NLP Ass-3

Transformer

2021114010

Therory

Q1

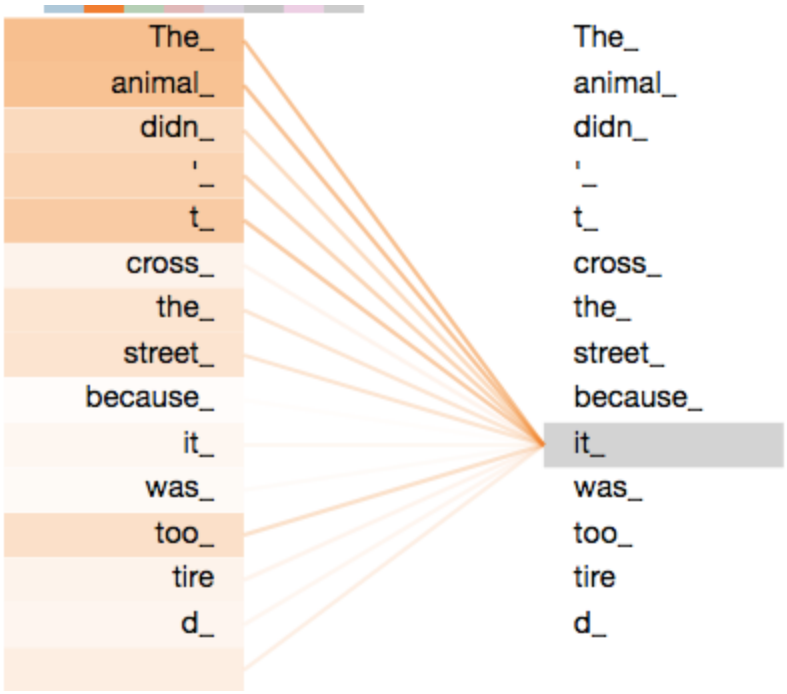
The Attention mechanism enables the transformers to have extremely long-term memory. A transformer model can “***attend***” or “***focus***” on all previous tokens that have been generated.

Self Attention :- Self-Attention is when it takes into consideration the relationship among words within the same sentence

For example the sentence

”`The animal didn't cross the street because it was too tired`”

When the model is processing the word “it”, self-attention allows it to associate “it” with “animal”.model processes each word (each position in the input sequence), self attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word.



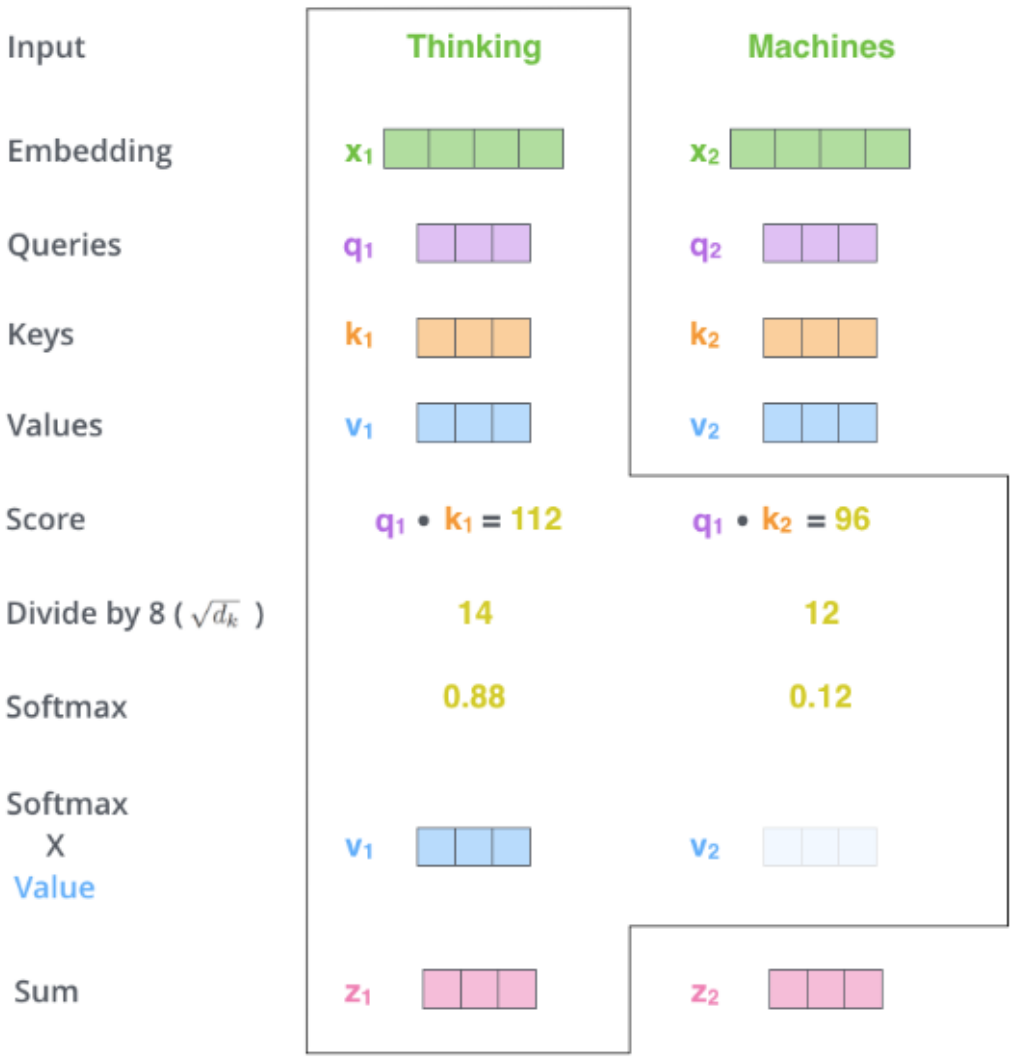
The main Purpose of Self-attention : -

1. Allow model to attend to different part of the same input sequence .
2. The purpose of self-attention is to allow transformer models to capture long-range dependencies in sequences .This is because self-attention allows the model to directly compare any two elements in the sequence, regardless of their distance apart.

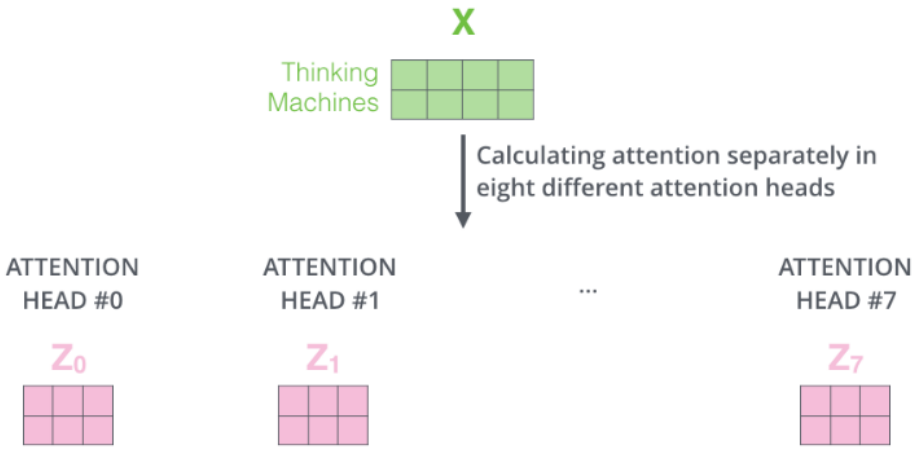
Self attention Mechanism capturing dependencies in sequences

1. It uses Key , Value and Query Values which are obtained by Multiplying Input sequence to Trained Metrices
2. Perform MAthematical Operations on it , whole process is done for every pair of word in the sentence to calculate the attention of 1 word on each word .

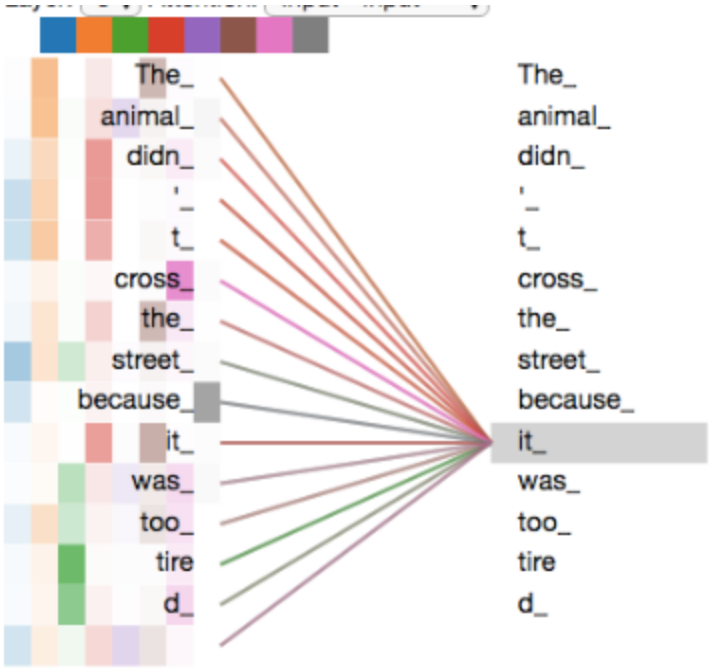
   For example Encoding `Thinking Machines` with attention



1. Multiheaded Attention :- the Paper Attention is all you need Works not just on 1 attention head but mutiple attention heads
    a. THe reason was that though the encoding contains a little bit of every other encoding, but it could be dominated by the actual word itself.
    b. So now we have n different key value query for each sentence



Example with 8 heads , the attention is different fro different heads and we use the concatinated version to capture essence of each head .



Q2

In a transformer, self-attention operates on the entire sequence simultaneously, which means that without positional information, the model would treat words with the same

content but different positions in the sequence as identical. Positional encodings provide the model with information about the position or order of words within the sequence, allowing it to distinguish between words at different positions.

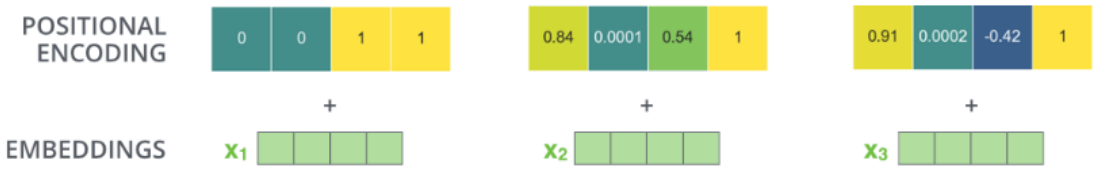
For positional Encoding we Use Vectors . These vectors follow a specific pattern that the model learns, which helps it determine the position of each word, or the distance between different words in the sequence

We then Concat the these Positional Encoding to the Output encodings of the Attention Model

and send forward to the next layer .

Input to Transformer Layers: The combined embeddings are then fed into the transformer model, which consists of multiple layers of self-attention and feedforward neural networks. The positional information helps the self-attention mechanism consider not only the content of the words but also their positions, enabling the model to capture dependencies and relationships within the sequence effectively.

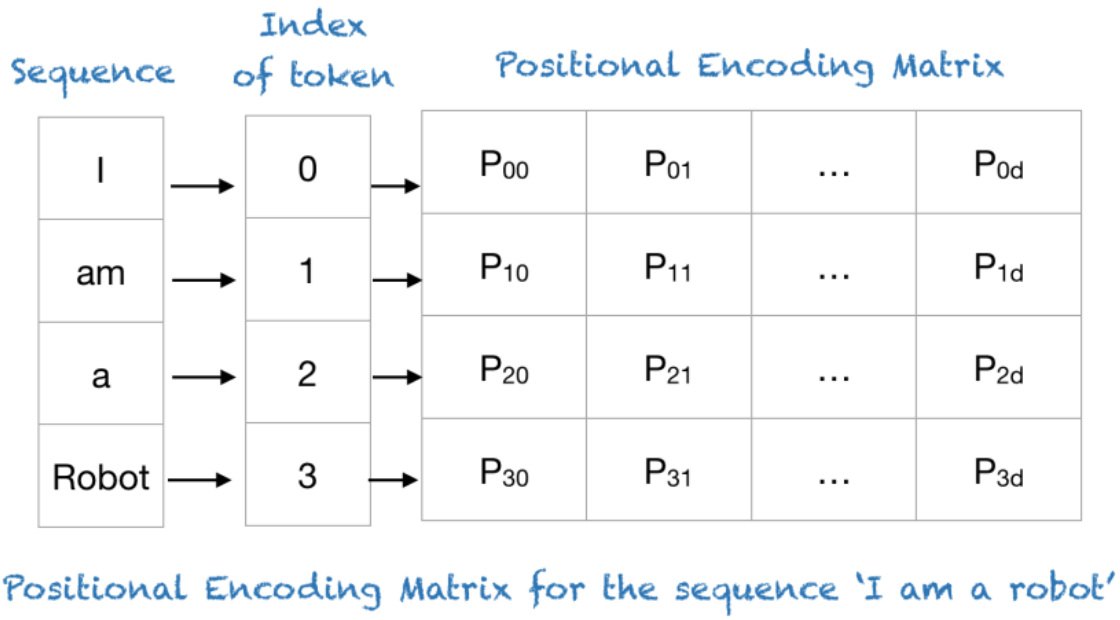
If we assumed the embedding has a dimensionality of 4, the actual positional encodings would look like this:



adding these values to the embeddings provides meaningful distances between the embedding vectors once they're projected into Q/K/V vectors and during dot-product attention.

These values of Positional Encoding Do not have meaning of their own , Rather they show the Relative Distance between 2 words

example



Positional Encoding Matrix for the sequence 'I am a robot'

Hyperparameters

```
"learning_rate": 0.0001,
        "Batch_size" :8,
        "Epochs" :5,
        "Embedding_dim" :512,
        "num_layers" :2,

optimizer = torch.optim.Adam(model.parameters(), lr=0.0001)
#loss function
criterion = nn.CrossEntropyLoss(ignore_index=0)
```

Learning Rate was Kept High due to less training Data , It may lead to everfitting but with this much Data it was the Best Step .

Thats why there is a Significant Difference in Training and Testing Accuracries and Bleu Scores

Accuracy , F1 , Bleu scores

Train Data

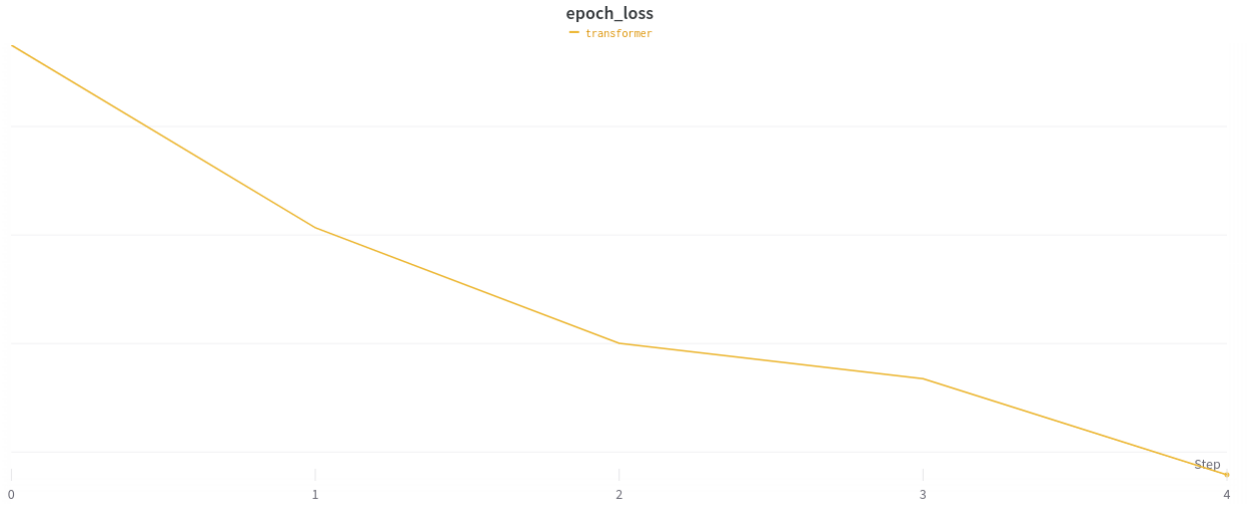
Bleu Score :- 0.008519611919693203

Test Data

bleu Score : - 5.464854192628201e-05

Graphs

Loss



Accuracy