Generative AI

with Large Language Models (LLMs)

Organized by
Department of Mathematics, CEG, Anna University

Outcome of the Workshop

1. Gain Foundational Knowledge of how Generative AI works
2. Describe Transformer Architecture and how LLMs are trained and fine-tuned
3. Describe Retrieval Augmentation Generation and compare it with Fine Tuning
4. Understand key stages in a typical LLM-based generative AI project lifecycle
5. Recent Developments in Generative AI / LLM space

Topics Covered

1. Introduction to Generative AI
2. Warm-up Test
3. Tech Foundations of Generative AI
4. Foundation Models
5. Self-Supervised Learning
6. Fine-tuning Foundation Models
7. Retrieval Augmentation Generation
8. Fine-Tuning vs Retrieval Augmentation Generation
9. Lifecycle of a Generative AI project
1. Exit Test

Vikram Pratap Singh | Founder, Zestifai Technologies

@Zestifai Tech: Building Conversational AI Powered Virtual Assistants for eCommerce and Consulting businesses through their GenAI adoption journey

@VFS Global: Established “Product Innovation & Business Transformation” Dept. Envisioned, designed and led the development of a “Global Digital Platform” for Visa Processing.

@STG International: Established “Center of Excellence” and led the design and development of eLearning and Learning Center Management Systems.

Key Skills: Generative AI, DeepLearning, Machine Learning, Software Architecture, Product Management

Qualification: Master's in Computer Science and PMP, CBIP, TOGAF Certifications among others

Linkedin: https://www.linkedin.com/in/vikrampsingh/ | **Website**: https://www.zestifai.com

Hands On / Follow Along Exercises

Lab

1. Prompt Engineering
2. Fine Tuning
3. Retrieval Augmentation Generation

Logistics

1. Hugging Face
2. PyTorch
3. Jupyter Notebook

References

The workshop slides, quizzes and exercises heavily depended on following resources. Due credit to respective authors and publishers. These resources are also recommended to workshop participants for further study.

1. Generative AI with LLMs Course by AWS and DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/
2. Transformer and BERT Model Course by Google and DeepLearning.AI: https://www.coursera.org/learn/transformer-models-and-bert-modelb
3. Fine tuning Large Language Models by Lamini and DeepLearning.AI: https://www.coursera.org/projects/finetuning-large-language-models-project
4. Serrano Academy Youtube Channel: https://www.youtube.com/@SerranoAcademy
5. Paper: Attention is all you need: https://arxiv.org/abs/1706.03762
6. Paper: Low-Rank Adaptation of LLMs - https://arxiv.org/abs/2106.09685
7. Paper: The Era of 1-bit LLMs - https://arxiv.org/abs/2402.17764

Warm-up Test

Question 1: What is Generative AI?

A. Generative AI is a type of artificial intelligence (AI) that can create new content, such as discrete numbers, classes, and probabilities. It does this by learning from existing data and then using that knowledge to generate new and unique outputs.
B. Generative AI is a type of artificial intelligence (AI) that can only create new content, such as text, images, audio, and video by learning from new data and then using that knowledge to predict a discrete, supervised learning output.
C. Generative AI is a type of artificial intelligence (AI) that can only create new content, such as text, images, audio, and video by learning from new data and then using that knowledge to predict a classification output.
D. Generative AI is a type of artificial intelligence (AI) that can create new content, such as text, images, audio, and video. It does this by learning from existing data and then using that knowledge to generate new and unique outputs.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 2: What is an example of both a generative AI model and a discriminative AI model?

A. A generative AI model could be trained on a dataset of images of cats and then used to generate new images of cats. A discriminative AI model could be trained on a dataset of images of cats and dogs and then used to classify new images as either cats or dogs.
B. A generative AI model could be trained on a dataset of images of cats and then used to classify new images of cats. A discriminative AI model could be trained on a dataset of images of cats and dogs and then used to predict new images as either cats or dogs.
C. A generative AI model does not need to be trained on a dataset of images of cats and then used to generate new images of cats, because the images were already generated by using AI. A discriminative AI model could be trained on a dataset of images of cats and dogs and then used to classify new images as either cats or dogs.
D. A generative AI model could be trained on a dataset of images of cats and then used to cluster images of cats. A discriminative AI model could be trained on a dataset of images of cats and dogs and then used to predict as either cats or dogs.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 3: What are foundation models in Generative AI?

A. A foundation model is a small AI model pretrained on a small quantity of data that was "designed to be adapted" (or fine-tuned) to a wide range of downstream tasks, such as sentiment analysis, image captioning, and object recognition.
B. A foundation model is a large AI model both post and pre-trained on a vast quantity of data that was "designed to be adapted" (or fine-tuned) to a wide range of downstream tasks, such as sentiment analysis, image captioning, and object recognition.
C. A foundation model is a large AI model post-trained on a vast quantity of data that was "designed to be adapted" (or fine-tuned) to a wide range of downstream tasks, such as sentiment analysis, image captioning, and object recognition.
D. A foundation model is a large AI model pretrained on a vast quantity of data that was "designed to be adapted" (or fine-tuned) to a wide range of upstream tasks, such as sentiment analysis, image captioning, and object recognition.
E. A foundation model is a large AI model pretrained on a vast quantity of data that was "designed to be adapted" (or fine-tuned) to a wide range of downstream tasks, such as sentiment analysis, image captioning, and object recognition.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 4: Hallucinations are words or phrases that are generated by the model that are often nonsensical or grammatically incorrect. What are some factors that can cause hallucinations? Select three options.

A. The model is trained on noisy or dirty data.
B. The model is not given enough context.
C. The model is trained on too much data.
D. The model is not trained on enough data

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 5: What is a prompt?

A. A prompt is a short piece of code that is given to the large language model as input, and it can be used to control the output of the model in many ways.
B. A prompt is a long piece of text that is given to the large language model as input, and it cannot be used to control the output of the model.
C. A prompt is a short piece of text that is given to the large language model as input, and it can be used to control the output of the model in many ways.
D. A prompt is a short piece of text that is given to the large language model as input, and it can be used to control the input of the model in many ways.
E. A prompt is a short piece of text that is given to the small language model (SLM) as input, and it can be used to control the output of the model in many ways.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Warm Up Quiz Answers

Q1:

Answer: Answer: D

Generative AI is a type of artificial intelligence (AI) that can create new content, such as text, images, audio, and video. It does this by learning from existing data and then using that knowledge to generate new and unique outputs.

Q2:

Answer: A

A generative AI model could be trained on a dataset of images of cats and then used to generate new images of cats. A discriminative AI model could be trained on a dataset of images of cats and dogs and then used to classify new images as either cats or dogs.

Q3:

Answer: E

A foundation model is a large AI model pre-trained on a vast quantity of data that is "designed to be adapted" (or fine-tuned) to a wide range of downstream tasks, such as sentiment analysis, image captioning, and object recognition.

Q4:

Answer: A, B, D

Q5:

Answer: C

1. Introduction to Generative AI

Generative AI

- Refers to any AI system whose primary function is to **generate content.**
- This is in contrast to AI systems that perform other functions, such as
  - Classifying data (e.g., assigning labels to images)
  - Grouping data (e.g., identifying customer segments with similar behavior)
  - Predicting (e.g. predicting Housing or Stock Price)

Examples

- Image generators (such as Midjourney or Stable Diffusion)
- Large language models (such as GPT-4, PaLM, or Claude)
- Code generation tools (such as Copilot, AWS Whisperer, Devin)
- Video Generation (such as Sora from OpenAI)

1. Introduction to Generative AI

Large language Models (LLMs)

- Refers to the type AI Systems that works with language.
- The “large” term refers language models with large number parameters. Research has found that using more data and computational power to train models with more parameters consistently results in better performance.

Examples

- Large language models (such as GPT-4, PaLM, or Claude)

Nevertheless, it is still a somewhat vague descriptor. Does a model trained on programming code count as LLms? How about those LLMs that take images as inputs

1. Introduction to Generative AI

Foundation Models (FMs)

- Refers to the type AI Systems with broad capabilities that can be adapted to a range of different, more specific purposes.
- The original model provides a base (hence “foundation”) on which other things can be built.
- Examples of foundation models include many of the same systems listed as LLMs

At present, “foundation model” is often used interchangeably with “large language model” because language models are currently the clearest example of systems with broad capabilities that can be adapted for specific purposes. E.g. GPT 3.5 served as the foundation for ChatGPT

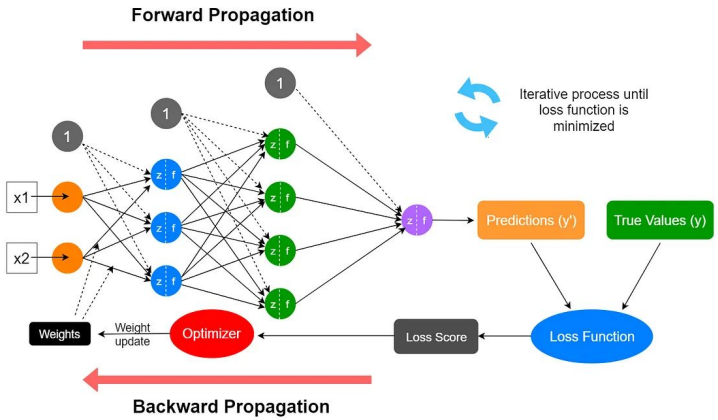
2. Tech Foundations of Generative AI

1. Transformer Architecture
    a. Self-Attention Mechanism
    b. Parallel Processing
2. Pre-Training
    a. Tokenization
    b. Self-Supervised Learning
3. Prompt Engineering
    a. Zero Shot
    b. One Shot
    c. Few Shot
4. Fine-Tuning
    a. Full Fine tuning (Instruction Fine Tuning)
    b. Partial Fine Tuning (PEFT: LoRA, QLoRA)
5. Retrieval Augmentation Generation

2. Tech Foundations of Generative AI: Weights and Biases

Neural networks

- Has **Weights** and **Biases**, and during training, its goal is to adjust those weights to reduce a **Loss Function.** Typically represented as matrices for each layer
- During the training process, these parameters are adjusted **iteratively** to minimize the difference between the predicted output and the actual output, thereby improving the model's performance
- Employs **Backpropagation** algorithm to adjust the weights of the connections in the network in order to minimize the error between the predicted output and the actual output of the network.



Images: https://medium.com/data-science-365/overview-of-a-neural-networks-learning-process-61690a502fa

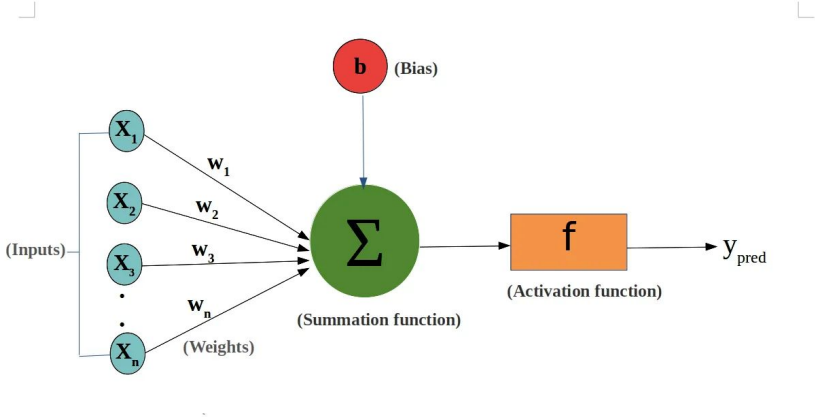
2. Tech Foundations of Generative AI: Weights and Biases

Weights

- Fundamental components that enable the network to learn from data during training process.
- Represent the strength of connections between neurons in adjacent layers and attached with each input/feature and they convey the importance of that corresponding feature in predicting the final output.

Biases

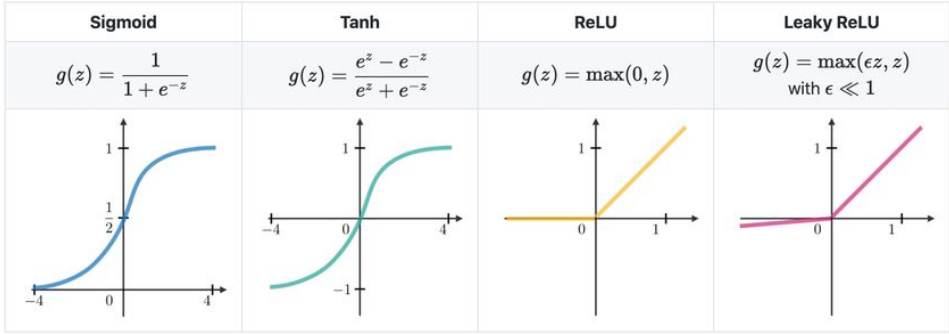
- Additional parameters in neural networks that are added to the weighted sum of inputs to the neurons.
- Bias is used for shifting the activation function towards left or right, you can compare this to y-intercept in the line equation. Allow the neural network to learn the best output given a certain input.

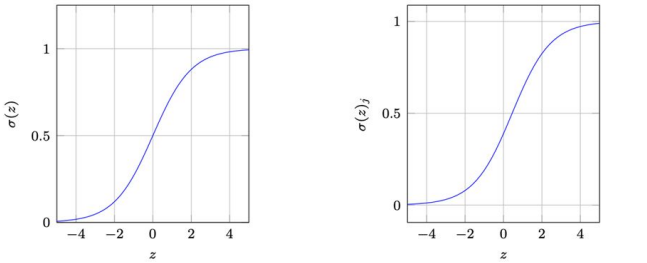


2. Tech Foundations of Generative AI: Activation Functions

Activation Function:

- It is used to introduce non-linearity in the model.
- It determines whether a neuron should be activated or not based on the input it receives.
- **Rectified Linear Unit (ReLU)**: Linear for positive values and zero for negative values, which helps with the vanishing gradient problem and speeds up training.
- **Sigmoid**: S-shaped curve, squashes the input into a range between 0 and 1.
- **Softmax**: Used in the output layer of a neural network for multi-class classification, squashes the inputs into a probability distribution.





(a) Sigmoid activation function. (b) Softmax activation function.

2. Tech Foundations of Generative AI: Weight Initialization

Weights Initialization Technique

Random Initialization, Xavier/Glorot Initialization, He Initialization, Uniform Initialization

Weight Size

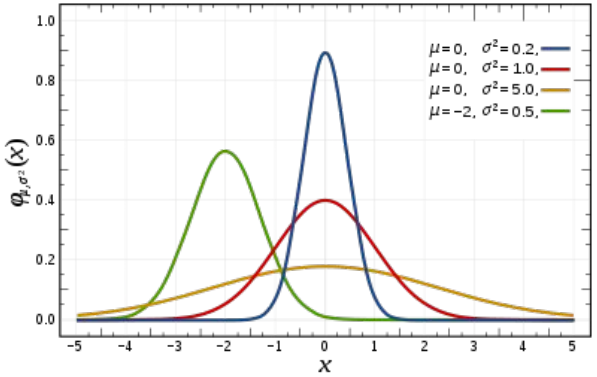
Typically represented using floating-point numbers, which can be stored using different precisions. Example: 4-bit, 8-bit, 32-bit (single precision) or 64-bit (double precision).

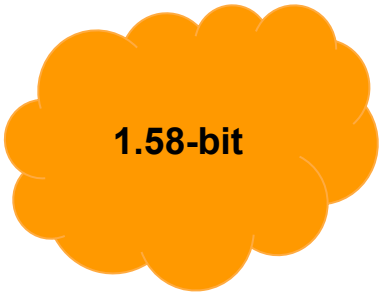
Weights Values

Random initialization from a normal distribution: with a small standard deviation, such as 0.01 or 0.1.

For example, if we initialize weights from a Normal / Gaussian distribution with mean 0 and standard deviation 0.01:

- Approximately 68% of the values will fall within one standard deviation of the mean.
- Approximately 95% of the values will fall within two standard deviations of the mean.
- Approximately 99.7% of the values will fall within three standard deviations of the mean.





2. Tech Foundations of GenAI: Language Model History

2013: Word2Vec and N-Grams

2014: RNN/LSTM

2015: Attention Mechanism

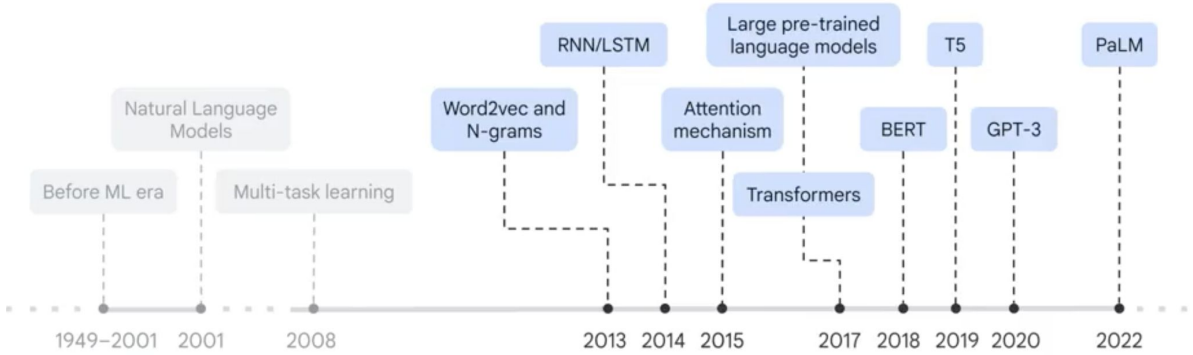
2017: Transformers → Large Pre-Trained Language Models

2018: BERT

2019: T5

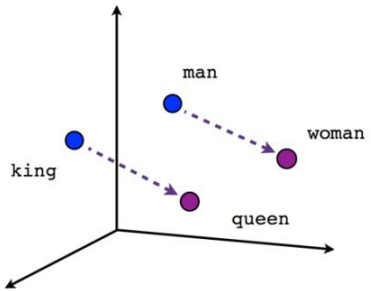
2020: GPT3

2022: PaLM

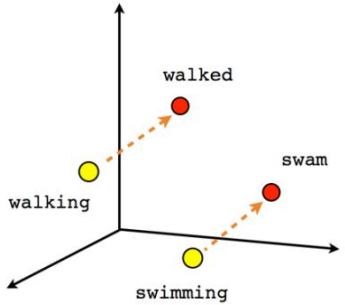


2. Tech Foundations of GenAI: Word2Vec

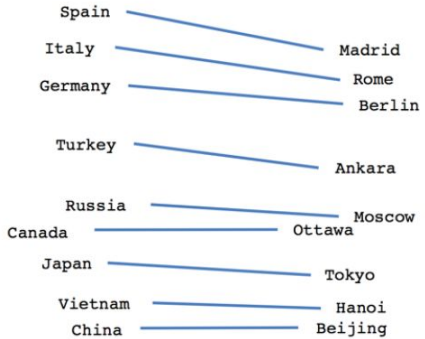
- Technique in natural language processing (NLP) used to represent words as dense, continuous-valued vectors in a high-dimensional space.
- These word embeddings capture semantic and syntactic similarities between words, enabling machines to understand and process natural language more effectively.
- Word2Vec word embedding space is a hyperparameter that needs to be specified before training the model and the choice of dimensionality depends on Task-specific Requirements, Size of Vocabulary, Available Computational Resources etc.



Male-Female



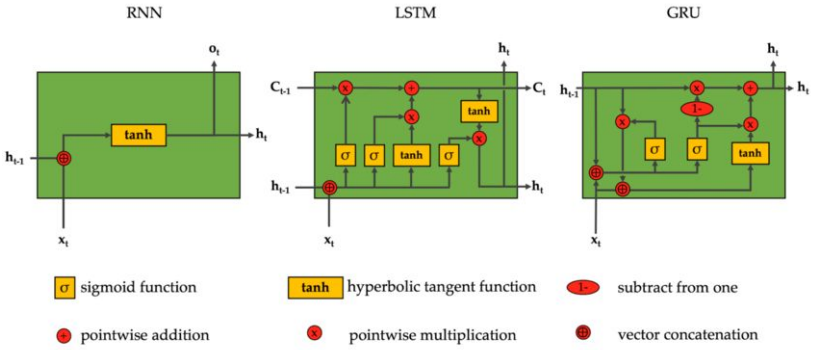
Verb tense



Country-Capital

2. Tech Foundations of GenAI: RNN → LSTM → GRU

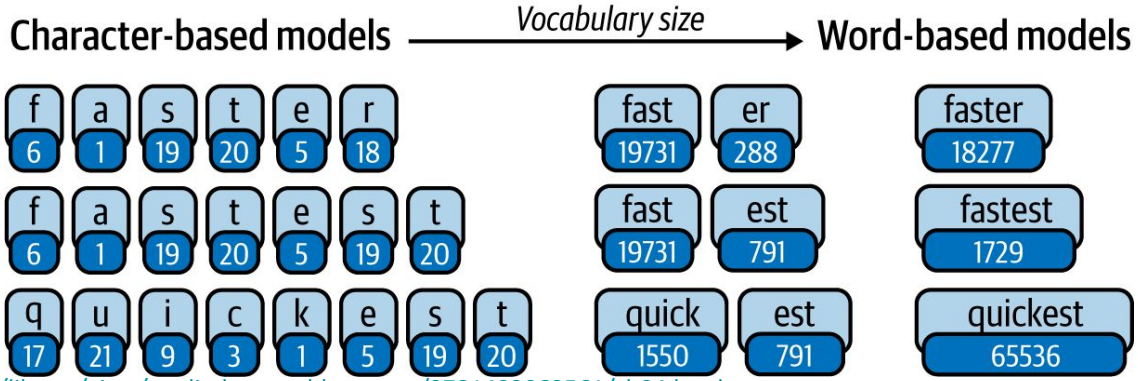
- Recurrent Neural Networks (RNNs) are a class of neural networks specifically designed to handle sequential data, such as time-series data, text, speech, and video.
- RNNs have connections that form directed cycles, allowing them to exhibit temporal dynamics and capture dependencies between inputs across time steps.
- Traditional RNNs suffer from the vanishing gradient problem, which makes it difficult to capture long-range dependencies in sequences.
- Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are two popular variants of RNNs that address this issue.



Images: https://www.researchgate.net/figure/Computation-wise-comparison-of-RNN-LSTM-and-GRU-nodes_fig4_342316332

2. Tech Foundations of GenAI: Tokenization

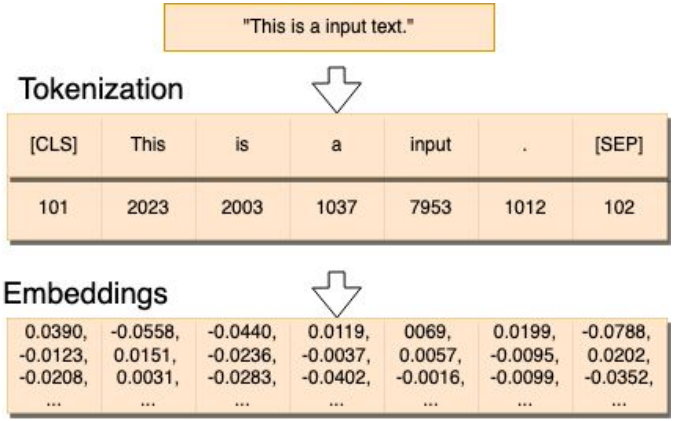
- Tokenization is the process of breaking down a sequence of text into smaller units called tokens. It lays the foundation for further analysis and processing.
- A fundamental step that breaks down unstructured text into discrete elements/numbers.
- With this numeric representation we can train the model to perform various tasks, such as classification, sentiment analysis, or language generation.
- These tokens can be words, characters, or even subwords, depending on the chosen tokenization strategy.

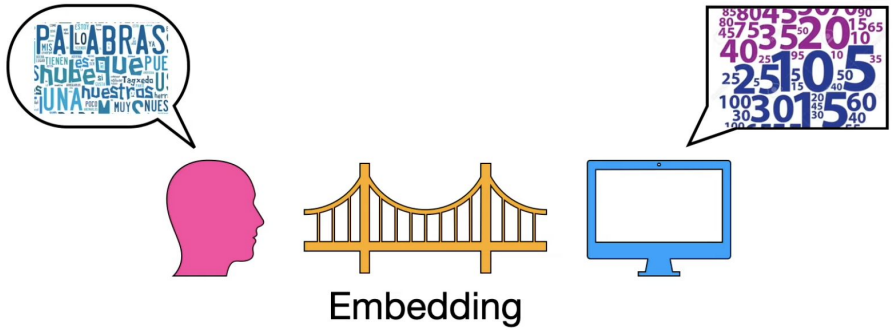


Refer: https://www.oreilly.com/library/view/applied-natural-language/9781492062561/ch04.html

2. Tech Foundations of GenAI: Tokenization

- Tokenization is cutting input data into parts that can be mapped (embedded) into a vector space.
- Special tokens are appended to the sequence e.g. class token ([CLS]) used for classification embedding in BERT transformer.
- Tokens are mapped to vectors which are passed into neural neural networks.
- Token sequence position itself is often vectorized and added to the word embeddings (positional encodings).

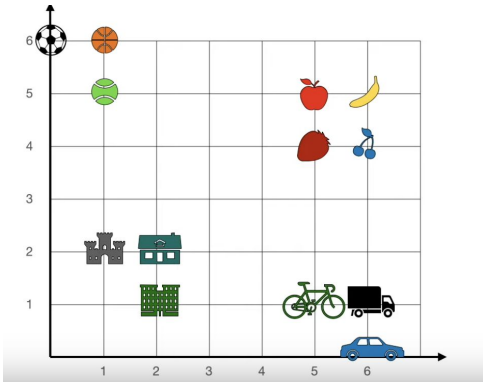




Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ

2. Tech Foundations of GenAI: Embeddings

- In the context of natural language processing (NLP), embeddings refer to dense, continuous-valued vector representations of words or tokens in a high-dimensional space.
- These embeddings capture semantic and syntactic similarities between words, enabling machine learning models to understand and process natural language more effectively.
- Word embeddings represented by a vector of real numbers, as opposed to one-hot encoding, which represents each word as a sparse binary vector.



Word	Numbers			
A	-0.82	-0.32	…	0.23
Aardvark	0.419	1.28	…	-0.06
…			…	
Zygote	-0.74	-1.02	…	1.35

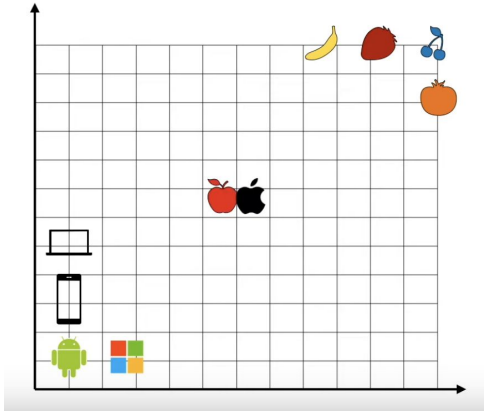
Images: https://www.youtube.com/watch?v=gaWMOYf4ri8&t=2333s

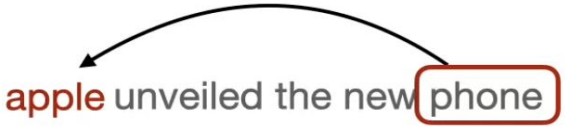
2. Tech Foundations of GenAI: Positional Encoding

- Positional encoding is a technique used in transformer-based architectures to addresses the lack of sequential order information in the transformer model by providing a way to encode the positional information of tokens within a sequence.
- Transformer architecture, the input tokens are processed independently of their positions within the sequence, which can result in the loss of sequential order information.
- Positional encoding is introduced to overcome the limitation by adding positional information to the input embeddings before feeding them into the transformer model.
- Positional encoding is usually represented as a matrix of the same dimension as the input embeddings. Each row of the matrix corresponds to a position in the sequence, and each column corresponds to a dimension in the embedding space.
- The positional encoding matrix is added element-wise to the input embeddings, effectively injecting positional information into the embeddings.
- The positional encoding matrix is computed based on a set of sinusoidal functions of different frequencies and phases.

2. Tech Foundations of GenAI: Attention

- How to ensure that apple fruit finds it place near top right and apple company finds its place near left bottom - Attention!
- Attention uses context to solve such problems
- In the first sentence “Please buy an apple and orange”, its orange which suggests that apple is fruit
- In the second sentence “Apple unveiled the new phone”, its the phone that suggests that this apple refers to apple company.

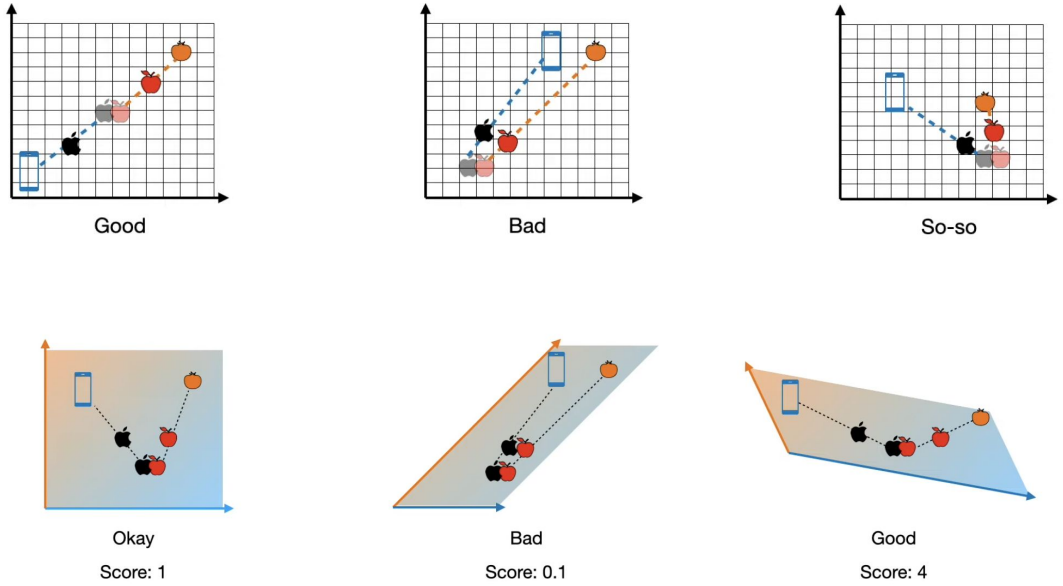




Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ

2. Tech Foundations of GenAI: Multi-head Attention

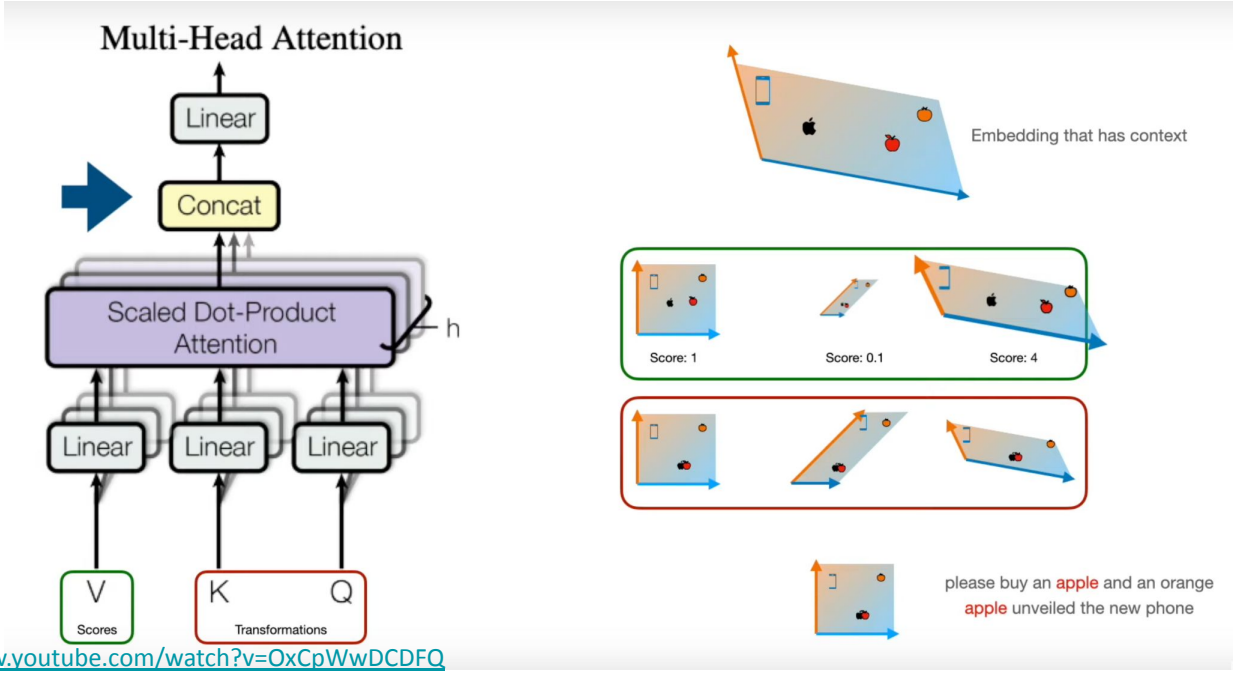
- We can have multiple embeddings instead of one and combine them to have better results. That leads to multi-head attention.
- We can build embeddings from existing embeddings using linear transformations.
- Linear transformations could be achieved through Rotation, Stretch, Sheer and their combinations
- Good embeddings get good score and bad gets low score



Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ

2. Tech Foundations of GenAI: Multi-head Attention

As we train the neural network, we train these embeddings and train their score and that is where Key, Query and Value Matrices comes in.



Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ

2. Tech Foundations of GenAI: Attention Maths

Similarity Measures

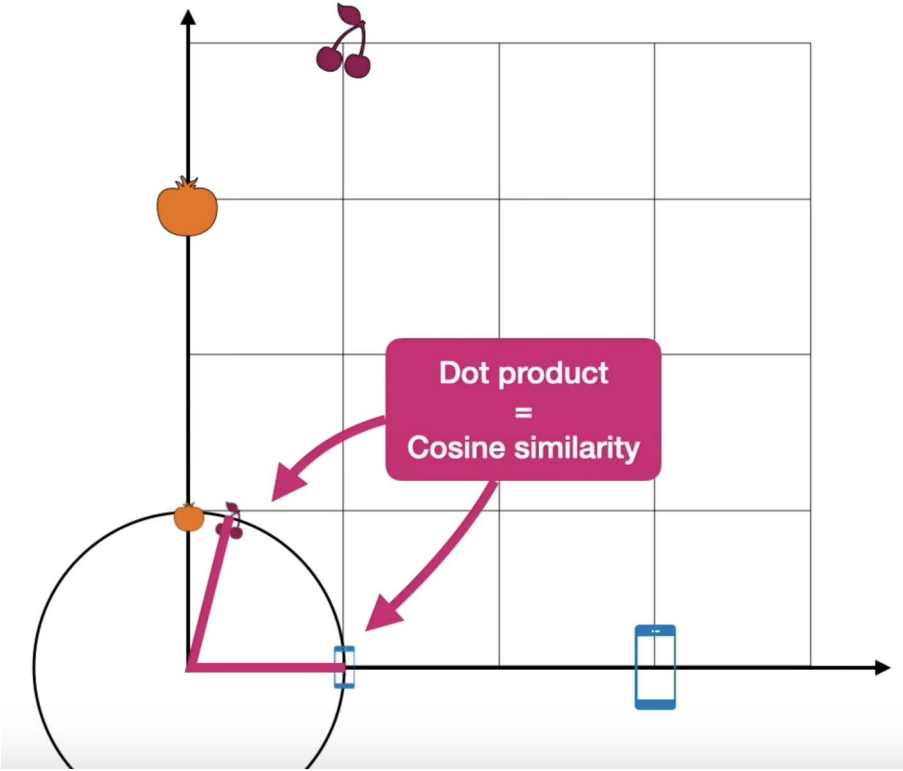
- Dot Product
- Cosine Similarity

Both are same when we scale down such that all vectors have unit length i.e they are same upto a scalar.

- Scaled Dot Product

Divide the dot product by square root of the length of the vector.

To calculate attention, we generally use scaled dot product.



Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ

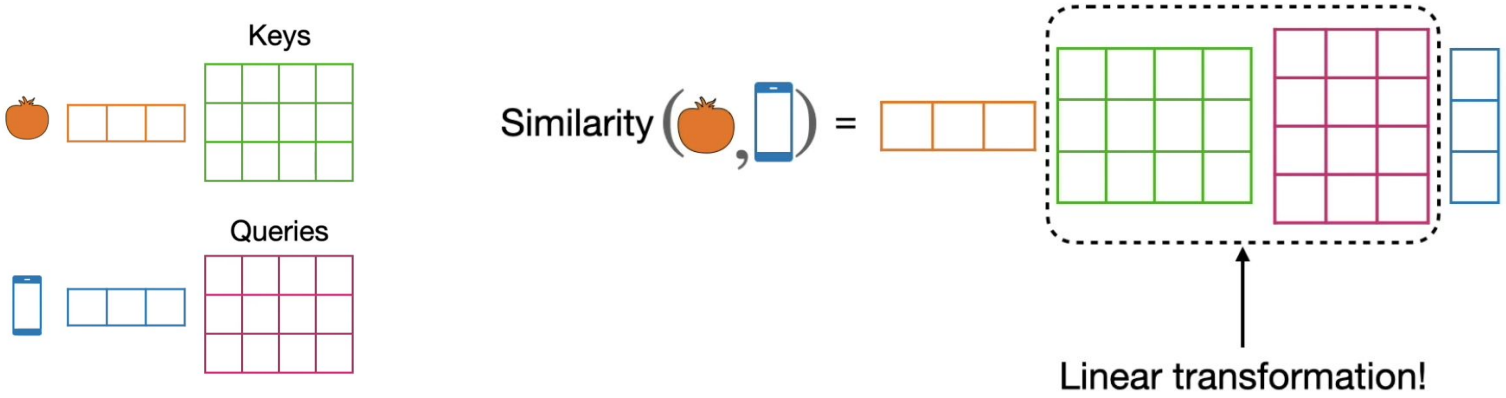
2. Tech Foundations of GenAI: Attention Mechanism

Steps to compute Attention

1. **Similarity Calculation:** The model compares the query with each Key in the encoded input sequence. This can be done using a dot product or a more complex function.
2. **Attention Weights:** The similarity scores are normalized to get weights between 0 and 1. These weights indicate how much attention the model should pay to each element in the input sequence.
3. **Context Vector:** The model creates a context vector by taking a weighted sum of the Values from the input sequence, using the attention weights as the weights. This context vector summarizes the most relevant information from the input based on the current query.
4. **Decoder Update:** The decoder uses the context vector along with other information to generate the next output element (like a word in a translation).

2. Tech Foundations of GenAI: Attention Mechanism

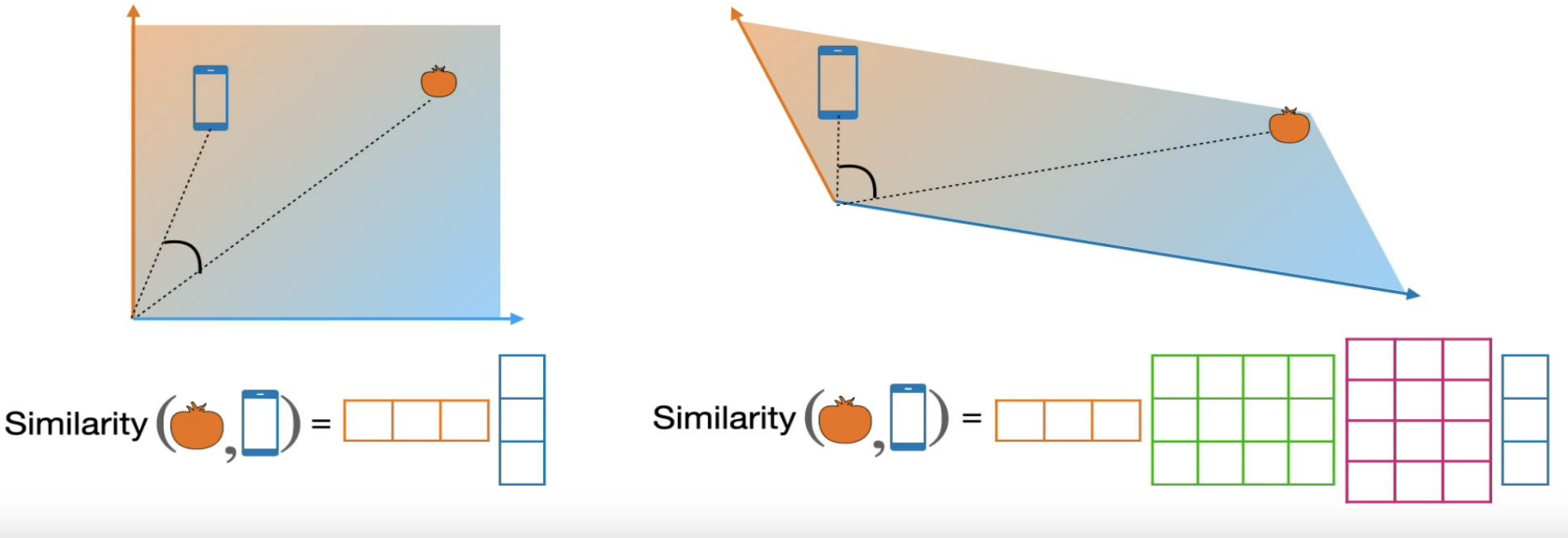
- It enables LLMs to focus on different parts of input data with varying degrees of importance, allowing for more effective learning and processing of sequential data such as text or time-series data.
- The attention mechanism allows the model to dynamically weigh the importance of each input token based on its relevance to the current context, rather than treating all tokens equally.
- **Key Components of attention:** Keys, Query and Values
- Keys and Query Matrix helps us find good embeddings where we can do attention and find really good information and all this is achieved using linear transformation. Keys and Query modify the embeddings.



Images: https://www.youtube.com/watch?v=UPtG_38Oq8o

2. Tech Foundations of GenAI: Attention Mechanism

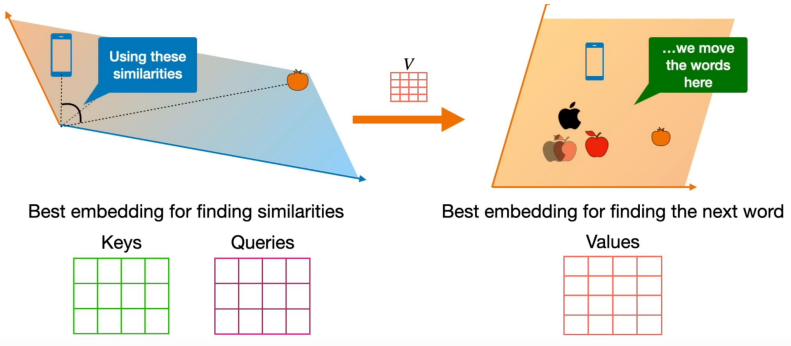
- Similarity on a transformed embeddings. Keys and Queries matrix turn the embeddings into one that is best for calculating similarities.
- Keys and Queries Matrix turn the embeddings into that is best for calculator similarities.

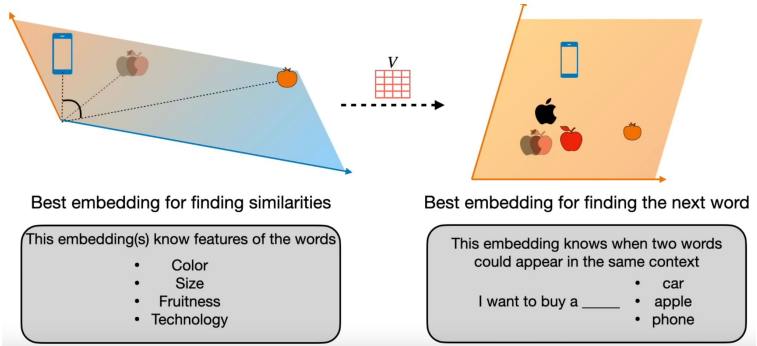


Images: https://www.youtube.com/watch?v=UPtG_38Oq8o

2. Tech Foundations of GenAI: Attention Mechanism

- On the left is the best embeddings for finding the similarities and on the right is the best embeddings for finding the next word. Embeddings on the left is found by Keys and Queries Matrices and the embeddings on the right is found by the values.
- Note: Transformer is best at finding the next word in the sentence and keeps finding next word until it builds required text.
- Keys and Queries matrices capture the granular features of the word embeddings. That how keys and Queries matrix give the best embeddings to apply attention.
- The Value matrix takes the embeddings on the left and multiplies every vector to get the embeddings on the right. Multiplication with Value matrix yields another transformation.

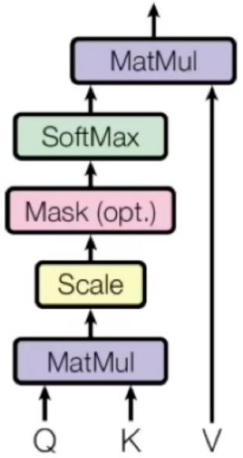




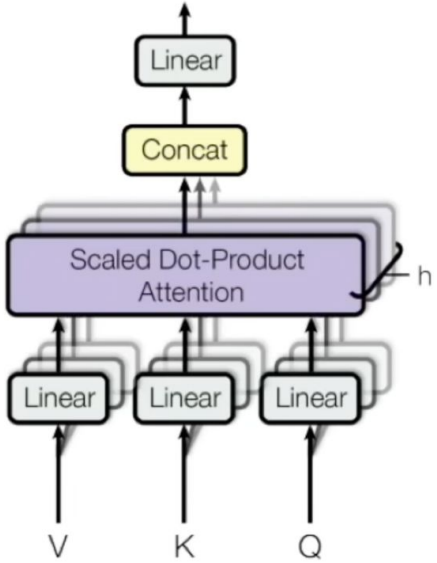
Images: https://www.youtube.com/watch?v=UPtG_38Oq8o

2. Tech Foundations of GenAI: Attention

Scaled Dot-Product Attention



Multi-Head Attention

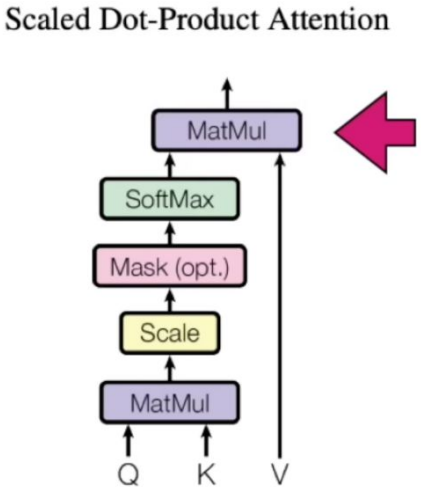


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

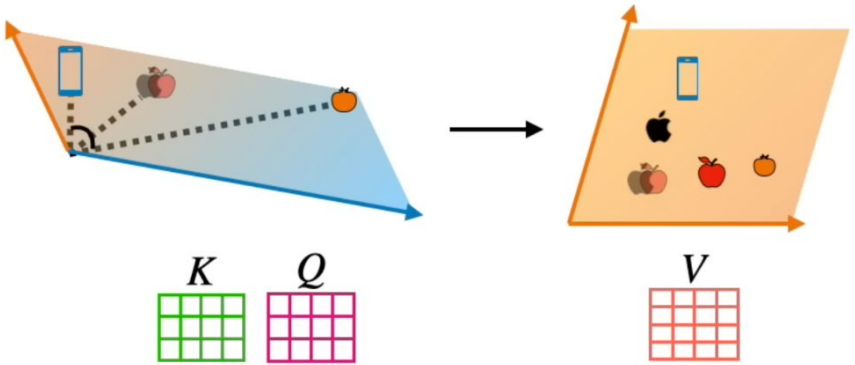
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ&t=1s

2. Tech Foundations of GenAI: Self Attention

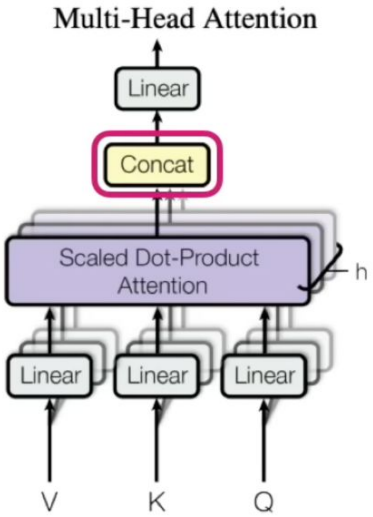


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

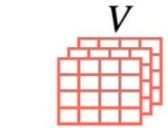


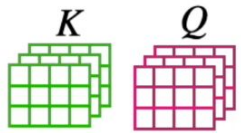
Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ&t=1s

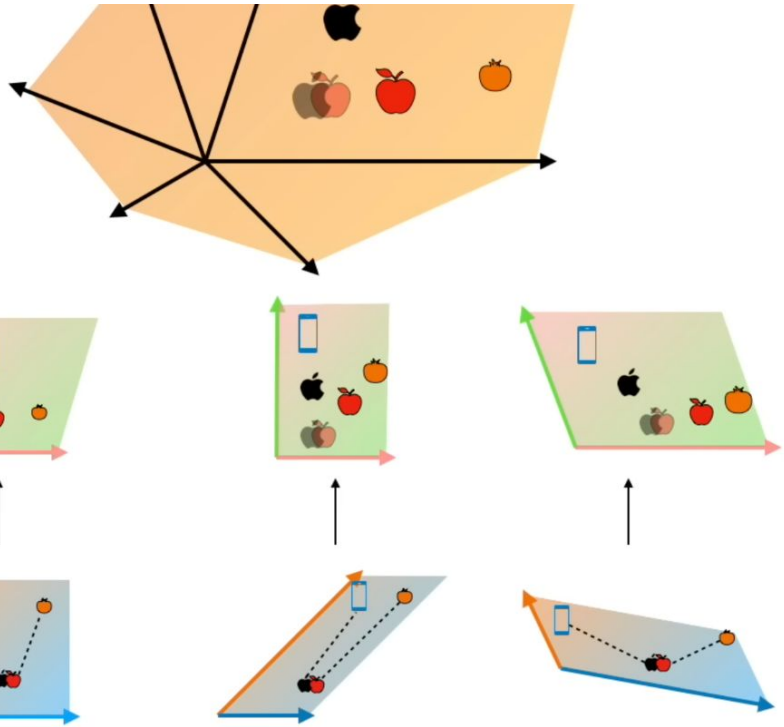
2. Tech Foundations of GenAI: Multi-head Attention



$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

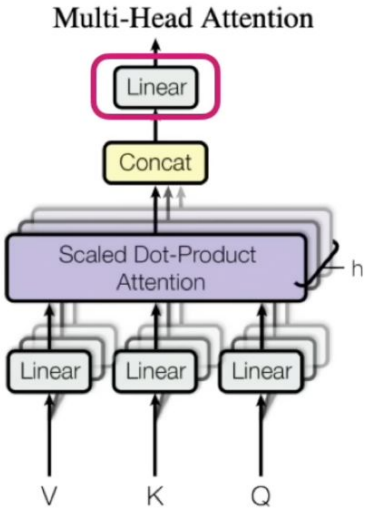




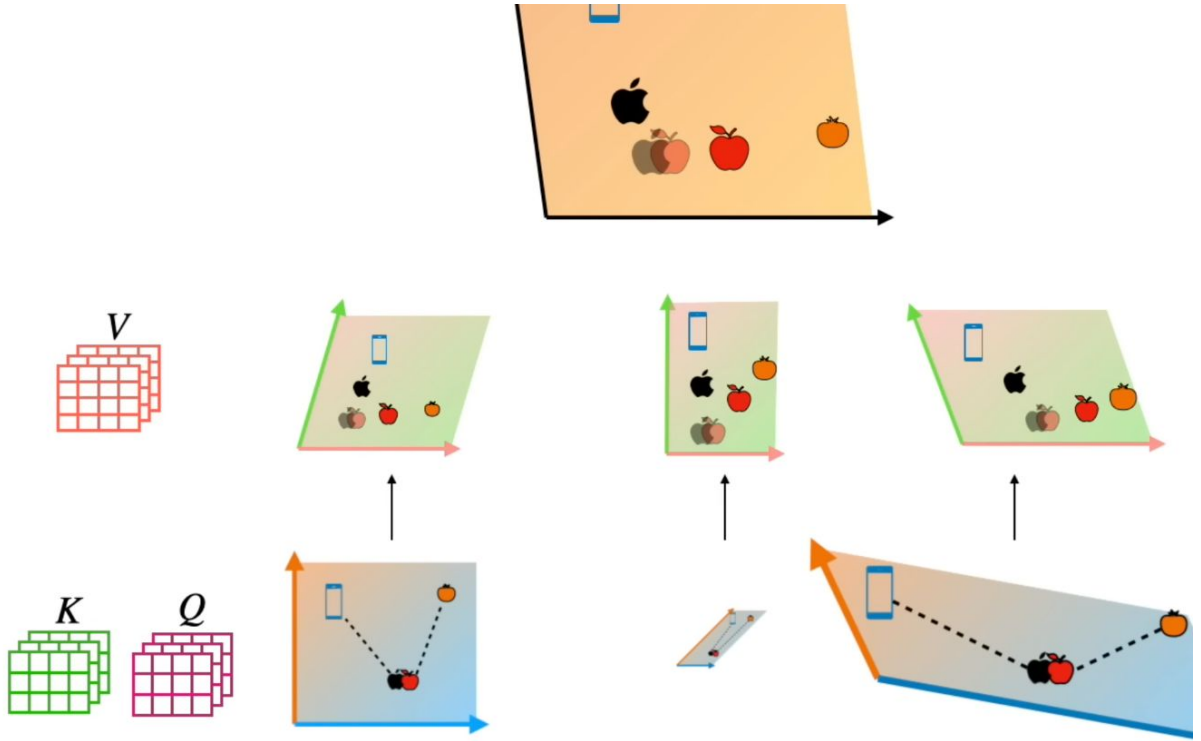


Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ&t=1s

2. Tech Foundations of GenAI: Multi-head Attention



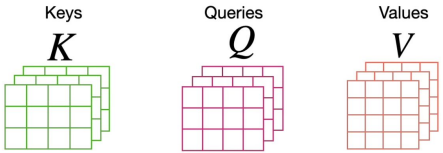
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

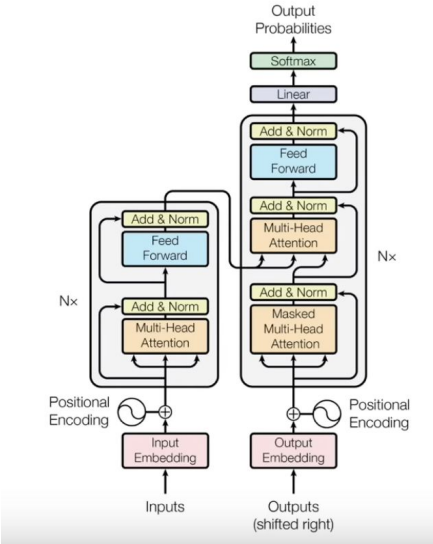


Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ&t=1s

2. Tech Foundations of GenAI: Query, Key and Value Matrices

- The K, Q and V matrices are derived from the input embeddings and trained using transformer model
- These matrices are used to compute attention scores, which determine the importance of different parts of the input sequence.
- The input embeddings or hidden states are linearly projected into three separate sets of vectors: query vectors, key vectors, and value vectors; implemented using learnable weight matrices.
- Each set of vectors (queries, keys, and values) has its own weight matrices, which are learned during the training of the model.





Images: https://www.youtube.com/watch?v=OxCpWwDCDFQ&t=1s

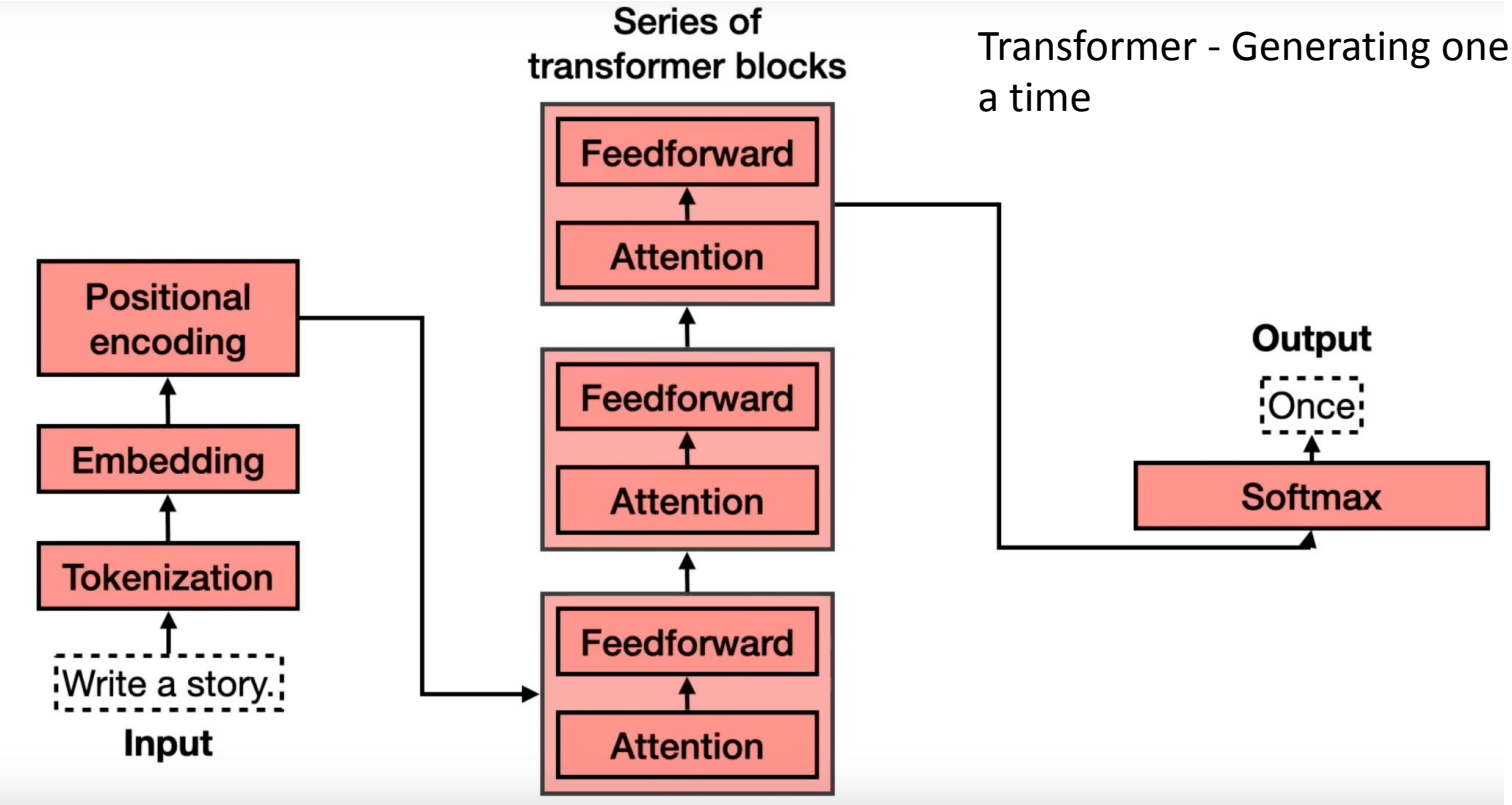
2. Tech Foundations of GenAI: Query, Key and Value Matrices

- The query matrix Q, key matrix K, and value matrix V are obtained by multiplying the input embeddings by their respective weight matrices.
- Here, Input represents the input embeddings, and W_Q, W_K, and W_V are the weight matrices for queries, keys, and values, respectively.

$$Q = \text{Input} \times W_Q$$
$$K = \text{Input} \times W_K$$
$$V = \text{Input} \times W_V$$

- The query, key, and value matrices are used to compute attention scores between tokens in the input sequence.

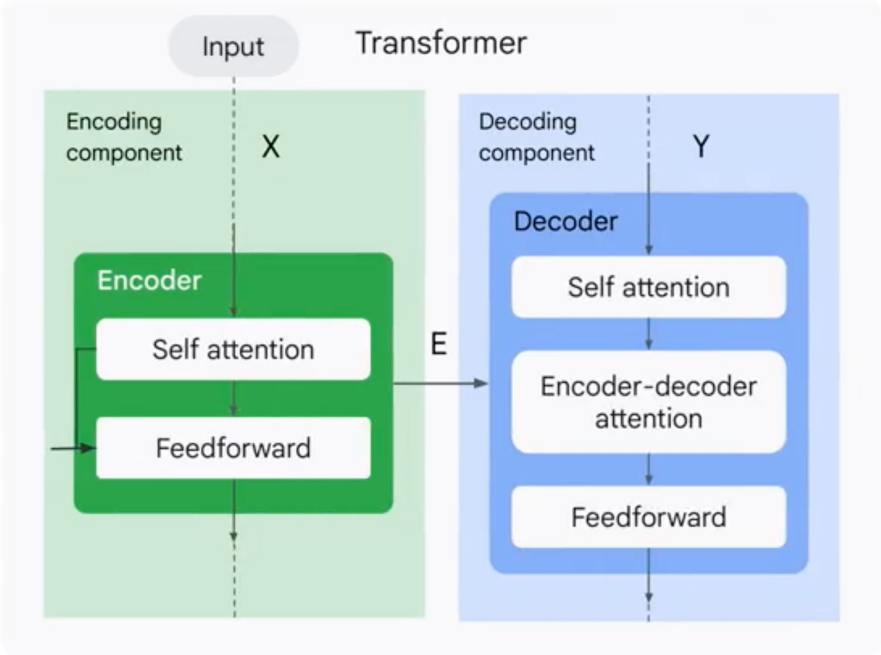
2. Tech Foundations of GenAI: Transformer



Images: https://www.youtube.com/watch?v=qaWMOYf4ri8&t=2337s

2. Tech Foundations of GenAI: Transformer

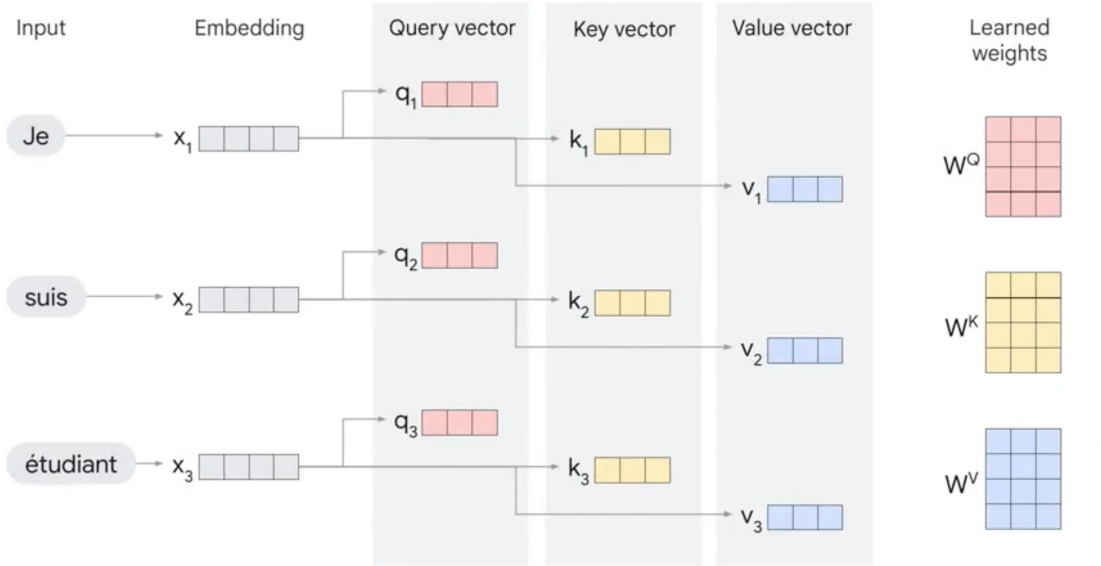
- A transformer is a encoder-decoder model that uses the attention mechanism. It takes advantage of parallelisation and its built with Attention Mechanism at its core.
- The Transformer models consist of encoder and decoder. The encoder encodes the input sequence and passes it to the decoder and the decoder decodes a representation for a relevant task.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

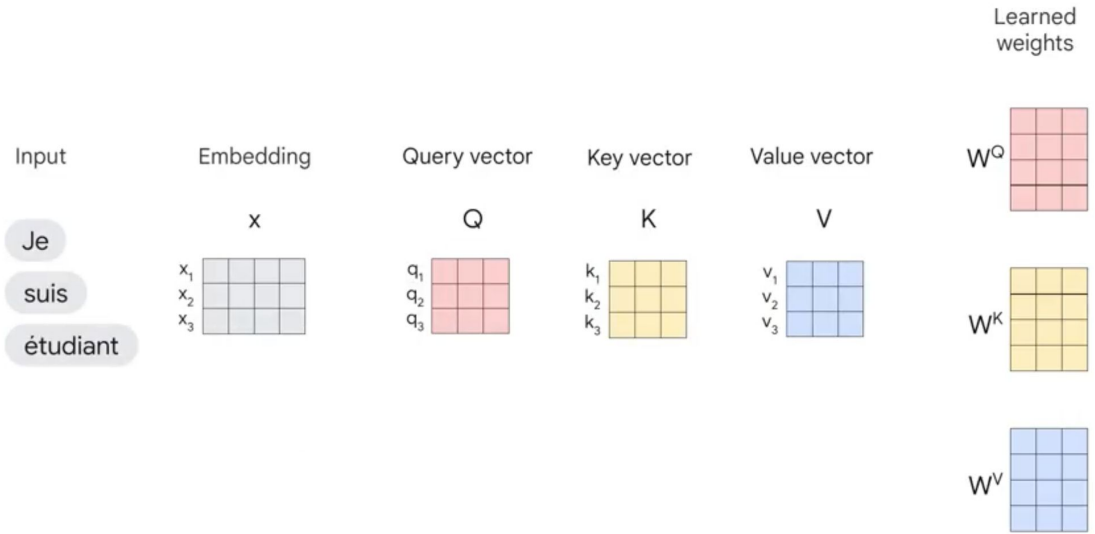
In the self attention layer, the input embedding is broken up into query, key and values vectors. These vectors are computed using weights that the transformer learns during the training process.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

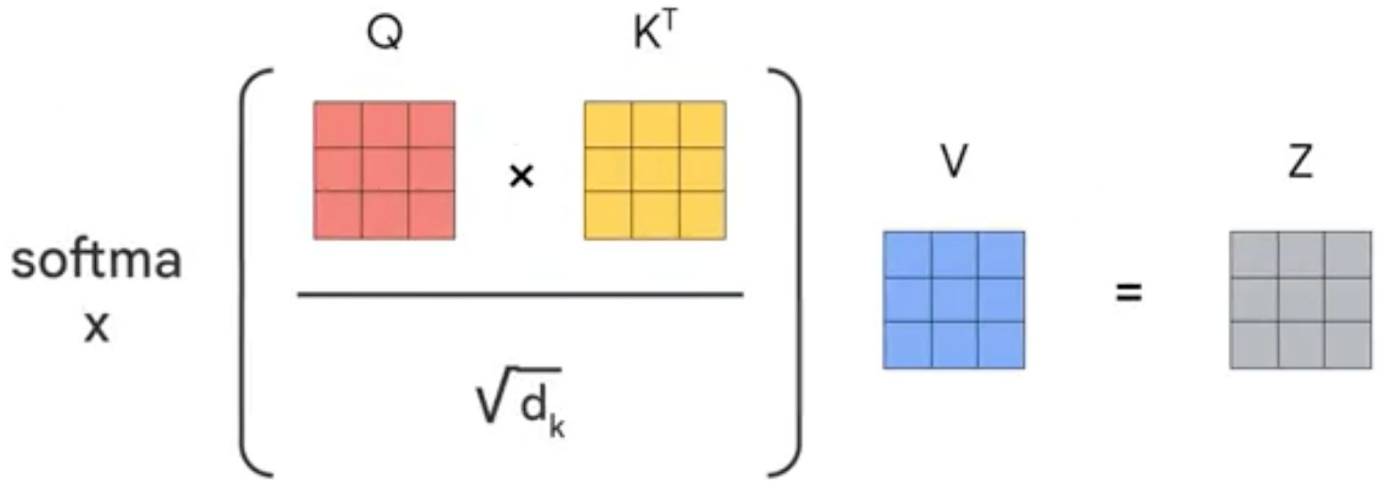
All of these computations happen in parallel in the model in the form of matrix computations



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

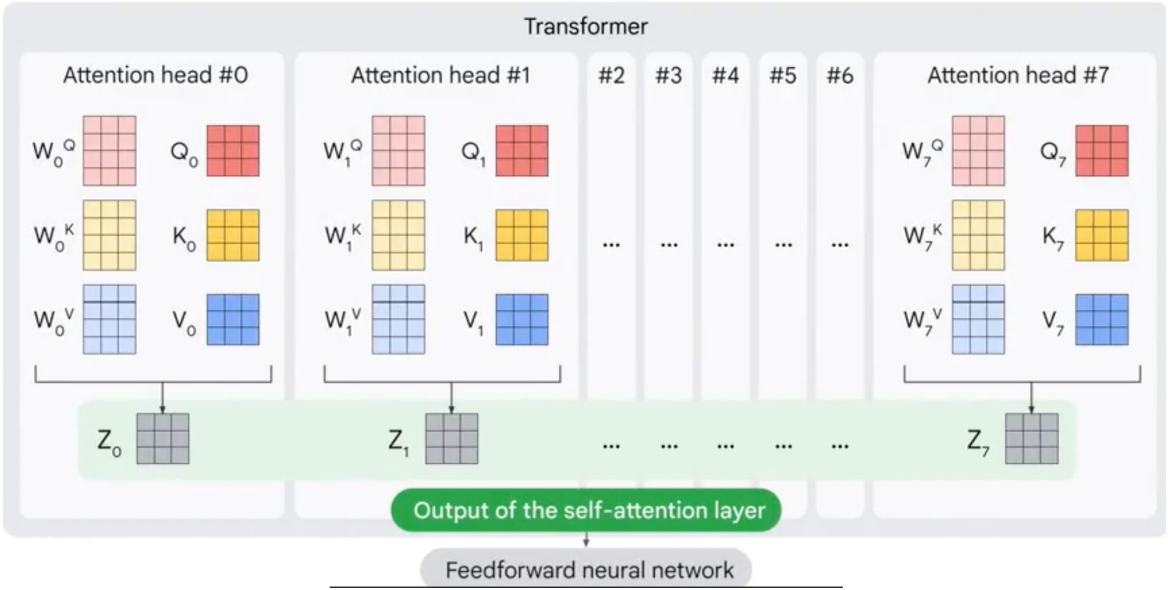
Once we have the query, key and value vectors, the next step is to multiply each value vector by by the softmax score in preparation to sum them up.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

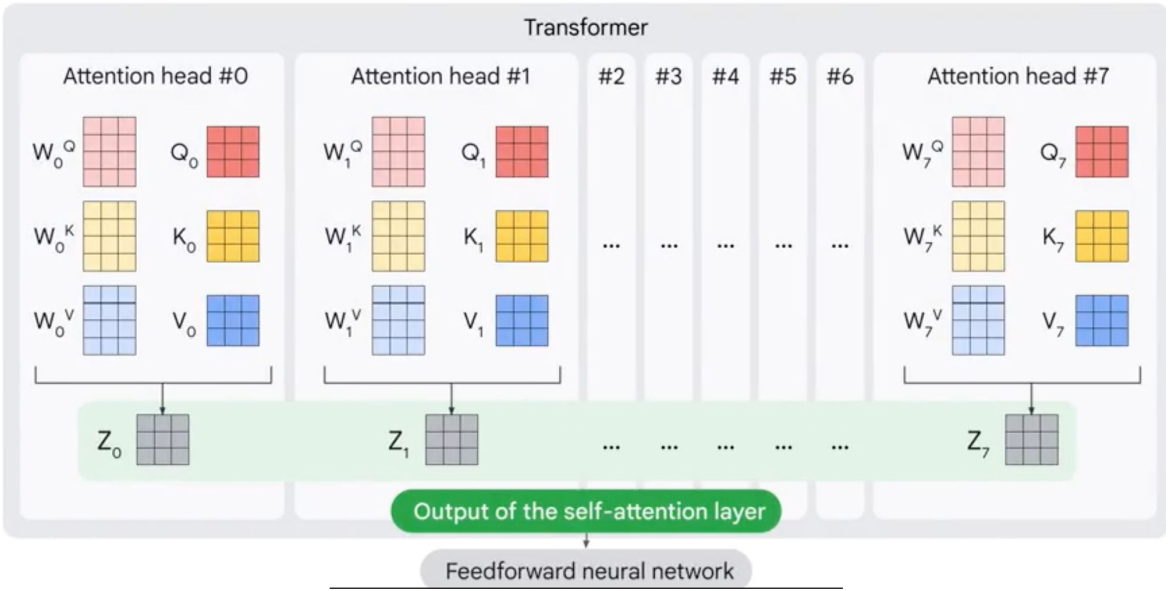
Next we have to sum the weighted value vectors, which produces the the output of self-attention layer at this position for the first word.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

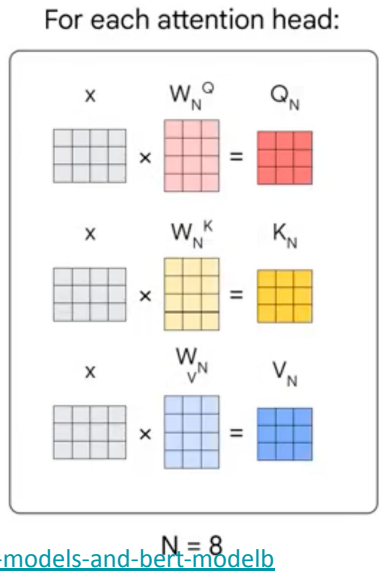
Next we have to sum the weighted value vectors, which produces the the output of self-attention layer at this position for the first word.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

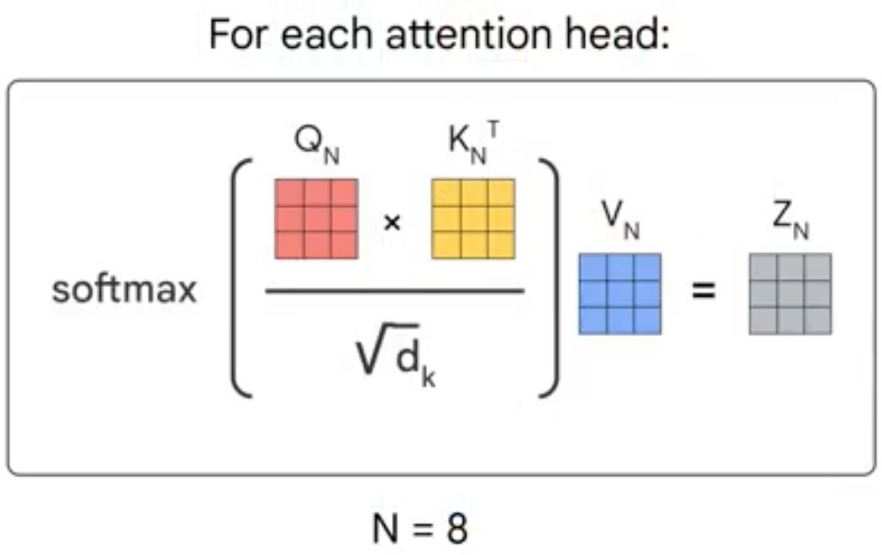
Perform multi headed attention (8 times in this case), and multiply the embedded words with respective weighted matrices.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

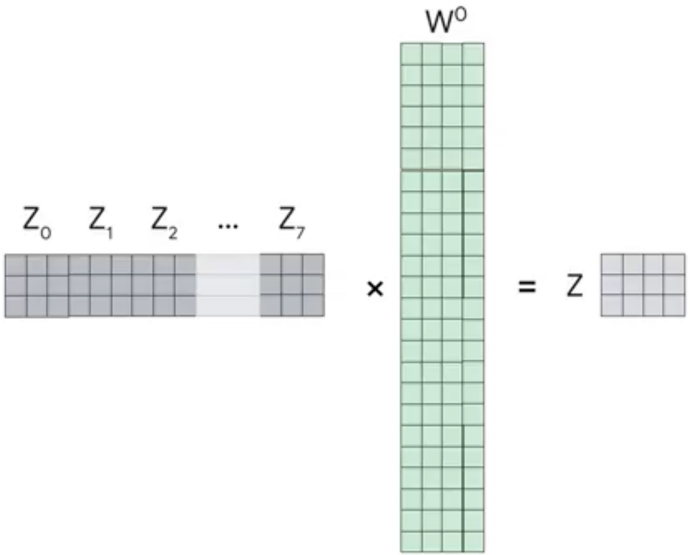
Calculate the attention using the resulting QKV matrices



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

Concatenate the matrices to produce the output matrix which is the same dimension as the final matrix that this layer initially got.



Refer: https://www.coursera.org/learn/transformer-models-and-bert-modelb

2. Tech Foundations of GenAI: Transformer

There are multiple variations of transformers

- Encoder-Decoder (BART)
- Decoder only (GPT3, GPT4)
- Encoder Only (BERT)

BERT (Bidirectional Encoder Representations from Transformer)

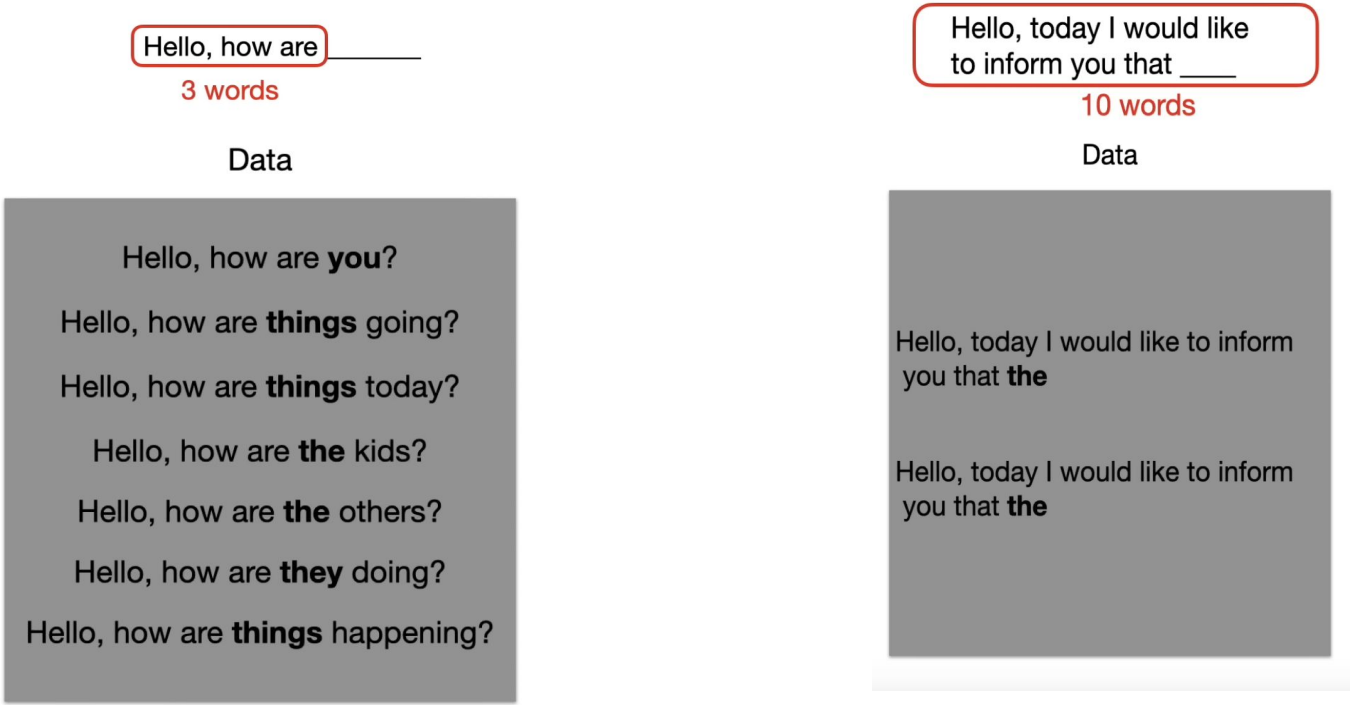
- It is Encoder only model
- It is one of the trained transformer models
- It powers google search

Originally BERT was trained on two different tasks

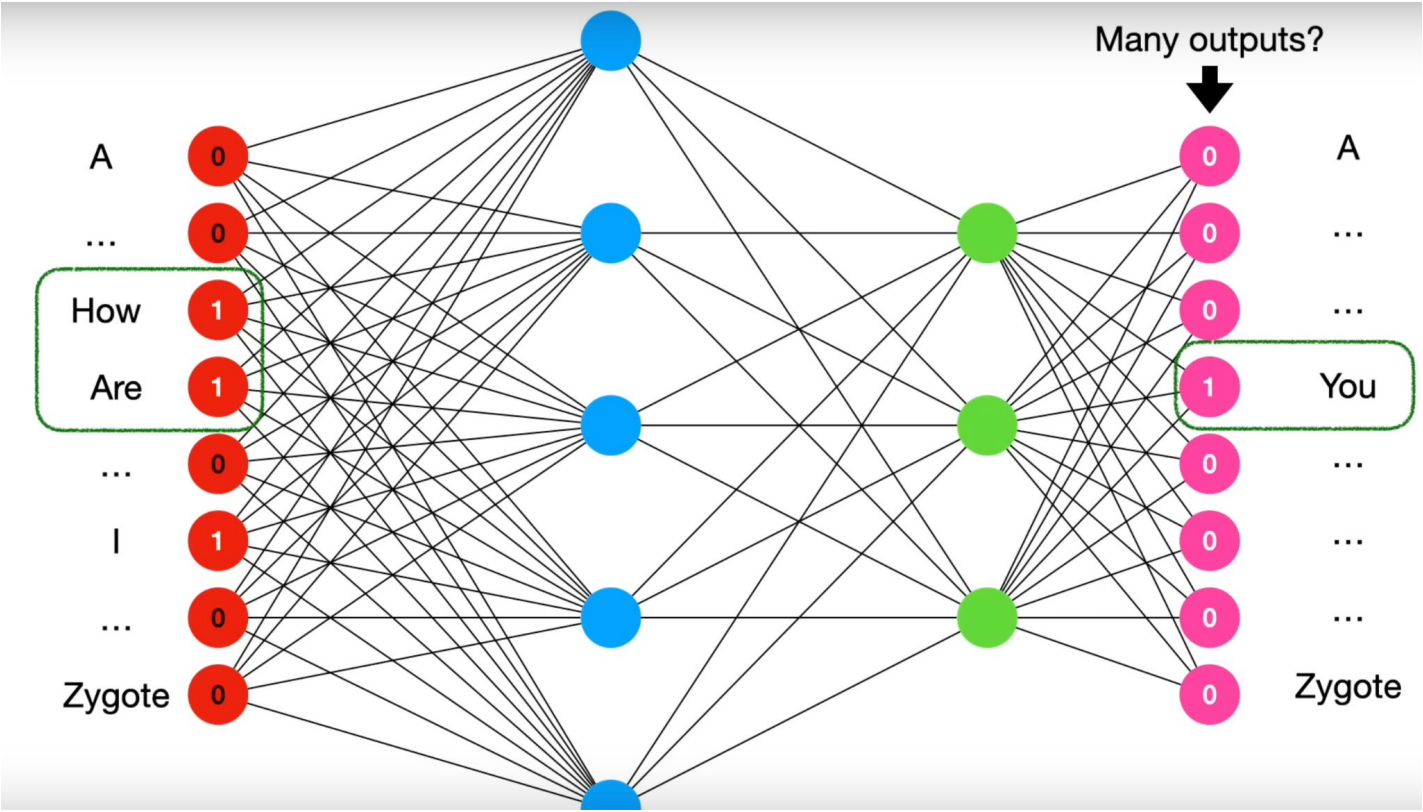
- **Masked Language Modeling (MLM)**: Here the some percentage of input words are masked and the model is trained to predict the masked word
- **Next Sentence Prediction (NSP)**: Here BERT aims to Learn the relationships between sentences and predicts the next sentence given the first one.

2. Tech Foundations of GenAI: n-grams

- N-grams are contiguous sequences of N items (tokens) from a given text or sequence of data.
- N-grams are used to capture the local context and syntactic structures of text data.

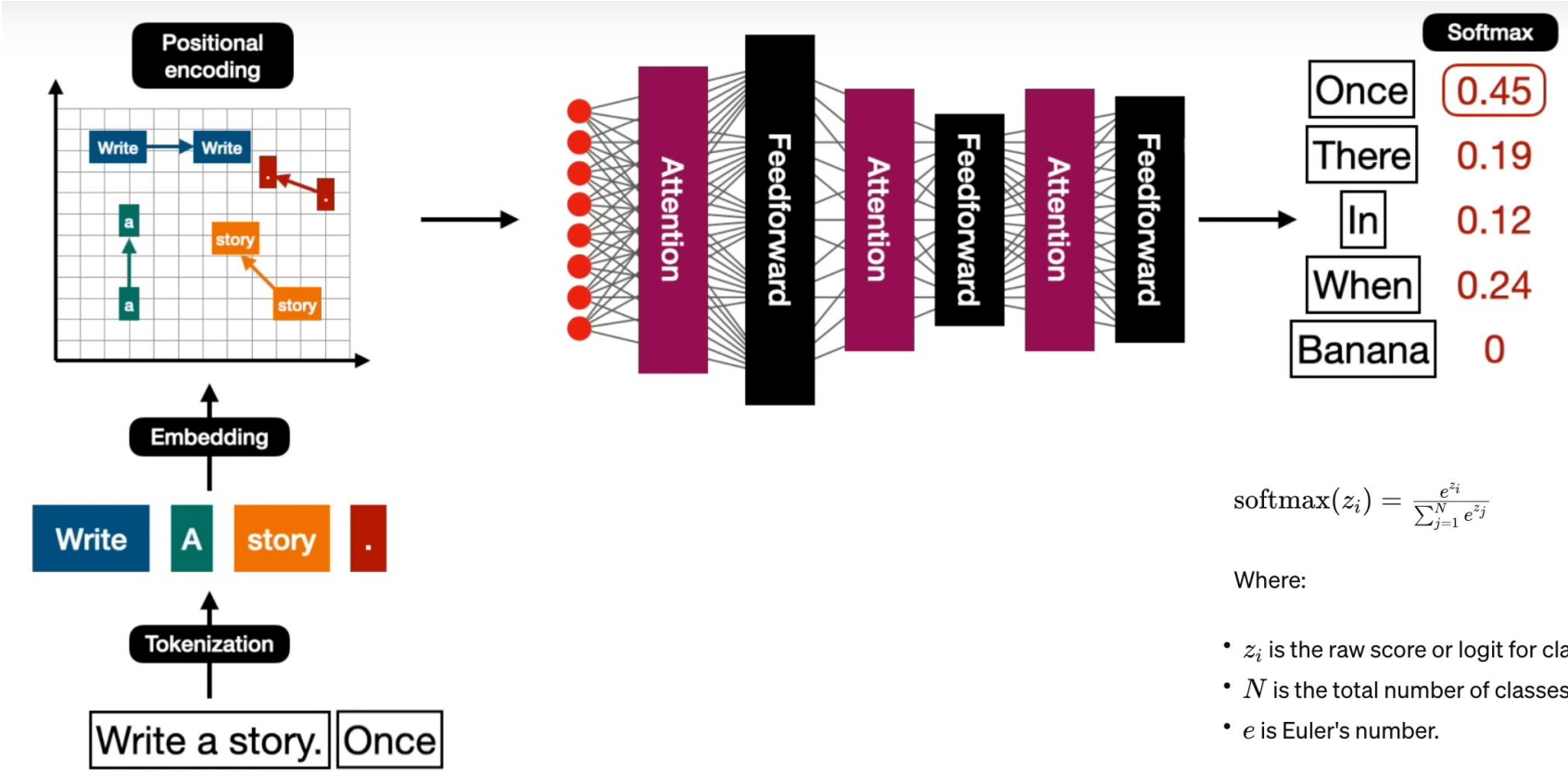


2. Tech Foundations of GenAI: Transformer



Images: https://www.youtube.com/watch?v=qaWMOYf4ri8&t=2337s

2. Tech Foundations of GenAI: Transformer



$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}$$

Where:

- z_i is the raw score or logit for class i.
- N is the total number of classes.
- e is Euler's number.

Images: https://www.youtube.com/watch?v=qaWMOYf4ri8&t=2337s

2. Tech Foundations of GenAI: Self Supervised Learning

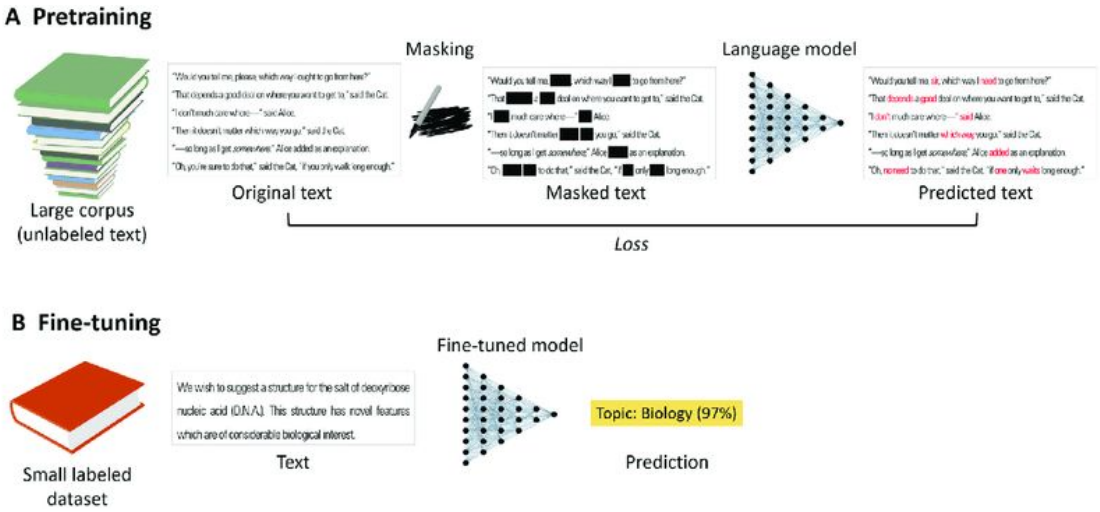
Self-supervised learning involves training a model to predict certain aspects of the input text **without relying on labeled data or explicit supervision**

Types of self-supervised learning used for training large language models

- **Masked Language Modeling (MLM)**: Certain tokens in the input text are masked, and the model is trained to predict the masked tokens based on the surrounding context. The model learns representations of words and phrases that capture their semantic and syntactic relationships.
- **Next Sentence Prediction (NSP)**: Model is trained to predict whether a given pair of sentences are consecutive or not. The model learns the relationships between sentences enabling it to capture higher-level semantic information beyond just individual words and phrases.

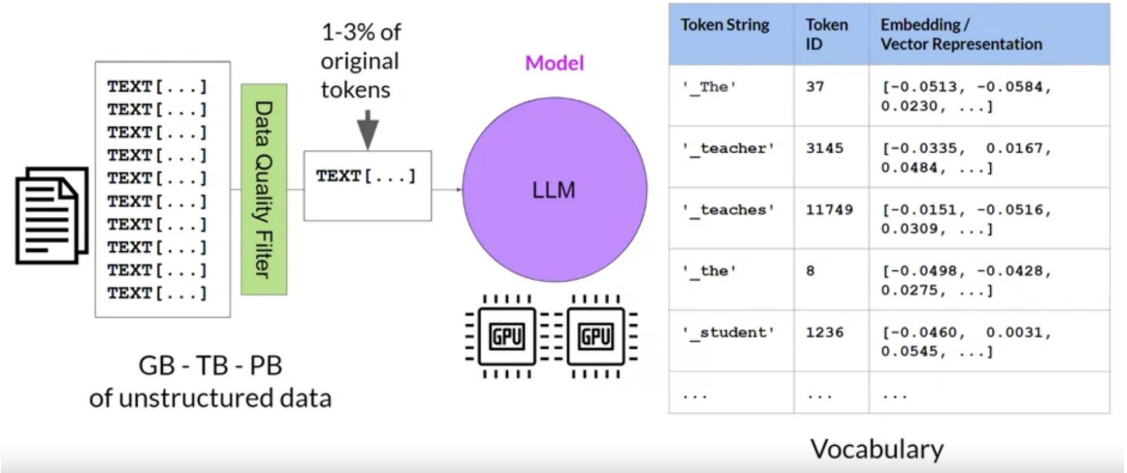
2. Tech Foundations of GenAI: Self Supervised Learning

- Models can learn rich, general-purpose representations of language that can be fine-tuned for specific downstream tasks, such as text classification, question answering, or language generation.
- Self-supervised learning used to train large language models like **GPT and BERT** (Bidirectional Encoder Representations from Transformers).



2. Tech Foundations of GenAI: Pre-Training

- LLMs encode a deep statistical representation of language. This understanding is developed during the models pre-training phase when the model learns from vast amounts of unstructured textual data.
- In this self-supervised learning step, the model internalizes the patterns and structures present in the language.



Token String	Token ID	Embedding / Vector Representation
'_The'	37	[-0.0513, -0.0584, 0.0230, ...]
'_teacher'	3145	[-0.0335, 0.0167, 0.0484, ...]
'_teaches'	11749	[-0.0151, -0.0516, 0.0309, ...]
'_the'	8	[-0.0498, -0.0428, 0.0275, ...]
'_student'	1236	[-0.0460, 0.0031, 0.0545, ...]
...	...	...

Vocabulary

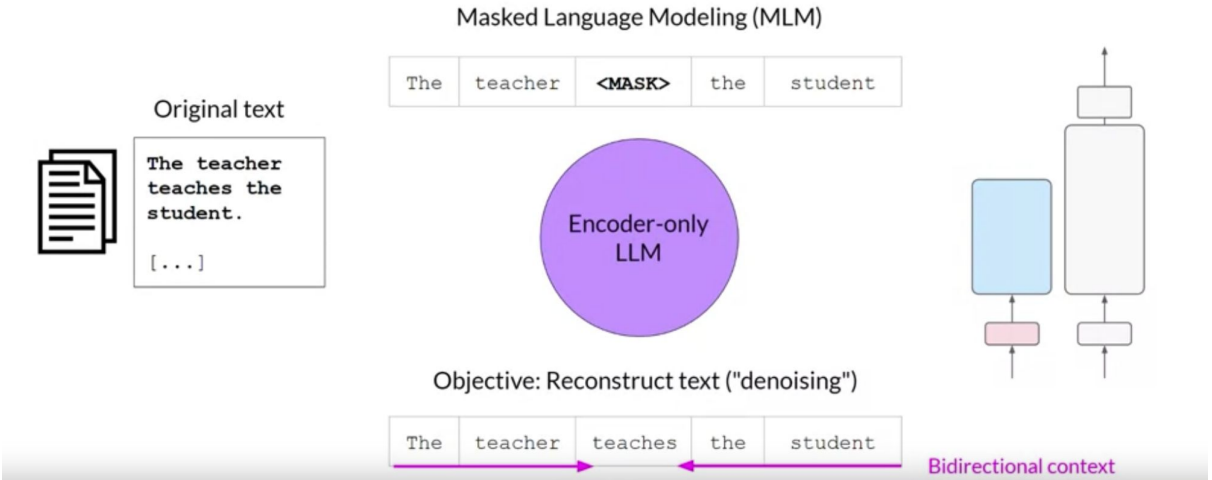
Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

2. Tech Foundations of GenAI: Types of Transformer Model

- There are three variance of the transformer model;
  - Encoder-only
  - Encoder-decoder models, and
  - Decode-only.
- **Encoder-only models** also known as **Autoencoding** models, and they are pre-trained using masked language modeling. Here, tokens in the input sequence or randomly mask, and the training objective is to predict the mask tokens in order to reconstruct the original sentence. This is also called a denoising objective.
- Autoencoding models have bi-directional representations of the input sequence, meaning that the model has an understanding of the full context of a token and not just of the words that come before.

2. Tech Foundations of GenAI: Pre-Training

- We can use them to carry out sentence classification tasks, for example, sentiment analysis.
- E.g. BERT and RoBERTa



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

2. Tech Foundations of GenAI: Pre-Training

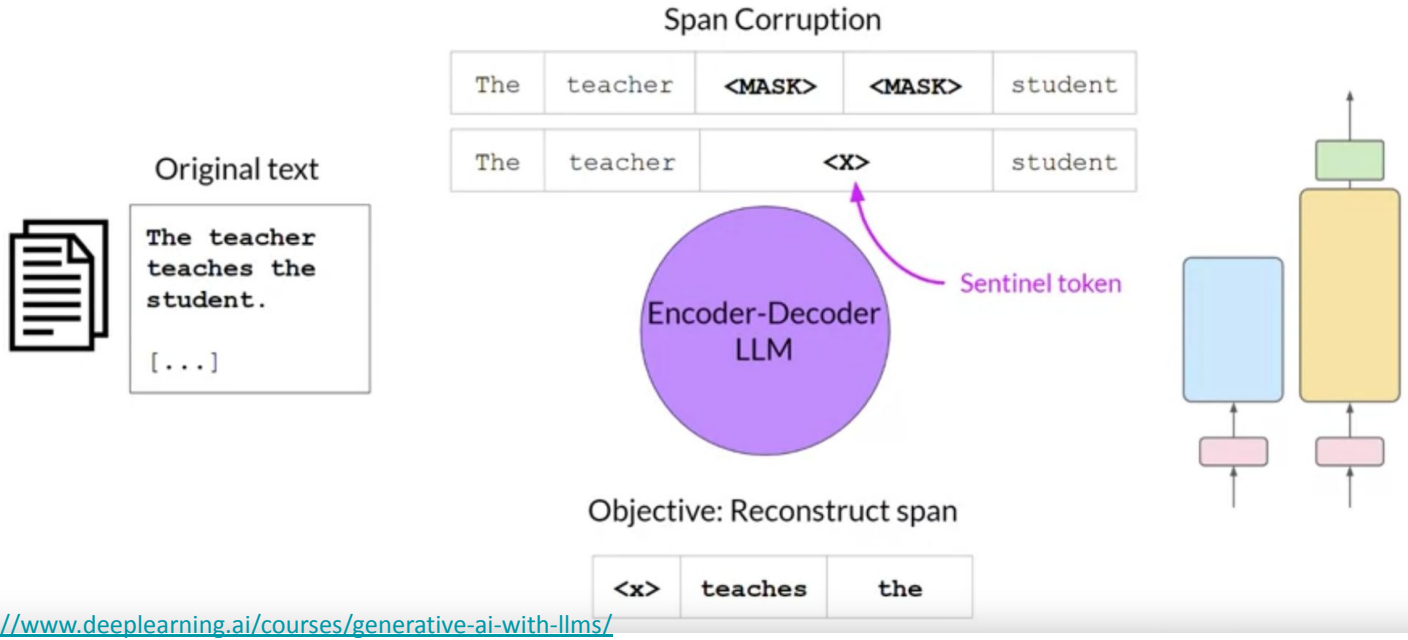
- **Decoder-only** or autoregressive models, which are pre-trained using causal language modeling. Here, the training objective is to predict the next token based on the previous sequence of tokens.
- Decoder-based autoregressive models, mask the input sequence and can only see the input tokens leading up to the token in question. The model has no knowledge of the end of the sentence.
- The model iterates over the input sequence one by one to predict the following token, this means that the context is unidirectional.
- Models of this type make use of the decoder component off the original architecture without the encoder. Decoder-only models are often used for text generation.
- Well known examples of decoder-based autoregressive models are GPT and BLOOM.

2. Tech Foundations of GenAI: Pre-Training

- **Sequence-to-sequence model** uses both the **encoder and decoder** parts off the original transformer architecture.
- Model pre-trains the encoder using span corruption, i.e. masks random sequences of input tokens. Those mass sequences are then replaced with a unique Sentinel token (x).
- Sentinel tokens are special tokens added to the vocabulary, but do not correspond to any actual word from the input text. The decoder is then tasked with reconstructing the mask token sequences auto-regressively.

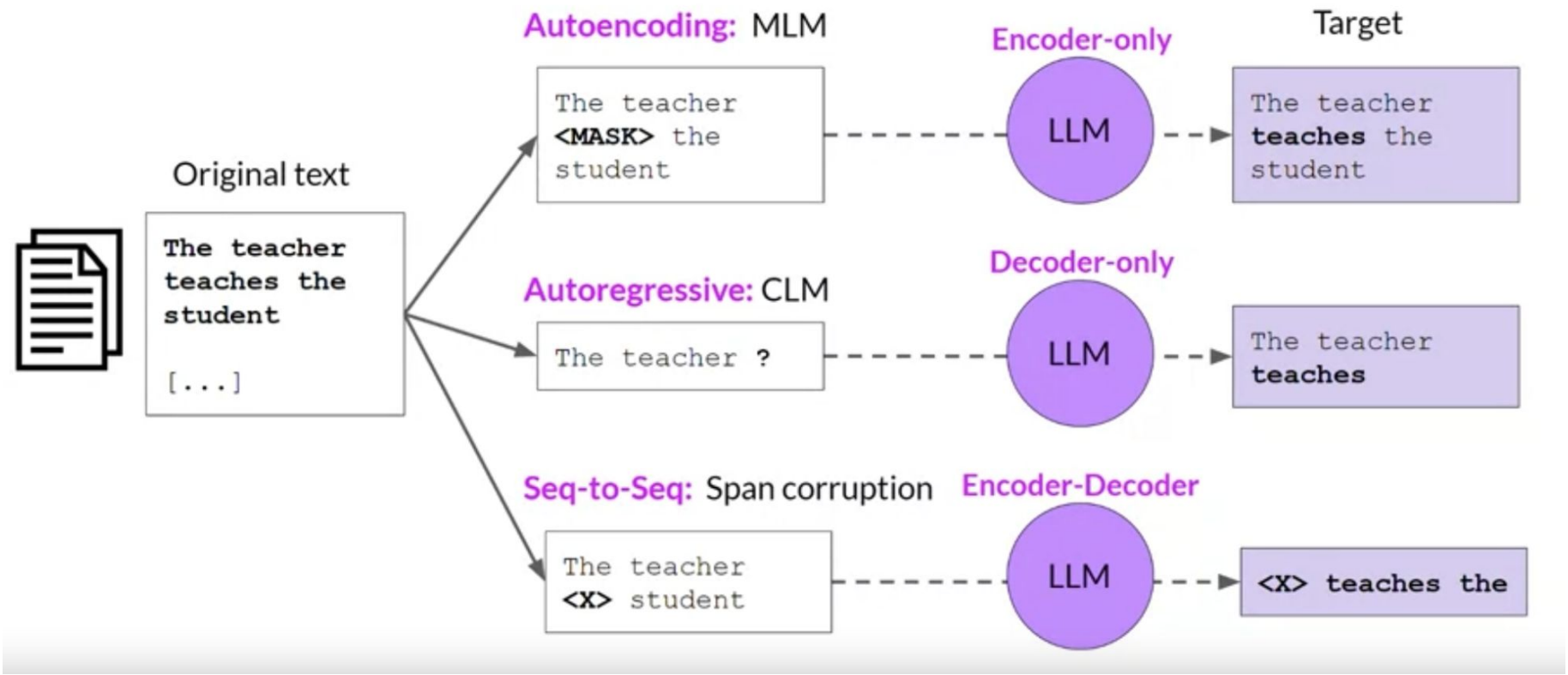
2. Tech Foundations of GenAI: Pre-Training

- Sequence-to-sequence models for translation, summarization, and question-answering.
- A popular sequence-to-sequence model T5 BART.



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

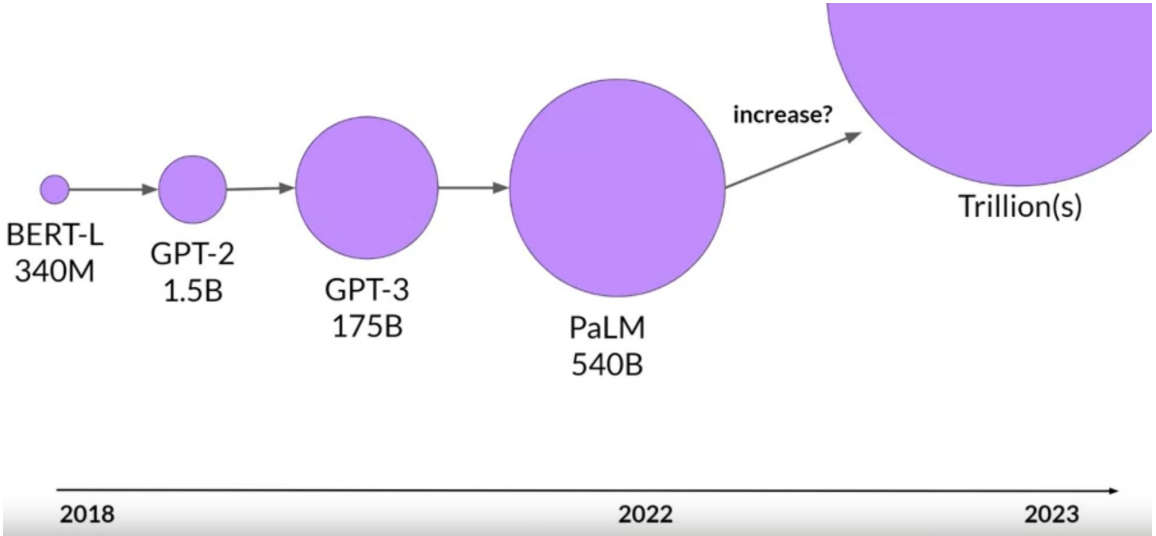
2. Tech Foundations of GenAI: Pre-Training



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

2. Tech Foundations of GenAI: larger the model, the better?

- Researchers have found that the larger a model, the more likely it is to work as you needed to without additional in-context learning or further training.
- This steady increase in model size actually led some researchers to hypothesize the existence of a new Moore's law for LLMs.



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

3. Foundation Models (FMs) / Large Language Models (LLMs)

Foundation Models (FMs) or Large Language Models (LLMs) are a form of generative artificial intelligence (generative AI). They generate output from one or more inputs (prompts) in the form of human language instructions.

- LLMs are **large, complex, deep, neural network** based models
- LLMs are primarily based primarily built with a Decoder-only **Transformers based Architecture.**
- LLMs are trained using **Self-Supervised Learning and Semi-Supervised Learning** techniques.

Note: LLMs typically do not directly utilize or rely on Generative Adversarial Networks (GANs) or Variational Autoencoders (VAEs) for their training or operation. Although researchers have explored integrating GANs or VAEs into LLM architectures to enhance their generative capabilities

3. Foundation Models (FMs) / Large Language Models (LLMs)

Foundation Models

- **Large scale deep neural networks** models (with millions or even billions of parameters) trained on vast amounts of **diverse data** (such as **large text corpora**) to learn general representations of the input data domain.
- **Train a single general purpose large model** instead of many smaller task specific models.
- **Self-Supervised Learning** to learn inherent structure or relationships within the input data itself, without requiring explicit supervision or labeled data
- **Enables Transfer Learning** - knowledge learned from pre-training on a large dataset is transferred to downstream tasks with minimal task-specific data.
- **Pre-trained foundation models** as a starting point and then **fine-tune them on task-specific data** to adapt them to particular tasks.

3. Foundation Models (FMs) / Large Language Models (LLMs)

Examples of Foundation Models - All based on Transformer Architecture

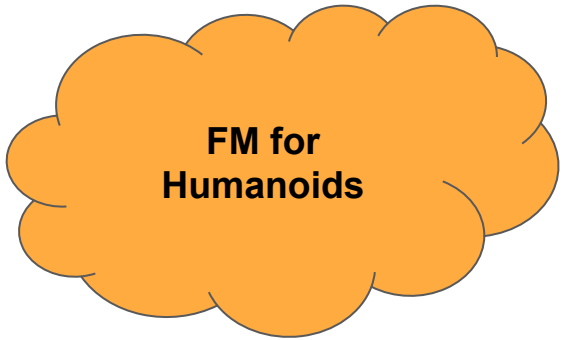
- **Google's BERT (Bidirectional Encoder Representations from Transformers)** - Released in 2018, it was trained using 340 million parameters and a 16 GB training dataset.
- **Google's T5 (Text-To-Text Transfer Transformer)** - Released in 2019, it had 11 billion parameters and was trained on a large **Common Crawl dataset**.
- **OpenAI's GPT-4 (Generative Pre-trained Transformer)** - Released in 2023, it was trained using 170 trillion parameters and a 45 GB training dataset.

FMs, such Anthropic's **Claude 2** and Meta's **Llama 2**, **Mistral** and the text-to-image model like Stability AI's **Stable Diffusion** can perform a range of tasks, like writing blog posts, generating images, solving math problems, engaging in dialog, and answering questions etc.

3. Foundation Models (FMs) / Large Language Models (LLMs)

Potential use of FMs include automating tasks and processes. For example

- **Customer support**
- **Language translation**
- **Content generation**
- **High-resolution image generation**
- **Robotics and many more**



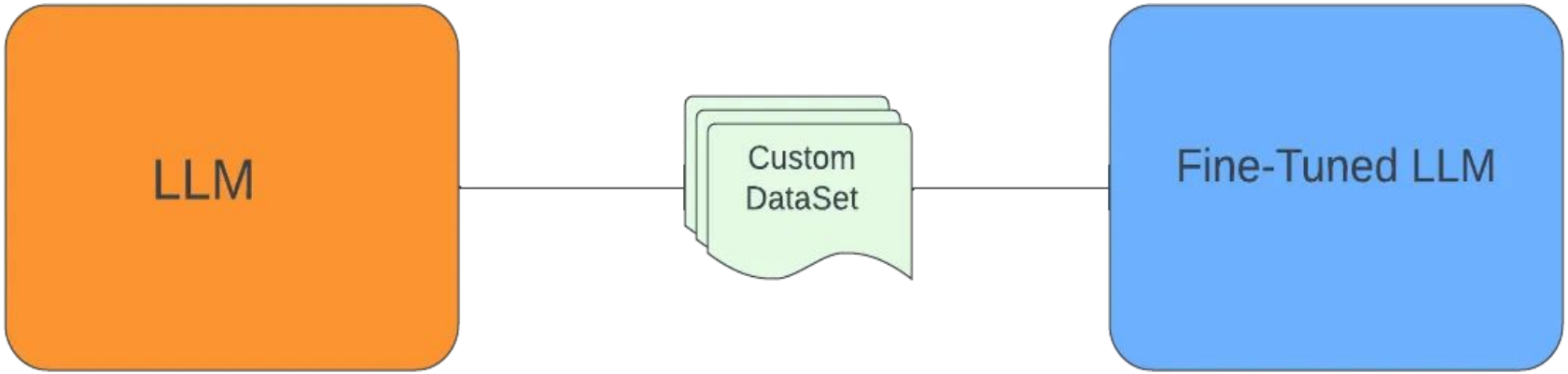
It’s faster and cheaper for data scientists to use pre-trained FMs to develop new ML applications rather than train unique ML models from the ground up.

4. Prompt Engineering

- Prompt engineering is a technique used to design and refine the prompts or conditioning text provided to large language models.
- Prompt engineering is particularly relevant in settings where fine-tuning or adapting pre-trained language models for specific tasks is not feasible or practical.
- Effective prompts are carefully crafted to provide the necessary context and constraints to guide the model towards generating relevant and accurate responses.
- Iterative refinement helps improve the quality and performance of the model by fine-tuning the prompts based on empirical observations and feedback.

5. Fine Tuning

- Fine tuning is performed on a custom/domain/task specific dataset.
- Fine tuning is a process of taking a general purpose LLM model and converting it to a specialised ones. For ex: **Instruction fine tuning** is one of the variants of fine tuning that made GPT3 into ChatGPT.



5. Fine Tuning

- Fine tuning is **one of the three ways to tune the LLM to behave** and perform on a specific task on hand. The other ways are “Prompt Engineering” and “Retrieval Augmentation Generation”
- Fine tuning **comes after pre training process**. While pre training is a Un/Semi/Self supervised training on large, unlabelled and general purpose corpus, fine tuning is usually a supervised training on a small, labelled task specific dataset.
- **Fine Tuning steps**: Selecting a pre-trained model → Gathering and pre processing Relevant Dataset → Fine Tuning
- Fine-tuning LLMs is commonly **used in NLP tasks such as sentiment analysis, named entity recognition, summarization, translation** etc. where understanding context and generating coherent language is crucial.

5. Fine Tuning

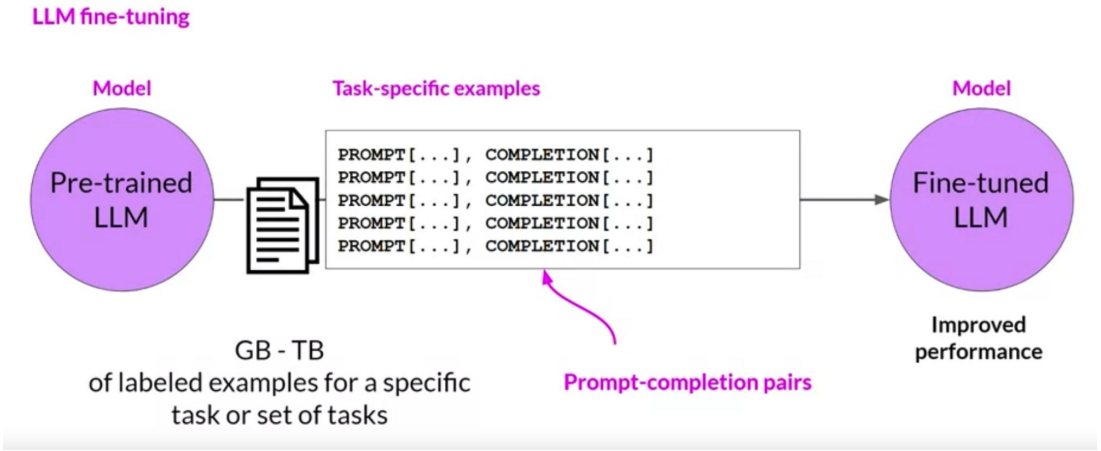
Two types of Fine tuning

- **Full Fine tuning** (Instruction Fine Tuning): Is a type that updates all model weights, creating a new version with improved capabilities. It requires sufficient memory and computational resources, similar to pre-training, to handle the storage and processing of gradients, optimizers, and other components during training.
- **Partial Fine Tuning** (PEFT: LoRA, QLoRA): Is a form of fine-tuning that only updates only a subset of parameters, effectively “freezing” the rest. This reduces the number of trainable parameters, making memory requirements more manageable and preventing **catastrophic forgetting**. PEFT maintains the original LLM weights, avoiding the loss of previously learned information.

Note: Memory allocation is not only required for storing the model but also for essential parameters during training, presenting a challenge for simple hardware.

5. Fine Tuning

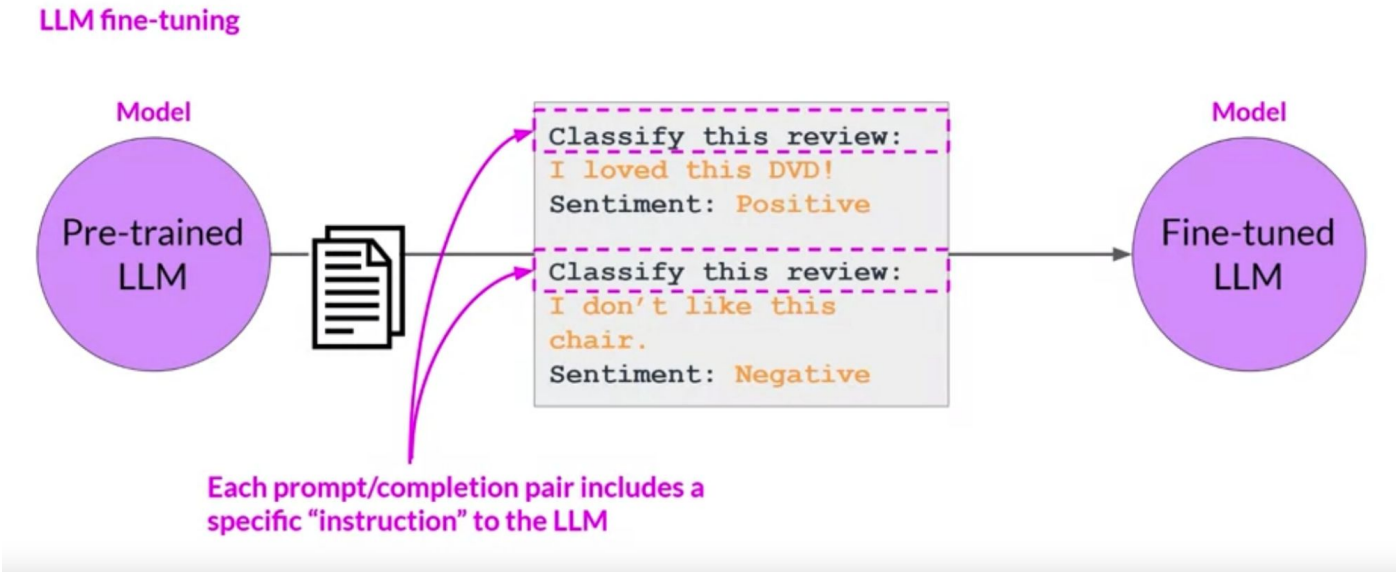
- In contrast to pre-training, where you train the LLM using vast amounts of unstructured textual data via self supervised learning, fine-tuning is a supervised learning process where you use a data set of labeled examples to update the weights of the LLM.
- The labeled examples are prompt completion pairs, the fine-tuning process extends the training of the model to improve its ability to generate good completions for a specific task.



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

5. Fine Tuning - Instruction fine tuning

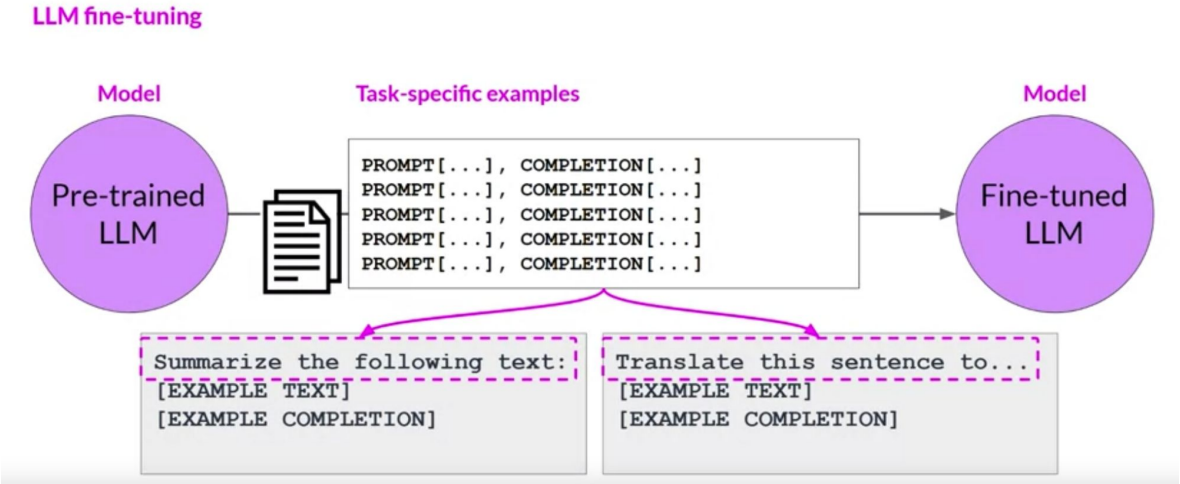
- Instruction fine-tuning trains the model using examples that demonstrate how it should respond to a specific instruction.
- Instruction fine-tuning, where all of the model's weights are updated is known as full fine-tuning.



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

5. Fine Tuning

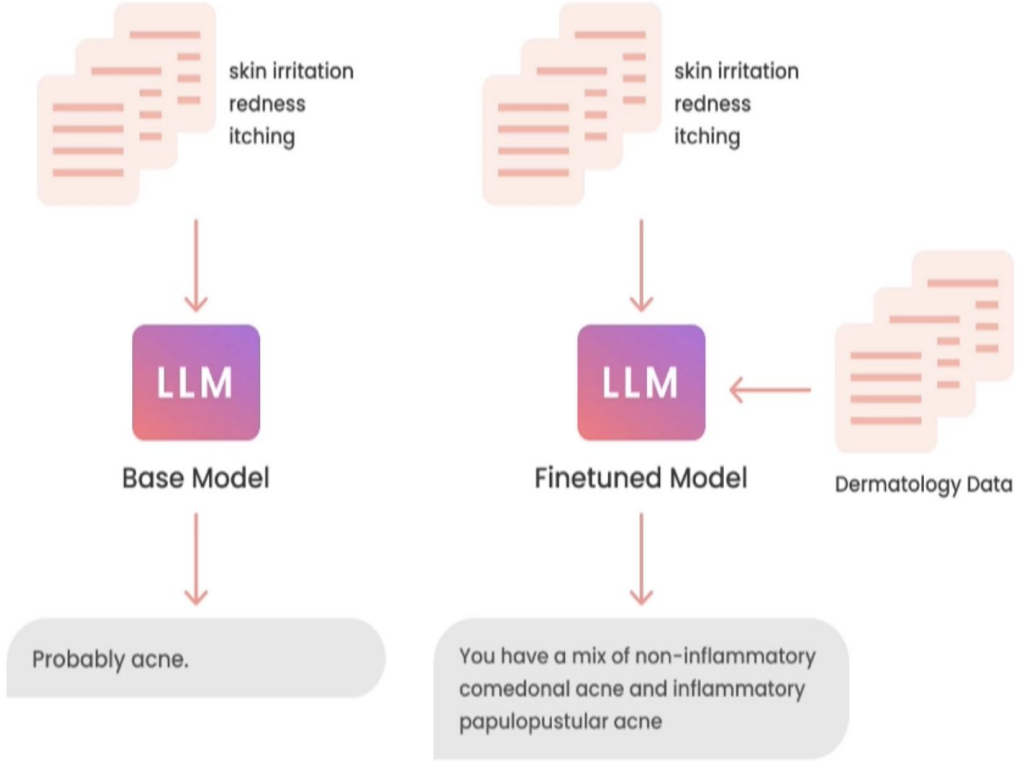
- To fine tune the model to improve its summarization ability, you'd build up a data set of examples that begin with the instruction summarize, the following text or a similar phrase.
- And if you are improving the model's translation skills, your examples would include instructions like translate this sentence.



Refer: https://www.deeplearning.ai/courses/generative-ai-with-llms/

5. Fine Tuning: What does it do?

- It lets us put more data than what fits into the prompt.
- It gets the model to learn the data instead of getting an access to it
- It steers the model to produce more consistent output
- It reduces hallucinations
- It customises the model to more specific use case
- It process is similar to models pre-training.



Refer: https://www.coursera.org/projects/finetuning-large-language-models-project

5. Fine Tuning: Compared to Prompt Engineering

Fine-tuning

Pros:

- No data to get started
- Small upfront cost
- No technical knowledge
- Connect data through Retrieval (RAG)

Cons:

- Fits relatively less data
- Forgets data
- Hallucinate

Prompt Engineering

Pros:

- Nearly unlimited data fits
- Learn new information
- Correct incorrect information
- Can use RAG too

Cons:

- More high quality data
- Upfront compute cost
- Need knowledge to process data and fine tuning

5. Fine Tuning: Benefits

- While Prompting is good for generic side projects and prototype, Fine-tuning is preferred for domain-specific, enterprise, production usage, privacy etc.
- Demonstrates better performance in terms of reduced hallucinations and increased consistency.
- By fine-tuning a pre-trained model on a specific task, we can adapt and refine its learned representations to better suit the characteristics of the target task or domain.
- Fine-tuning typically requires less training time and data compared to training a model from scratch.
- We can fine-tune different layers or parts of the pre-trained model depending on the task requirements and the amount of available training data.

5. Fine Tuning

PEFT

- Technique used to fine-tune large pre-trained language models in a computationally efficient manner. It aims to achieve good performance on downstream tasks while reducing the computational cost and resource requirements associated with fine-tuning large models.
- PEFT often involves making architectural modifications to the pre-trained model to reduce its computational complexity while preserving its representational capacity.
- For example, the model may be pruned, quantized, or compressed to reduce the number of parameters or the size of the model.

5. Fine Tuning

LoRA

- Instead of fine-tuning all the weights that constitute the weight matrix of the pre-trained large language model, two smaller matrices that approximate this larger matrix are fine-tuned.
- These matrices constitute the LoRA adapter. This fine-tuned adapter is then loaded into the pre-trained model and used for inference.
- LoRA fine-tuning for a specific task or use case yields an unchanged original LLM and a considerably smaller “LoRA adapter,” often representing a single-digit percentage of the original LLM size (in MBs rather than GBs).
- During inference, the LoRA adapter is combined with its original LLM. The advantage is that the LoRA adapters reuses the original LLM, reducing overall memory reqt.

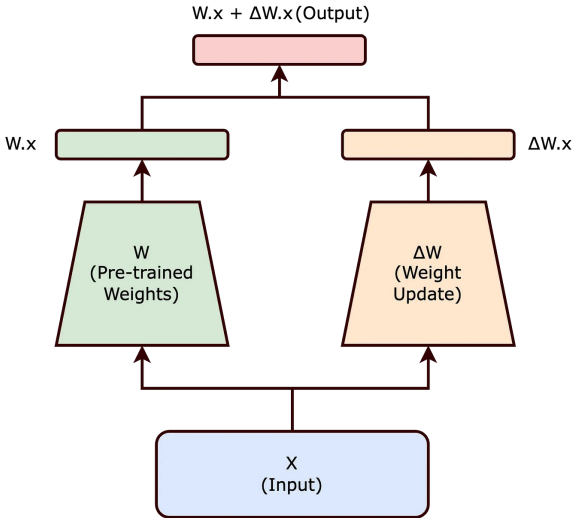
5. Fine Tuning

QLoRA

- QLoRA represents a more memory-efficient iteration of LoRA. **QLoRA** quantizing the weights of the LoRA adapters (smaller matrices) to lower precision (e.g., 4-bit instead of 8-bit) making LoRA more memory efficient i.e. reduces memory footprint and storage requirements
- Despite this reduction in bit precision, QLoRA maintains a comparable level of effectiveness to LoRA.

5. Fine Tuning: Traditional / Full Fine-tuning

Here we modify a pre-trained neural networks weights to adapt to a new task. This adjustment involves altering the original weight matrix (W) of the network. The changes made to (W) during fine-tuning are collective ly represented by (Δ W), such that the updated weights can be expressed as (W + Δ W).



Refer: https://towardsdatascience.com/understanding-lora-low-rank-adaptation-for-finetuning-large-models-936bce1a07c6

5. Fine Tuning: Low Rank Adaptation

Here, rather than modifying (W) directly, the LoRA approach seeks to decompose (ΔW); rank hypothesis suggests that significant changes to the neural network can be captured using a lower-dimensional representation.

LoRA proposes representing (Δ W) as the product of two smaller matrices, (A) and (B), with a lower rank. The updated weight matrix (W') thus becomes:

[W' = W + BA]

In this equation, (W) remains frozen (i.e., it is not updated during training). The matrices (B) and (A) are of lower dimensionality, with their product (BA) representing a low-rank approximation of (Δ W).

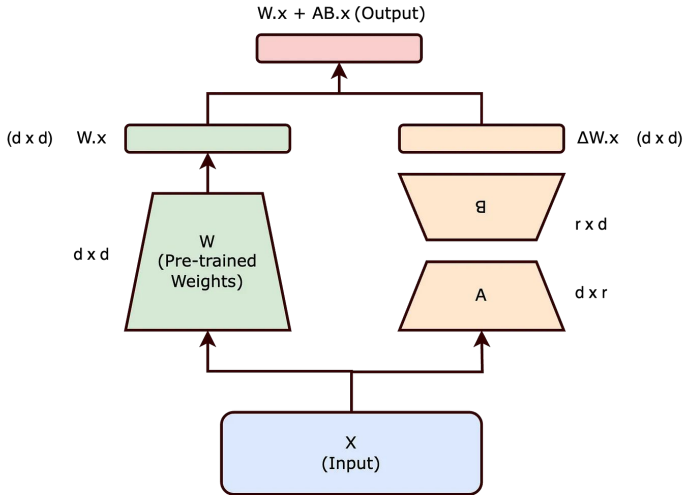
Refer: https://towardsdatascience.com/understanding-lora-low-rank-adaptation-for-finetuning-large-models-936bce1a07c6

5. Fine Tuning

- The rank of a matrix refers to the number of linearly independent rows or columns in the matrix.
- The reduced dimensionality of low-rank matrices allows for more compact representations and efficient storage and computation.
- By choosing matrices (A) and (B) to have a lower rank (r), the number of trainable parameters is significantly reduced.

Low-Rank Adaptation (LoRA) method benefits

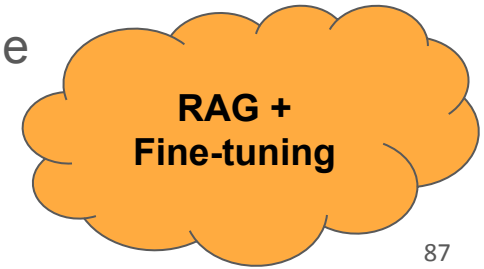
- Reduced Memory Footprint
- Faster Training and Adaptation
- Feasibility for Smaller Hardware:
- Scaling to Larger Models



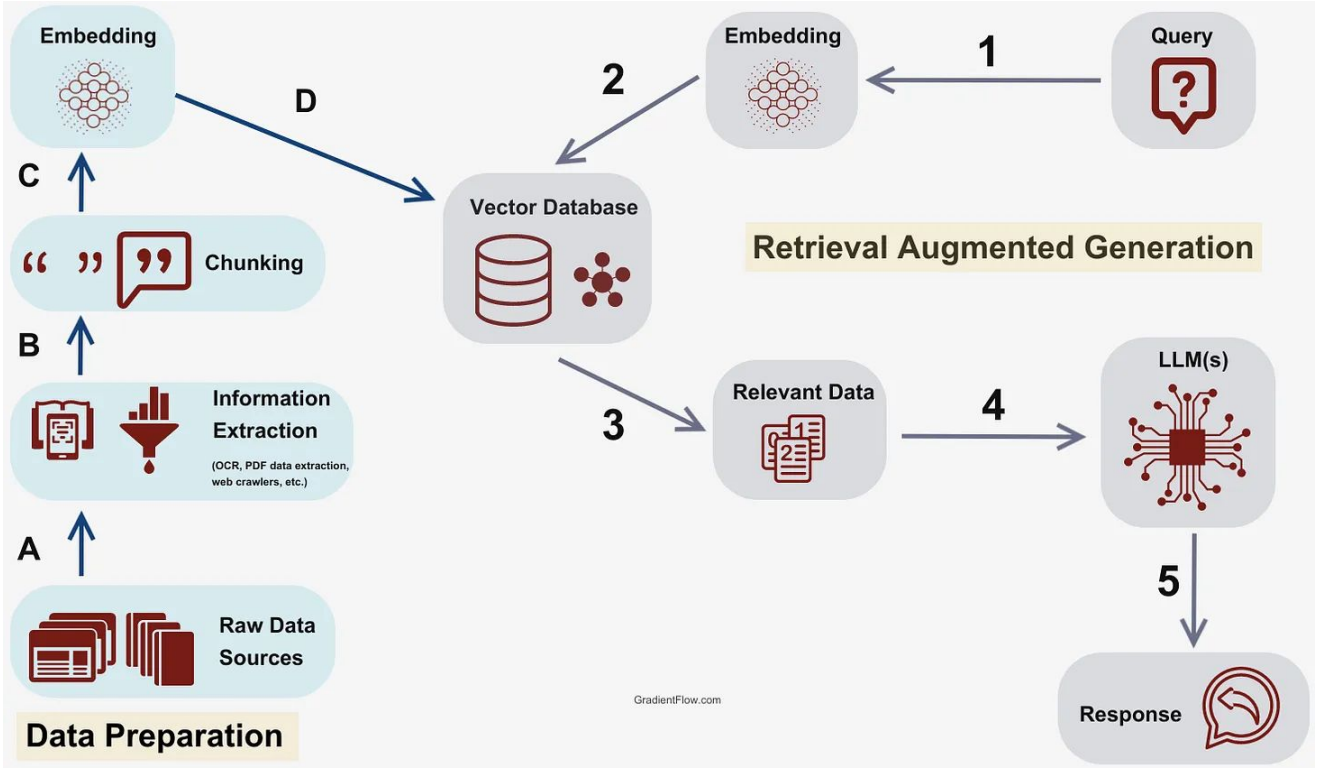
Refer: https://towardsdatascience.com/understanding-lora-low-rank-adaptation-for-finetuning-large-models-936bce1a07c6

6. Retrieval Augmentation Generation (RAG)

- Combines elements of both retrieval-based and generation-based approaches to produce high-quality and contextually relevant text responses.
- In a **retrieval-based approach,** the system retrieves pre-existing responses or snippets of text from a large database (such as a knowledge base or a corpus of text) in response to user queries.
- In a **generation-based approach**, the system generates responses from scratch based on the input query or context.
- The system first retrieves a set of candidate passages or documents from a large corpus of text using a retrieval mechanism.Next, it generates a response by leveraging both the retrieved passages and the input query or context.

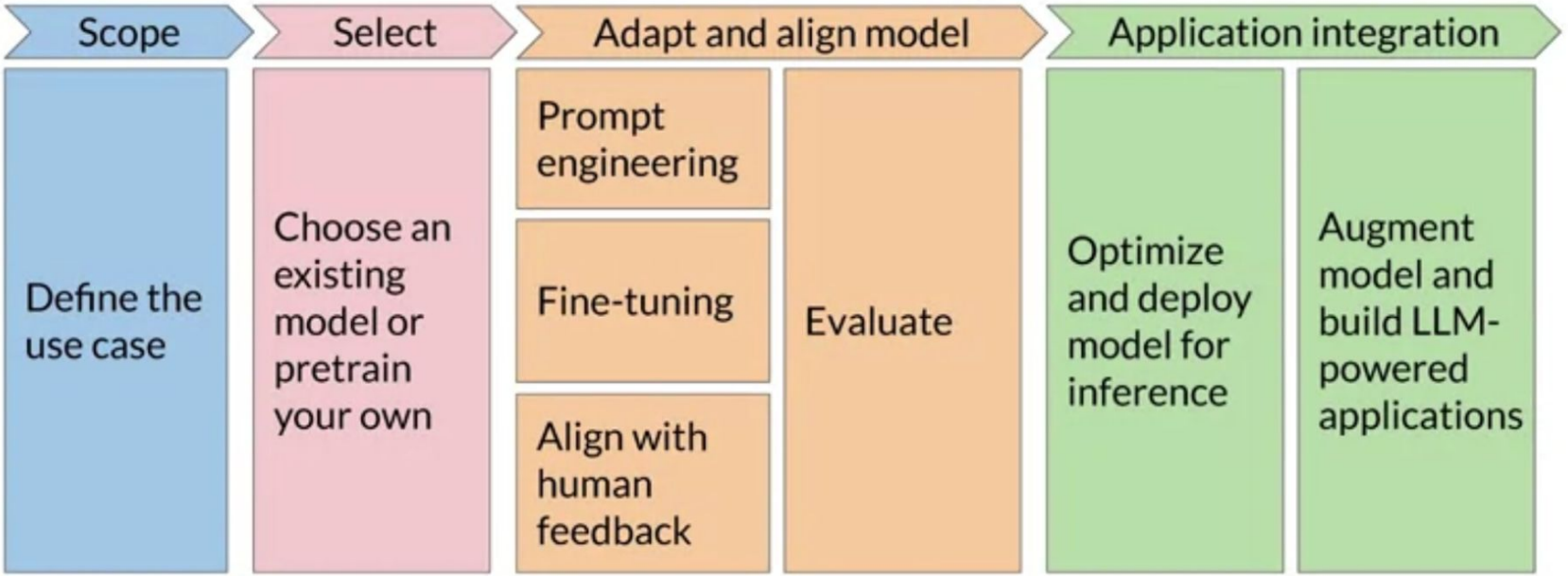


6. Retrieval Augmentation Generation (RAG)



Images: https://gradientflow.substack.com/p/best-practices-in-retrieval-augmented

7. Life Cycle of a Generative AI Project



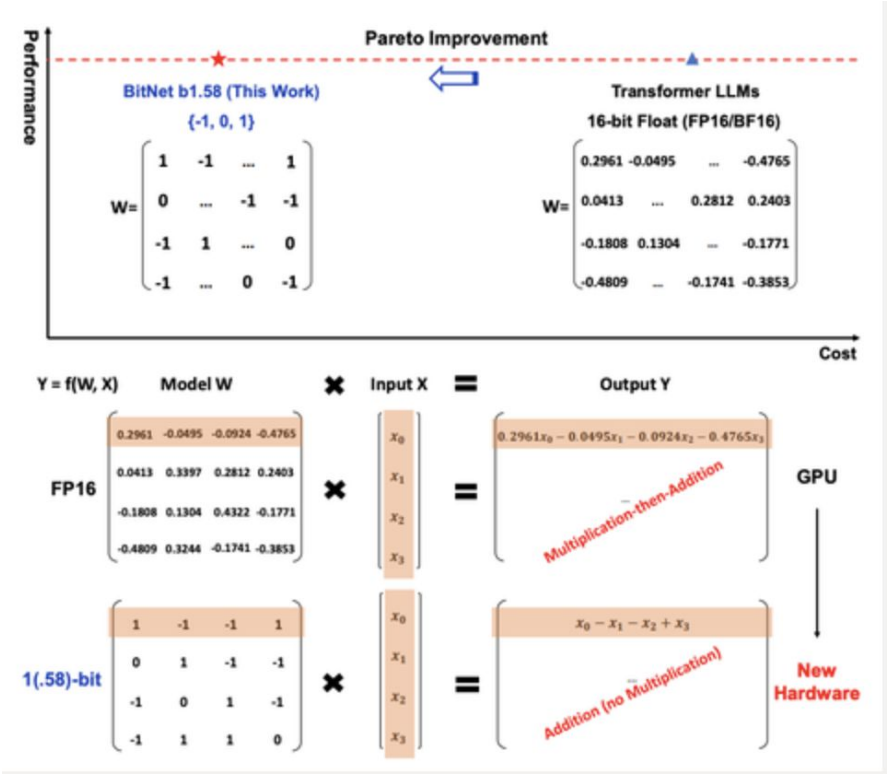
Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

8. Recent Developments

1. 1-bit LLM
2. RAFT
3. GROOT

The Era of 1-bit LLMs

- Refer: "The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits".
- Weight are represented as a trit - 1.58 bits {-1; 0; 1}, without losing performance of a Full Precision Transformer LLM with the same model size and training tokens.
- Instead of multiplying, we just add (since multiplying by 1, 0, or -1 is just a plus, minus, or nothing).
- This makes them 2.7x faster, use 3.5x less GPU memory, and 71x less energy.



Refer: https://arxiv.org/abs/2402.17764

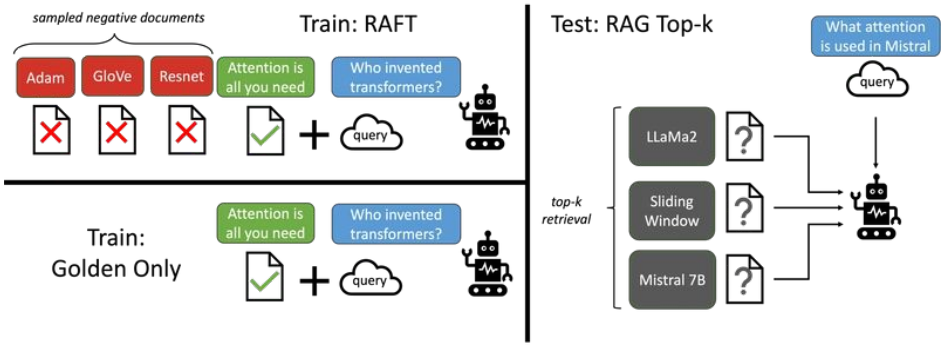
RAFT: Retrieval Augmented Fine Tuning

- Refer: RAFT: Adapting Language Model to Domain Specific RAG
- **Fine-tuning is akin to a "closed-book exam**," where the model relies solely on the knowledge acquired during training to answer queries. Can lead to approximation and hallucination
- **RAG is like an "open-book exam**," where the model retrieves relevant documents to generate answers. May retrieve "distractor" documents that are semantically similar but not relevant.
- **RAG involves storing those documents in a vector database** and (at query time) finding documents based on their semantic similarity with the question and bringing them (including distrator docs) into the context of the LLM for in context learning.
- **RAFT** given a question, and a set of retrieved documents, we train the model to ignore those documents that don't help in answering the question, which we call, distractor documents.
- **RAFT's chain-of-thought-style response** helps improve the model's ability to reason apart from citing the right sequence from the relevant document that would help answer the question.

RAFT: Retrieval Augmented Fine Tuning

Retrieval-Augmented Fine-Tuning” combines the benefits of Retrieval Augmented Generation and Fine-Tuning for better domain adaptation.

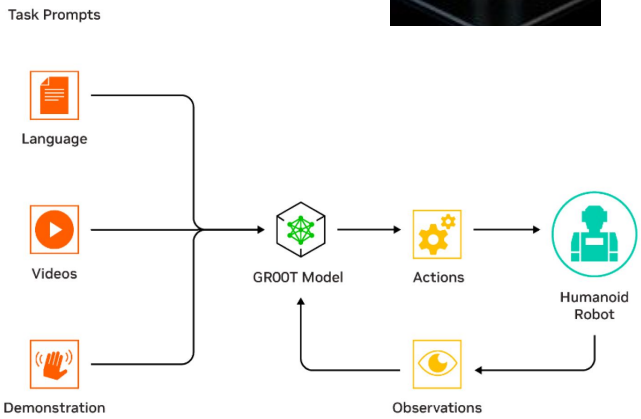
- It allows the model to "study" the documents beforehand, process involving creating a synthetic dataset that includes questions, relevant and irrelevant documents, answers, and Chain-of-Thought explanations.
- The it fine-tunes the model on this dataset, RAFT primes the LLM on domain knowledge and style while improving the quality of generated answers from the retrieved context.



Refer: https://arxiv.org/abs/2403.10131

GROOT: General-purpose Robot 00 Technology

- Nvidia's **AI Model for Humanoid Robot Development.**
- **GROOT** powered robots will be designed to understand human language and emulate human gestures through **imitation learning**
- Robots can interact with their environment, exhibit human-like behavior and perform complex tasks.
- Employs **Reinforcement Learning**, echoing the essence of human learning through exploration and guided by feedback and rewards.
- GR00T model takes **multimodal instructions** and past interactions as input and produces the actions for the robot to execute.



Refer: https://developer.nvidia.com/project-gr00t

Exit Test

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 1: Interacting with Large Language Models (LLMs) differs from traditional machine learning models. Working with LLMs involves natural language input, known as a _____, resulting in output from the Large Language Model, known as the _____ .

Choose the answer that correctly fill in the blanks.

A. Prompt, fine-tuned LLM
B. Prompt, completion
C. Prediction request, prediction response
D. Tunable request, completion

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 2: Large Language Models (LLMs) are capable of performing multiple tasks supporting a variety of use cases. Which of the following tasks supports the use case of converting code comments into executable code?

A. Invoke actions from text
B. Text summarization
C. Information Retrieval
D. Translation

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 3: What is the *self-attention* that powers the transformer architecture?

A. The ability of the transformer to analyze its own performance and make adjustments accordingly.
B. A technique used to improve the generalization capabilities of a model by training it on diverse datasets.
C. A mechanism that allows a model to focus on different parts of the input sequence during computation.
D. A measure of how well a model can understand and generate human-like language.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 4: Which of the following stages are part of the generative AI model lifecycle? (Select all that apply)

A. Selecting a candidate model and potentially pre-training a custom model.
B. Deploying the model into the infrastructure and integrating it with the application.
C. Manipulating the model to align with specific project needs.
D. Performing regularization
E. Defining the problem and identifying relevant datasets.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 5: "RNNs are better than Transformers for generative AI Tasks." Is this true or false?

A. True
B. False

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 6: Which transformer-based model architecture has the objective of guessing a masked token based on the previous sequence of tokens by building bidirectional representations of the input sequence.

A. Autoencoder
B. Autoregressive
C. Sequence-to-sequence

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 7: Which transformer-based model architecture is well-suited to the task of text translation?

A. Autoencoder
B. Sequence-to-sequence
C. Autoregressive

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 8: Scaling laws for pre-training large language models consider several aspects to maximize performance of a model within a set of constraints and available scaling choices. Select all alternatives that should be considered for scaling when performing model pre-training?

A. Compute budget: Compute constraints
B. Dataset size: Number of tokens
C. Batch size: Number of samples per iteration
D. Model size: Number of parameters

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 9: Fill in the blanks: _____ involves using many prompt-completion examples as the labeled training dataset to continue training the model by updating its weights. This is different from _____ where you provide prompt-completion examples during inference.

A. In-context learning, Instruction fine-tuning
B. Prompt engineering, Pre-training
C. Pre-training, Instruction fine-tuning
D. Instruction fine-tuning, In-context learning

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 10: Fine-tuning a model on a single task can improve model performance specifically on that task; however, it can also degrade the performance of other tasks as a side effect. This phenomenon is known as:

A. Catastrophic forgetting
B. Instruction bias
C. Catastrophic loss
D. Model toxicity

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 11: Which evaluation metric below focuses on precision in matching generated output to the reference text and is used for text translation?

A. HELM
B. ROUGE-2
C. BLEU
D. ROUGE-1

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 12: Which of the following best describes how LoRA works?

A. LoRA trains a smaller, distilled version of the pre-trained LLM to reduce model size
B. LoRA freezes all weights in the original model layers and introduces new components which are trained on new data.
C. LoRA decomposes weights into two smaller rank matrices and trains those instead of the full model weights.
D. LoRA continues the original pre-training objective on new data to update the weights of the original model.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 13: What is a soft prompt in the context of LLMs (Large Language Models)?

A. A set of trainable tokens that are added to a prompt and whose values are updated during additional training to improve performance on specific tasks.
B. A strict and explicit input text that serves as a starting point for the model's generation.
C. A technique to limit the creativity of the model and enforce specific output patterns.
D. A method to control the model's behavior by adjusting the learning rate during training.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 14: "Prompt Tuning is a technique used to adjust all hyperparameters of a language model." Is this true or false?

A. True
B. False

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 15: "PEFT methods can reduce the memory needed for fine-tuning dramatically, sometimes to just 12-20% of the memory needed for full fine-tuning." Is this true or false?

A. True
B. False

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 16: What is a transformer model?

A. A deep learning model that uses self-attention to learn relationships between different parts of a sequence.
B. A computer vision model that uses fully connected layers to learn relationships between different parts of an image.
C. A machine learning model that uses recurrent neural networks to learn relationships between different parts of a sequence.
D. A natural language processing model that uses convolutions to learn relationships between different parts of a sequence.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 17: What is the attention mechanism?

A. A way of identifying the topic of a sentence
B. A way of predicting the next word in a sentence
C. A way of determining the similarity between two sentences
D. A way of determining the importance of each word in a sentence for the translation of another sentence

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 18: What are the encoder and decoder components of a transformer model?

A. The encoder ingests an input sequence and produces a sequence of images. The decoder takes in the images from the encoder and produces an output sequence.
B. The encoder ingests an input sequence and produces a sequence of tokens. The decoder takes in the tokens from the encoder and produces an output sequence.
C. The encoder ingests an input sequence and produces a sequence of hidden states. The decoder takes in the hidden states from the encoder and produces an output sequence.
D. The encoder ingests an input sequence and produces a single hidden state. The decoder takes in the hidden state from the encoder and produces an output sequence.

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 19: What are the two sublayers of each encoder in a Transformer model?

A. Embedding and classification
B. Self-attention and feedforward
C. Recurrent and feedforward
D. Convolution and pooling

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Question 20: What are the three different embeddings that are generated from an input sentence in a Transformer model?

A. Embedding, classification, and next sentence embeddings
B. Convolution, pooling, and recurrent embeddings
C. Token, segment, and position embeddings
D. Recurrent, feedforward, and attention embeddings

Refer: Generative AI with LLMs Course by DeepLearning.AI: https://www.deeplearning.ai/courses/generative-ai-with-llms/

Exit Quiz Answers

Q1:

Answer B.

The input for working with LLMs is referred to as the prompt and the output from the LLM is referred to as the completion.

Q2:

Answer: D

Q3

Answer: C

Self-attention is a key component in models like Transformers, where it enables the model to attend to different words in the input sequence to capture their relationships and dependencies.

Q4:

Answer:

A: Selecting a candidate model and potentially pre-training a custom model are important stages in the generative AI model lifecycle.

B: Once we have a model performing to our needs, we can deploy it into the infrastructure and integrate it with the application.

C: It is likely we will have to manipulate the model in some way to align it with the specific needs of the project.

E: It is crucial to define the problem being solved and identify relevant datasets instrumental to the project.

Exit Quiz Answers

Q5
Answer: False: While RNNs can be used for generative AI tasks, they struggle with compute and memory, making it hard to keep context in longer texts. The transformers architecture is more parallelizable and its dynamic attention mechanism helps to capture long-range dependencies in the input.

Q6
Answer: A
Autoencoder models are pre-trained using masked language modeling. They use randomly masked tokens in the input sequence and the pretraining objective is to predict the masked tokens to reconstruct the original sentence

Q7
Answer: B
Sequence-to-sequence models use both the encoder and decoders in the transformer-based architecture making them best suited for tasks such as translation, text summarization, and question answering.

Q8
Answer:
A: The compute budget plays a crucial role in scaling during pre-training. When faced with a limited compute budget, we may need to impose restrictions on either the model size or the dataset size.
B: The size of the pre-training data is an important factor to consider when scaling with compute constraints. This is because the size of the dataset directly affects the computational requirements during pre-training, and having a larger dataset generally leads to improved model performance.
D: The size of the model in terms of number of parameters is a key scaling choice to consider with compute constraints because the number of parameters directly impacts the compute needs required during pre-training.

Exit Quiz Answers

Q9
Answer: D

Q10
Answer: A

Q11
Answer: C

BLEU focuses on precision and text translation while Rouge focuses on text summarization.

The evaluation metric that focuses on precision in matching the generated output to the reference text and is commonly used for text translation tasks is called "BLEU" (Bilingual Evaluation Understudy).

Q12

Answer: C

LoRA represents large weight matrices as two smaller, rank decomposition matrices, and trains those instead of the full weights. The product of these smaller matrices is then added to the original weights for inference.

Exit Quiz Answers

Q13
Answer: A

A soft prompt refers to aa set of trainable tokens that are added to a prompt. Unlike the tokens that represent language, these tokens can take on any value within the embedding space. The token values may not be interpretable by humans, but are located in the embedding space close to words related to the language prompt or task to be completed.

Q14
Answer: False

Prompt Tuning focuses on optimizing the prompts given to the model using trainable tokens that don’t correspond directly to human language. The number of tokens you choose to train, however, would be a hyperparameter of your training process.

Q15
Answer: True

By training a smaller number parameters, whether through selecting a subset of model layers to train, adding new, small components to the model architecture, or through the inclusion of soft prompts, the amount of memory needed for training is reduced compared to full fine-tuning.

Thank You!