What is Generative AI?

Generative ai is types of ai that can generate new content such text, images, music, video etc, based on patterns and structures it learns from large datasets.

Finetuning Techniques of LLMs:

1. ***Full parameter finetuning (Pretraining)***
2. ***Parameter Efficient Fine Tuning (PEFT)***

Parameter Efficient Fine Tuning (PEFT)

PEFT Methods:

1. **LoRA**
2. **QLoRA**

LoRA and QLoRA are fine-tuning techniques for large neural networks that **use low-rank decomposition and quantization** to make advanced AI models more accessible and sustainable.

Fine Tuning Existing Models

1. When dealing with domain-specific use cases that require model adaptations, Fine Tuning becomes required.
2. Fine-Tuning allows us to leverage existing pre-trained foundation models and adapt them to specific tasks or domains.
3. Fine tuning the model requires a lot of training data, huge infrastructure and effort.
4. In the process of full parameter fine-tuning of LLMs, there is a risk of ***catastrophic forgetting***, where previously acquired knowledge from pretraining is lost.
5. There are various techniques such as **Parameter Efficient Fine Tuning (PEFT**), which provide a way to perform modular, fine-tuning with optimal resources and cost.

Parameter Efficient Fine Tuning (PEFT)

1. PEFT is a technique designed to fine-tune models while minimizing the need for extensive resources and cost.
2. PEFT is a great choice when dealing with domain-specific tasks that necessitate model adaptation.

3. By employing PEFT, we can strike a balance between retaining valuable knowledge from the pre-trained model and adapting it effectively to the target task with fewer parameters.
4. There are various ways of achieving Parameter efficient fine-tuning. Low Rank Parameter or LoRA & QLoRA are most widely used and effective.

Low-Rank Parameters

This is one of the most widely used methods, where a set of parameters are modularly added to the network, with lower dimensional space. Instead of modifying the whole network, only these modular low-rank networks are modified, to achieve the results.

Low-Rank Adaptation (LoRA)

1. Low-Rank Adaptation provides the modular approach towards to fine-tuning a model for domains specific tasks and provides the capability of transfer learning.
2. LoRA technique can be implemented with fewer resources and are memory efficient.

In the following picture you can see the dimension/rank decomposition, that reduces the memory footprint considerably.

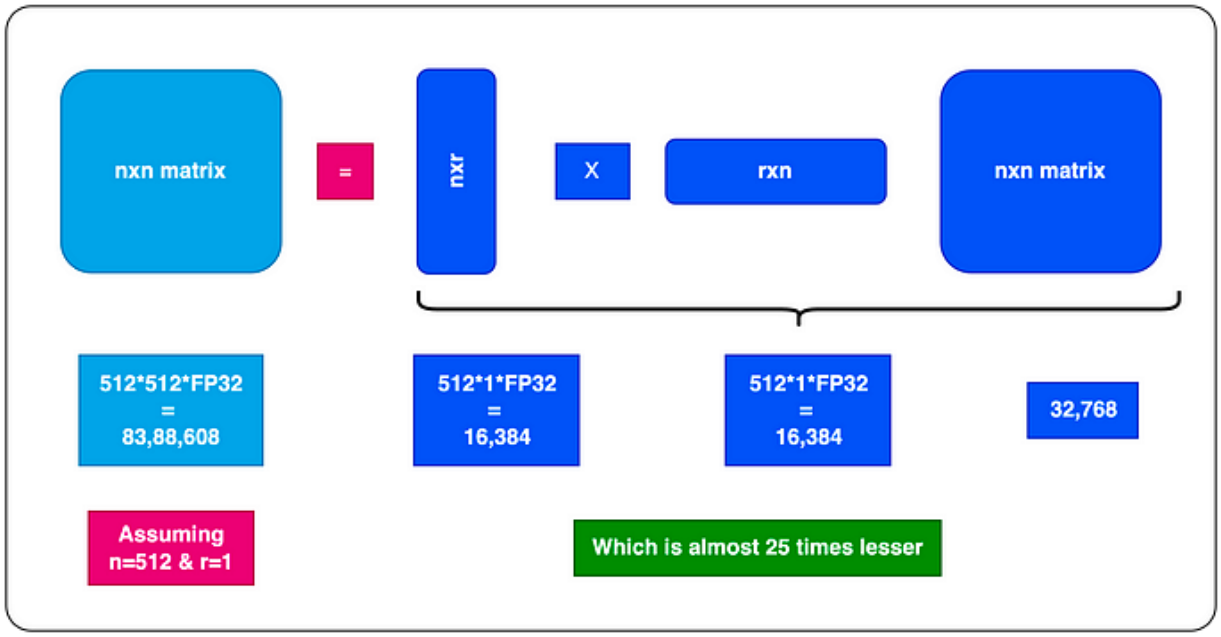

Figure 12

LoRA network for Training

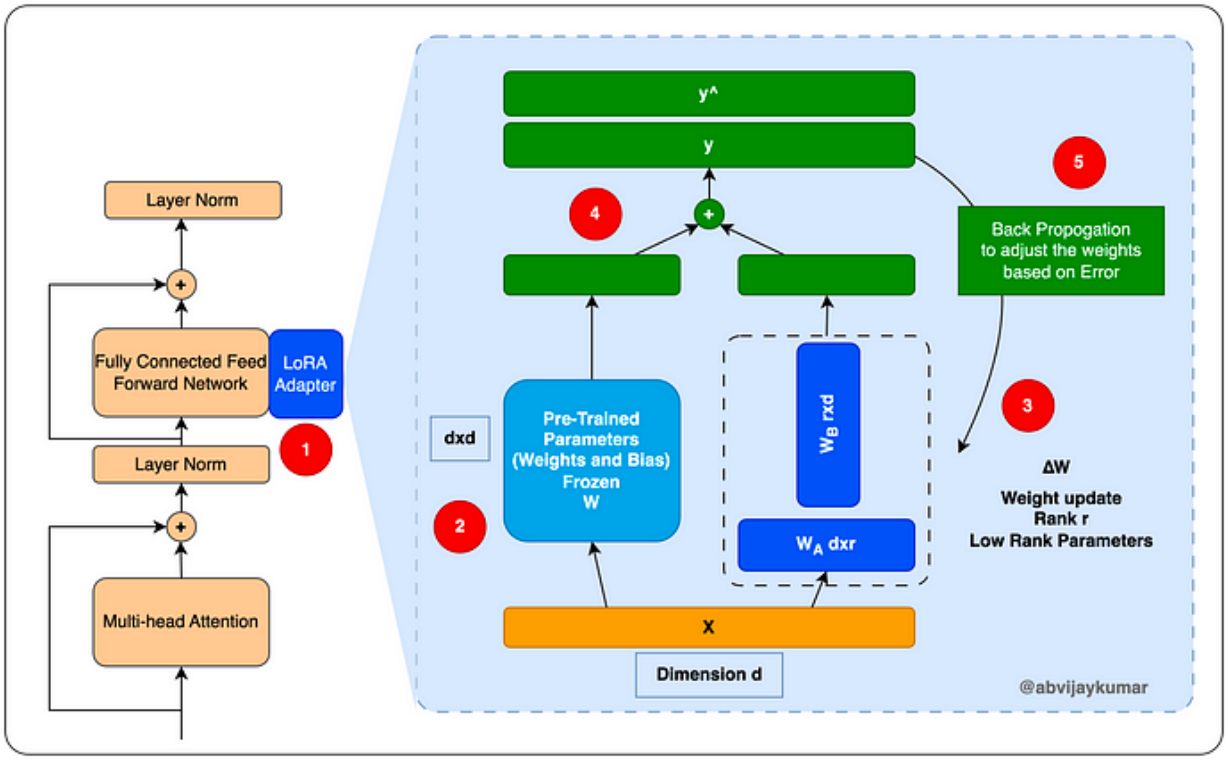

Figure 2

Steps:

1. **LoRA can be implemented as an adapter designed to enhance and expand the existing neural network layers. It introduces an additional layer of trainable parameters (weights) while maintaining the original parameters in a frozen state. These trainable parameters possess a substantially reduced rank (dimension) compared to the dimensions of the original network. This is the mechanism through which LoRa simplifies and speeds up adapting the original models for domain-specific tasks. Now, let's take a closer look at the components within the LORA adapter network.**
2. **The pre-trained parameters of the original model (*W*) are frozen. During training, these weights will not be modified.**
3. **A new set of parameters is concurrently added to the networks WA and WB. These networks utilize low-rank weight vectors, where the dimensions of these vectors are represented as dxr and rxd. Here, 'd' stands for the dimension of the original frozen network parameters vector, while 'r' signifies the chosen low-rank or lower dimension. The value of 'r' is always smaller, and the smaller the 'r', the more expedited and simplified the model training process becomes. Determining the appropriate value for 'r' is a pivotal decision in LoRA. Opting for a lower value results in faster and more cost-effective model training, though it may not yield optimal results. Conversely, selecting**

a higher value for 'r' extends the training time and cost, but enhances the model's capability to handle more complex tasks.

4. **The results of the original network and the low-rank network are computed with a dot product, which results in a weight matrix of n dimension, which is used to generate the result.**
5. **This result is then compared with the expected results (during training) to calculate the loss function and WA and WB weights are adjusted based on the loss function as part of backpropagation like standard neural networks.**

Types of RAG:

Agents:

Autogen

React

List of all agents:

Langchain: tools

Llamaindex: tools

Quantization

Ggml/gguf

Gptq

AwQ

Onxx

Prompt engineering:

Document Retravel and chunking methods:

Evaluation

1. **Raga**
2. **Langsmith**
3. **Prompt flow**

Feedback and human in the loop –

SLM –

Text and multi model llm

Model Name	Model Size	Context window
Llm1	2B	4140
Llm2	3B	8080
Llm3	4B	1290