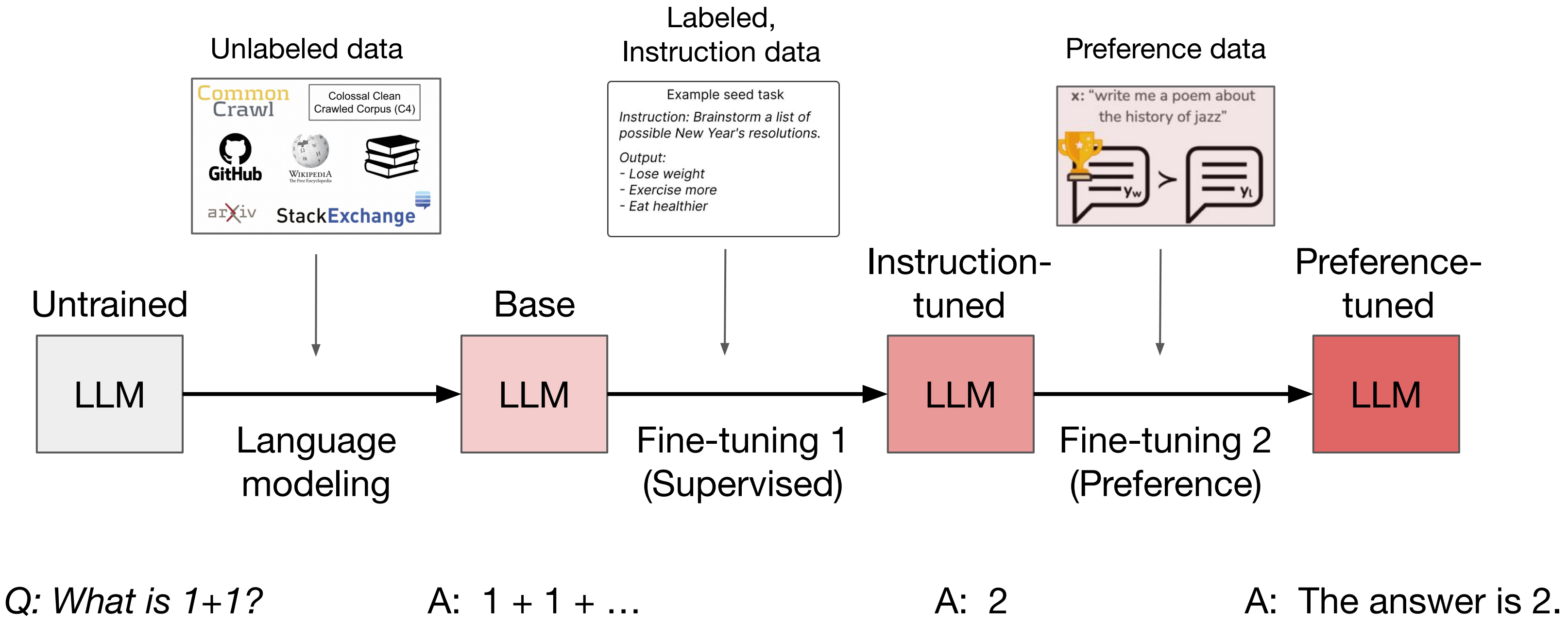
INFO-I590 Fundamentals and Applications of LLMs

Fine-Tuning and Instruction Tuning

Jisun An

Three steps of creating a high-quality LLM



Pre-training

- Self-supervised objective for language modeling
- Use as much data as you can find
- Biggest model you can afford
- Goal: a model that understands many linguistic properties
    - Grammar
    - World knowledge (e.g., “The president of the USA is ___”)
    - Emergent properties
- We are not focusing on a specific task or application

Fine-tuning

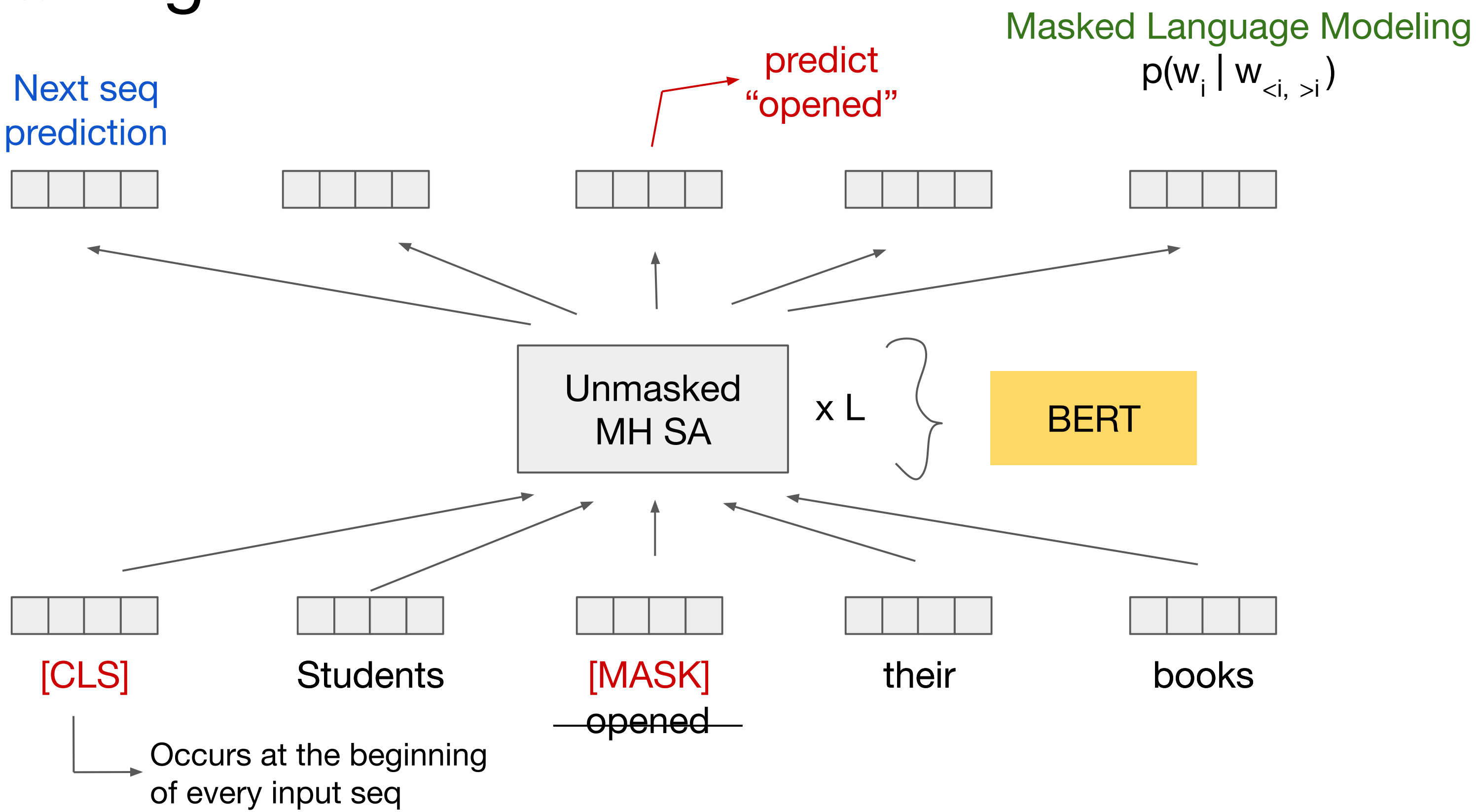
- Smaller labeled dataset corresponding to a single task/domain of interest
- Goal: maximize performance on this task/domain
- Parameter adaptation
  - Parameter-efficient adaptation

Fine-Tuning

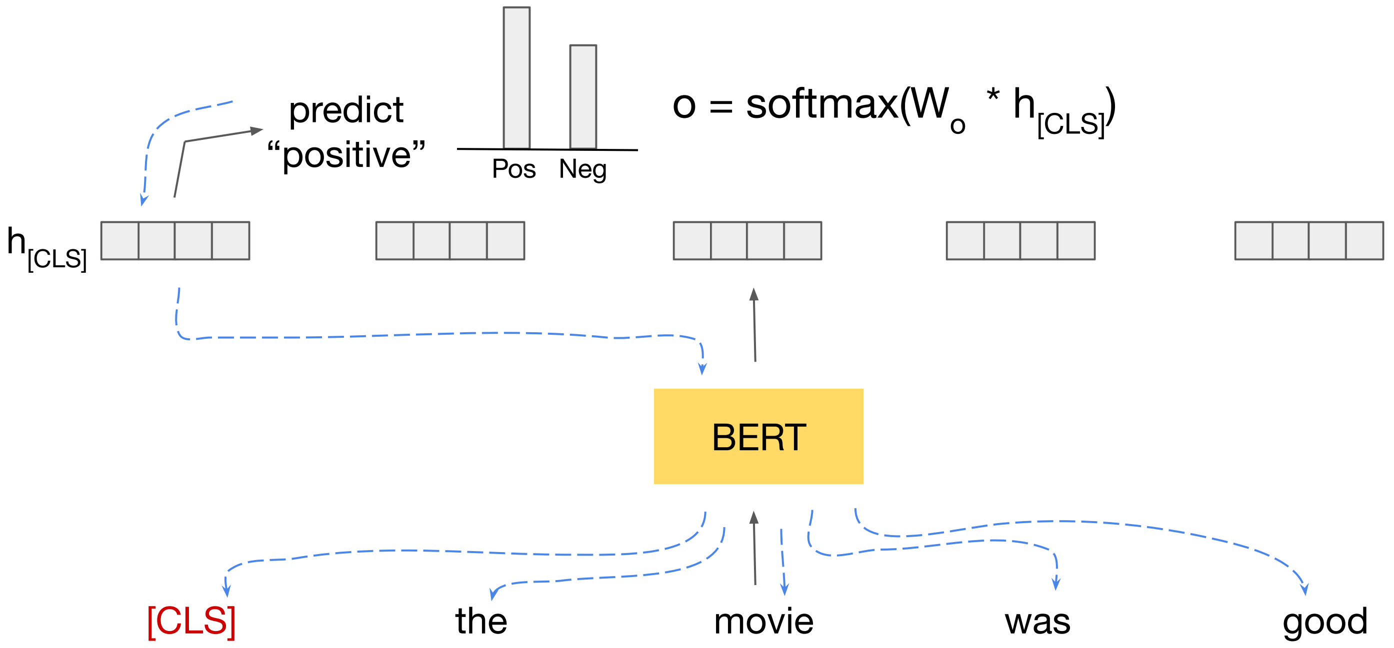
BERT

- Example of an encoder-only transformer
- Pre-training: training objective is self-supervised: “masked LM”
- Fine-tuning: process of adapting a pretrained model to a particular downstream task

Pre-training BERT

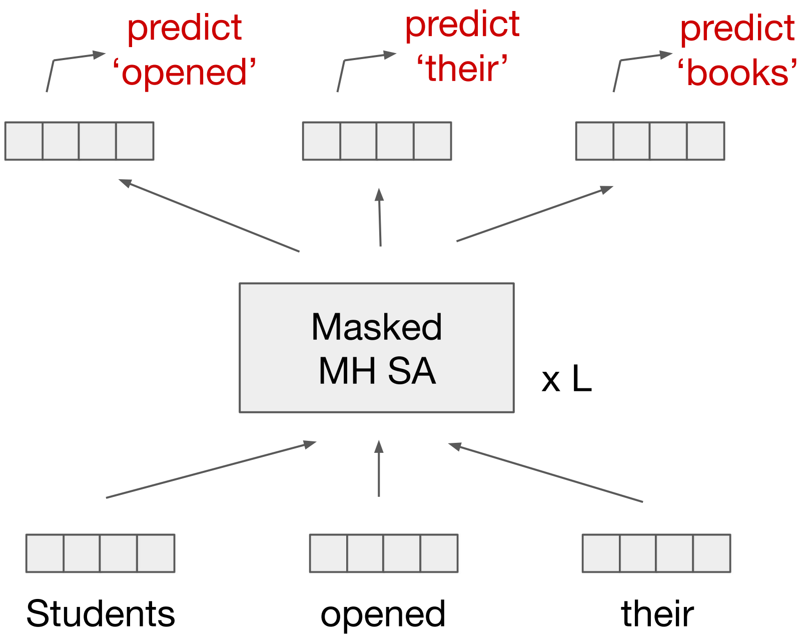


Fine-tuning: Sentiment analysis, Input → (Pos or Neg)



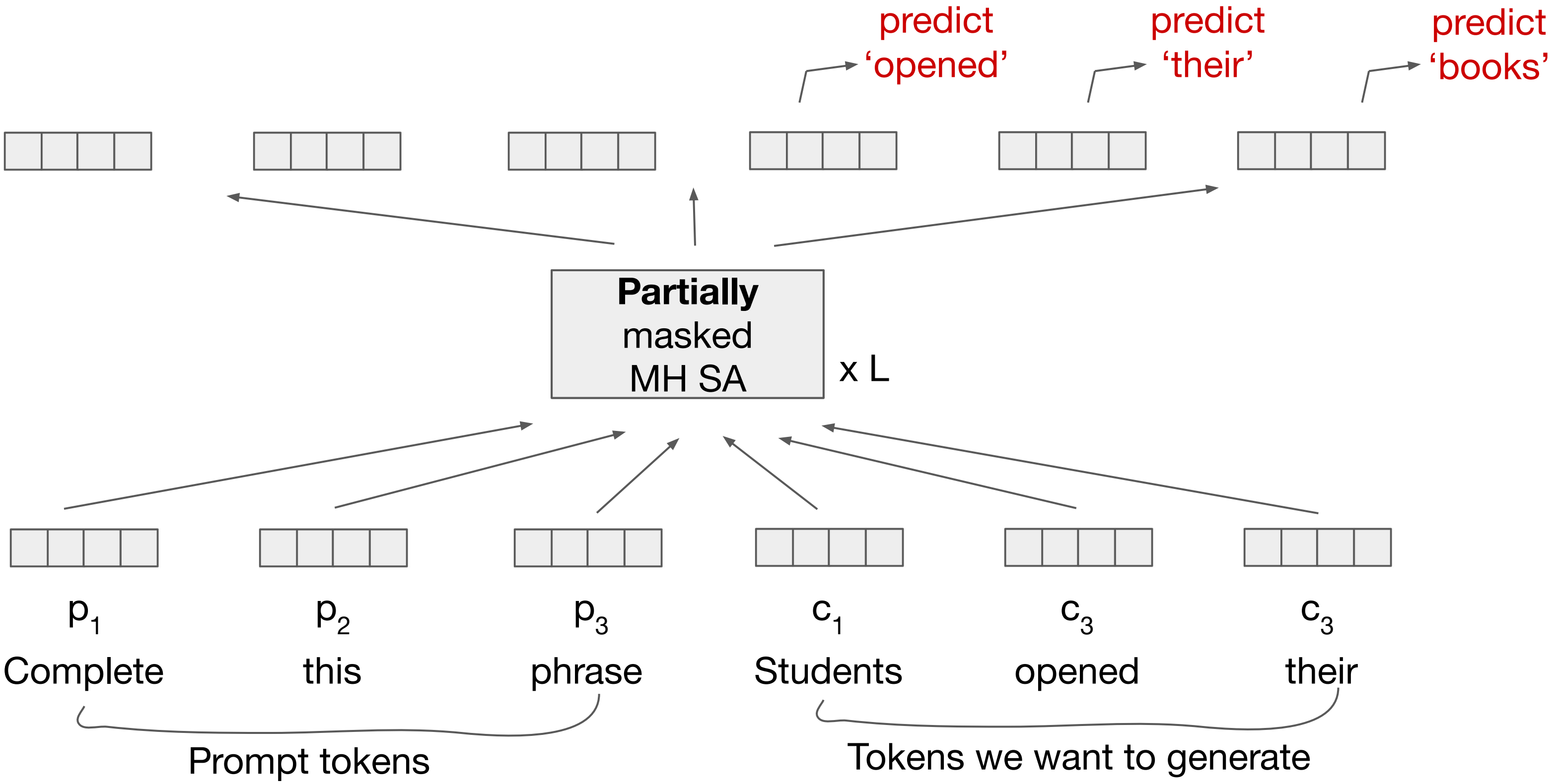
- Fine-tuning is NOT self-supervised. It generally requires a labeled training data for the downstream task. But, it uses far less data than pretraining.

Pre-training Decoder-Only Model



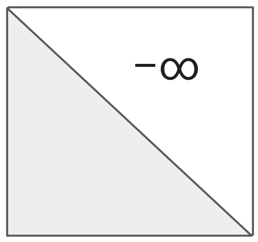
⇒ Useful for
text generation

Prefix LM

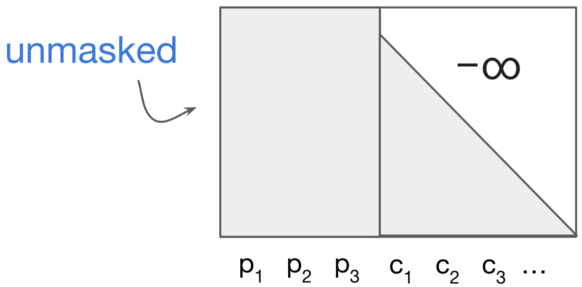

predict
'opened'
predict
'their'
predict
'books'
Partially
masked
MH SA
x L
p1
Complete
p2
this
p3
phrase
c1
Students
c3
opened
c3
their
Prompt tokens
Tokens we want to generate

Prefix LM

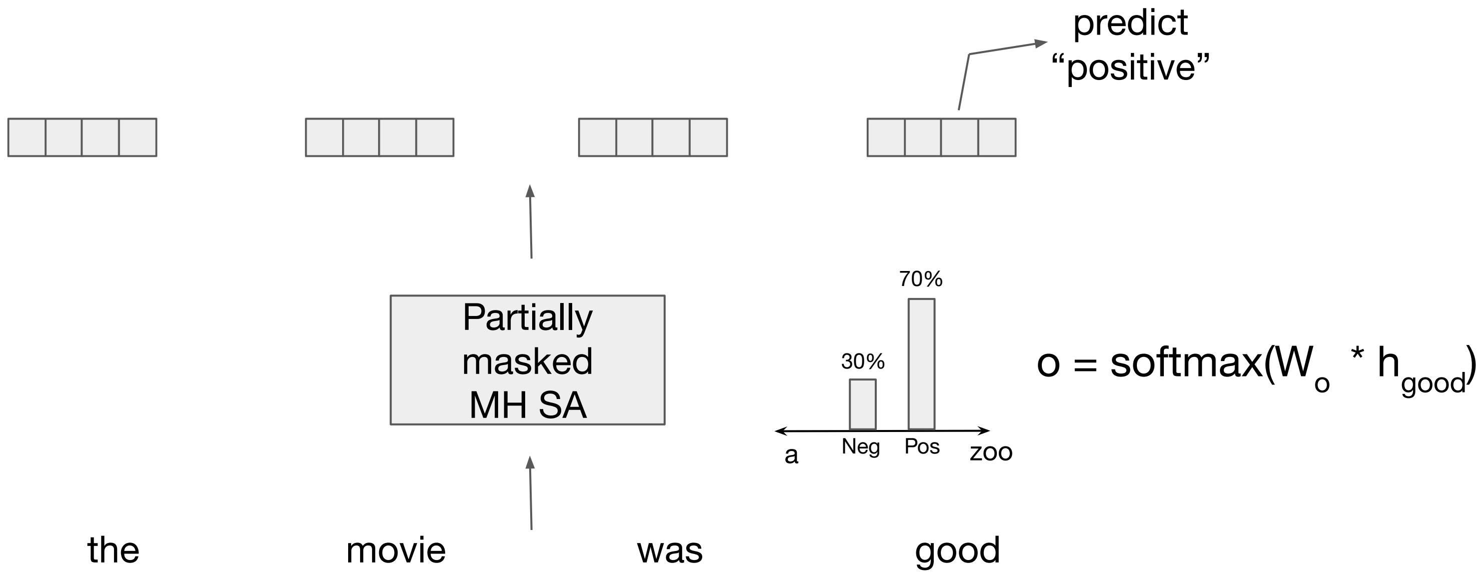
Decoder mask



Prefix LM mask

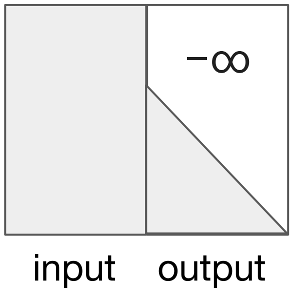


Fine-tuning a decoder-only LM for a classification



- Fine-tuning a pretrained decoder model is useful for text generation tasks.
- No new parameters.

Fine-tuning a decoder-only LM for text generation



$\underbrace{\text{The movie was good}}_{\text{input}}$ $\underbrace{\text{Positive because of “good”}}_{\text{output}}$

Instruction Tuning

Instruction-tuning (1)

- Fine-tuning (supervised fine-tuning (SFT))
- Goal: Make the pretrained model more capable of following instructions
- Method: standard fine-tuning on a special dataset

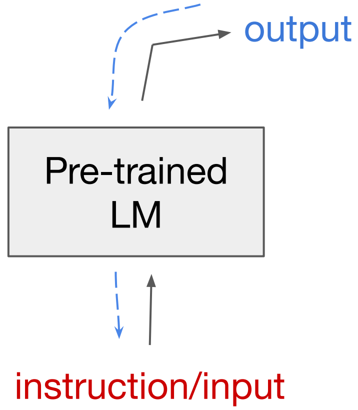
Instruction-tuning (2)

- Collect a dataset of instructions on what tasks to solve, and outputs of that task for a few examples
  - Sentiments analysis, summarization, questions and answers, etc

Instruction: Please answer the following question and provide a detailed justification.

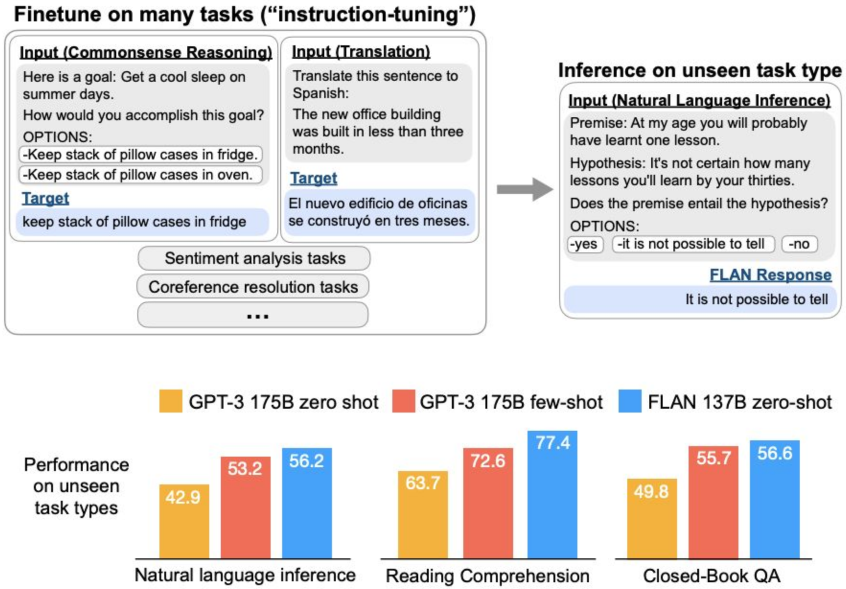
Input: What was the mobile number of Jisun?

Output: I can’t answer that because it is private information.



Instruction-tuning (3)

- Instruction tuning focuses on many different tasks at once, not just one
- Instruction tuning improves generalization on tasks outside of the fine-tuning data



The first instruction-tuned model: FLAN

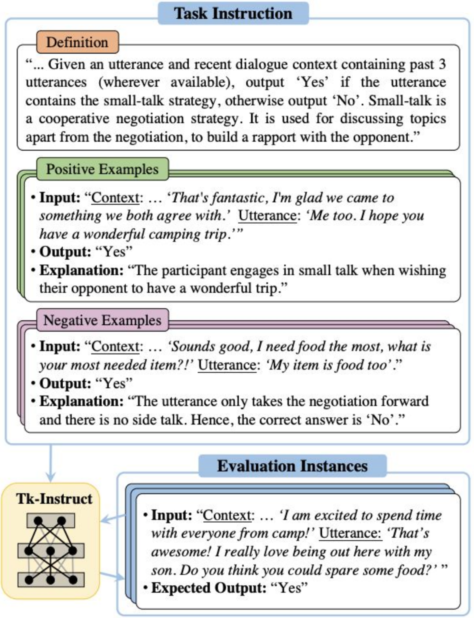
Wei et al. 2022. Finetuned Language Models Are Zero-Shot Learners. https://arxiv.org/abs/2109.01652

Instruction Tuned Models

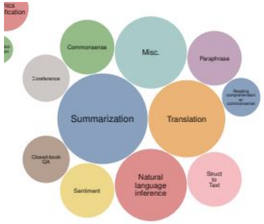
- **FLAN-T5**: huggingface/google/flan-t5-xxl
  - Encoder-decoder model based on T5
  - 11B parameters
- **LLaMa-2 Chat**: huggingface/meta-llama/Llama-2-70b-chat-hf
  - Decoder-only model
  - 70B parameters
- **Mixtral instruct**: huggingface/mistralai/Mixtral-8x7B-Instruct-v0.1
  - Decode-only mixture of experts model
  - 45B parameters
- *(smaller versions also available - Mistral, LLaMa2-7B)*

Natural Instructions

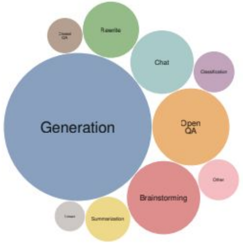
- 1,616 diverse NLP tasks and their expert-written instructions



(a) SUP-NATINST (this work)



(d) FLAN

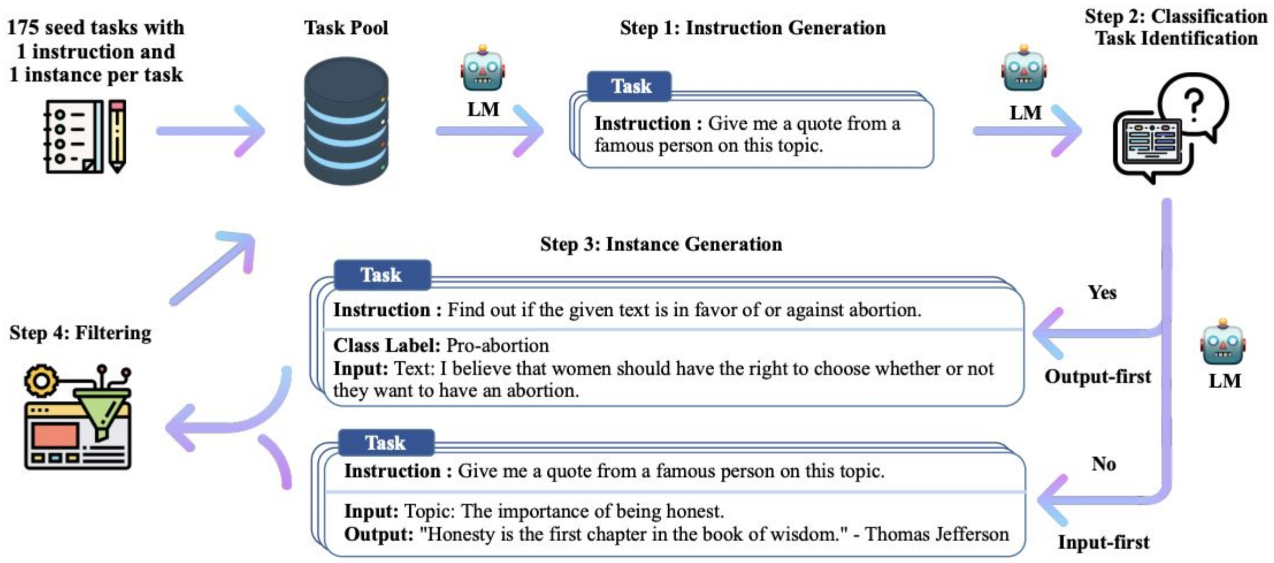


(e) INSTRUCTGPT

Wang et al. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. https://arxiv.org/abs/2204.07705

Self-Instruct

- It is possible to automatically generate instruction tuning datasets, e.g. self-instruct (Wang et al. 2022)



Wang et al. 2022. Self-instruct: Aligning language models with self-generated instructions. https://arxiv.org/abs/2212.10560

LIMA: Less is More

- LIMA: a 65B parameter LLaMa fine-tuned on only 1,000 carefully curated prompts and responses, without any reinforcement learning or human preference modeling.
- Only limited instruction-tuning data is necessary to teach models to produce high quality output.

Source	#Examples	Avg Input Len.	Avg Output Len.
Training			
Stack Exchange (STEM)	200	117	523
Stack Exchange (Other)	200	119	530
wikiHow	200	12	1,811
Pushshift r/WritingPrompts	150	34	274
Natural Instructions	50	236	92
Paper Authors (Group A)	200	40	334
Dev			
Paper Authors (Group A)	50	36	N/A
Test			
Pushshift r/AskReddit	70	30	N/A
Paper Authors (Group B)	230	31	N/A

Table 1: Sources of training prompts (inputs) and responses (outputs), and test prompts. The total amount of training data is roughly 750,000 tokens, split over exactly 1,000 sequences.

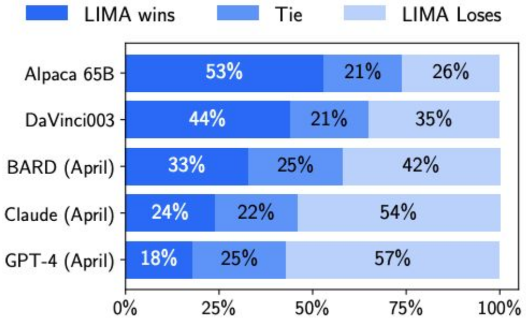
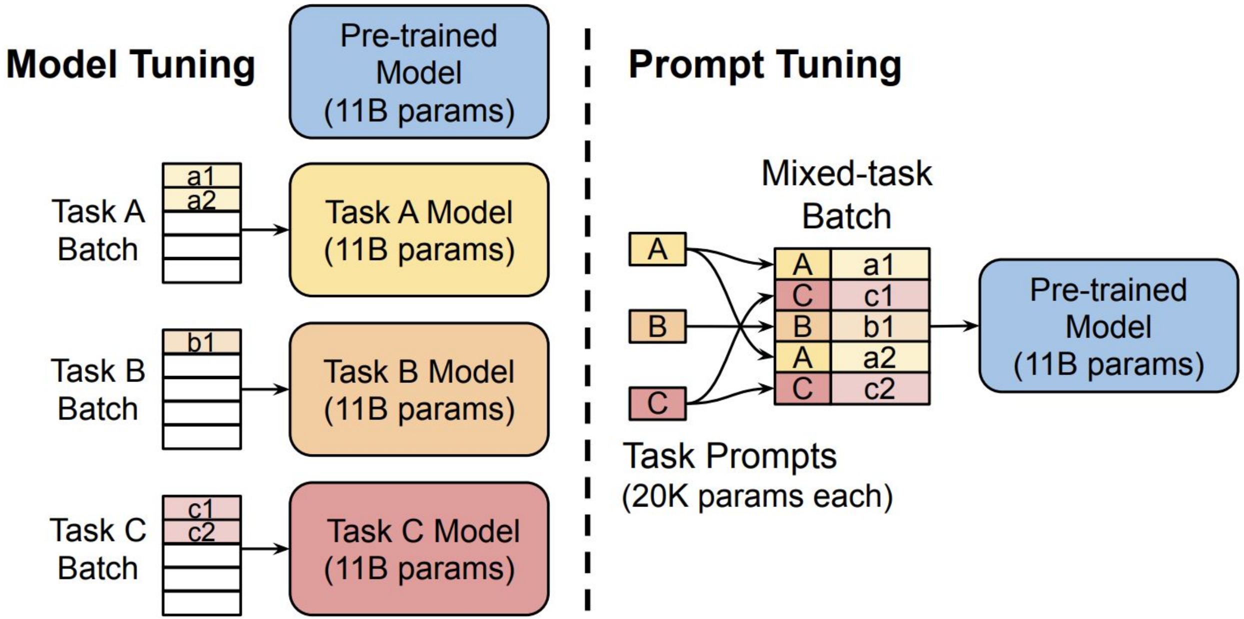


Figure 1: Human preference evaluation, comparing LIMA to 5 different baselines across 300 test prompts.

Zhou et al. 2023. LIMA: Less Is More for Alignment. https://arxiv.org/abs/2305.11206

Parameter-Efficient Fine-Tuning (PEFT)

Why do we need parameter-efficient fine-tuning strategies?



Lester et al. 2022. The Power of Scale for Parameter-Efficient Prompt Tuning. https://arxiv.org/abs/2104.08691

Parameter-efficient fine-tuning (PEFT)

- High-level idea: Don’t tune all of the parameters, but just some!
  - we want to avoid modifying most of the pretrained model’s parameters during fine-tuning
- PEFT strategies
  - Prompting (requires adjusting *zero* parameters)
  - Prompt tuning
  - LoRA (Low-Rank Adaptation)

Prompting

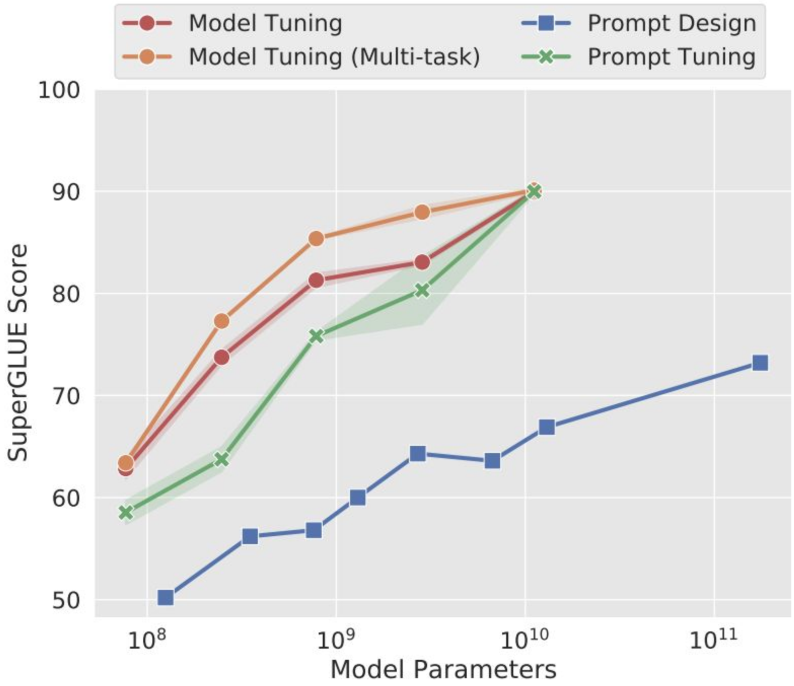
- Requires adjusting zero parameters to solve a downstream task

What is the sentiment of the below sentence? Answer with either
"positive" or "negative"
<input sentence>
Output: positive

Prompt engineering

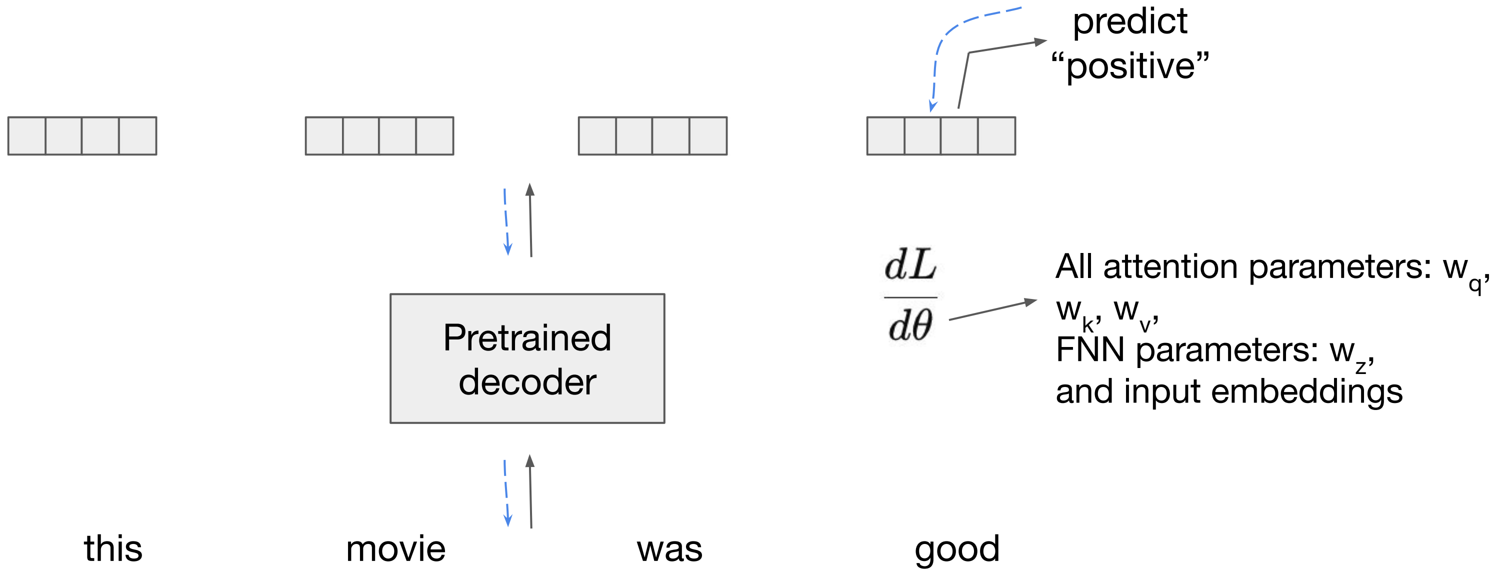
- Limitations of prompting
  - Hard to solve very complex reasoning/understanding tasks
  - Requirements for the pretrained model are immense
    - Huge-scale pretraining
    - High quality large scale instruction turning
    - RLHF, requires access to very expensive human preference datasets

Prompting vs Fine-tuning

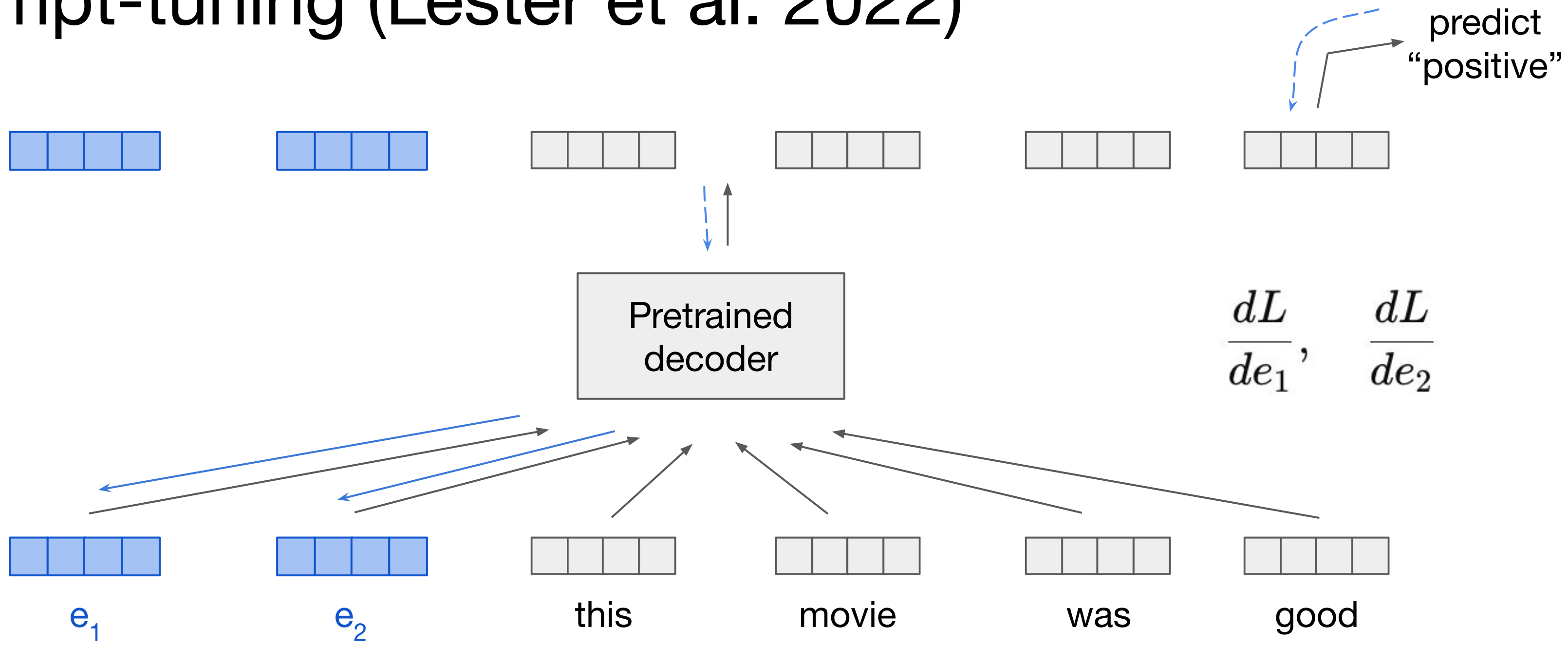


Lester et al. 2022. The Power of Scale for Parameter-Efficient Prompt Tuning. https://arxiv.org/abs/2104.08691

Review of full model fine-tuning

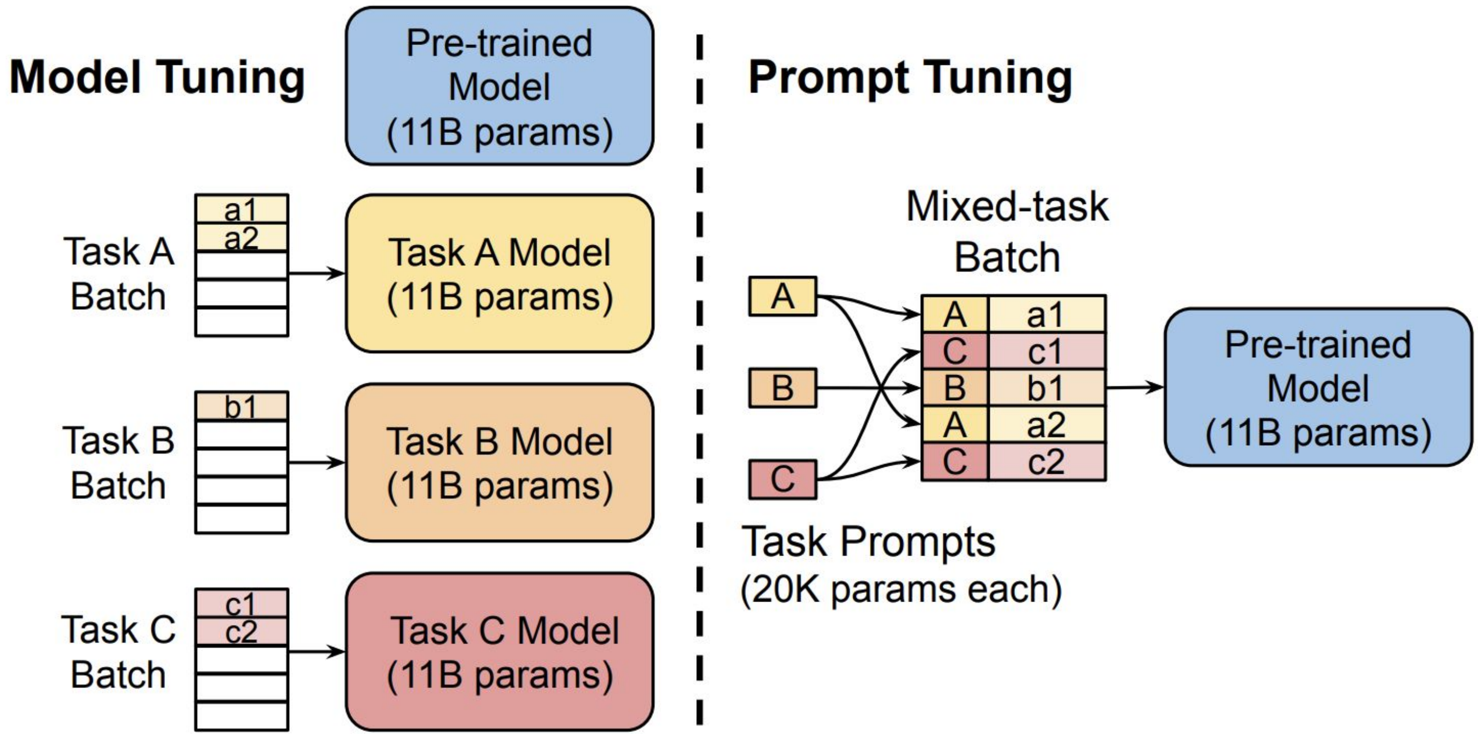


Prompt-tuning (Lester et al. 2022)



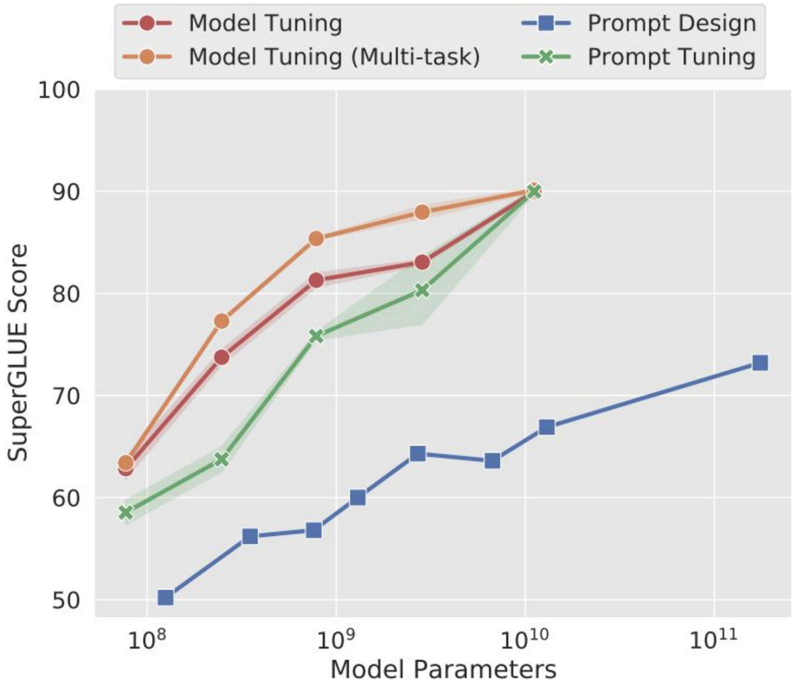
Update: keep all pretrained parameters frozen, only do: $e_1^{\text{new}} = e_1^{\text{old}} - \eta \frac{dL}{de_1}, \; e_2^{\text{new}} = e_2^{\text{old}} - \eta \frac{dL}{de_2}$

Lester et al. 2022. The Power of Scale for Parameter-Efficient Prompt Tuning. https://arxiv.org/abs/2104.08691



Lester et al. 2022. The Power of Scale for Parameter-Efficient Prompt Tuning. https://arxiv.org/abs/2104.08691

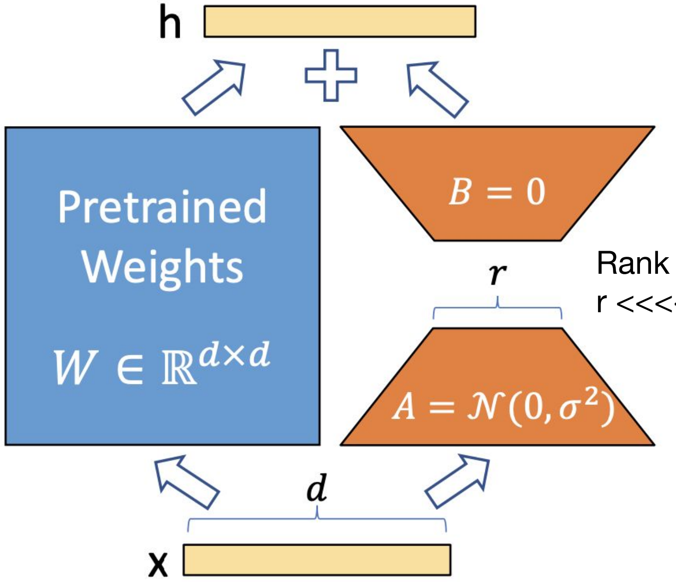
Prompt Tuning vs Model Tuning



Lester et al. 2022. The Power of Scale for Parameter-Efficient Prompt Tuning. https://arxiv.org/abs/2104.08691

LoRA - Low-Rank Adaptation (Hu et al. 2021)

- Freeze pre-trained weights, train low-rank approximation of difference from pre-trained weights
- Advantage: after training, just add in to pre-trained weights – no new components!



Rank parameter,
r <<<<<<< d

Hu et al. 2021. LoRA: Low-Rank Adaptation of Large Language Models. https://arxiv.org/abs/2106.09685

Low-Rank Matrix

- A low-rank matrix is a matrix whose rank (i.e., the number of linearly independent rows or columns) is much smaller than its full dimension.

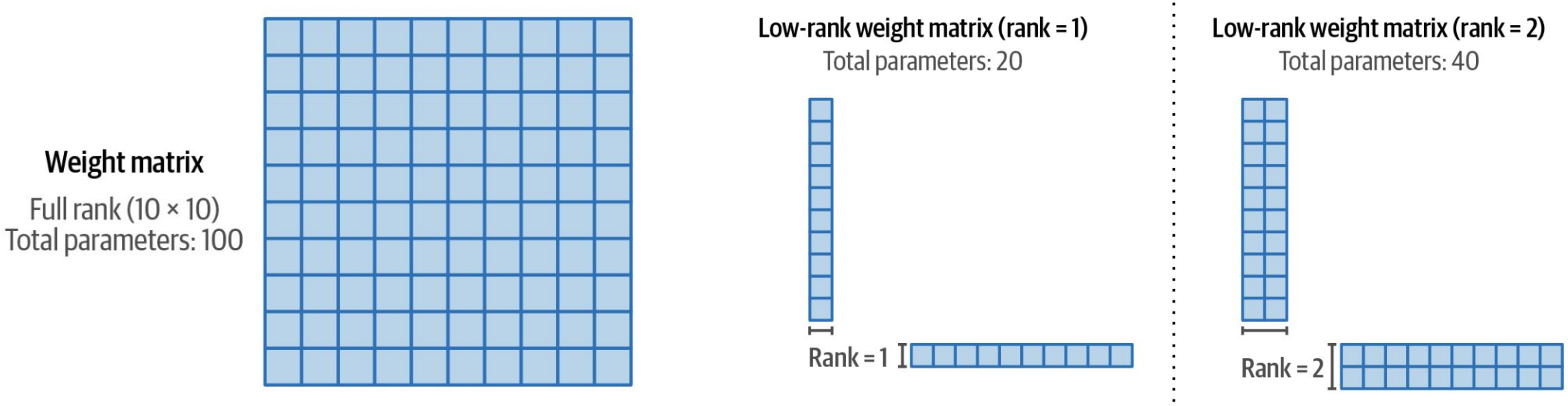
A matrix $A \in \mathbb{R}^{m \times n}$ is **low-rank** if there exists a decomposition:

$$A = UV^T$$

where:

- $U \in \mathbb{R}^{m \times r}$,
- $V \in \mathbb{R}^{n \times r}$,
- $r \ll \min(m, n)$, meaning that the rank is significantly smaller than the matrix's full size.

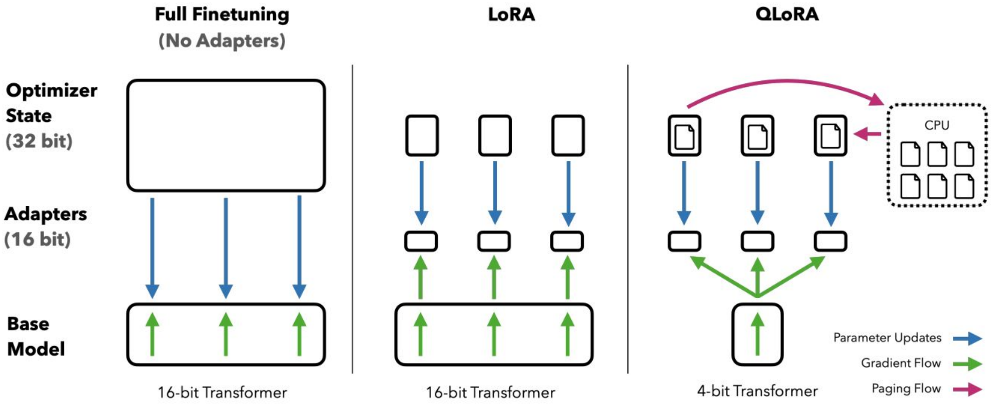
Low-Rank Matrix Example

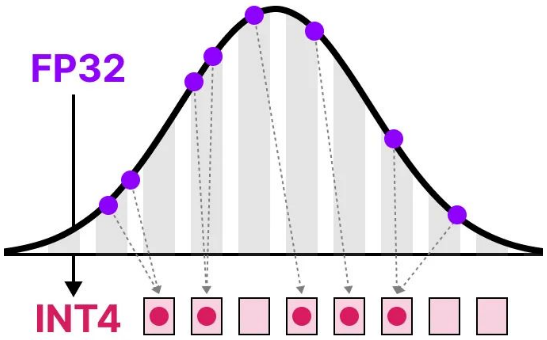


Img: Hands on Large Language Models

Q-LoRA (Dettmers et al. 2023)

- Further compress memory requirements for training by
  - **4-bit quantization** of the model
  - Use of CPU memory paging to prevent OOM
  - Can train a 65B model on a 48GB GPU!





Dettmers et al. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. https://arxiv.org/abs/2305.14314
Img: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-quantization

Any Questions?