights

$$\text{Pretrained Weights} + A \times B$$

Advantage (Rank=8)

$512 \times 64 = 32,768$ params

$512 \times 8 = 4,096$ (A)

$8 \times 64 = 512$ (B)

86% reduction in params

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

Task A

$$\text{Input} * \text{Task A Matrix} = \text{Output}$$

The diagram shows a yellow input vector being multiplied by a vertical yellow bar (Task A matrix) to produce a yellow output vector. Below this, a box contains a blue snowflake icon (representing the frozen pretrained weights) plus a yellow box (representing the Task A matrix product).

Task B

$$\text{Input} * \text{Task B Matrix} = \text{Output}$$

The diagram shows a green input vector being multiplied by a vertical green bar (Task B matrix) to produce a green output vector. Below this, a box contains a blue snowflake icon (representing the frozen pretrained weights) plus a green box (representing the Task B matrix product).

Prompt Tuning

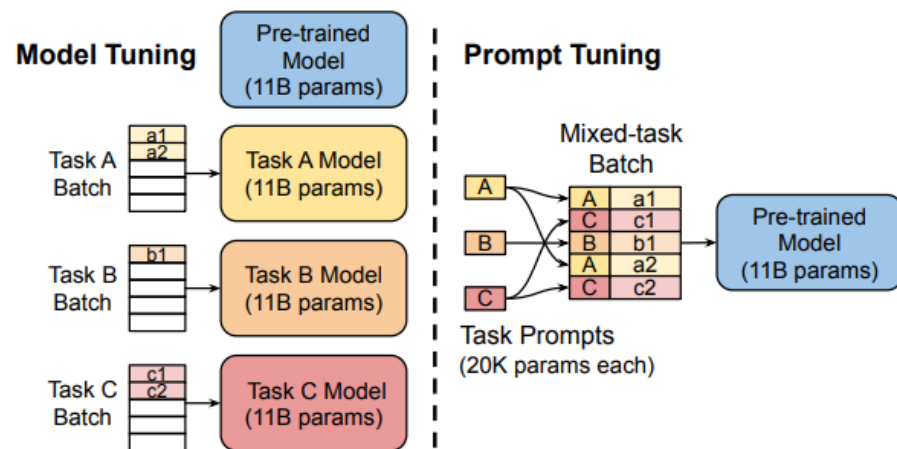
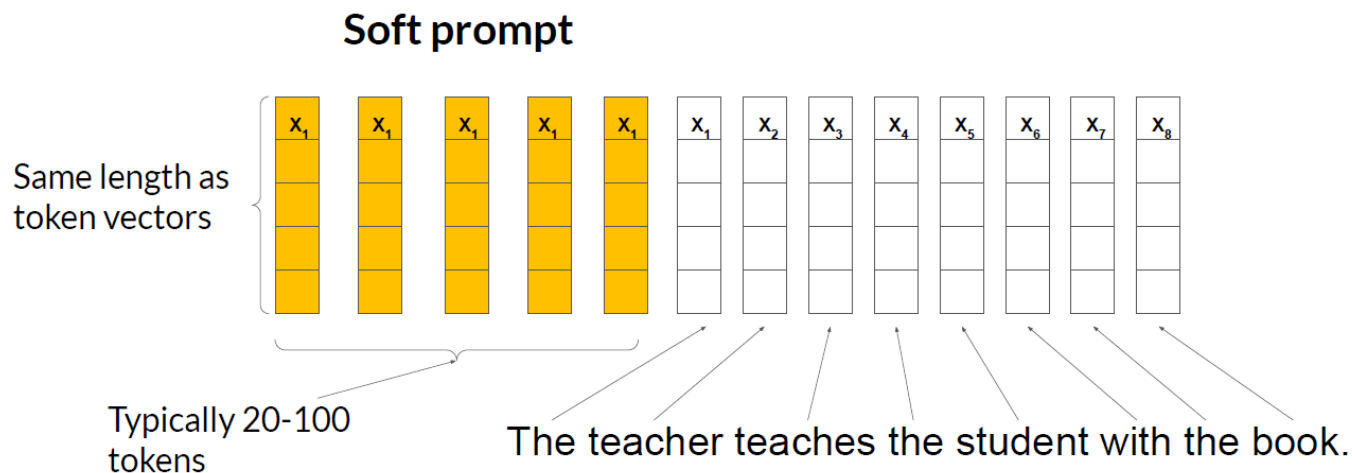


Figure 2: **Model tuning** requires making a task-specific copy of the entire pre-trained model for each downstream task and inference must be performed in separate batches. **Prompt tuning** only requires storing a small task-specific prompt for each task, and enables mixed-task inference using the original pre-trained model. With a T5 “XXL” model, each copy of the tuned model requires 11 billion parameters. By contrast, our tuned prompts would only require 20,480 parameters per task—a reduction of *over five orders of magnitude*—assuming a prompt length of 5 tokens.

Prompt Tuning

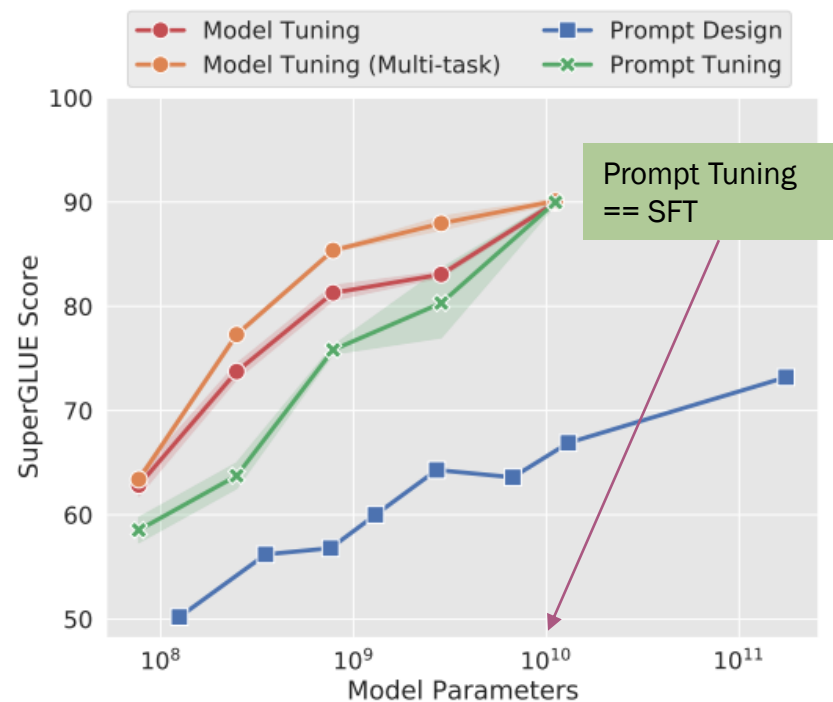
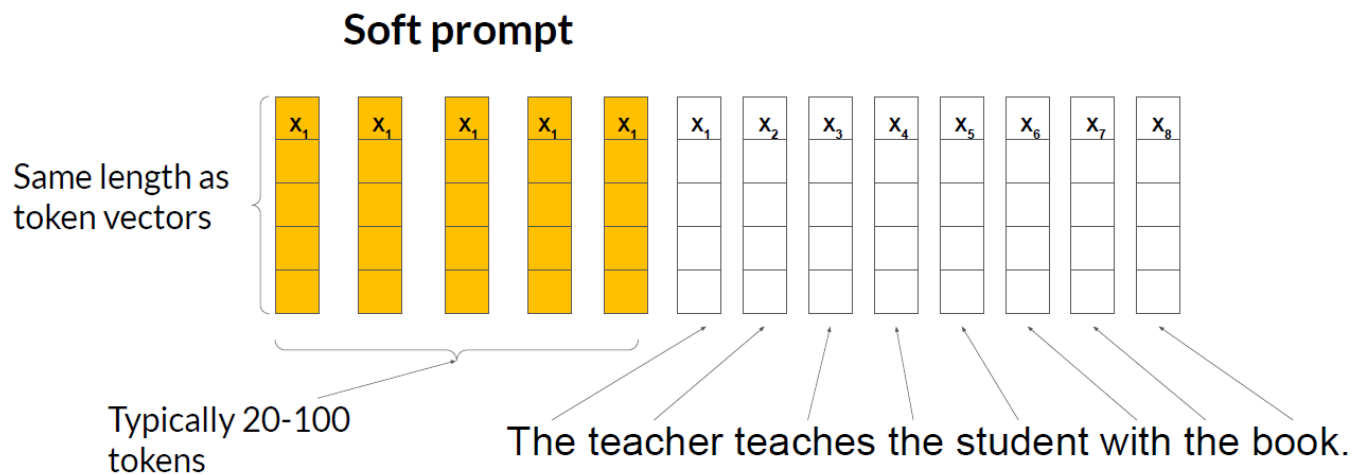


Figure 1: Standard **model tuning** of T5 achieves strong performance, but requires storing separate copies of the model for each end task. Our **prompt tuning** of T5 matches the quality of model tuning as size increases, while enabling the reuse of a single frozen model for all tasks. Our approach significantly outperforms few-shot **prompt design** using GPT-3. We show mean and standard deviation across 3 runs for tuning methods.

Guidelines: Finetuning for Use-case



Use instruct models for Finetuning for single task.

Example, Llama-chat. FLAN-T5 (not base models)

Use PEFT techniques, oppose to Full Finetuning

Example: Plug LORA adapters for specific use-case

Do not go for Full Finetuning. Finding right hyperparameter is difficult.

Unless you have the **compute budget** and **lot of training data**.

Quality of training data is very important

Research proves high quality datasets performs well even if quantity is less.



Thank you

