Fine Tuning

What, why and how finetuning ?

What?

- Fintuning is basically used to improve LLM performance
- If we are using rag with llm if you are not getting proper answer then we need to use FineTuning
- Pretained model – any llms (state of art model) – it has trained with their own data set and its fixed
- In real case we have to use our won dataset with respected to our domain.

Transfer learning model –

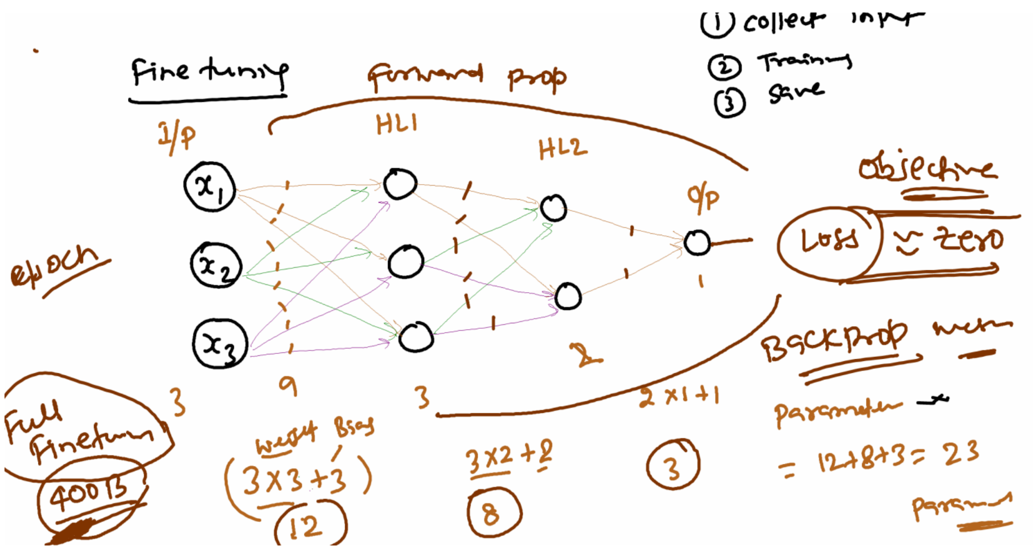
Instead of creating the model gain we can use the existing model again because model is already trained with billions on data its not possible to bult our won data llm only change is we need to do is how the way out data output should come.

Why ?

- We will use the same pretrained the model but we need to change the output or different way of output of the model we can achieve it using finetuning.
- Using pretrained model data set + extra dataset (our own)

How?

– You need to understand multi layer perceptron, forward propagation and backward propagation.



Full finetuning is very very expensive and difficult we need lots of resources like huge GPU machines if we 400B of data and we have to maintain loss value should be equal to zero.

1) Collect input data
2) Training
3) save

PEFT – Parameter efficient Finetuning

- Full finetuning is impossible for Pretrained LLM model, so we will use the PEFT model
- PEFT is a customized like we can use or add tiny set of extra weight to the existing fixed pretrained data set.

Pretrained LLM Data	**+**	Extra dataset like our own dataset (new hidden layer)

- And for the new extra hidden layer we will do fine tuning so that we can do training for this small data so we can use less gpu and computation memory
- Here the concept comes **LORA and QLORA** – We are doing finetuning for only extra dataset

LORA – Low Rank Adaptation

Example:

Old weight (Wo) = 512 * 512 = 262144

New weight (Wn) = Wo + Wn = old + new weights = 512*4 + 4* 512 = 10240 (considered 4*4 matrix)

- In 262144 we are considering 10240 params only it is around 3.9% (huge)
- In general use case we have to consider only 1.5 to 2%
- For finetuning we are just adding new weights to the existing Pretrained model

Does LORA will reduce the performance of LLM?
It will improve the model because we are using existing pretrained model and also we are adding extra weights definitely it will improve the performance.

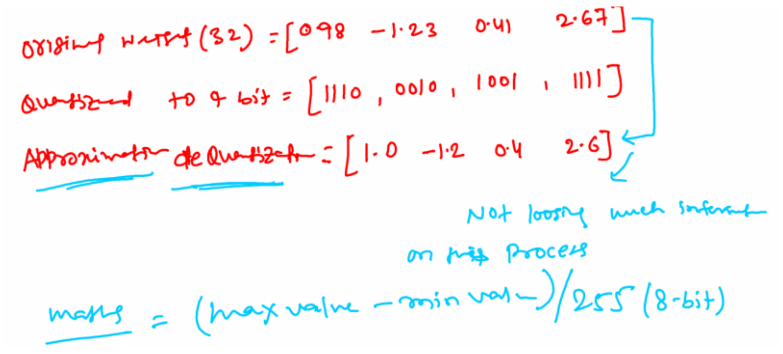
QLORA – Quantized + LORA (Low Rank Finetuning)

- Quantized means reducing the weight (compress the original model weight)

Example –

Original value 32 bits by the help of quantization reduce the weights to 4 bits/ 8 bits because Storing 32 bits takes more memory

- We are only compressing new weights
- Here we are just minimizing the bit size not the parameters, parameters remain so no loss of params or data



Original weight (32) = [0.98 -1.23 0.41 2.67]

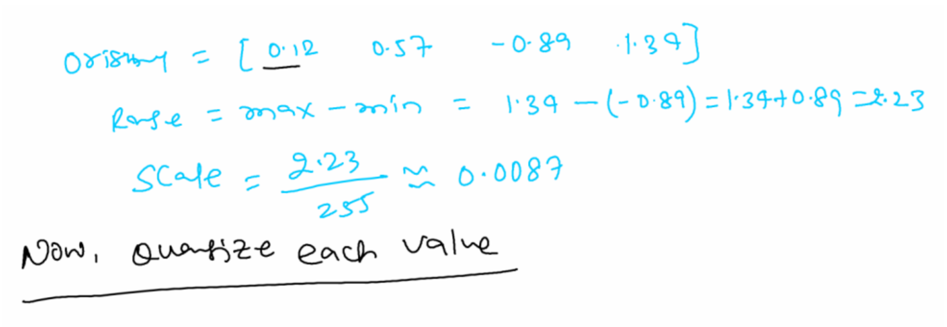
Quantized to 4 bit = [1110, 0010, 1001, 1111]

Approximation dequantization = [1.0 -1.2 0.4 2.6]

Not loosing much information on this process

maths = (max value − min value)/255 (8-bit)

How maths formula works, consider below example



Original = [0.12 0.57 -0.89 1.39]

Range = max − min = 1.39 − (−0.89) = 1.39 + 0.89 = 2.23

$$\text{Scale} = \frac{2.23}{255} \simeq 0.0087$$

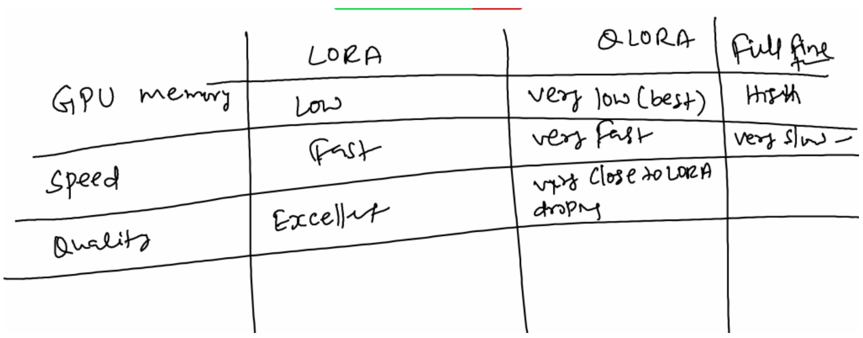
Now, Quantize each value

Advantages:

- memory saving 32 to 4 bits
- Edge device or light device
- Using by quantization method
- Using by approximation or dequantization method
- QLORA is cost less compared to LORA

QLORA gives better performance than LORA because reduce of params but there is some difference in QLORA, it will give hallucination rarely and there is a loss of some params because we are taking approximate values when doing quantization.

If we have enough memory, then we will use LORA instead of QLORA.



	LORA	QLORA	Full fine
GPU memory	Low	very low (best)	High
Speed	Fast	very fast	very slow −
Quality	Excellent	very close to LORA dropping	

Methods Used in Finetuning

Means which finetuning we have to use.

Supervised Finetuning (SFT)

Where we are using instruction and response kind of LLM

Example: Chatbot – Q/A

GT - General Finetuning

Unsupervised in nature , like general statements

Example: Hi, How are you?

RL – Reinforcement learning

Reinforcement learning human feedback -RLHF

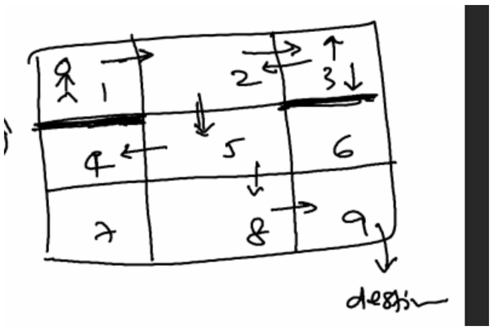
We have to follow the fastest path to reach the destination, like automatic car if steps wrong penalty else reward.

Response tuning –

Text- I am good at GenAI

Rejected text – In some of the subject of GenAI , I am not good

- So, basically if we are getting the human feedback of the general text then we have to accept rejected text and give the feedback (like I am good at all except java).
- Whenever we build a bot when we are checking wheather my bot is giving good or bad answer then we have to use RT method because we are collecting human feedbacks also so that we can train the model better.



DPO – Direct preference optimization

Learn the policy , when we have preferred answers there and also rejected answers there, so that we can push the model with preferred answers.

DPO is enhancement of RT , we are enhancing the DPO by adding prompt extra to RT

MCP – set of rules

Prompt – what are generators in python?

Text bot – topic in generator

Rejected text – are you talking about iterators or generators

Finetuning we are applying to all prompt, bot text, rejected text

ORPO – Odd-Ratio Preference Optimization

It can handle streaming data, data is not properly aligned – ab comparison means bot text and rejected text

Supervised and unsupervised data given then we have to use ORPO

ST and GT