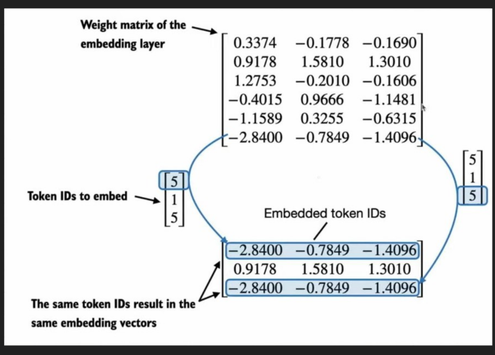
Lecture - 11 : Positional Embedding

① Until now, we looked at token embedding.

② In the embedding layer, the same token ID gets mapped to the same vector representation regardless of where the token ID is positioned in the input sequence.

eg. • The cat sat on the mat.
• on the mat the cat sat.

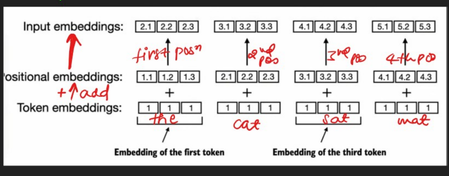


③ It is helpful to inject additional position information to the LLM.

④ There are two types of positional embedding

Absolute

For each pos^n in input sequence, a unique embedding is added to the token's embedding to convey it's exact location



NOTE: The positional vector have the same dimention as the original token embedding.

Relative

The emphasis is on the relative position or distance between tokens.

The model learns the the relationship in terms of "how far apart" rathers than at which exact position.

ADVANTAGE

model can generalised better to sequence of varying lengths, even if it has not seen such lengths during training.

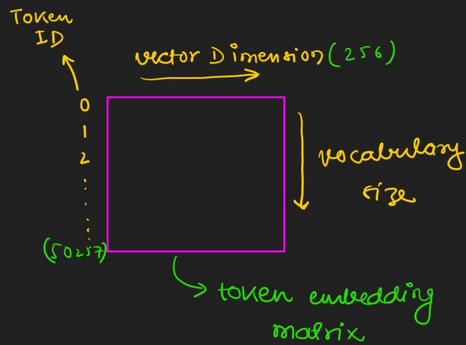
⑤ Both types of positional encoding enables LLM to understand the order and relationship b/w tokens, ensuring more accurate and context aware prediction.

⑥ The choice b/w the two depends on specific application and nature of data being processed.

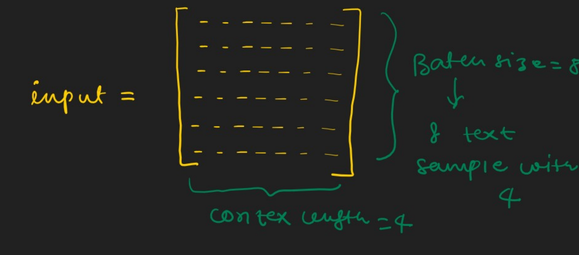
- Absolute: Suitable when fixed order of token is crucial, such as a sequence generation.
- Relative: Suitable for tasks like language modelling over long sequence, where the same phrase can appear in different parts of the sequence.

⑦ Open AI's GPT model use absolute positional embedding that are optimized during the training process.
This optimization is part of the model training itself.

⑧ Implementing positional embedding in a hands a manner:

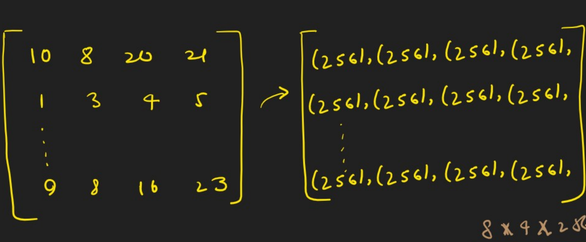


⑨ Get input data from dataloader

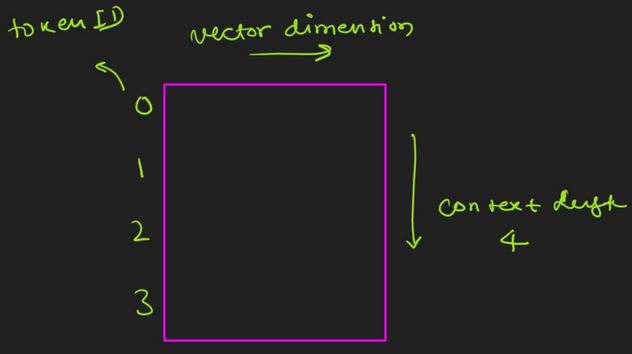


One embedding vector of 256 length is generated for each token in input.

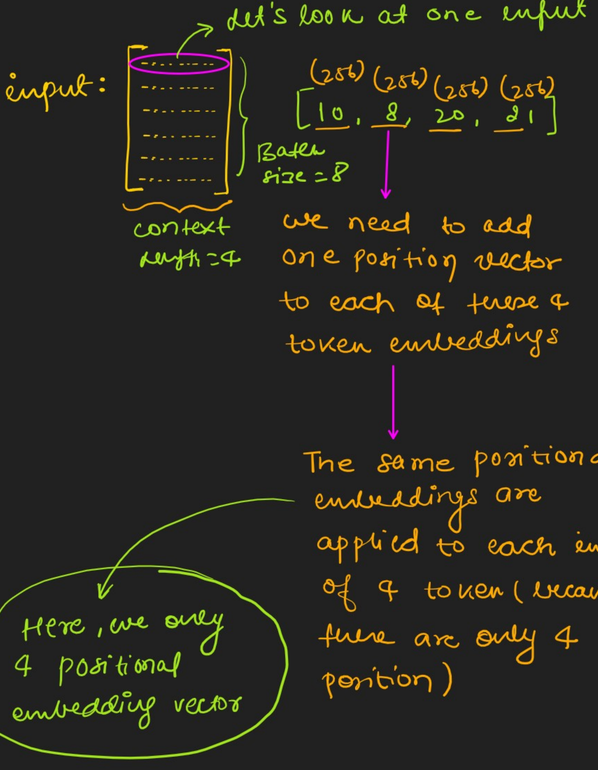
⑩ Inserting positional embedding



(11) Create another embedding layer for positional embedding



(12) Input



(13) Generate the 4 positional embedding vectors from the positional embedding matrix.

→ [4, 256]

(14) Add the token embedding + position embedding

$$\underbrace{\begin{bmatrix} (256)\ (256)\ (256)\ (256) \\ (256)\ (256)\ (256)\ (256) \end{bmatrix}}_{8\times4\times256} + \underbrace{\begin{bmatrix} (256)\ (256)\ (256)\ (256) \\ \\ \end{bmatrix}}_{4\times256}$$

vector embedding + position embedding

= Input Embeddings (8 x 4 x 256)

final training input the LLM.