

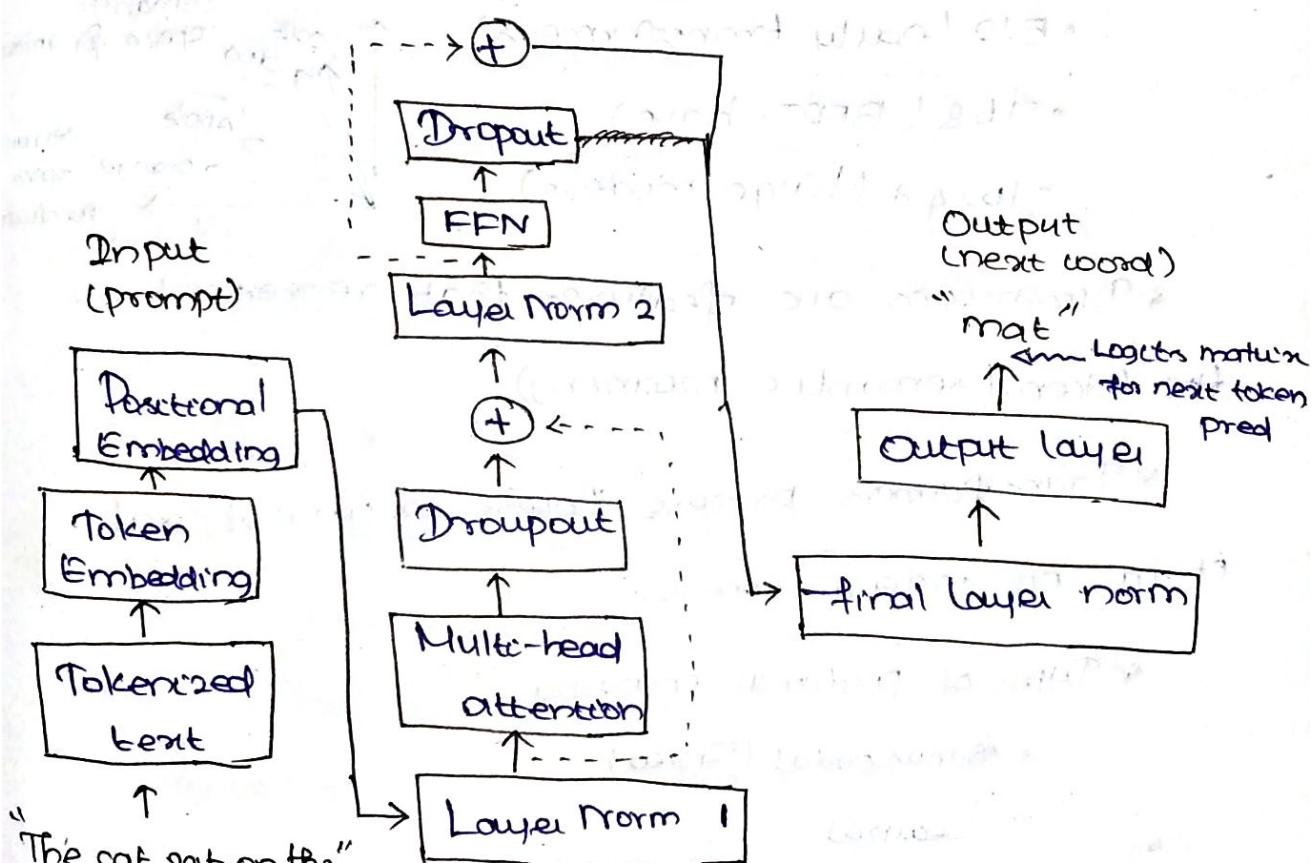
# Transformers for Vision Bootcamp

## Lec-1 Journey of a single token

\* LLM performs next word prediction iteratively to print the passage.

Decoder only architecture,

Transformer block



"The cat sat on the"

Input Block:

\* token is the smallest discrete unit a transformer processes. Tokens are produced by tokenizer, not by the model.

\* Common tokenization strategies,

- 1) Word based X
- 2) Subword based (Byte-Pair encoding) ✓
- 3) Character X

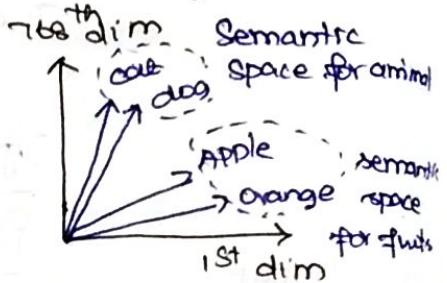
\* Each token is mapped to a unique ID in the vocabulary

\* Token ID does not carry any meaning, they are just the indices.

\* Then the token IDs are converted into token embedding (dense vectors) Every token ID will have its predefined embedding.

\* Typical dimensions of dense vectors,

- 512 (early transformers)
- 768 (BERT-base)
- 1024+ (large models)



\* Dimensions are features that represent a token. (semantic meaning)

\* Transformers process tokens in parallel and it has no order awareness.

\* Types of positional encoding

- Sinusoidal (fixed)
- Learned.

\* Positional embedding is added to the token embedding.

Input embedding = Token embedding + Positional embedding.

## Transformer block:

### 1) Layer Normalization:

\* It is like normalizing/standardizing the data

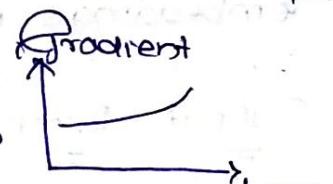
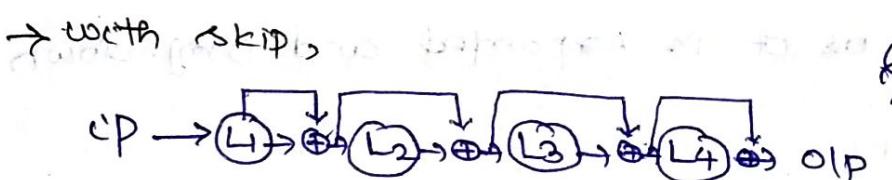
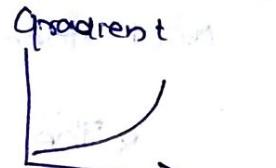
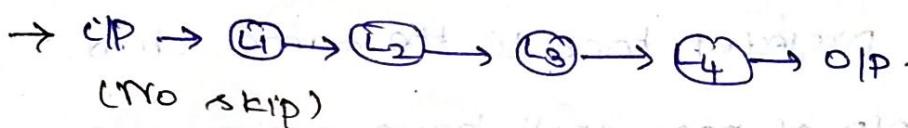
by making mean = 0, std. deviation = 1

$$\text{norm} = \frac{x - \mu}{\sigma} \quad \mu \rightarrow \text{mean}$$

### 2) Skip Connection:

\* It solves the problem of vanishing gradients.

It was introduced first in the ResNet paper.



\* Skip connection made the training much efficient, as central layer will have much contribution for gradient.

### 3) Dropout:

\* It is used to prevent overfitting by making some node inactive in a probabilistic way.

\* It forces some lazy neurons to learn from the data.

#### 4) Multi-head attention (briefly)

\* This layer computes attention scores which represents how each token is relevant to each other tokens.

#### 5) Feed forward NN:

\* When the input reaches this FNN, the magnitude must be changed but the dim is still the same.

\* In general, the dim is projected to  $4 \times \text{dim}$  and then projected back to the same dim.

\* The OUP of FNN will have richer contextual embeddings as it is expanded and proj. down.

#### Output Block:

\* The OUP from FNN reaches the final layer norm which is exactly same as prev layer norm

\* In the output layer, the OUP dimension is projected to the dimensionality of the vocab dictionary.

\* The next token is predicted using softmax probabilities.