Fine Tunning LLM Models

① Quantization

① Full Precision / Half Precision → related to data types / how the data is stored in the memory.

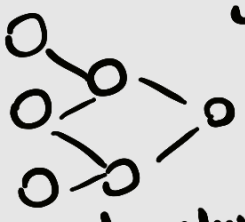
② Calibration - Model Quantization
↳ Problems

③ Modes of Quantization
↳ Post Training Quantization
↳ Quantization Aware Training

Data → weight and parameters

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

weights → matrix



neural networks.

with LLM models, the parameters are alot and keep increasing.
Llama 2 → 70 billion for example.

every value are stored in the memory in the form of 32 bits.
FP 32
(Full precision/ single precision) like floating points.

let's say here we have the value 7.23, and this number is stored as 32 bits in the memory.

we can converts this 32 bit to int 8. 8 bits
we can run the model in our resources more easily.

inferencing → giving input and getting output.

quantization helps inferencing.

@ similarly, in Tensorflow data are stored in the TF 32 bit format.

↳ In the scope of LLMs, after Quantization we can also fine tune the models. But this has some problems. When we are converting from 32 bits to lower bits, we may face some data loss and it leads to some accuracy loss. But there are specific techniques to overcome this.

Calibration : How we will be able to convert 32 bit into int 8. It answers "what is the formula required."

* How to perform Quantization? *

① Symmetric Quantization ② Asymmetric Quantization

⇓

Batch Normalization From Deep Learning.

technique of Symmetric Quantization.

↳ between all the layers, we apply batch normalization, so that all our weights are zero centered.

(ex) Symmetric Unsigned int 8 quantization.

[0.0 .... 1000] → Numbers → 32 bits → Uints

0 to 1000

8 bits

2^8

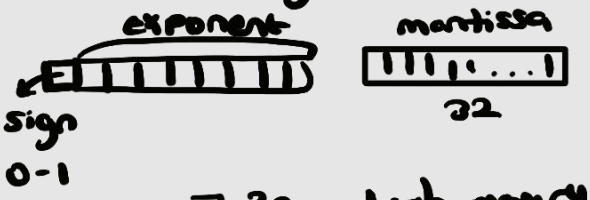
unsigned → no signs

0 - 255

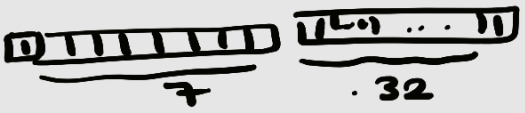
0.0 ——— 1000 x

0 ——— 255 q

Single precision Floating Point 32



7.32 high memory



FP 16 (half precision)

less memory

Min max scaler

0.0 → 0

1000 → 255 ↓ bits

$$scale = \frac{x_{max} - x_{min}}{q_{max} - q_{min}}$$

quantization

$$\frac{1000 - 0}{255 - 0} = 3.92$$ scale factor.

$$round\left(\frac{250}{3.92}\right) = 63.77 \rightarrow 64$$ symmetric

FP32 → Uint8

zero point = 0

important quantization parameter.

② Asymmetric Uint8 quantization

[-20.0 .... 1000.0]

[0 ... 255]

$$\frac{1000 - (-20)}{255} = 4.0$$ scale factor

$$round\left(\frac{-20}{4.0}\right) = -5$$ → but our range is 0 - 255

add +5 → -5 + 5 = 0 → add the same number.

↳ "zero point"

- if it was signed int8, range would be [-128 ... 127] and then apply the same formula.

→ the process we apply to the quantization is called callibration.

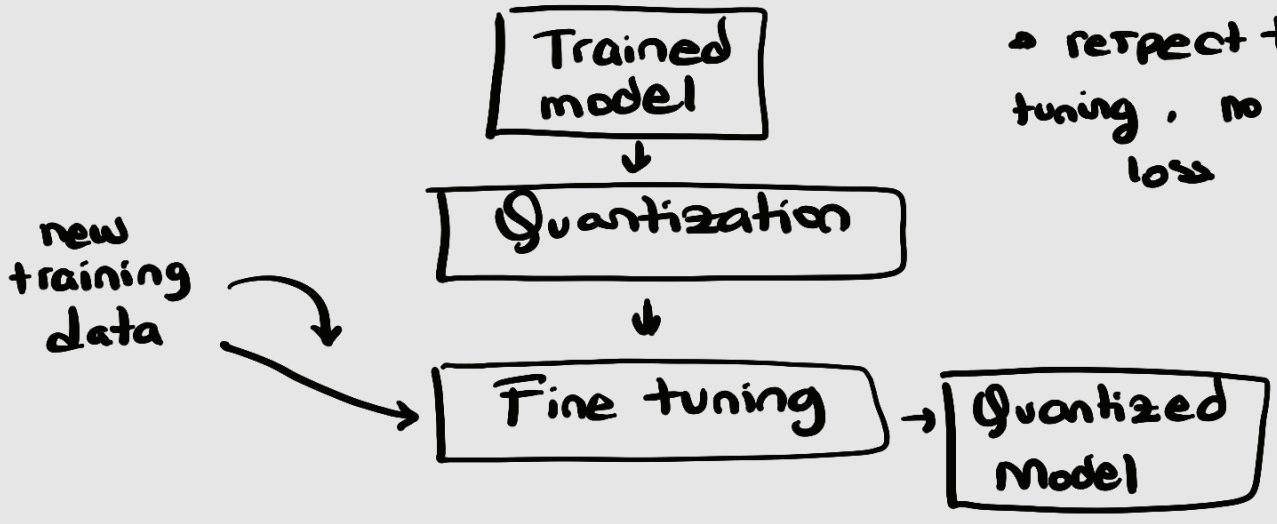
Modes of Quantization

Post training quantization (PTQ)

- there is loss of data
- accuracy decreases

Quantization aware training (QAT)

- respect to fine tuning, no accuracy loss



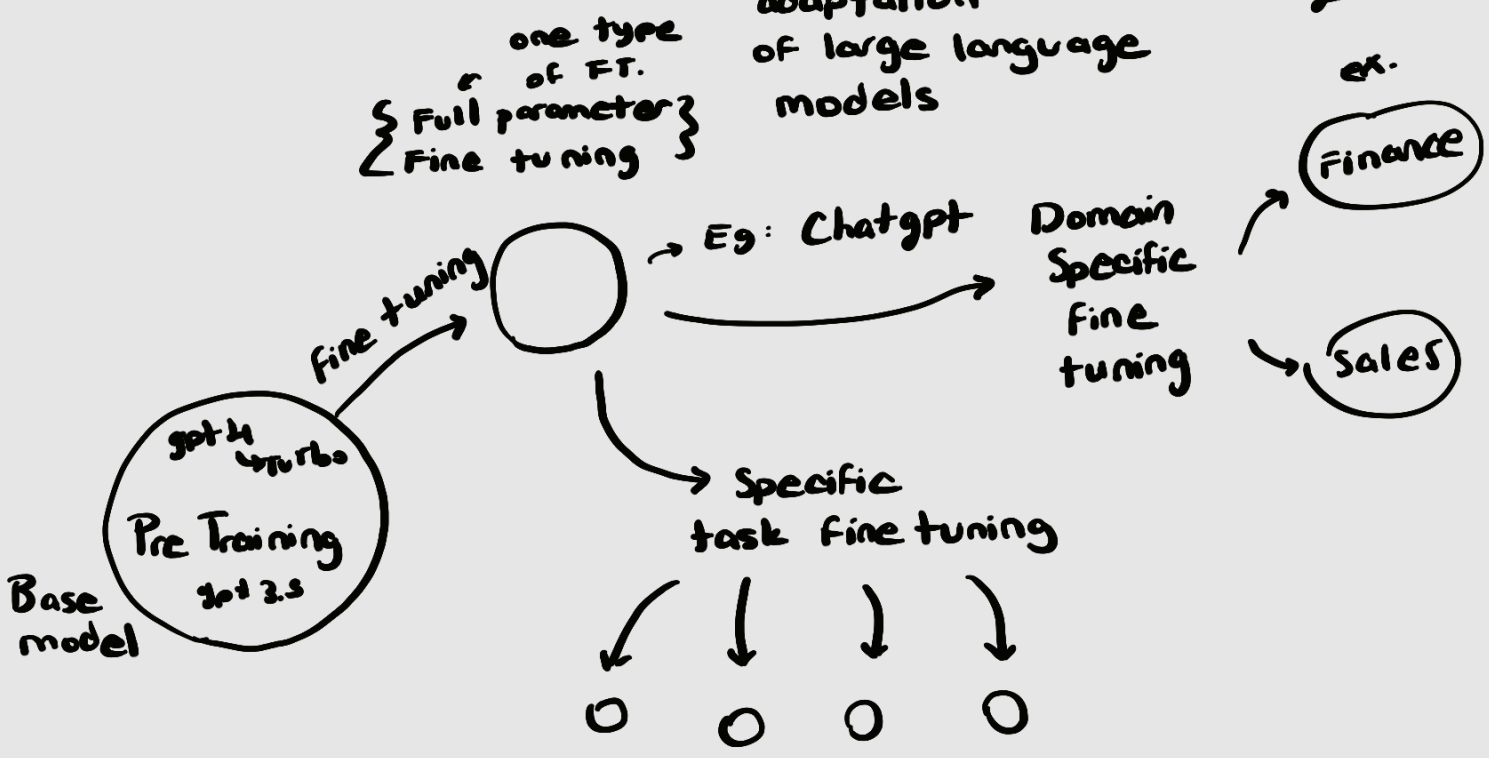
- QLora / Lora

Fine Tuning LLM - LoRA, QLoRA

LoRA → low-rank adaptation of large language models

QLoRA → Quantized LoRA

Training an LLM:



* pre-trained llm models (gpt 4 turbo etc) are base models.

Full Parameter Fine Tuning [challenges]

① need to update all model weights.

② Hardware Resource Constraint.

Downstream tasks
- ↳ model monitoring
- ↳ model inferencing
- ↳ GPU, RAM constraint

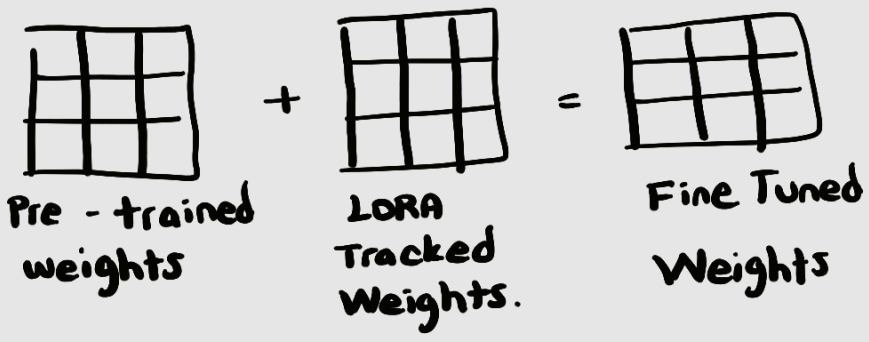
} challenges

Because of these challenges we use LoRA, QLoRA

What does LoRA do?

① Instead of updating all the weights, it will track changes.

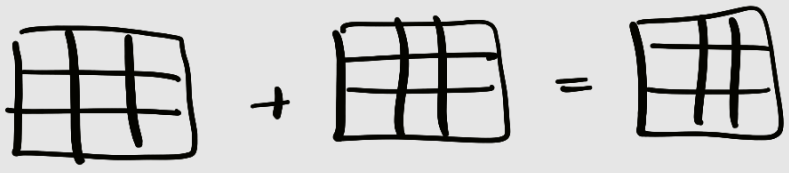
↳ track the changes of the new weights based on fine tuning.



* Lora uses a technique to track changes.

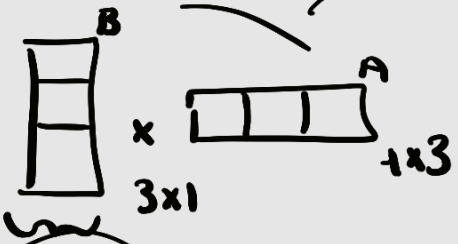
" Matrix Decomposition"

3x3 matrix will be saved on a two smaller matrix.



$W_0 + \Delta W = W_0 + BA$

matrix decomposition



Rank = 1

check online for calculation