Bert

Outline

1) Language Models
   - Training
   - Inference

2. Transformer Architecture (Encoder)
   - Embedding vectors
   - Positional Encoding
   - Self Attention and casual mask

3. BERT
   - The importance of the left and the right content
   - Bert pre-training
     → Masked Language Model task
     → Next Sentence Prediction task

4. BERT fine tuning
   - Text classification Task
   - Question Answering Task

What is a Language Model?

A language model is a probabilistic model that assign probabilities to sequence of words

$$P(\underbrace{\text{"china"}}_{\text{Next Token}} \mid \underbrace{\text{"Shanghai is a city in"}}_{\text{Prompt}})$$

We usually train a neural network to predict these probabilities. A neural network trained on a large corpora of text is known as a Large Language Model (LLM).

How to train a Language model?

Imagine we want to train a language model on Chinese poems, for example the following one:

English

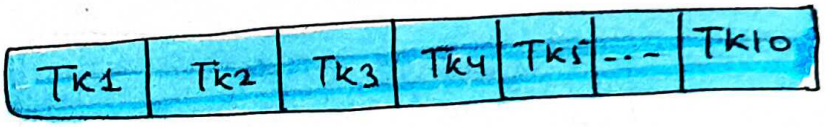
Before my bed lies a pool of moon bright
I could imagine that it's frost on the ground I look up and see the bright shining moon Bowing my head I am thinking of home.

How to train a language model?

Target Sequence (10 tokens) → Before my bed lies a pool of moon bright [EOS]

Cross Entropy Loss ⟶ Run **Backprop** to update the weights

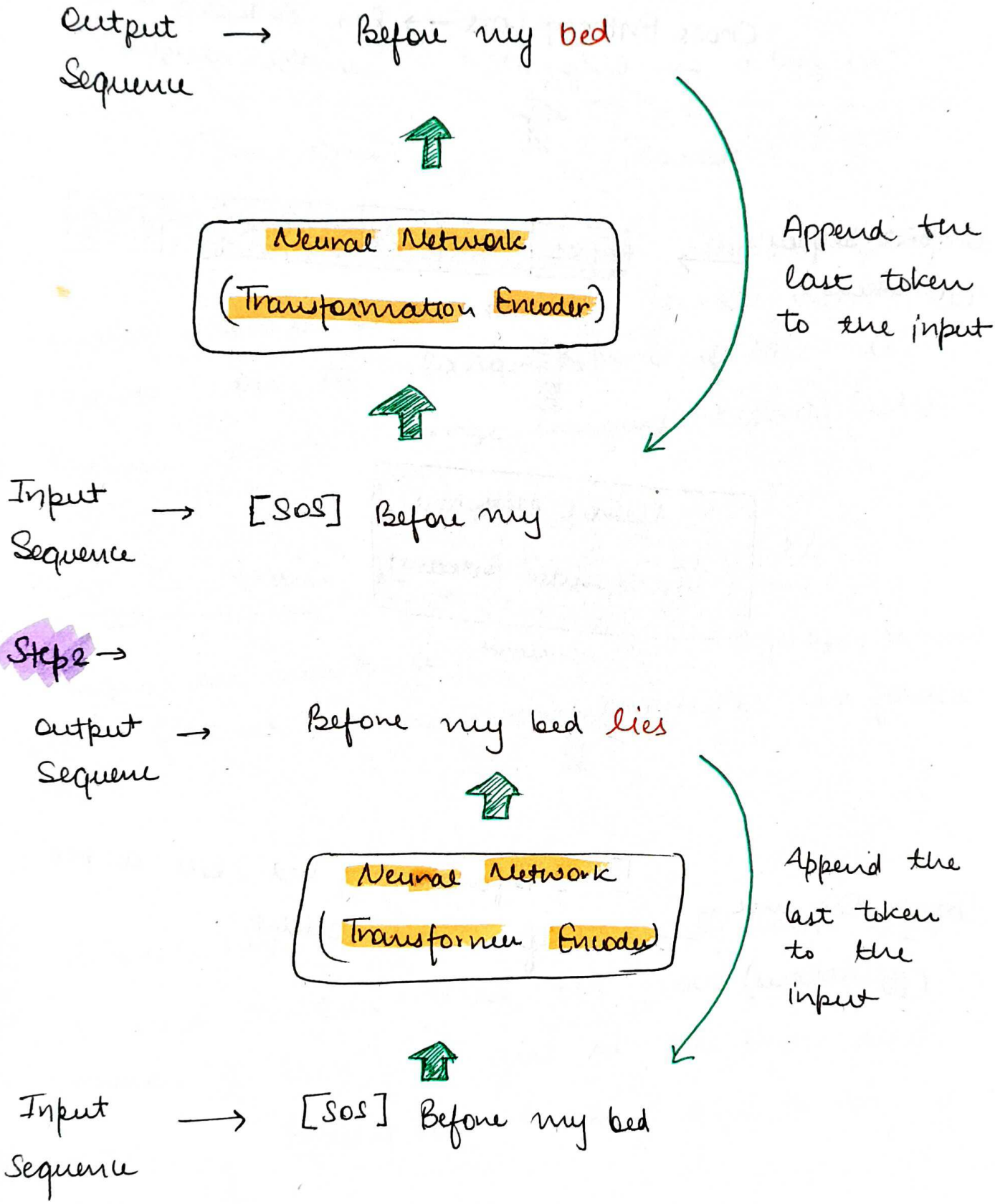
Output Sequence (10 tokens) ⟶



Input Sequence (10 tokens) ⟶ [SOS] Before my bed lies a pool of moon bright

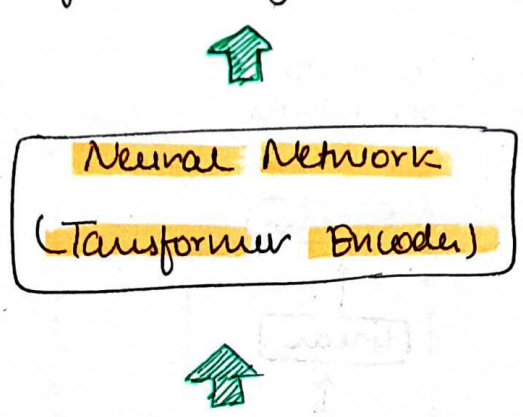
How to Inference a language model?

Steps →



Step3:-

Output Sequence → Before my bed lies a



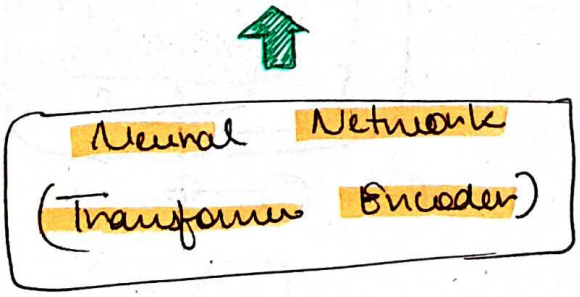
Append the last token to the input

Input Sequence → [SOS] Before my bed lies

⋮

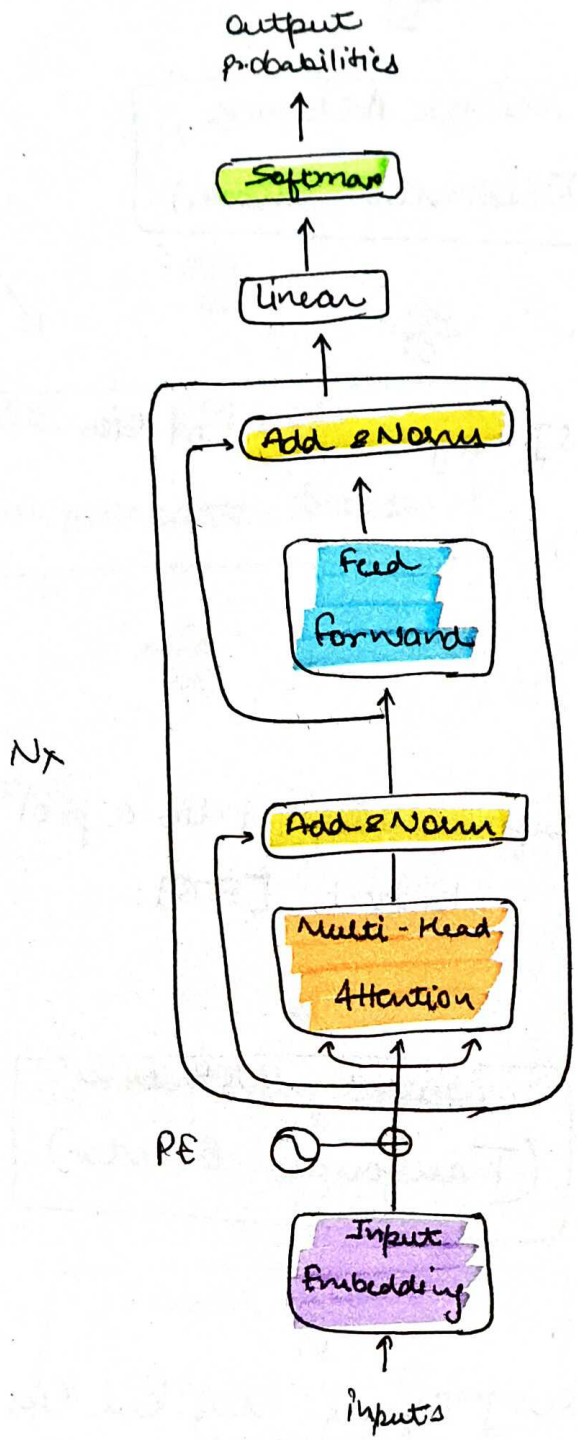
Step9:

Output Sequence → Before my bed lies a pool of moon bright [EOS]

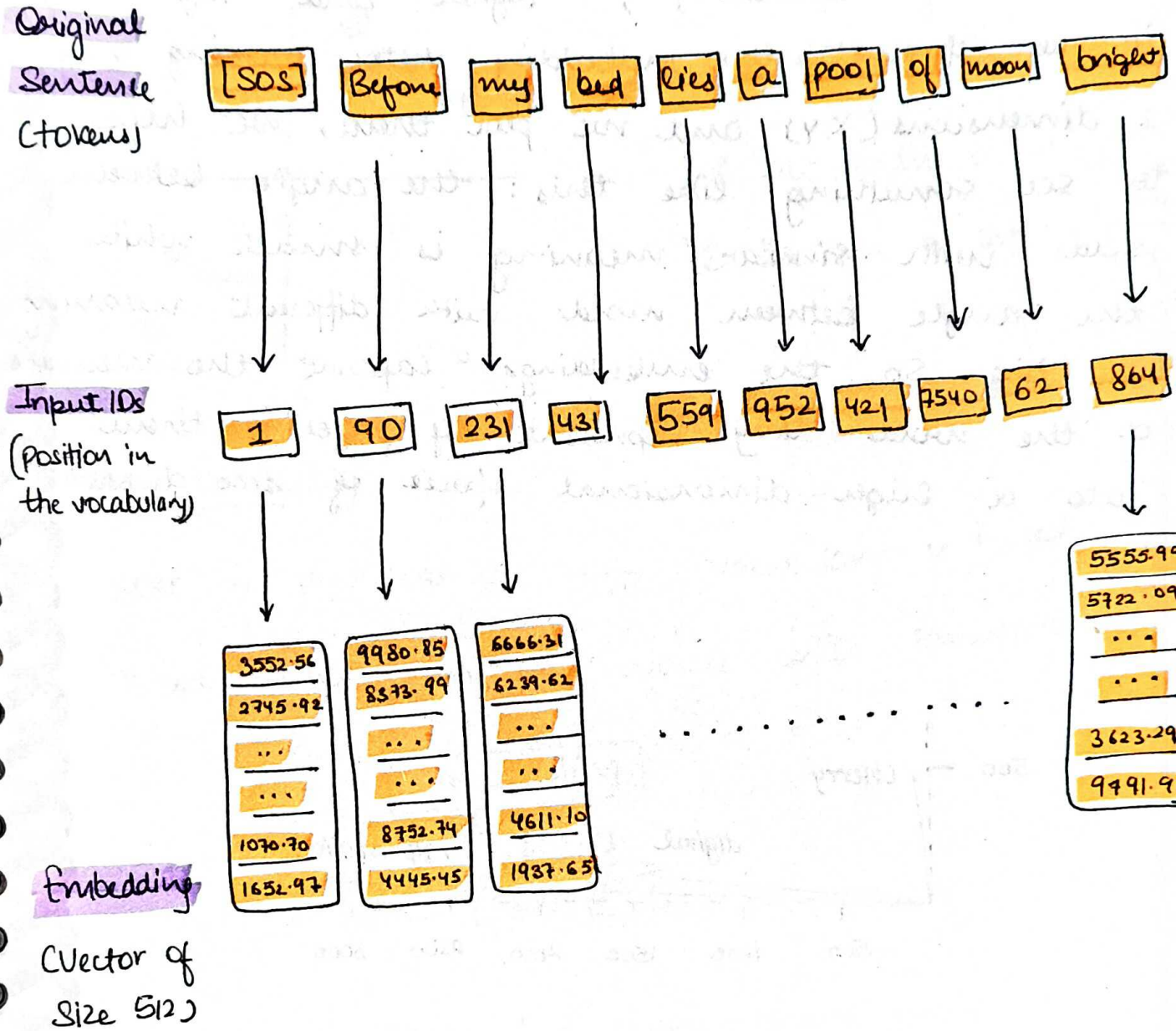


Input Sequence → [SOS] Before my bed lies a pool of moon bright.

Transformer Encoder



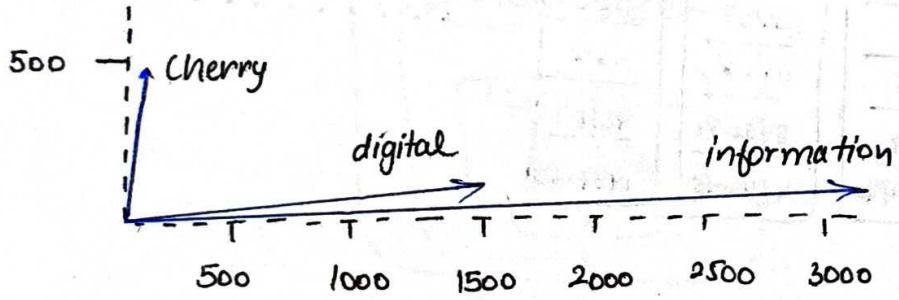
Let's convert the input into Input Embeddings!



We define $d_{model} = 512$, which represents the size of the embedding vector of each word

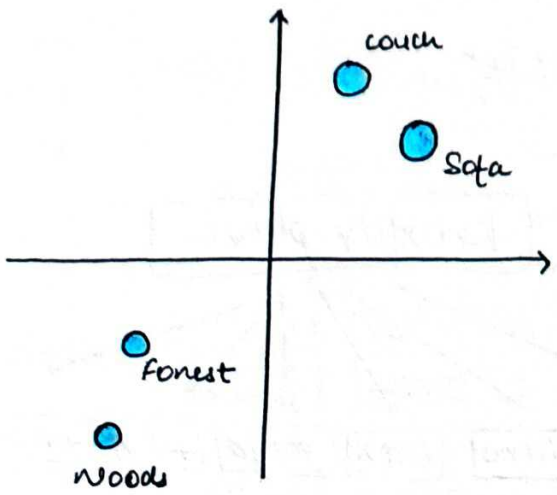
Why do we use vectors to represent words?

Given the words "cherry", "digital" and "information", if we represent the embedding vectors using only 2 dimensions (X,Y) and we plot them, we hope to see something like this: the angle between words with similar meaning is small while the angle between words with different meaning is big. So, the embeddings "capture" the meaning of the words they represent by projecting them into a high-dimensional space of size dmodel.



We commonly use the **cosine similarity**, which is based on the **dot product** between the two vectors.

Word Similarity



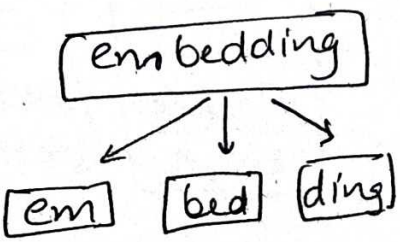
distance reflects words similarity

"I sat on the couch" → Neural Network

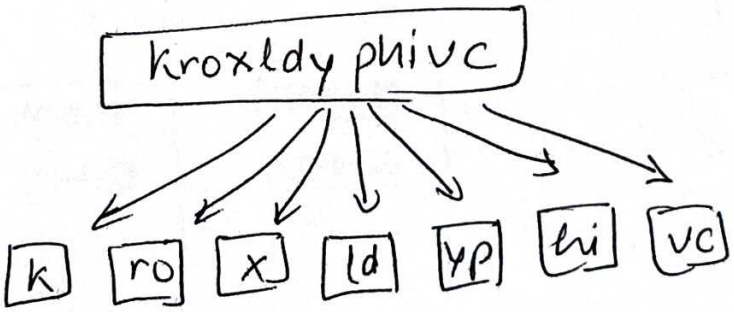
"I sat on the sofa" →

BERT's Vocabulary

1. BERT is pre-trained ⟶ Vocabulary is fixed
2. Break down unknown words into subwords:

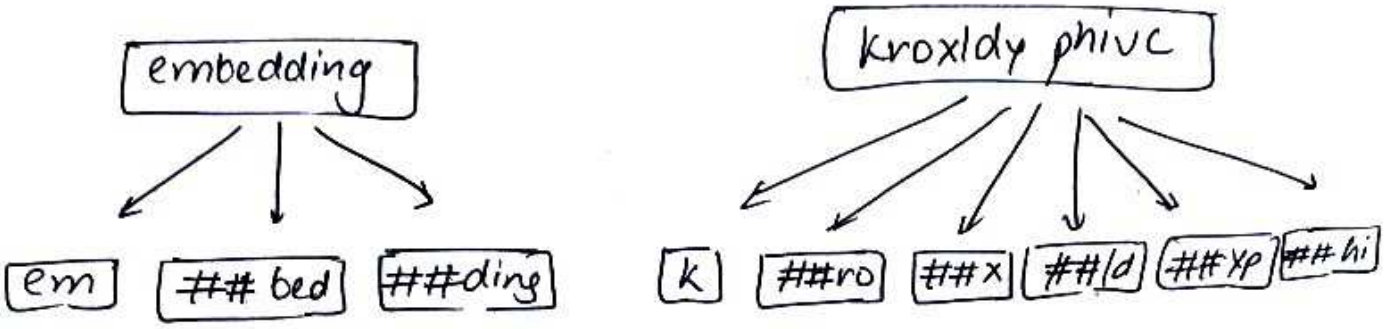


3. A subword exists for every character.

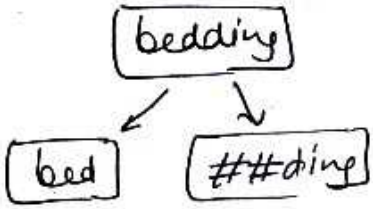


2 Types of Subword

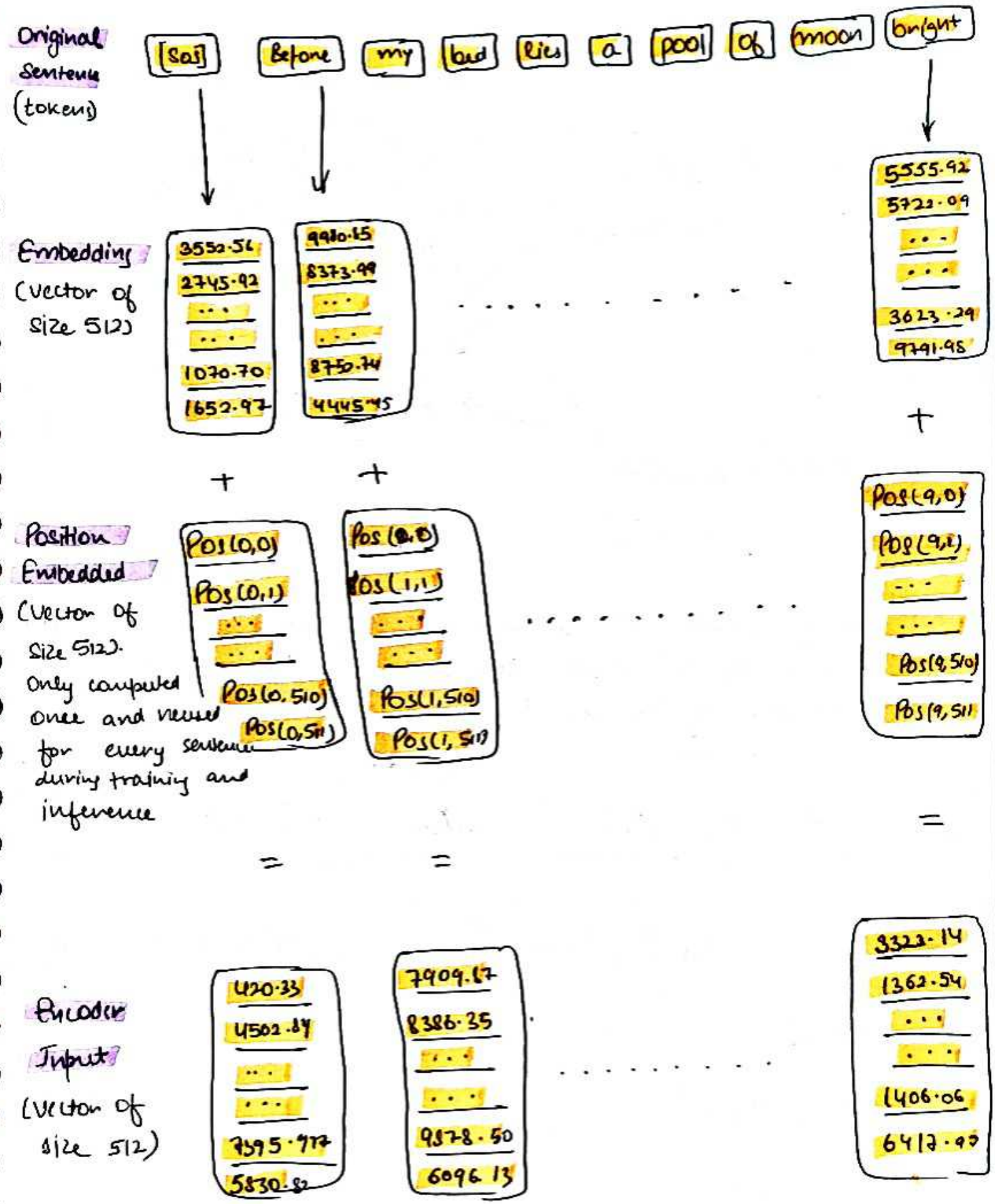
All subword start with "##" . . .



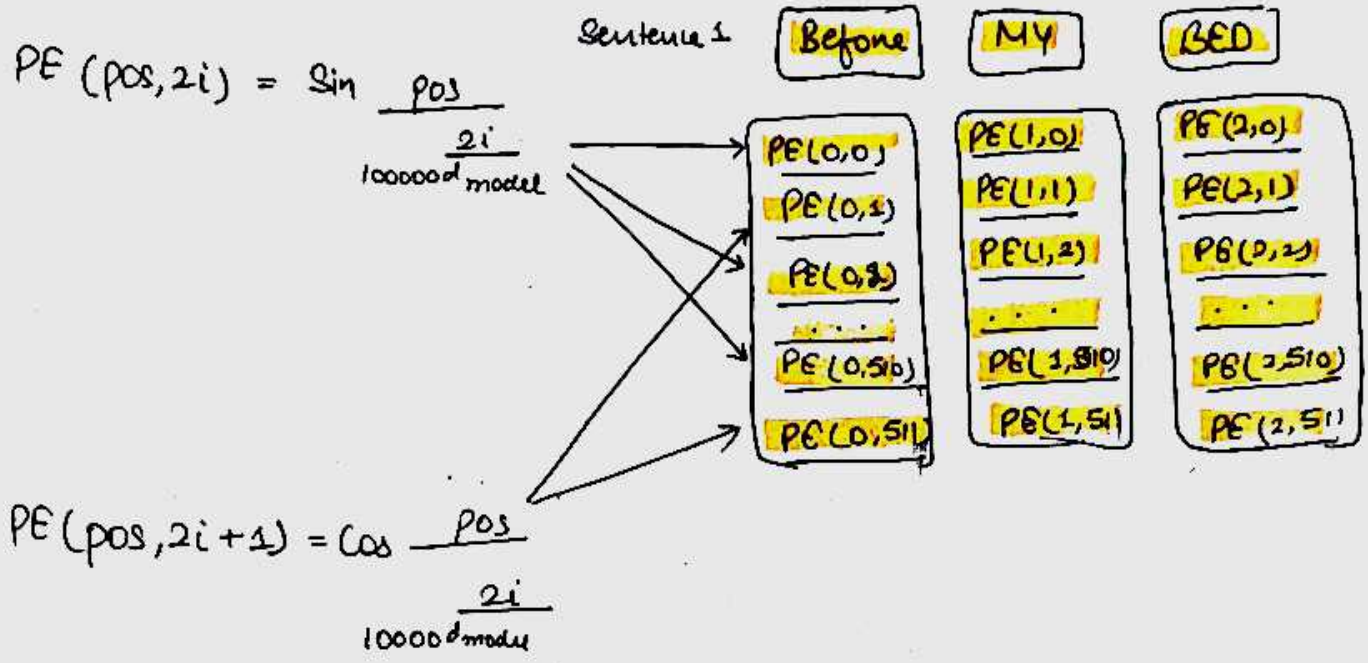
Except for the first subword in a word.



Let's add Positional Encoding

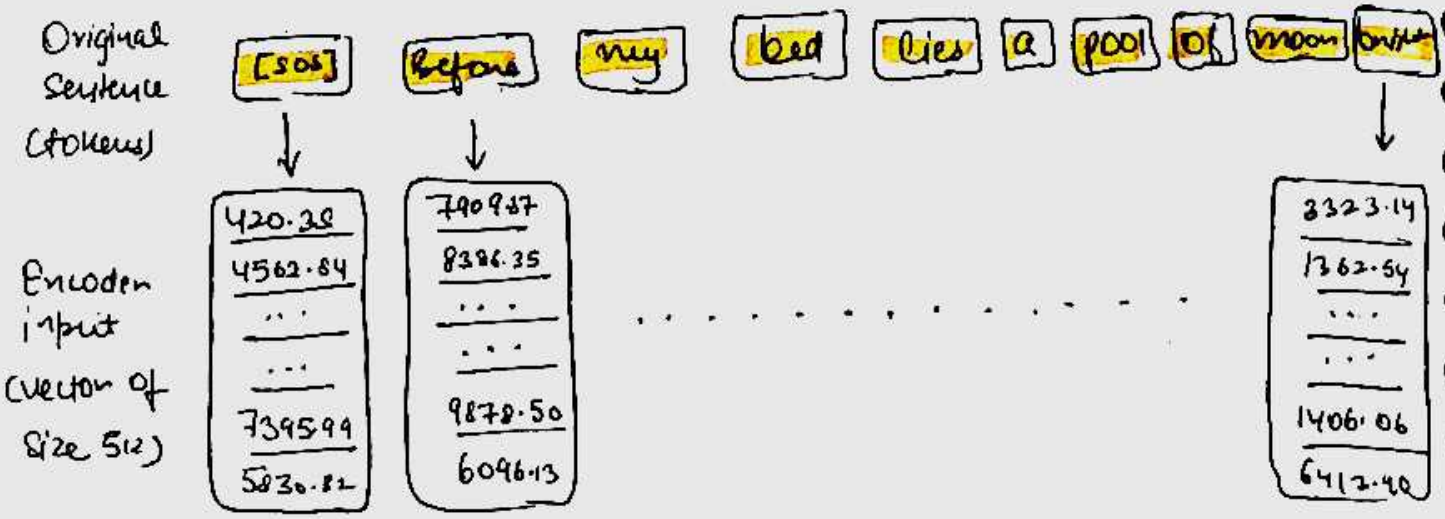


How to compute positional encodings?



We only need to compute the positional encodings once and then reuse them for every sentence, no matter if it is training or inference.

The Self-Attention mechanism: Input



[SOS]	420.38	4562.84	...	...	7395.97	5830.82
Before	7909.87	8386.35	...	...	9878.50	6096.13
my						
bed						
lies						
a						
pool						
of						
moon						
bright	3323.14	1362.54	...	...	1406.06	6417.90

Matrix of shape (10, 512) where each row represent a token in the input sequence.

The Self-attention mechanism : Q, K and V

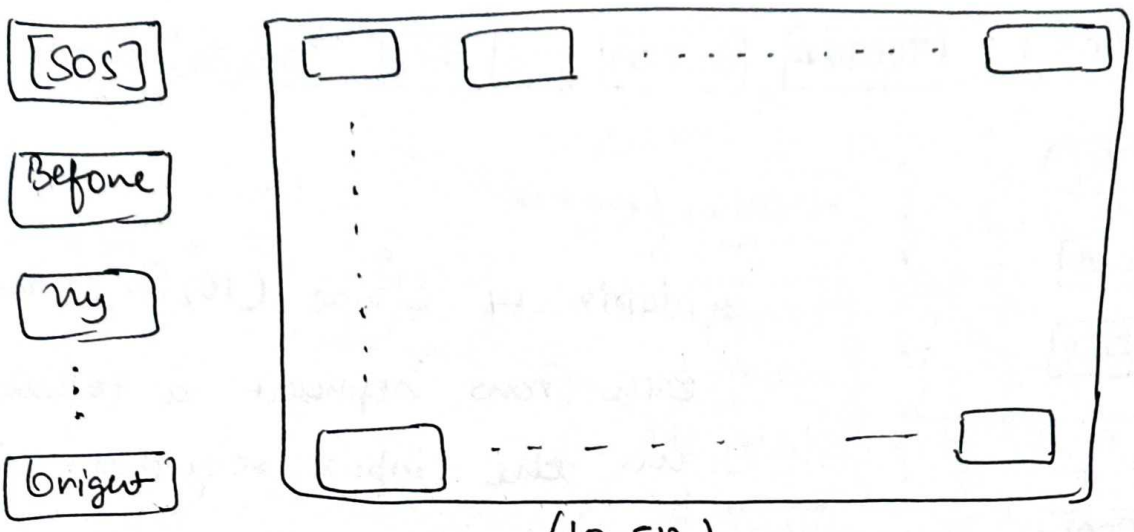
In a Large language Models (LLM) we employ the Self-Attention mechanism, which means the Query (Q), key k and Value (V) are the same matrix.

Query

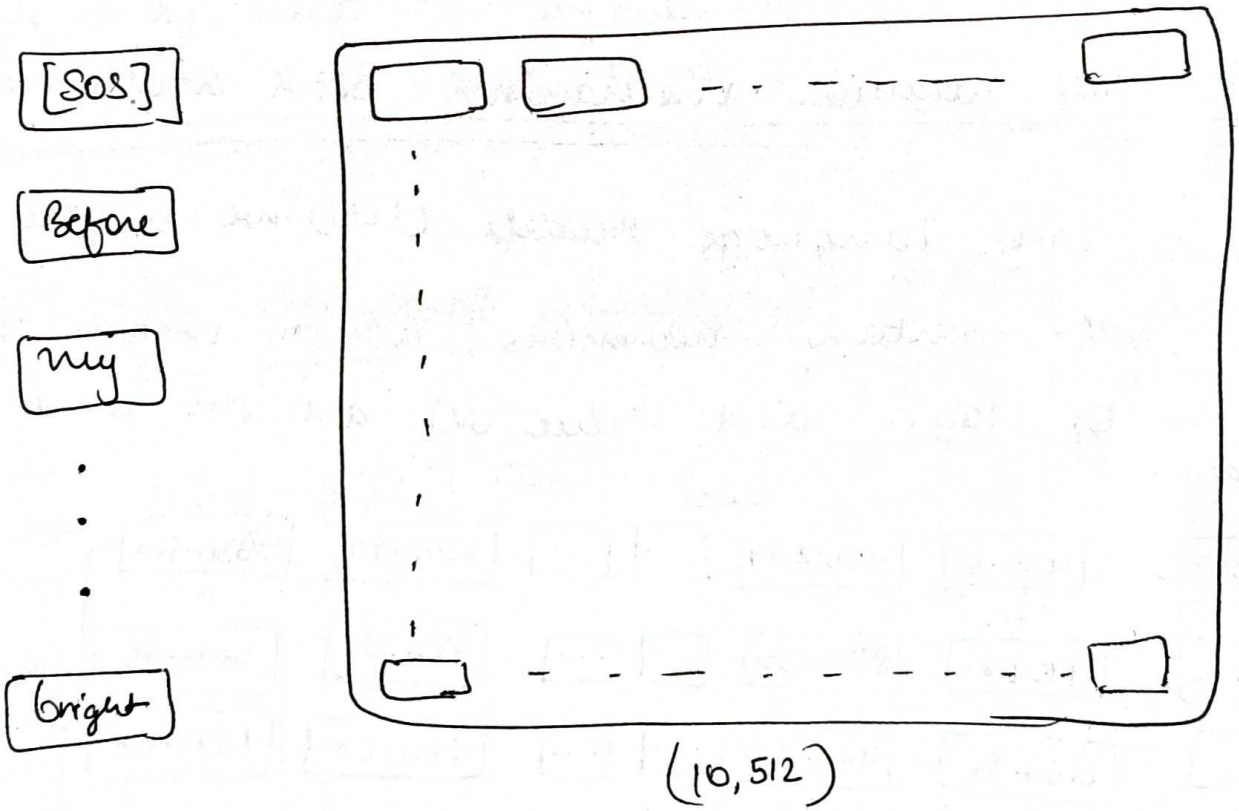
[SOS]	420.38	4562.84	...	...	7395.99	5830.82
Before	7909.87	8386.35	...	...	9878.50	6096.13
my	6167.86	1013.103	...	...	7411.60	1092.17
⋮	⋮					
bright	3323.14	1362.54	...	...	1406.061	6417.90

(10, 512)

Key



Value



The Self-attention mechanism

Self-Attention allows the model to relate words to each other. In our case $d_k = d_{model} = 512$.

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

$$\left[\frac{Q_{(10,512)} \times K^T_{(512,10)}}{\sqrt{512}}\right] \text{Softmax} = \downarrow$$

	[SOS]	Before	my	bed	lies	a	pool	of	moon	bright
SOS	0.62	0.19	0.02	0.02	0.04	0.01	0.00	0.09	0.00	0.02
Before	0.15	0.00	0.00	0.01	0.01	0.00	0.17	0.00	0.67	0.00
my	0.09	0.02	0.56	0.02	-	-	-	-	-	0.03
bed	0.10	0.06	0.03	0.00	-	-	-	-	-	0.04
lies	0.02	0.00	0.00	0.05	-	-	-	-	-	0.06
a	0.01	0.00	0.02	0.02	-	-	-	-	-	0.01
pool	0.00	0.16	0.02	0.00	-	-	-	-	-	0.04
of	0.22	0.00	0.01	0.05	-	-	-	-	-	0.04
moon	0.00	0.67	0.01	0.00	-	-	-	-	-	0.03
bright	0.06	0.00	0.03	0.03	-	-	-	-	-	0.01

The Self-attention mechanism: the reason behind the casual mask.

A language model is a probabilistic model that assign probabilities to sequence of words. In practice, a language model allows us to compute the following:

$$P[\text{"China"} \mid \text{"Shanghai is a city in"}]$$

"China": Next Token

"Shanghai is a city in": Prompt

Shanghai is a city in China, it is also a financial center.

Left content

Right content

To model the probability distribution above, each word should only depend on words that come before it (left content).

We will see later that in BERT we make use of both, the left and the right content.

Self-Attention mechanism : Casual mask

	[SOS]	Before	my	bed	lies	a	Pool	of	moon	bright
SOS	5.45	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞
Before	4.28	2.46	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞
my	8.17	3.56	5.54	-∞	-∞	-∞	-∞	-∞	-∞	-∞
bed	6.71	4.13	6.26	0.79	-∞	-∞	-∞	-∞	-∞	-∞
lies	5.43	7.59	3.91	8.19	9.03	-∞	-∞	-∞	-∞	-∞
a	4.42	4.35	4.84	2.06	3.88	2.57	-∞	-∞	-∞	-∞
Pool	8.36	6.00	2.13	3.11	9.11	8.11	9.00	-∞	-∞	-∞
of	2.21	3.72	8.61	1.01	6.28	9.03	9.00	6.72	-∞	-∞
moon	4.08	6.22	5.09	2.11	5.93	0.11	0.31	8.11	4.20	-∞
bright	6.43	8.88	6.17	4.54	8.13	7.12	5.31	2.09	3.11	1.38

$$\frac{QK^T}{\sqrt{d_k}}$$

So, we use softmax to convert $-\infty$ into 0.00.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$

	SOS	Before	my	bed	lies	a	pool	of	moon	bright
SOS	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Before		0.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
my			0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00
bed				0.00	0.00	0.00	0.00	0.00	0.00	0.00
lies					0.75	0.00	0.00	0.00	0.00	0.00
a						0.48	0.00	0.00	0.00	0.00
pool							0.59	0.00	0.00	0.00
of								0.23	0.00	0.00
moon									0.03	0.00
bright										0.03

(10, 10)

$$\text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

(10, 10) × V (10, 512) = Attention output (10, 512)

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

Introducing BERT

BERT's architecture is made up of layers of encoders of the Transformer model:

- $BERT_{BASE}$
  - 12 encoder layers
  - The size of the hidden size of the feed forward layer is 3072.
  - 12 attention heads.
- $BERT_{Large}$
  - 24 encoder layers
  - The size of the hidden size of the forward layer is 4096.
  - 16 attention heads.

Difference with vanilla Transformer:

- The embedding vector is 768 and 1024 for the two models.
- Positional embeddings are absolute and learnt during training and limited to 512 position.
- The linear layer head changes according to the application

Uses the word piece tokenizer, which also allows sub-word tokens. The vocabulary size is ~ 30k tokens.

BERT vs GPT / LLaMA

BERT stands for Bidirectional Encoder Representation from Transformers.

1. Unlike common language models, BERT does not handle "special tasks" with prompts, but rather, it can be specialized on a particular task by means of fine-tuning.

2. Unlike common language models, BERT was been trained using the left content and the rigert content.

3. Unlike common language models, BERT is not built specially for text generation.

4. Unlike common language models, BERT has not been trained on the Next Token Prediction task, but rather, on the Masked Language Model and Next Sentence Prediction task.

common language models = GPT, LLaMA etc.

Tasks in GPT / LLAMA vs BERT

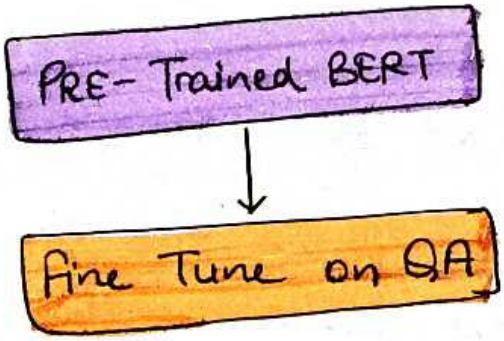
Question Answering in GPT / LLAMA : Prompt Engineering

Content: " Shanghai is a city in china, it is a also financial center, its fashion capital and industrial city".

Question: " What is the fashion capital of china? Answ with one word".

Ans: Shanghai

Question Answering in BERT : Fine Tuning



The importance of left content in human conversations

Left context is used for example in phone conversations.

User : Hello! My internet line is not working, could you send a technician?

Operator : Hello! Let me check. Meanwhile, can you try restarting your Wifi router?

User: I have already restared it but looks like the red light is not going away.

Operator: All right. I'll send someone.

The importance of right content in human conversations

Imagine there's a kid who just boke his mom's favourite necklace. The kid doesn't want to tell the truth to his mom, so he decides to make up a lie.

So, instead of saying directly: "your favourite necklace has broken"

The kid may say: "Mom, I just saw the cat playing in your room and your favourite necklace has broken."

Or it may say: "Mom, aliens came through your window with laser guns and your favourite necklace has broken."

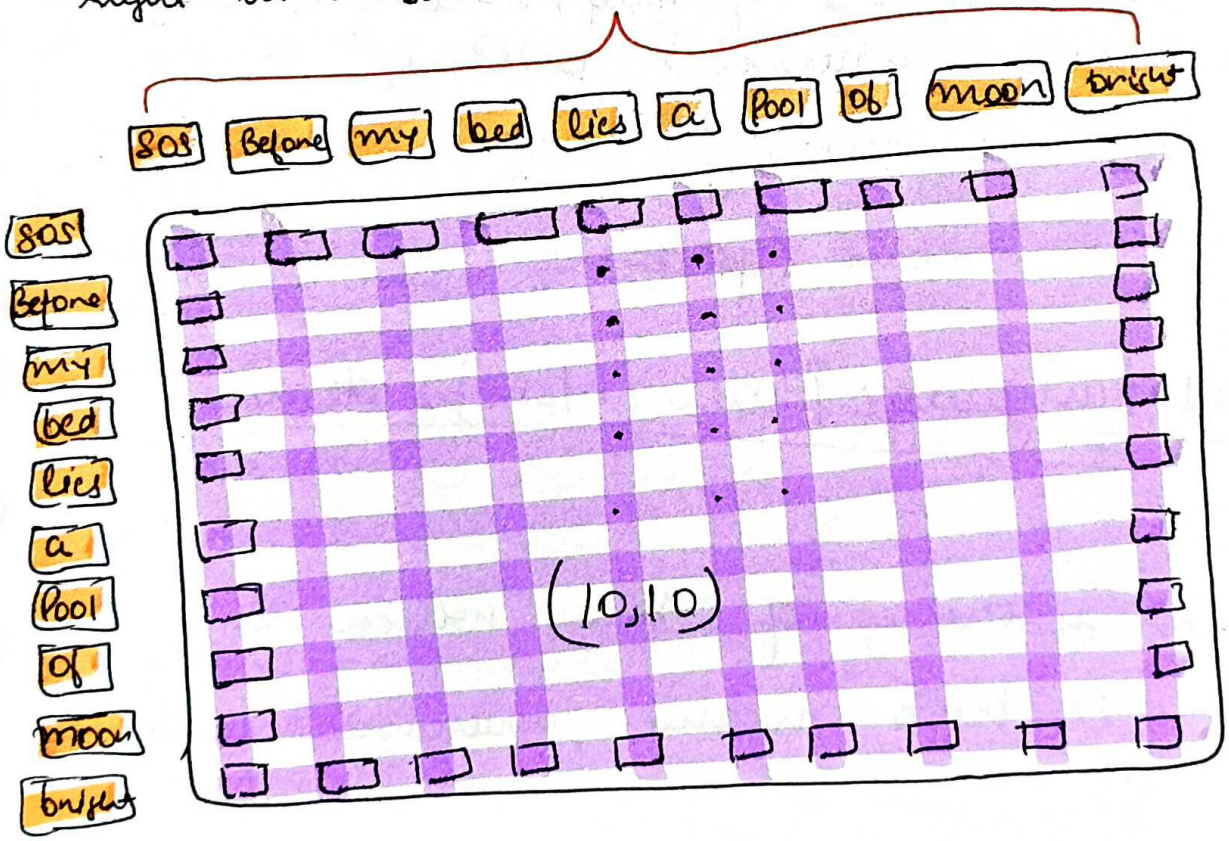
As you can see we conditioned the lie on what we want to say next: whatever the lie we make up, it will be always conditioned on the conclusion we want to arrive to (the necklace being broken).

Left and Right Context in BERT

$$\text{Attention}(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{dk}}\right)V$$

$$\text{Softmax}\left[\frac{Q_{(10,768)} \times K^T_{(768,10)}}{\sqrt{768}}\right] = \downarrow$$

This is the reason it is a Bidirectional Encoder.
Each token "attends" token to its left and tokens to its right in a sentence.



Masked Language Model (MLM)

Also known as the Close task. It means that randomly selected words in a sentence are masked, and the model must predict the right word given the left and right content.

Rome is the Capital of Italy, which is why it hosts many government buildings.

↓ Randomly select one one more tokens and replace them with the special token [MASK]

Rome is the [Mask] of Italy, which is why it hosts many government buildings

↓

Capital

Masked Language Model (MLM): details

Rome is the capitalof Italy, which is why it hosts many government buildings

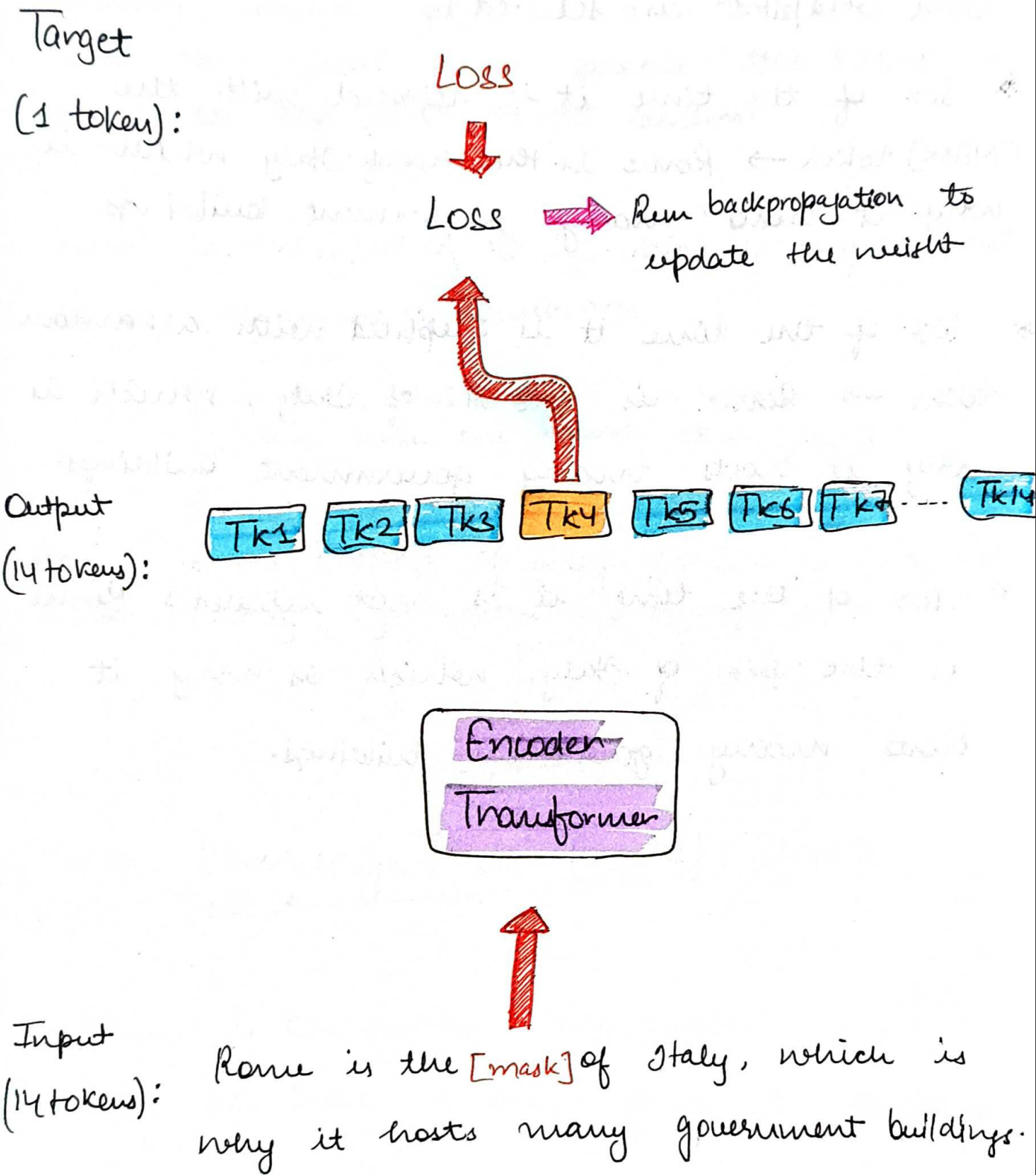
The pre-training procedure selects 15% of the tokens from the sentence to be masked. When a token is selected to be masked (Suppose the word "capital" is selected):

* 80% of the time it is replaced with the [MASK] token → Rome is the [Mask] of Italy, which is why it hosts many government buildings.

* 10% of the time it is replaced with a random token → Rome is the zebra of Italy, which is why it hosts many government buildings.

* 10% of the time it is not replaced → Rome is the capital of Italy, which is why it hosts many government buildings.

Masked Language Model (MLM): training



Next Sentence Prediction (NSP)

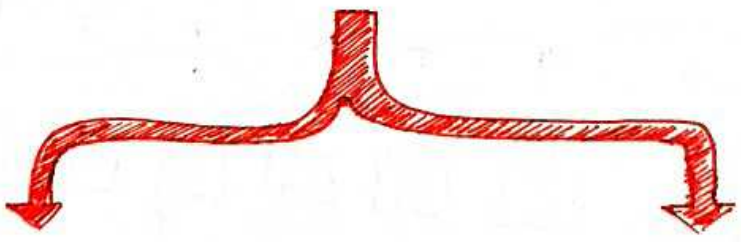
Many downstream applications (for example choosing the right answer given 4 choices) require learning relationships between sentence rather than single tokens, that's why BERT has been pre-trained on the Next Sentence Prediction tasks.

Sentence A ⟶ Before my bed lies a pool of moon bright
I could imagine that it's frost on the ground

Sentence B ⟶ I look up and see the bright shining moon
Bowing my head I am thinking of home

SentenceA = Before my bed lies a pool of moon bright

SentenceB = I look up and see the bright shining moon



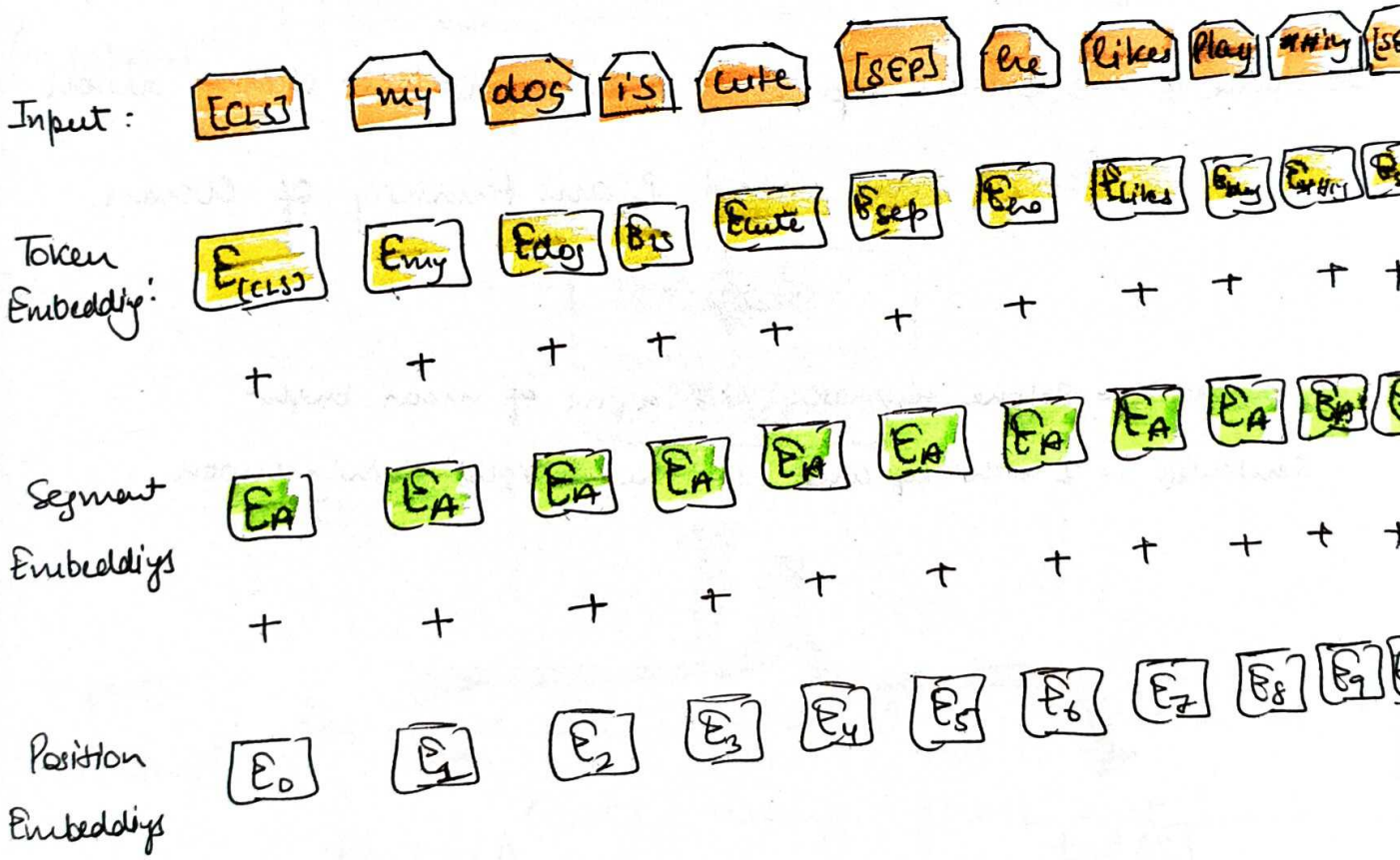
IsNext

NotNext

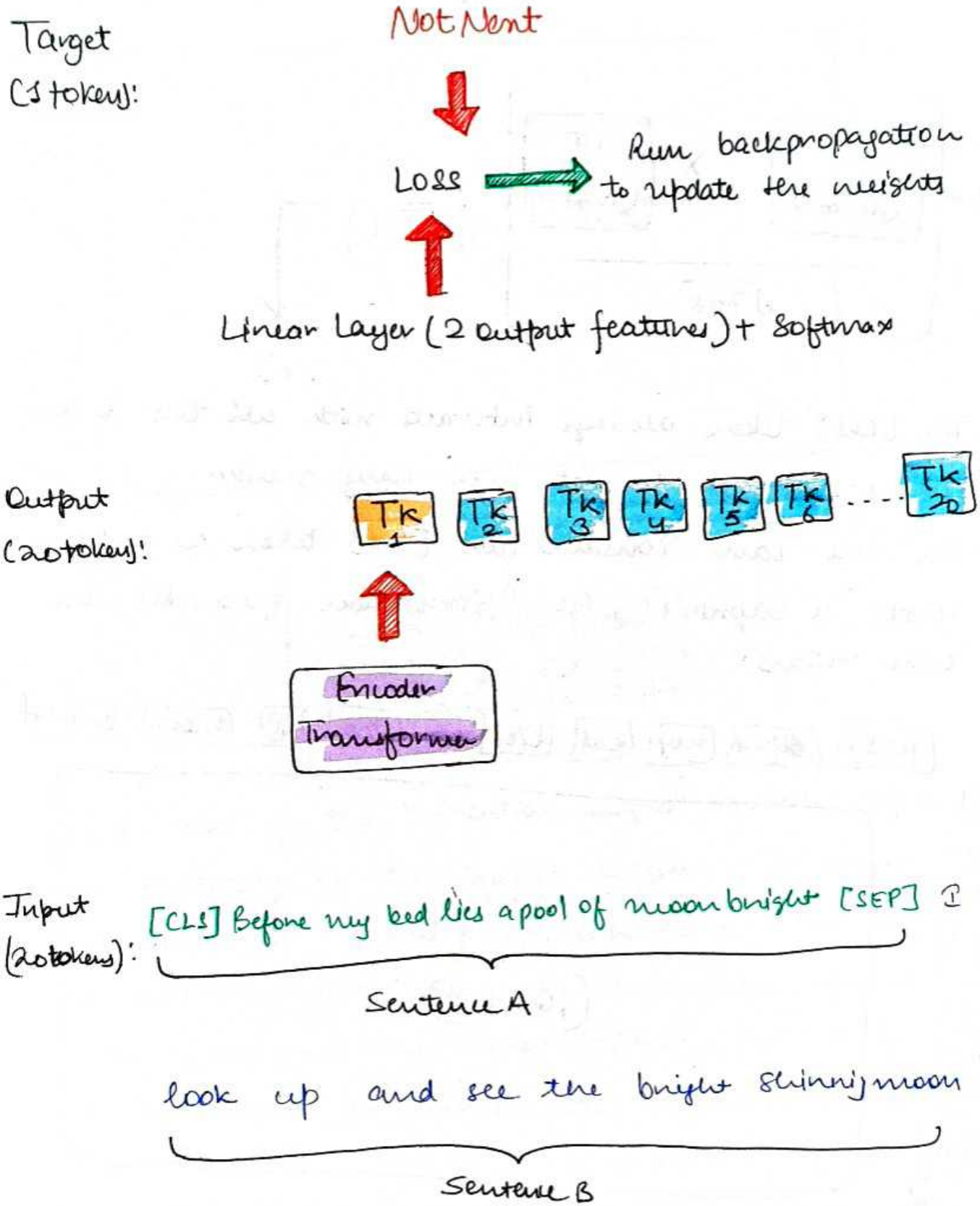
- 50% of the time, we select the actual next sentence.
- 50% of the time we select a random sentence from the text.

Next Sentence Prediction (NSP): Segmentation Embeddi

Given the sentence A and the sentence B, how can BERT understand which tokens belongs to the sentence A and which to the sentence B?
We introduced the segmentation embeddings!
We also introduce two special tokens: [CLS] and [SEP]



Next Sentence Prediction (NSP): training



[CLS] token in BERT

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

$$\text{Softmax}\left[\frac{Q_{(10,768)} \times K^T_{(768,10)}}{\sqrt{768}}\right] = \downarrow$$

The [CLS] token always interacts with all the other tokens, as we do not use any mask.

So, we can consider the [CLS] token as a token that "captures" the information from all the other tokens.

[CLS] Before my bed lies a pool of moon bright

CLS
Before
my
bed
lies
a
pool
of
moon
bright

(10, 10)

[CLS] token : Output Sequence

CLS | Before | my | bed | lies | a | pool | of | moon | bright

CLS
Before
my
bed
lies
a
pool
of
moon
bright

(10, 10) X

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

V (10, 768) = Attention Output (10, 768)

Each row of the "Attention Output" matrix represents the embedding of the output sequence: It captures not only the meaning of each token, not only its position, but also the interaction of each token with all the other tokens, but only the interaction for which the softmax scores is not zero. All the 512 dimension of each vector only depend on the attention scores that are non-zero.

Text Classification

Text classification is the task of assigning a label to a piece of text. For example imagine we are running an Internet provider and we recieve complaints from our customers. We may want to classify requests coming from users as hardware problems, software problems or billing issues

My router's led is not working. I tried changing the power socket but still nothing.

Hardware

My router's web page doesn't allow me to change password anymore... I tried restarting it but nothing.

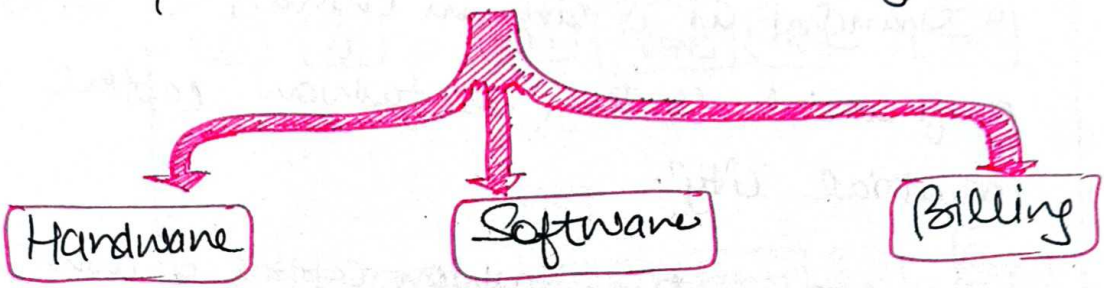
Software

In this month's bill I have been charged 100$ instead of the usual 60$, why is that?

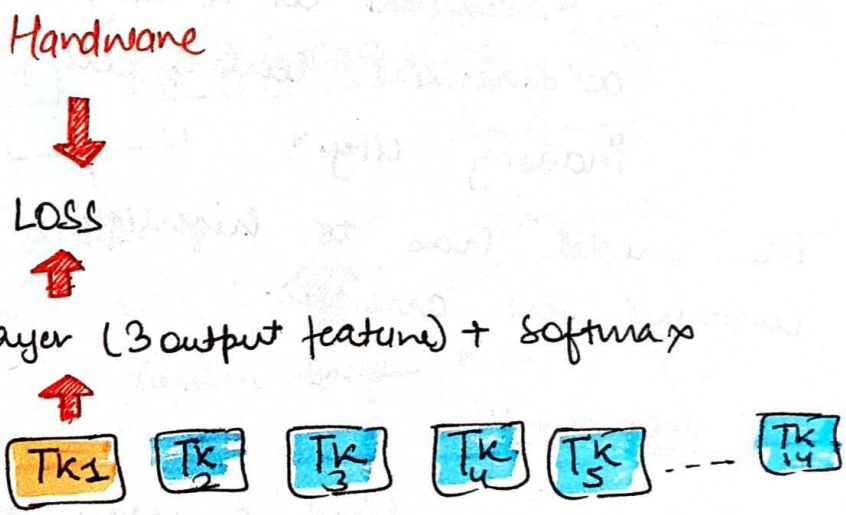
Billing

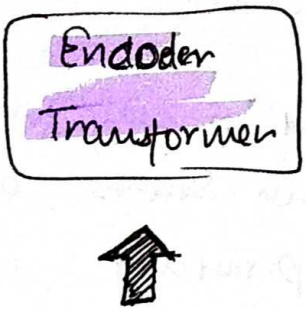
Text Classification: training

My router's led is not working. I tried changing the power socket but still nothing.



Target (1 tokens):





Input (16 tokens): [CLS] My router's led is not working. I tried changing the power socket but still nothing.

Question Answering

Question answering is the task of answering given a content.

Content: "Shanghai is a city in China, it is also a financial center, its fashion capital and industrial city".

Question: "what is the fashion capital of china?"

Answer: "Shanghai is a city in china, it is also a financial center, its fashion capital and industry city."

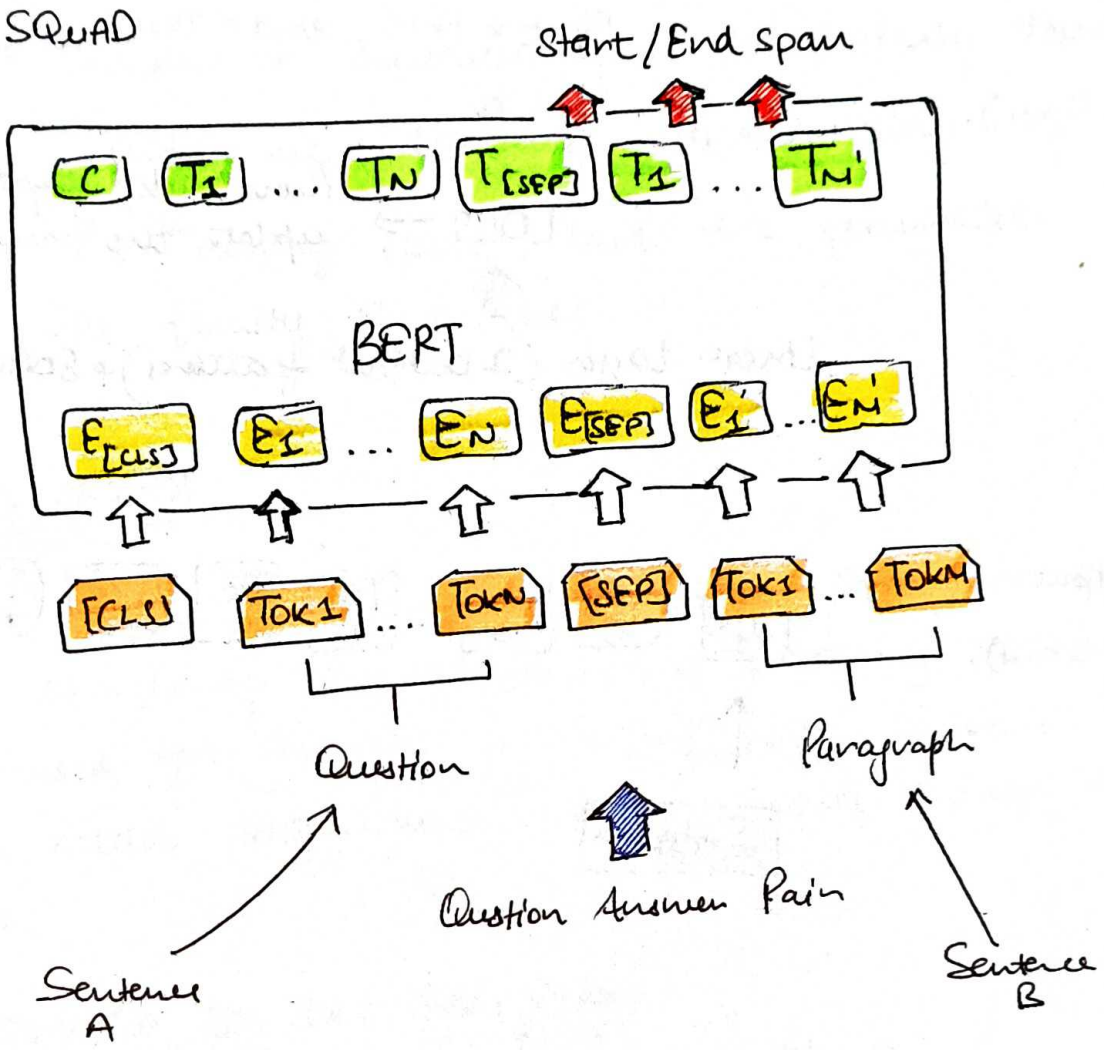
The model has to highlight the part of text that contains the answer.

Two problems:

1. We need to find a way for BERT to understand which part of the input is the content, which one is the questions.

2. We also need to find a way for BERT to tell us where the answer starts and where it ends in the content provided.

Question Answering : Sentence A and B



Question Answering : Start and end positions