12-1-2024

Fine Turning LLm models

① Quantization - is a common technique used to reduce the model size, through it can sometimes result in reduced accuracy.

② Quantization-aware training is a method that allows practitioners to apply quantization techniques without sacrificing accuracy.

③ It is done in model training process rather than after the fact. The model size can typically be reduced by two to four times & sometimes even more.

④ modes of Quantization → ① Post training quantization.
② Quantization aware training

definition - Quantization: Conversion from higher memory format to a lower memory format.

Convert
32 bits ⇒ int 8 inference
Full precision ⇒ half precession. & loss of data weight & parameter & accuracy

Calibration - model Quantization.

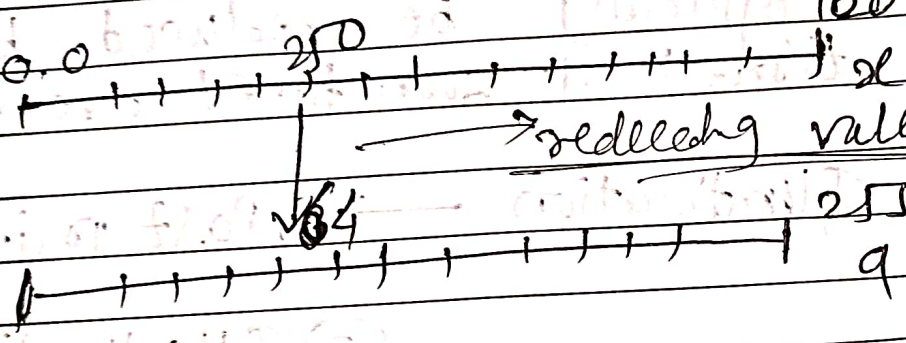
Types of Quantization

① Symmetric quantization ② Asymmetric Quantization

Batch normalization
mean = 0 & 6 = 1

Symmetric Usigned int 8 quantization.

→ [0.0 ---- ---- 1000] → NO → 32 bites → Uints
8 bits ⇒ 2^8
|| 0 – 255



min max scalar
0.0 → 0
1000 → 255

$$\text{scalefactor} = \frac{x_{max} - x_{min}}{q_{max} - q_{min}} = \frac{1000 - 0}{255 - 0} = 3.92$$

$$\text{round} = \frac{250}{3.92} = 64$$

Higher → Lower

zero point = 0
scale = 3.92

② Asymmetric Uint 8 quantization

[-20.0 .... 1000.0]

[0 .... 255]

-20.0 → 0

1000 → 255

$\frac{1000 + 20}{255} = 4.0$

round $= \left(\frac{-20}{4}\right) = (-5.0)$ $-5 + 5 = 0$

Zero point = 5
Scale Factor = 4.0

Modes of Quantization

① Post Training Quantization (PTQ)

weights + bias

→ Pre-trained model → Calibration → Quantized model → Use case

Calibration ↑ squeezing / reducing value

Loss of data, Accuracy decrease

② Quantization Aware Training (QAT) — without sacrificing Accuracy

Trained model ↓ Quantization ↓ Fine Tuning → Quantization model

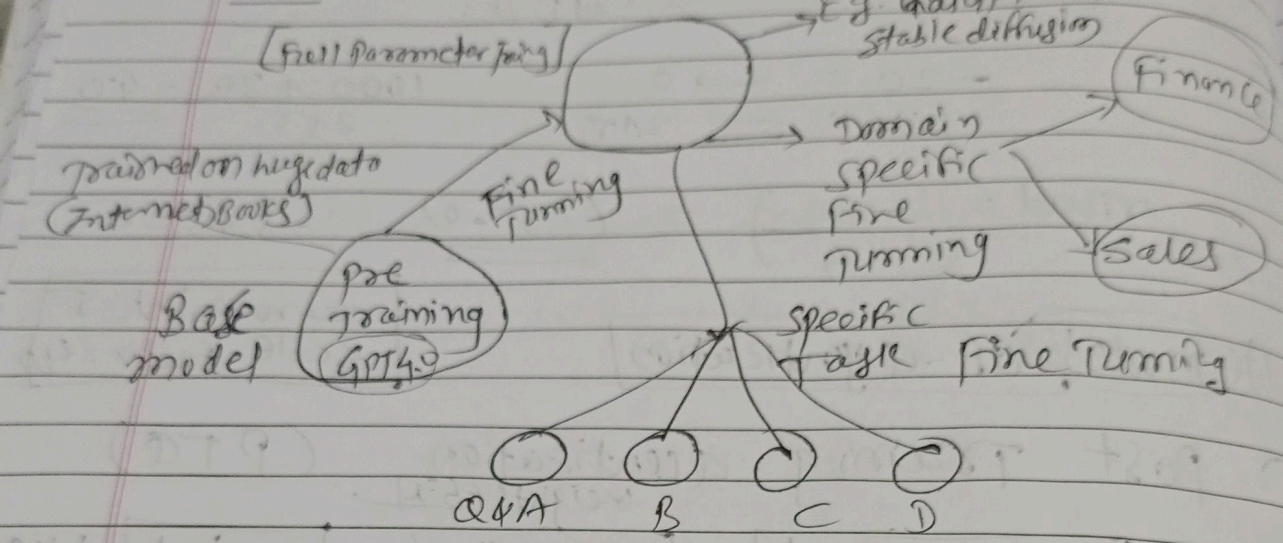
New training data → Fine Tuning

Fine tuning LLM model

#LORA, QLORA

Indepth mathematical Intuition

① LORA - Low order rank Adaptation



Full Parameter Fine Tuning [Challenges]

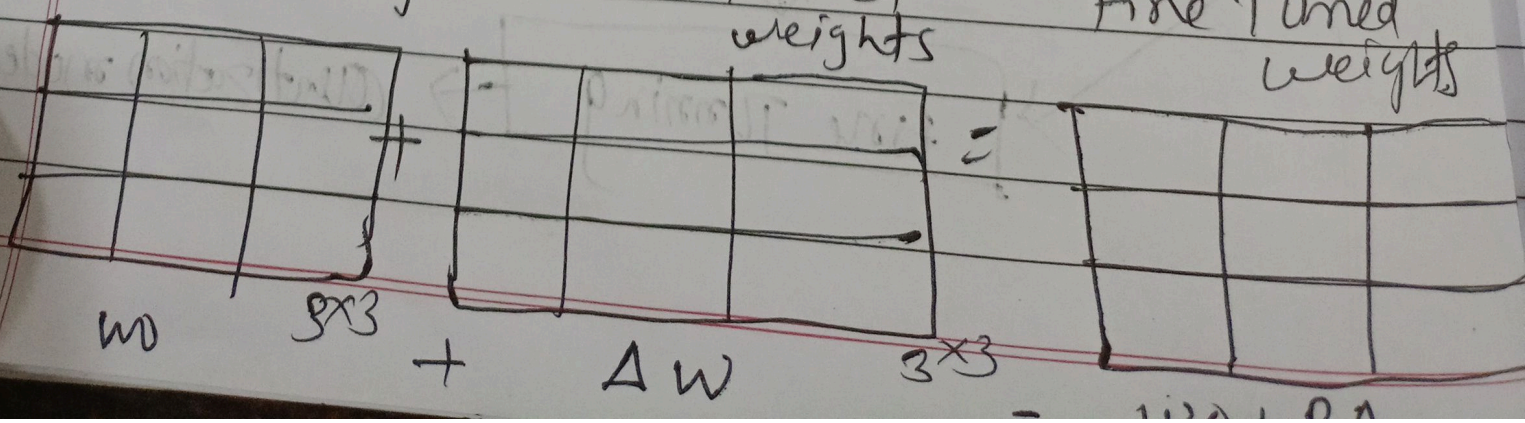
① Update all model weights (175B)

② Hardware Resource Constraints & downstream task → RAM, GPU, model training

To overcome full parameter fine tuning we use LoRA, QLoRA. (Lora 2.0)

LoRA — Instead of updating weights it tracks changes.

PreTrained weight + LoRA Tracked weights = Fine Tuned weights

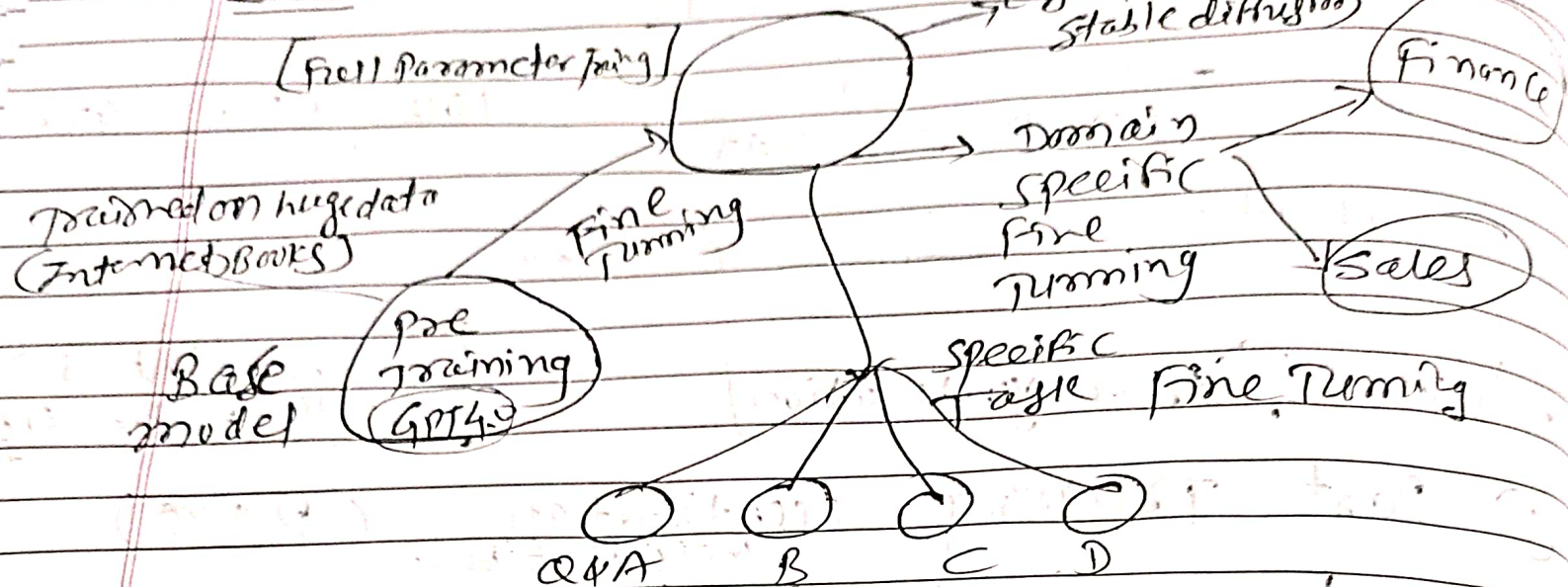


Fine tuning LLM model

LORA, QLORA

Indepth mathematical Intuition

① LORA - Low rank order Adaptation



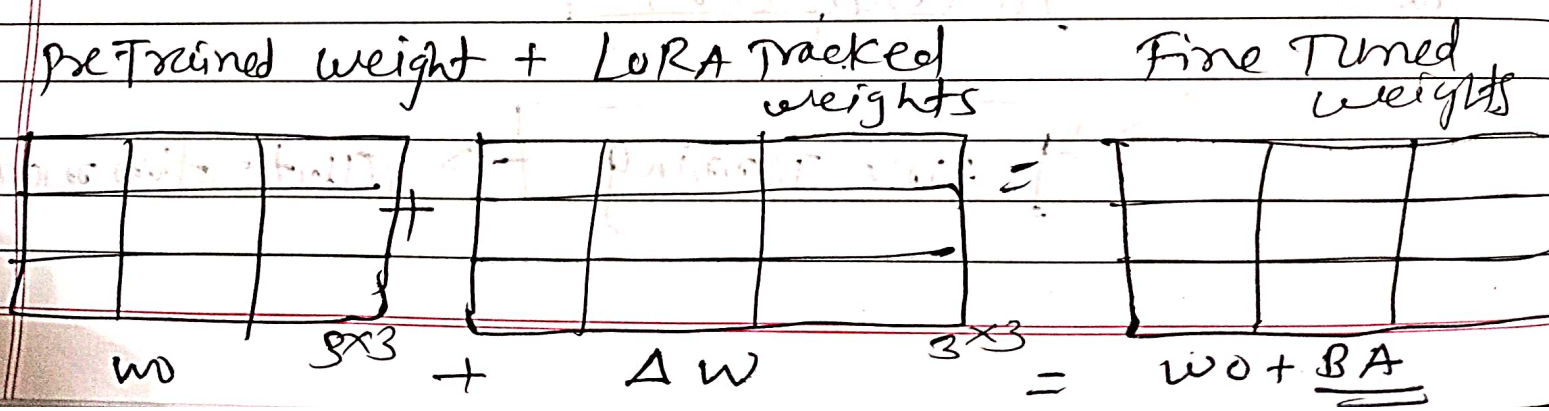
Full Parameter Fine Tuning [Challenges]

① Update all model weights (175B)

② Hardware Resource Constraints & downstream tasks → RAM, GPU, model [illegible]

To overcome Full Parameter Fine Tuning we use LORA, QLORA. (plora 2.0)

LORA — Instead of updating weights it tracks changes.



model complexe (things)
High Rank
matrix decomposition

Linear Algebra

LoRA

[$W_0 + \Delta W = W_0 + BA$]

[How to select Rank?]

(B) (A)

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}_{3\times 1} \times \begin{bmatrix} 4 & 5 & 6 \end{bmatrix}_{1\times 3}$$

Rank = 1

* **Rank matrix** = The Rank of matrix is the number of linearly independent rows.
or
The Number of linearly independent Columns

only zero matrix has rank zero.

Example =
$$\begin{bmatrix} 1 & 2 & 3 \\ 1 & 4 & 2 \\ 2 & 6 & 5 \end{bmatrix}_{3\times 3}$$

upper triangle matrix.

formation operation = $R_2 \rightarrow R_2 - R_1$
$R_3 \rightarrow R_3 - 2R_1$

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & -1 \\ 0 & 2 & 1 \end{bmatrix}$$

$R_3 \rightarrow R_3 - R_2$

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & -1 \\ 0 & 0 & 2 \end{bmatrix}$$

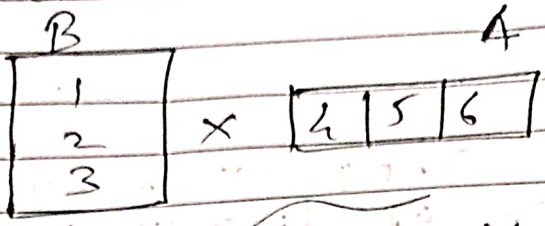
upper triangle

Rank of matrix = 3

x decomposition: less Parameter required, less matrix so, solve challanges

⭐ QLoRA = Quantization Low order Rank Adaption

matrix decomposition



Float 16Bit
↓↓ Convert
4Bit

Reduce the memory - Convert Higher memory Format to lower memory Format

Bits and bytes = Process of quantization

fine Tune = parameter efficient Transfer learning for NLP
(Peft)

Transformers = "Attention all you need"
"state of the art"

NN Architecture

Transformer Architecture

① Embedding - Text input divided into smaller unit called Tokens.

Token → numerical vectors
Converting

called embeddings

BitNet b1.58 Ternary {-1, 0, 1} 12-1-24

LLMOps & LLM pipeline

VEXT — Platform ⇒ LLMOps

Create AI Project

Instead of building any model from scratch use VEXT LLMOps without coding.

Fine tune Gemma Models in Keras using LoRA.

import keras
import keras_NLP

Backend Jax

(2) Transformer Block —

(1) Attention mechanism - It allows tokens to communicate with other tokens, capturing contextual info & relationship between them

(2) MLP — multilayer perceptron layer — FFNN that operates on each token idependently.

(3) output probabilities — Final linear & softmax layers transform the processed embeddings into probabilities enabling the model to make predicti about the next token in a seq.