FML Assignment-3
Generative AI - LLM

Hardik Prabhu
24105801g
(Big Data Analytics)

Week-1

* These Generative AI models are designed by mimicing human Activity
* So these LLM (Large Lagugs Model's) are trained on ~~trained~~ using trillions of word over many weeks & months with Large amounts of compute power
* LLM's are able to take natural Langugs or human written instructions & perform all the tasks which are assigned
  * The text passed to LLM is known as prompt
  * The space or memory avd. to prompt is called the context window
  * The output of the Model is called completion
* Act of using the model to generate text is known as Coference

LLM tasks & Use cases:-

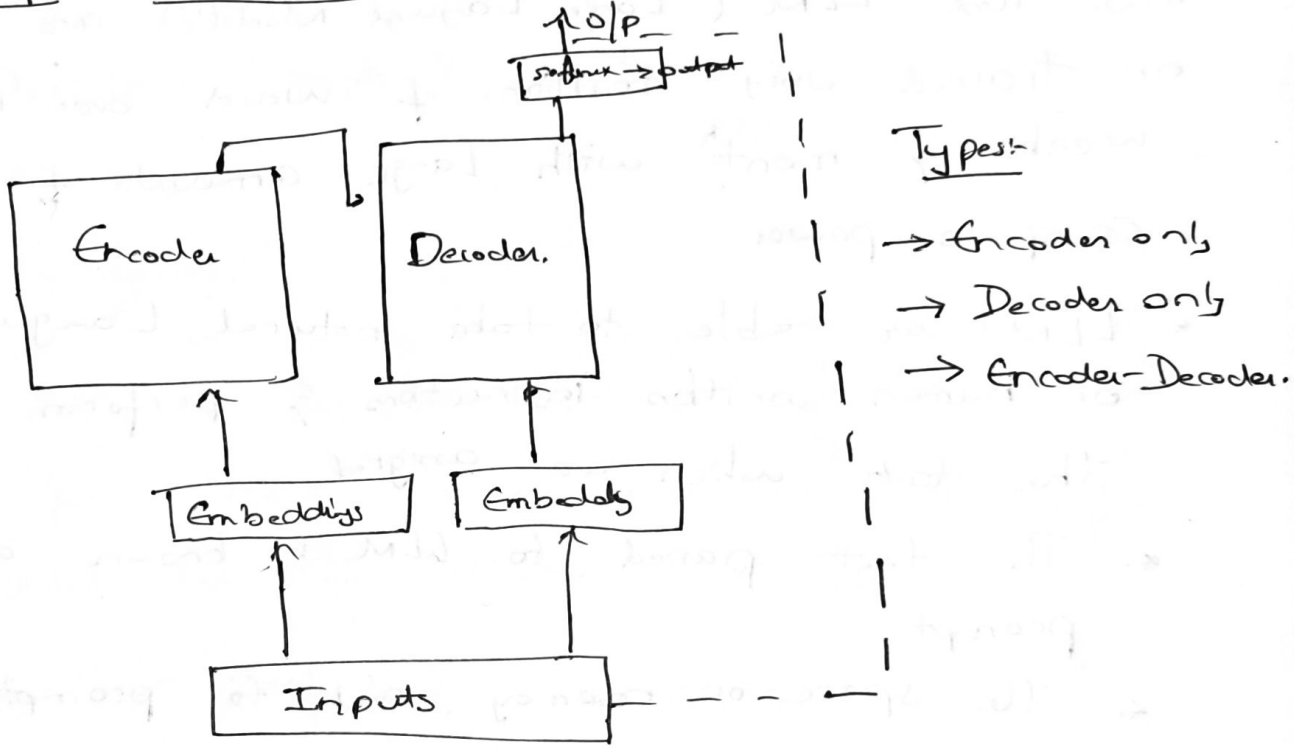
* It is used for Information retrieval
* Models can be tuned to peform speafic tasks
* previous RNN's used to generate text
* Used to check relation between prents

words,

transformer :-

* It checks the relations between each of the words
* Attention map used to define attention weights.

Simplified transformer Architecture :-



→ Inputs are the words which are converted into numbers as by tokenizer, where each number is a dictionary of all the words model can work with.

→ Tokenizer methods :-

→ token ID's match two complete words

→ token ID's used to represent pts of words.

→ Tokenizer used to train the model we must use the same tokenizer wh we

generate text

→ User inputs are converted into embeddings where the tokenizer is used to modify to accordingly to the desired output

* Encoder :- It encodes input i.e prompts with contextual understanding & produces one vector per input token

* Decoder :- Accepts input token & generates new tokens.

Prompt Engineering :-

* Work to develop & Improve the prompt is known as prompt Engineering

* In context Learning provides extra information needed for the model to perform better

zero shot Inference - provide no example
one shot Inference - provides one example.
few shot Inference - provides few example

Generative Configuration :-

* Max new tokens

→ Greedy vs random Sampling → Selects a token using a random word with highest probability is selected / weighted strategy.

top k

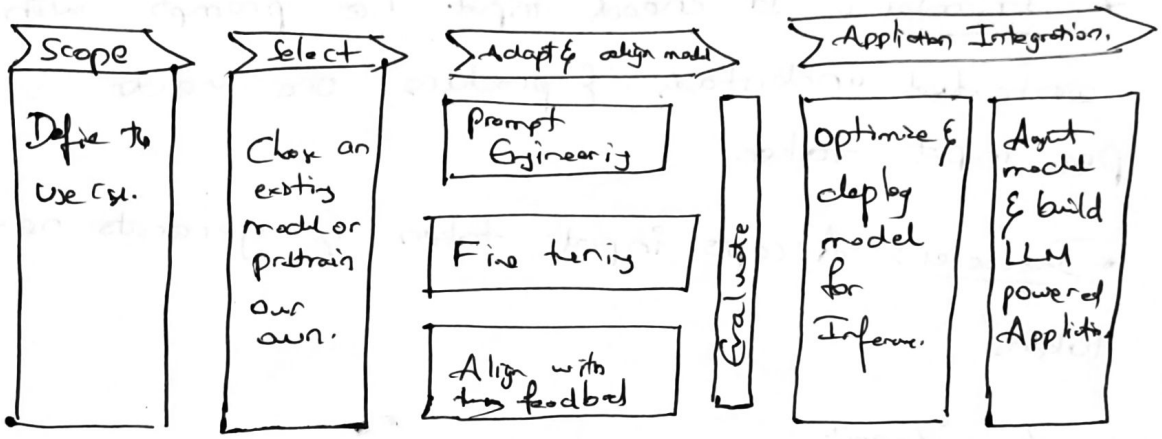
→ number of token to randomly choose from

top p

total probability that we want the model to choose from.

* temperature :- Higher the temperature higher the randomness.

Generative AI project Lifecycle :-



* Pre train LLM's :-
  * Autoencoding models are pre-trained using masked Language Modeling
  * Auto Regressive models are pretrained using Casual Language models
  * Sequence to sequence models are pretrained generally using Span Competition.

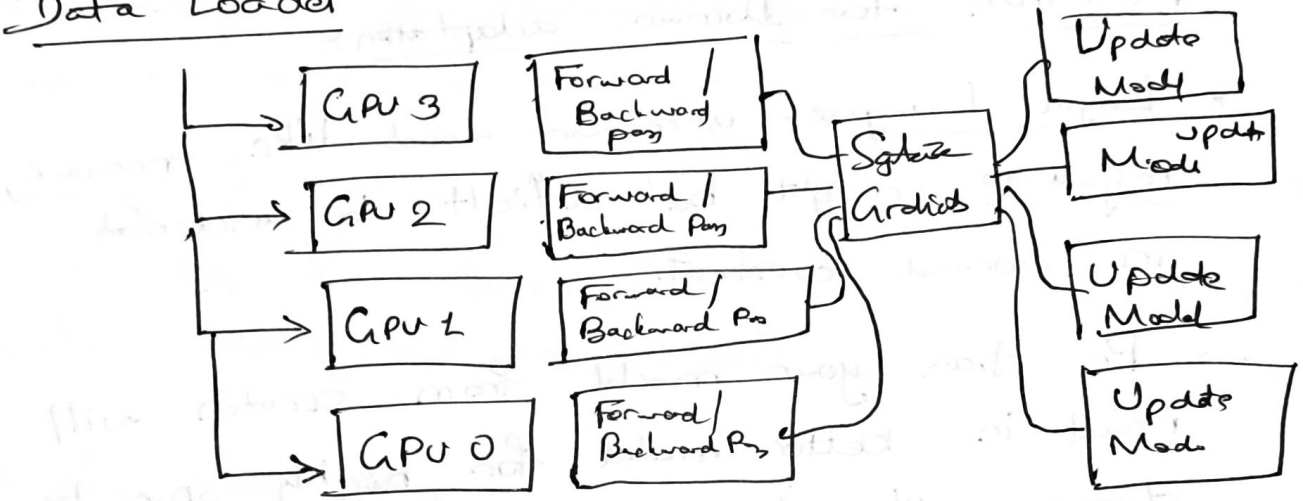
* Computational challenges :-
  → out of memory
  Quantization is used to reduce the problem
  → to reduce required memory to store & train models.
  → Projects original 32-bit floating point numbers into

lower precision spaces

Efficient Multi-GPU Compute strategies

Distributed Data Parallel

→ Zero Redundancy Optimizer



* In fully Shared Data Parallel requires to collect this data from all of the GPU's before the forward & Backward Pass.
* It helps to reduce overall GPU memory utilization.
* Supports offloading to CPU if needed.

Scaling Laws & Compute-optimal Models:-

* Goal is pretraining to maximize model performance which is minimize model loss.
* It can be done by scaling choice
  - → Dataset size (number of tokens)
  - → Model size (Number of parameters)

→ Constraint

↳ Compute budgets:-

→ "peta flop-day" } floating point operation performed at rate of 1 peta flopper second for one day

* Pre-Train for Domain adaptation:-

* Legal Langue:- uncommon words like mensam, reasjudicate, might be difficult to understand in normal context.

→ Pre train your model from scratch will result in better model for highly specialized domain like law, medicine or finance.

Ex:- Bloomberg GPT was specialized in financial tasks.

Week 2

Instruction Fine tuning:-

* Here Fine tuning LLM with Instruction prompt.

* This has disadvantage like in context Learning may not work for smaller model

* Example take up space in the context window.

Fine tuning is a supervised Learning process where you use a dataset of labelled examples to update the weights of LLM.

Instruction fine tuning:-

→ trains model to demonstrate how it should respond to a specific Instruction.

Prompt [] Completion []

these are feeded to the LLM (Large Laguage Model)

Steps for fine tuning

1) Prepare Your Dataset
2) train-test-Split
3) Pass it to the Model
4) LLM Completions.
5) Calculate loss using Cross-Entropy

Fine tuning on Style task:-

* Catastrophic forgetting:-

* Fine tuning can significantly increase the performance of a model on a specific task but can lead to reduction

in ability on other tasks.

How to avoid Catastrophic forgetting :-

* First check whether catastrophic forgetting actually impacts your use case.
* Fine tune on multiple task at the same time which would require more resources & compute time.
* Consider Parameter Efficient Fine-tuning (PEFT)
  → It preserves the weights of the original LLM and trains only a small number of task specific adapter layer & parameters.

Multi-task instruction fine tuning :-

* Instruction fine-tuning with FIAN (Fine tuned language)
* FLAN models refer to a specific set of Instruction fine tuning (FLAN-T5, PALM)
* FLAN-T5 :- It is a great, general purpose Instruct Model which was 473 datasets across 146 task categories, including Samsum A dialogue Dataset.

Model Evaluation Metrics:

$$\text{Accuracy} = \frac{\text{Correct Prediction}}{\text{Total Prediction}}$$

While this metric can be considered for small models but while LLM's can be much more challenges.

→ Example → "Mike really loves drinking tea" & "Mike likes sipping tea" are of same context, So how does measure the similarity.

* ROUGE (Recall oriented under study for gisting evaluation, It is primarily employed to assess the quality of automatically generated summaries by comparing them to human-generated reference summaries.

* BLEU Score: (Bilingual evaluation understudy) is an algorithm designed to evaluate the quality of machine-translated text again by comparing it to human-generated translations.

Terminology:-

Unigram → Sigle words

Bigram → Two words

n gram → Is a group of n-words.

Rouge-1 Precision = $\frac{\text{unigram matchs}}{\text{unigram in output}}$

Rouge 1 Recall = $\frac{\text{unigram matchs}}{\text{unigram in reference}}$

Rouge-1 F1 = $2 * \frac{\text{Precision} * \text{recall}}{\text{Precision} + \text{recall}}$

[Harmonic mean]

* Here in unigrams order is not considered which would create conflict in some cases to overcome this we use bigram.
* Another method is taken the longest common subsequence present in both generated output and the reference output

LCS (gen, ref)

* BLEU score is useful in evaluating the quality of machine-translated text
* It is calculated using average precision over multiple n-gram sizes & then averaged.

BLEU metric = Avg (precision across range of n-gram sizes)

Benchmarks:

→ GLUE

→ SuperGLUE

→ HELM

→ MMLU

→ BIG - bench

Benchmarks for massive models:

→ MMLU (Massive multitask language understanding)

→ Big Bench

→ Big Bench Hard

→ Lite

→ HELM (Holistic Evaluation of Language Models)

Parameter Efficient Fine Tuning (PEFT)

Full time tuning of Large Language Models is challenging as it requires updating trainable weights, optimizer states, gradients, Forward Activations, temporary memory which would often increase the storage size to petabytes.

→ PEFT only updates only a small subset of parameters

→ Some PEFT techniques freeze most of the model weights & focus on fine tuning a subset of model parameters

PEFT tradeoffs include

→ Parameter Efficiency
→ Trang Speed
→ Inference Costs
→ Model Performance
→ Memory Efficiency

PEFT Methods

* Selective :- Selects the subset of initial LLM parameters to time frame.
* Reparametrization :- of a model weights using a low-Scale rank representation
* Additive :- Add trainable layer or params to model

LORA :- Low Rank Adaption of LLM's

1. Freeze most of the orignal LLM weights
2. Inject 2 rank decomposition matrics
3. Train the weight of the smaller matrices.

Steps to update model for Inference:

1> Matrix Multiply the low rank matrices
2> Add to original weights.

* Applying LORA to just the self-Attention layers of model is often enough to fine tune for a task

Prompt tuning with soft prompts

* With the LORA the goal was to find an efficient way to update the weights of the model without having to train every single parameter again

* In prompt tuning we add additional tokens & leave it up to the supervised Learn process to determine their optimal values.

* The set of trainable tokens is called a soft prompt & it gets prepend to embedding vectors that represent your input text.

* Prompt tuning is a very parameter efficient strategy because only a few parameters are being trained

* Prompt tuning can be effective as full fine tuning for larger models.