LoRA Fine-Tuning

Dr. Karthik Mohan, March 5th 2025

Today's Talk

LoRA Fine-Tuning

Llama3 Herd

Herd of models including 405B LM, 70B, 8B, 1B versions and also Llama Guard 3 for input/output safety

Reference: https://arxiv.org/pdf/2407.21783

Llama 3 Herd of Models

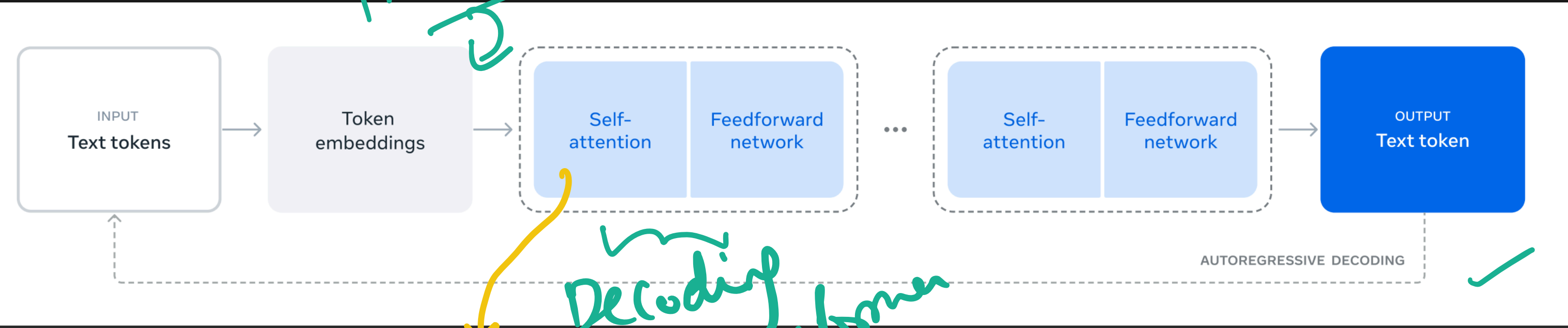
Instruct
Fine-Tuning

	Finetuned	Multilingual	Long context	Tool use	Release
Llama 3 8B	✗	✗[1]	✗	✗	April 2024
Llama 3 8B Instruct	✓	✗	✗	✗	April 2024
Llama 3 70B	✗	✗[1]	✗	✗	April 2024
Llama 3 70B Instruct	✓	✗	✗	✗	April 2024
Llama 3.1 8B	✗	✓	✓	✗	July 2024
Llama 3.1 8B Instruct	✓	✓	✓	✓	July 2024
Llama 3.1 70B	✗	✓	✓	✗	July 2024
Llama 3.1 70B Instruct	✓	✓	✓	✓	July 2024
Llama 3.1 405B	✗	✓	✓	✗	July 2024
Llama 3.1 405B Instruct	✓	✓	✓	✓	July 2024

Reference: https://arxiv.org/pdf/2407.21783

Llama3 Architecture

RoPE Embeddings



Decoding Transformer Block

W^q

$rank(W^q) = small$

LoRA Fine-Tuning

Low-Rank Adaptation of LLMs

Refers to an efficient fine-tuning procedure - where ALL weights of the LLM are frozen. But - New and relatively fewer weights are introduced for fine-tuning.

Reference: https://arxiv.org/pdf/2106.09685

Llama 3-8b Instruct

↓ LoRA fine-tune

Llama3-8b Instruct LoRA fine-tuned model

LoRA Fine-Tuning

Low-Rank Adaptation of LLMs

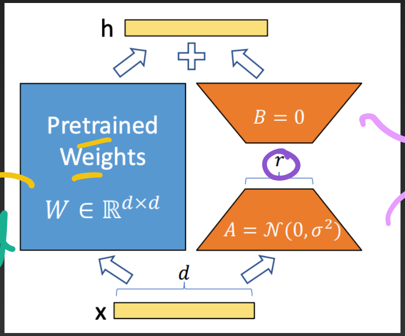
Refers to an efficient fine-tuning procedure - where ALL weights of the LLM are frozen. But - New and relatively fewer weights are introduced for fine-tuning.

Example

$q = W^q x_t$

W^q ← Instruct-tuned

Frozen



A & B are learned or fine-tuned

frozen

fine-tuned

$\tilde{W}^q = W^q + \underbrace{A^q B^q}_{\text{LoRA addition}}$

$\tilde{q}_t = \tilde{W}^q x_t$

Reference: https://arxiv.org/pdf/2106.09685

ICE #1

$W = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \rightarrow \text{rank} = 2$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \times 2 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

$\underline{W}_{d \times d} = \underline{A}^{d \times k} \; \underline{B}_{k \times d}$

Low-Rank Matrices

$W = AB$

Let W (dxd) be a matrix that can be thought of as a product of A (dxk) and B (kxd). What is the rank of the matrix W?

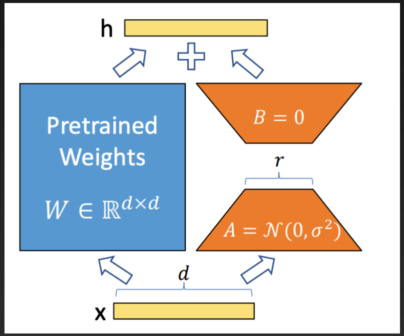
$k < d$

a) At least d

b) At most d

c) At least k

d) At most k ✓



$W = \begin{bmatrix} 1 & 4 \\ 3 & 12 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 \\ 3 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} 1 & 4 \end{bmatrix}}_{B}$

$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \rightarrow \text{rank} = 1$

$= \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} 1 & 1 \end{bmatrix}}_{B}$

$W = \begin{bmatrix} 1 & 4 \\ 3 & 12 \end{bmatrix} \rightarrow \text{rank} = 1$

$\begin{bmatrix} 4 \\ 12 \end{bmatrix} = 4 \begin{bmatrix} 1 \\ 3 \end{bmatrix}$

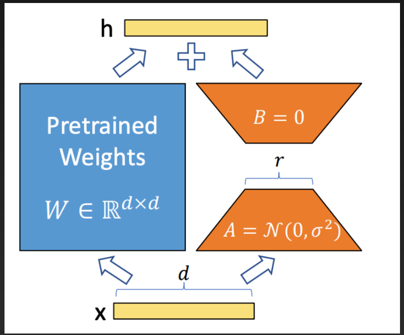
Reference: https://arxiv.org/pdf/2106.09685

LoRA Fine-Tuning Basis

Low-Rank Adaptation of LLMs

Based on the assumption that learned weight matrices in LLMs typically reside in "low-dimensional" subspaces. Thus learning a low-rank matrix can be a way to fine-tune.

Low-rank



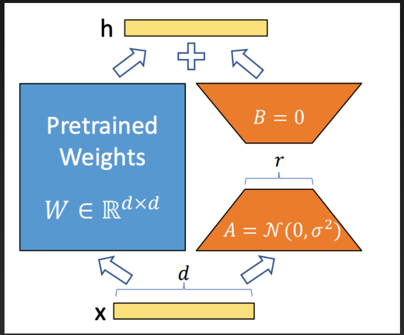
Reference: https://arxiv.org/pdf/2106.09685

ICE #2

LoRA application

In the Llama3 Transformer - What weight matrices does LoRA duplicate for fine-tuning?

a) Query matrix
b) Key Matrix
c) Value Matrix
d) MLP matrices
e) All of the above ✓

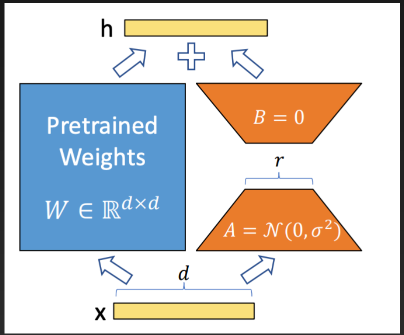


Reference: https://arxiv.org/pdf/2106.09685

LoRA Fine-Tuning Features

Low-Rank Adaptation of LLMs

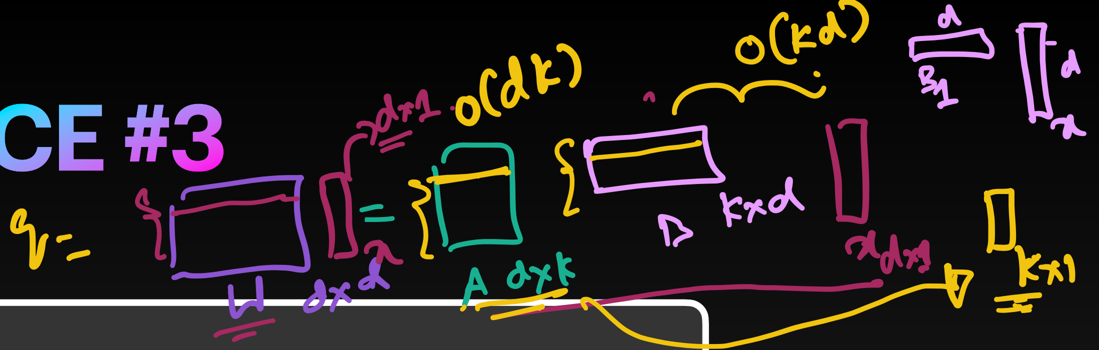
* Can be used to fine-tune “any” LLM model by freezing entire model
* Only the new low-rank weights are fine-tuned
* Final model is the existing weights + the LoRA adaptor weights
* Latency is same at inference time - As the new weights get added in
* Orthogonal to partial freezing and fine-tuning paradigm
* For GPT-3 175B - reduced RAM requirement from 1.2TB to 350GB.
* With r = 4, reduced the checkpoint size of the fine-tuned model reduced from 350GB to 35MB!



Reference: https://arxiv.org/pdf/2106.09685

$O(Wx) = O(d^2)$

ICE #3

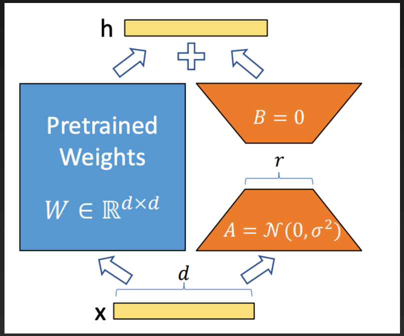


Low-Rank Matrices

Let W (dxd) be a matrix that can be thought of as a product of A (dxk) and B (kxd). Let's say we have a token embedding, x of a token T, that lives in d dimensions. If W represents the query matrix - What is the computational complexity of computing the query vector q from the token T?

a) O(d*d)
b) O(d*d*k)
c) O(k*k)
d) O(d*k)

$(O(dk) + O(kd))$
$= O(dk)$



Reference: https://arxiv.org/pdf/2106.09685

ICE #4

W 4000x4000

A 4000x5

5x4000

4000x10

Factor = 400

LoRA fine-tuning

t → [ ]$_{4000 \times 1}$

Let the embedding dimension, d be 4000. Assume that we do LoRA fine-tuning with a LoRA rank, k of 5. If the LLM model we are fine-tuning is a 8b Llama 3 model. How many new parameters are we introducing with the LoRA fine-tuning?

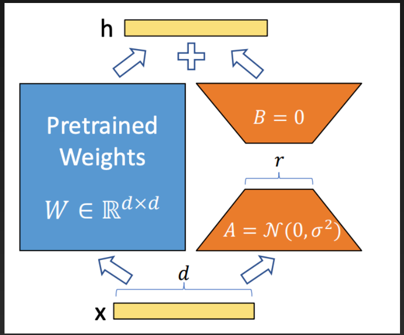
a) 10 MM

b) 20 MM ← 8b/400 ≈ 20MM

c) 30 MM

d) 40 MM

Factor of Reduction param



Reference: https://arxiv.org/pdf/2106.09685

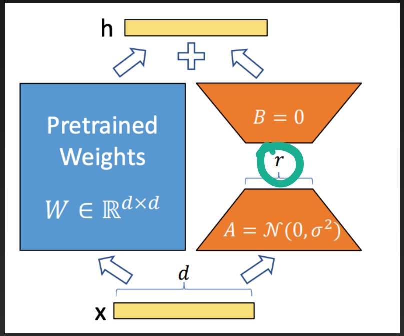
LoRA Fine-Tuning vs Partial Freezing

d = 4000

r = 10, 100, 500, 1000

LoRA vs Partial Freezing

* Computer vision models, for example get fine-tuned by freezing all but last 3 layers of CNN model on the fine-tuned data set
* In the context of LLMs - What are the pros/cons of LoRA as compared to the partial freezing of weights approach for fine-tuning?



Reference: https://arxiv.org/pdf/2106.09685

ICE #5

For Recommender System Problem

Low Rank Matrix Factorization

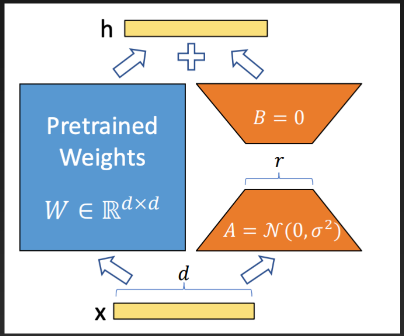
Recall the context of low-rank factorization of data matrix into users factors and movies factors. Let X = UV be this factorization. Where (i,j) element of X represents whether user i liked movie j or not (1 for like and 0 for not). In this case if we have millions of users and 100k movies - X is a large matrix. But typically the column dimension of U is limited to 50 or 100. Why would 100 dimensions be sufficient?

a) X It's a low rank factorization - so 100 should be sufficient

b) Its computationally expensive to consider 1000 dimensions or more

c) The user factors and movie factors have a common theme of genres and there are not too many genre combos

d) It works experimentally and hence 100 is sufficient



Reference: https://arxiv.org/pdf/2106.09685

MP 3 → Part 1 & Kaggle

↓

LoRA Fine-Tuning

- Enough RAM ?