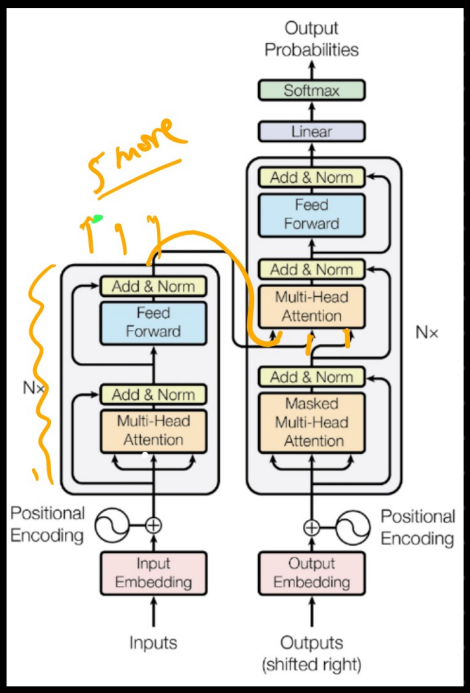
Transformer

Transformer complete summary



Encoder Side:
1. input sequence
2. input embedding
3. positional encoding
4. self-attention, multi-head attention
5. add and layer norm
6. feed forward network(artificial neural network)
7. then again addition(residual connection) and layer normalization
8. Nx (multiple encoder block(in original research paper there was 6 block and all the block were identical means in every block same process happening))

Decoder Side:
1. output sequence[shifted right means one additional token will add]
2. output sequence embedding[this is word embedding and the dimension of each word is 512]
3. positional encoding for making sure the position of word since we are passing the input parallelly
4. masked multi-head attention
5. addition and layer normalization
6. residual connection
7.multihead attention[it is a cross attention since we are getting input from mask multiheaded attention layer and from encoder as well]
8. then again addition(residual connection) and layer normalization
9. passing output to feed forward network(artificial neural network)
10. addition(residual) and normalization
11. linear activation function
12. then SoftMax activation function
13. final output in the form of probability
14. multiple block of decoder(6 encoder according to the research paper)

this transformer architecture have two sides first at the time of training and second at the time of inferencing

↳ Prediction (testing)

1. Positional encoding
2. Self attention & multihead attention (Self-)
3. Residual & layer norm.
4. mask self attention
5. cross attention

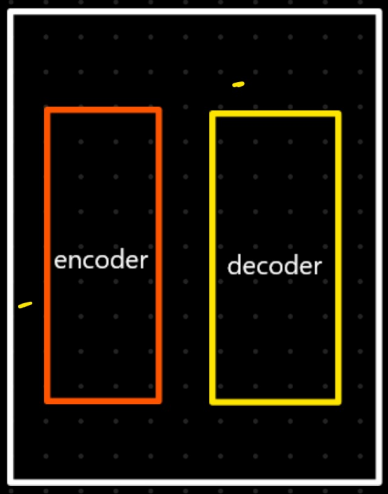
} tomorrow (decoder)

(translation)

GPT → GPT3 3.5 & 4V (generation)

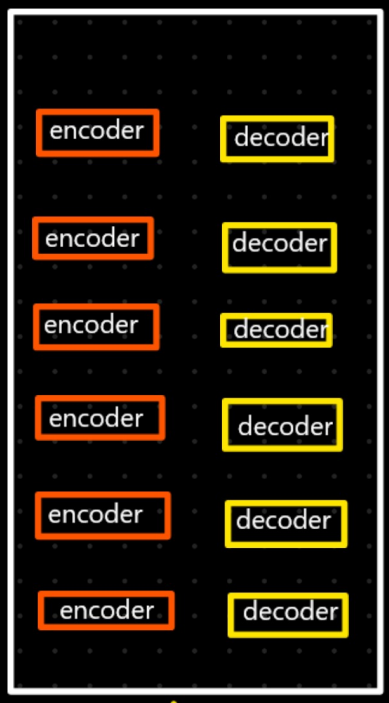
1. UCMFIT — lang. modelling → next word pred
2. Attention all you need

Lets Start with encoder



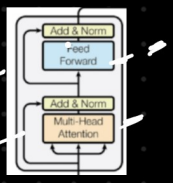
Original Paper

hyperparameter



transformer

ANN



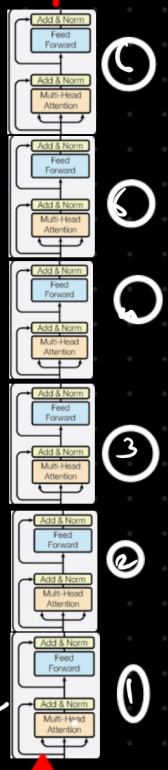
feed forward NN

self attention

encoder

output

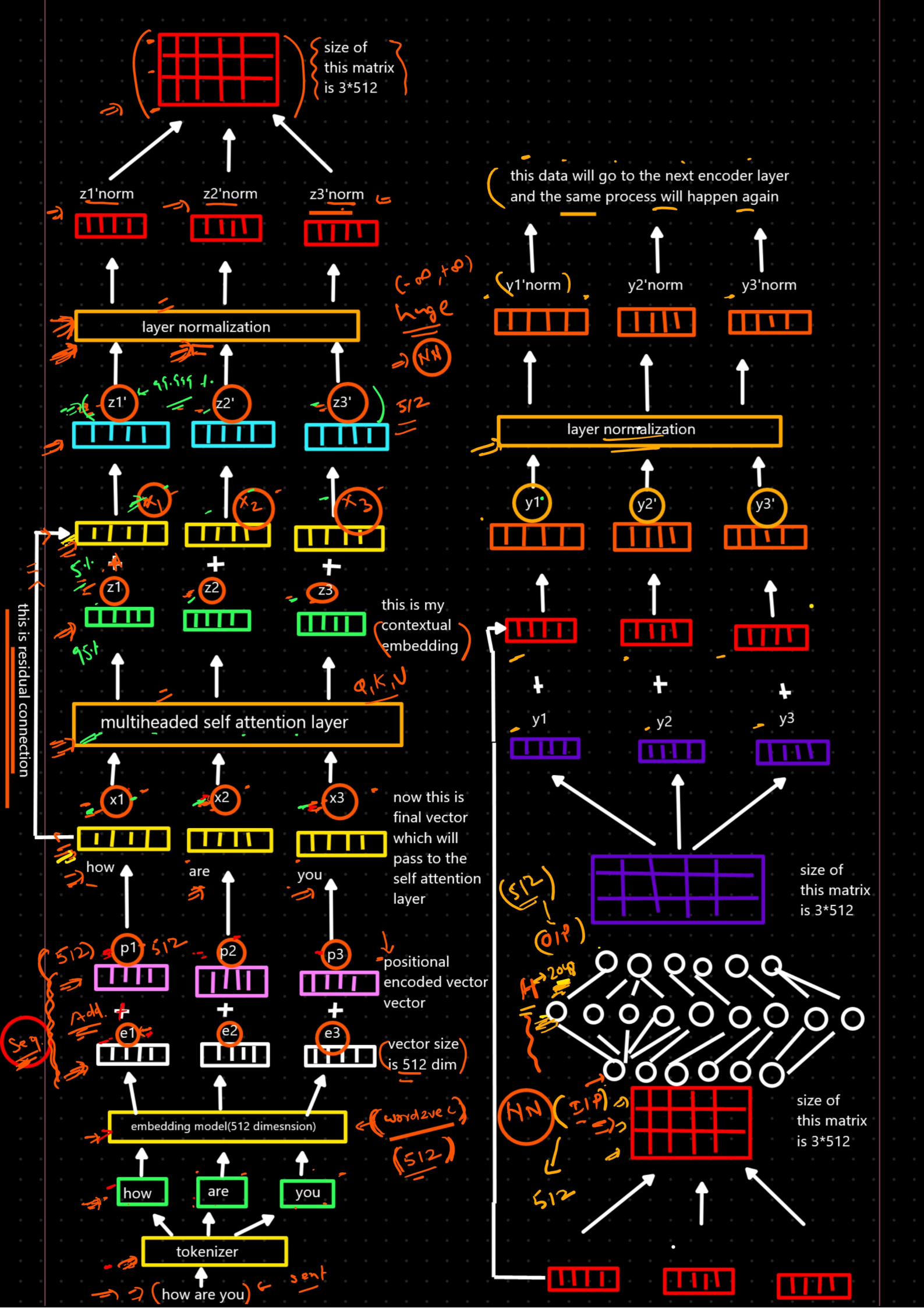
this output we will pass to the decoder



input

how are you

this input could be one single sentence or multiple sentence[batch data]


size of this matrix is 3*512
z1'norm
z2'norm
z3'norm
layer normalization
z1'
z2'
z3'
x1
x2
x3
z1
z2
z3
this is my contextual embedding
multiheaded self attention layer
this is residual connection
x1
x2
x3
how
are
you
now this is final vector which will pass to the self attention layer
p1
p2
p3
positional encoded vector
e1
e2
e3
vector size is 512 dim
embedding model(512 dimesnsion)
how
are
you
tokenizer
(how are you)
this data will go to the next encoder layer and the same process will happen again
y1'norm
y2'norm
y3'norm
layer normalization
y1'
y2'
y3'
y1
y2
y3
size of this matrix is 3*512
size of this matrix is 3*512

here 3 general questions → research Paper ✗

1. why they have used residual connection
2. why used feed forward neural network
3. why they took 6 encoder block

Residual Conn → skip connection = stable training

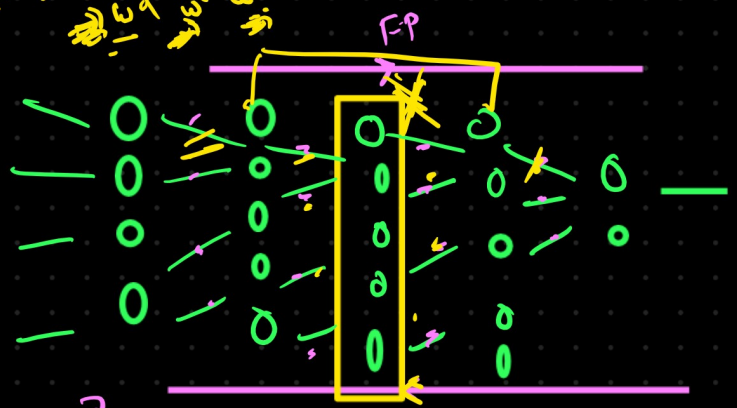
Resnet ⇒ skip connection

vanishing gradient Problem

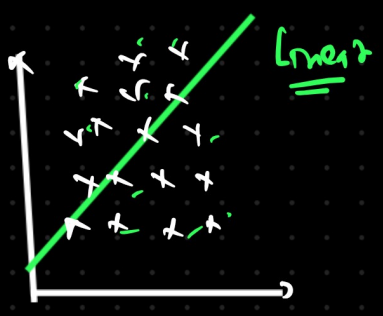
① embedding → self attention → (Contextual embedding) + (original embedding) ⇒ (Super info) — info rich

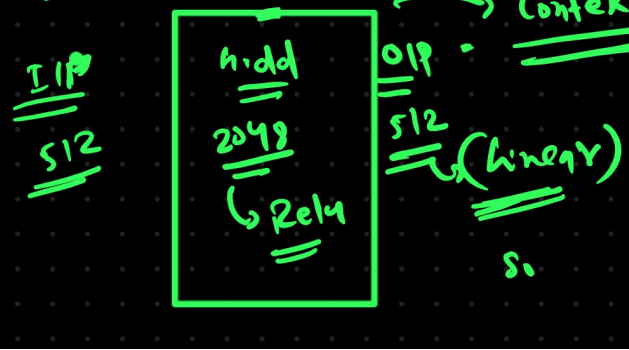
F-P

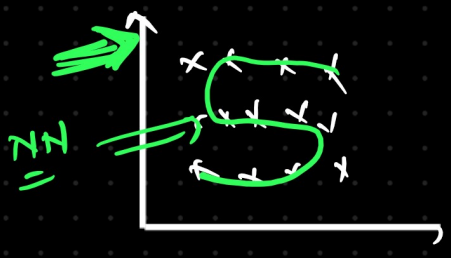
② training ⇒ backpropagation (weight) ⇒ self-attention w_q w_v w_k



② ANN ⇒ Non-linear information







Residual Connection → Skip connection → Identical conn

FP ⇒ original embedding + Contextual embedding ⇒ (info rich)
(BP) ⇒ Self attention

training ⇒ Skip layer (weight) ⇒ vanishing grad ⇒ stable training

⇒ Attn ⇒ Non-linearity ⇒ hidden layer ⇒ Relu ⇒ Non linear
↓
(more pattern)

o/p ⇒ linear activation fn

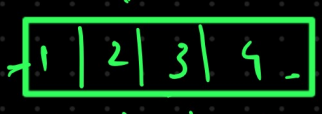
Why 6 encoder & 6 decoder ⇒ hyperparameter

! experimental

(Concat)

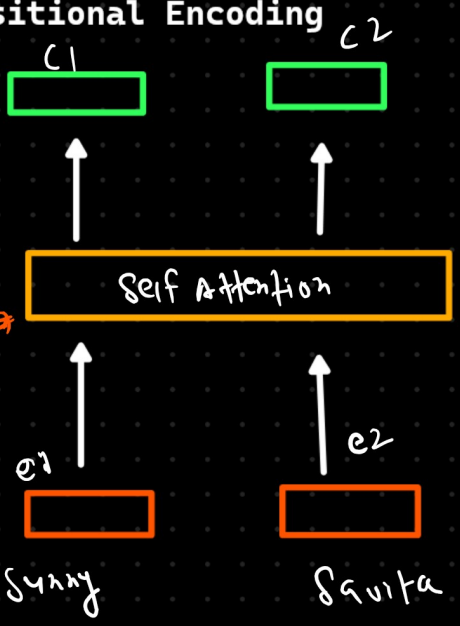
dimension ⇒ (increasing)

Addition



← New vector

Positional Encoding



$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{2i/d_{model}}})$$

$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{2i/d_{model}}})$$

Sunny Savita

(sequentially) ⇒ RNN → t=1 Sunny
t=2 Savita

Parallely

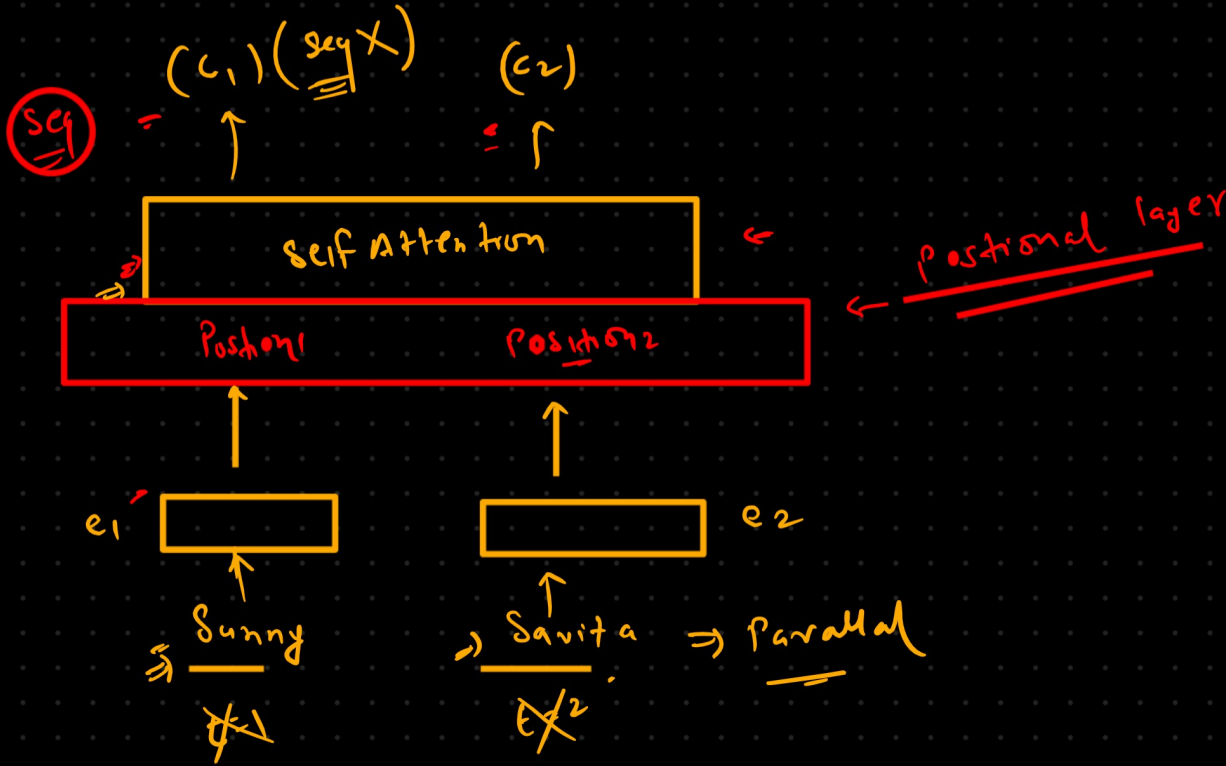
Sunny Savita

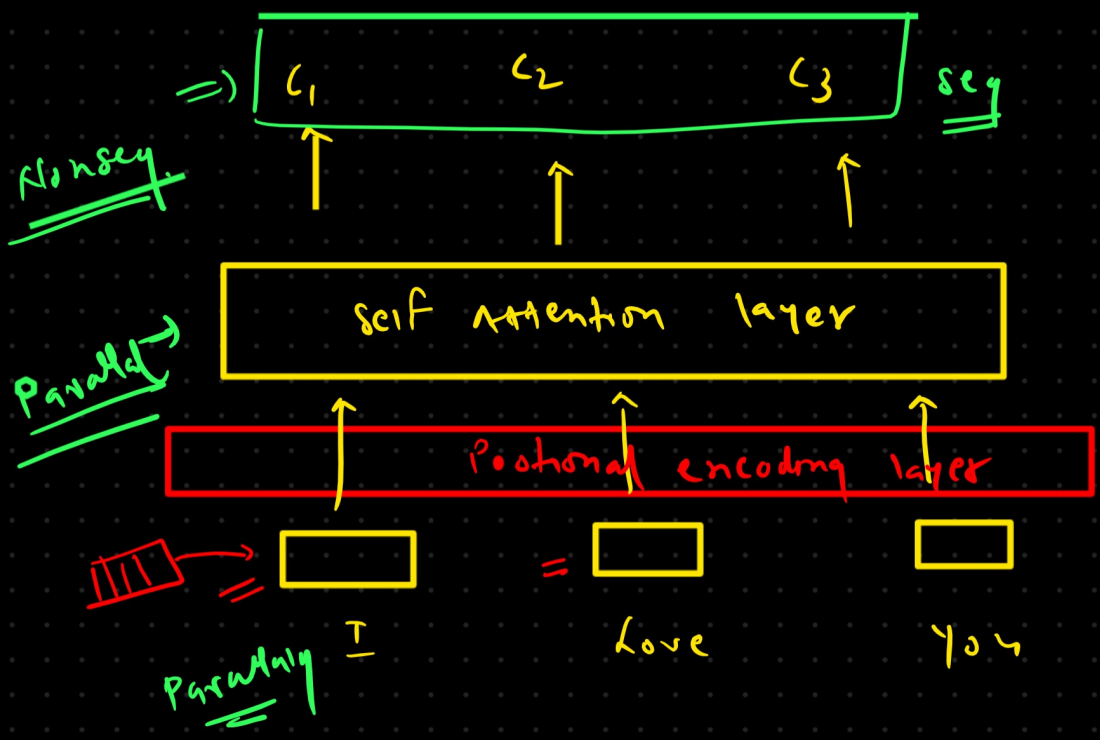
self attention(seq)
⇒ parallel.

1. NLP basics
2. history (RNN, GRU, LSTM, ERD, ERD Attention)
3. Transformer
4. Pytorch
5. transformer Code in Pytorch.
   - text
   - multimodal
   - Audio based
   - video based
6. API (OpenAI, Gemini, huggingface)
7. Finetuning (Bert, Llama, mistral, GPT ....)
   - quantization
   - lora, qlora
   - Preference optimization → RLHF
   - framework
   - instruction
   - SFT (TRL)

Seq → Positional = vector

✗ Sunny is GenAI taking & teaching transform. ✗





(Self Attention) → (masked self attention) → (cross Attention) ⑧

weight matrix — little difference — 1.5 lacs

↓

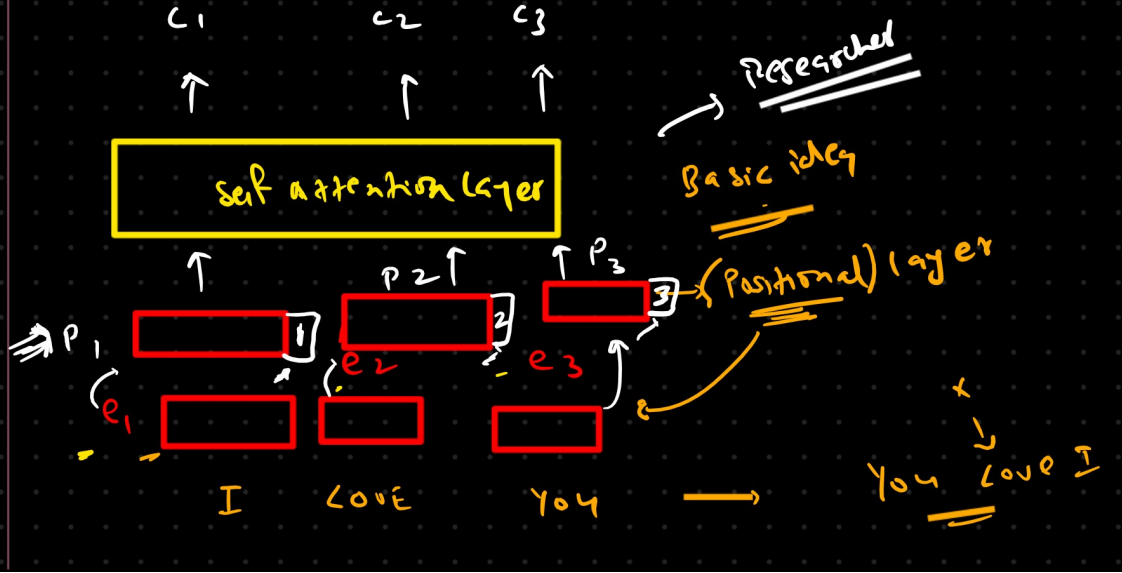
multiple weight matrix

① q, K, V

↓

W

⇒ (decoder)



Passing → book ⇒ 3 word 3L 30L

X

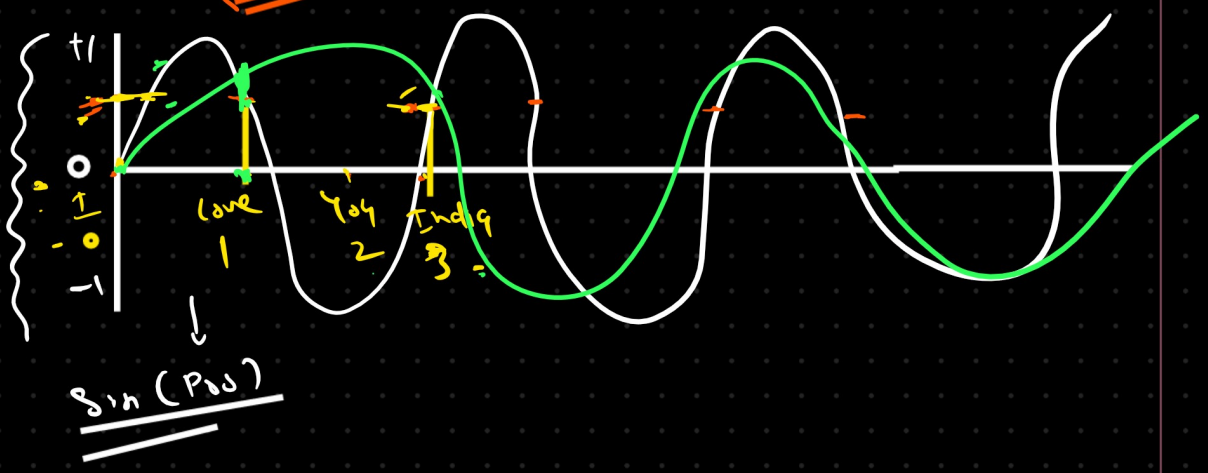
desperate

(MH) (Self Attention)

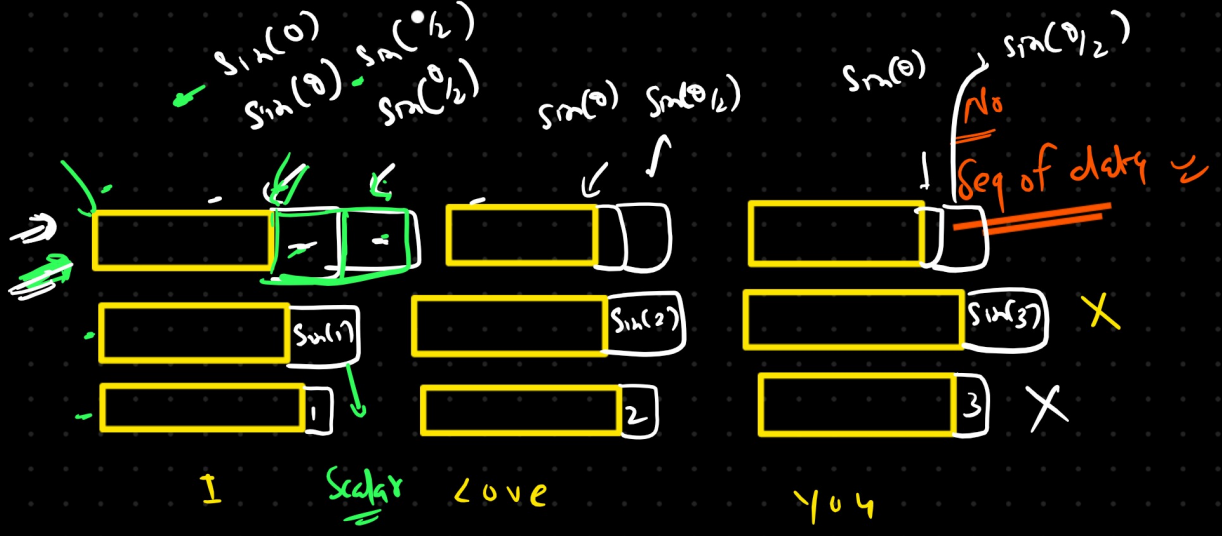
X

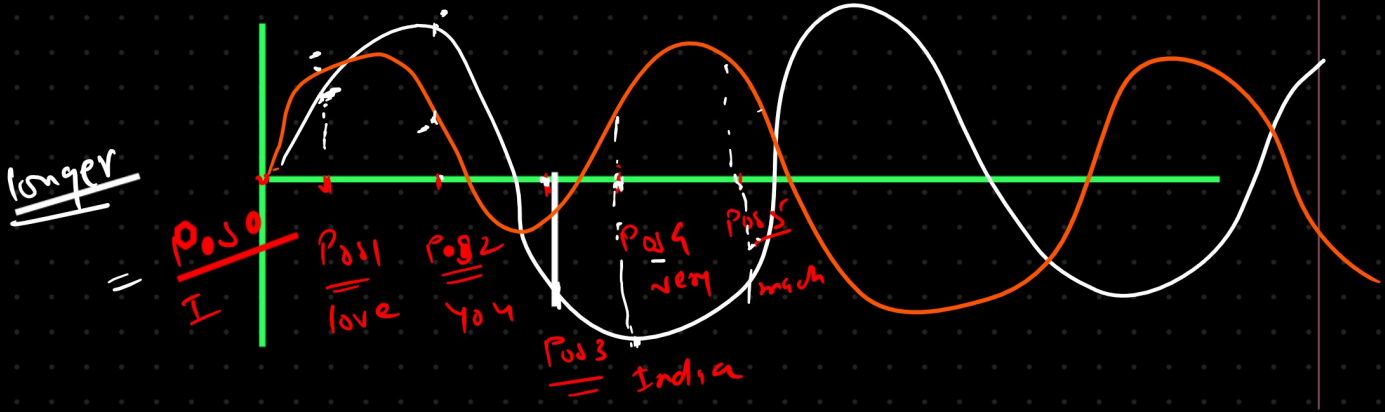
(Periodic)

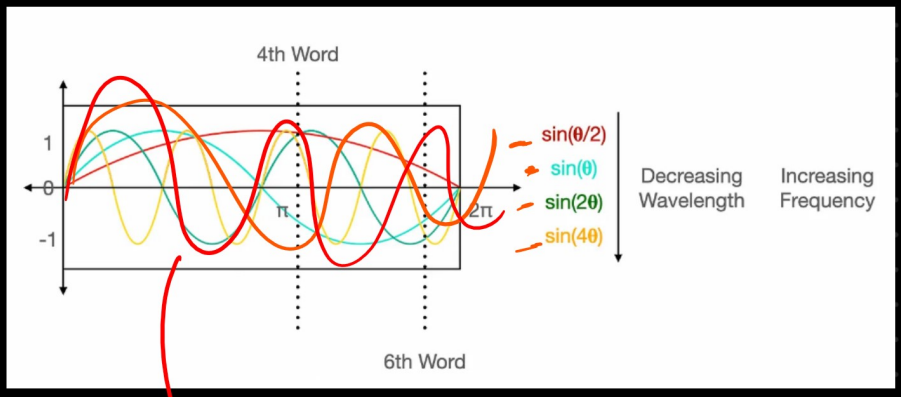
Sin



6th Word



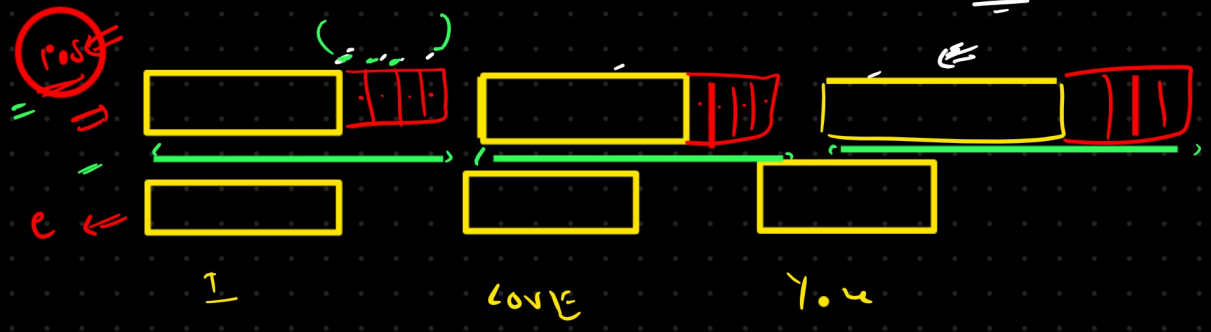




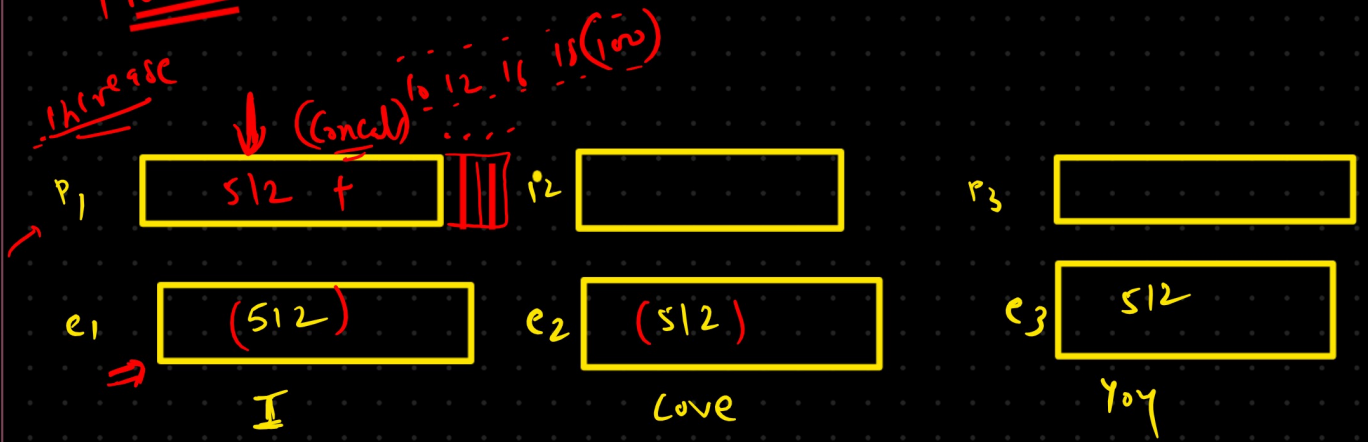
4 more value

very very long sent

cos(0)



Problem (Researcher)



$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{2i/d_{model}}})$$
$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{2i/d_{model}}})$$

→ 512 (manually 512) — 6

seq. of data

P_1 = 512 + e_1 512 → I

P_2 = 512 + e_2 512 → love

P_3 = 512 + e_3 512 → you

Pos 0 → great

Pos 1 → india

model word2vec dim 6

e_1 → 0 1 0 1 0 1 → P_1

x_1

dim 6 e_2 → 0.8 0.5 0.04 0.9 0.1 0.99 =

Pos = 1

for i = 0 → Pos = 0

$PE(0,0) = \sin\left(\frac{0}{10000^{2\times 0/6}}\right) = \sin(0) = 0$

$PE(0,1) = \cos\left(\frac{0}{10000^{2\times 0/6}}\right) = \cos(0) = 1$

For i = 1

$PE(0,2) = \sin\left(\frac{0}{10000^{2/6}}\right) = \sin\left(\frac{0}{10000^{1/3}}\right) = 0$

$PE(0,3) = \cos\left(\frac{0}{10000^{2/6}}\right) = \cos\left(\frac{0}{10000^{1/3}}\right) = 1$

For i = 2

$PE(0,4) = \sin\left(\frac{0}{10000^{4/6}}\right) = \sin\left(\frac{0}{10000^{2/3}}\right) = 0$

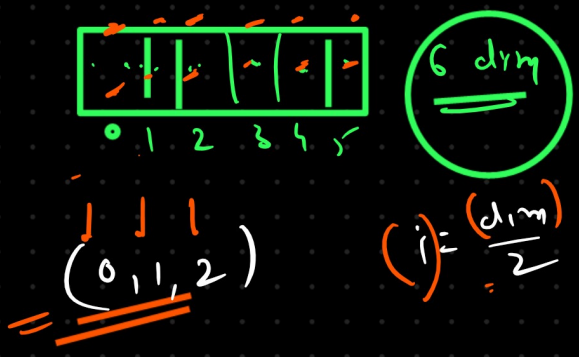
$PE(0,5) = \cos\left(\frac{0}{10000^{4/6}}\right) = \cos\left(\frac{0}{10000^{2/3}}\right) = 1$

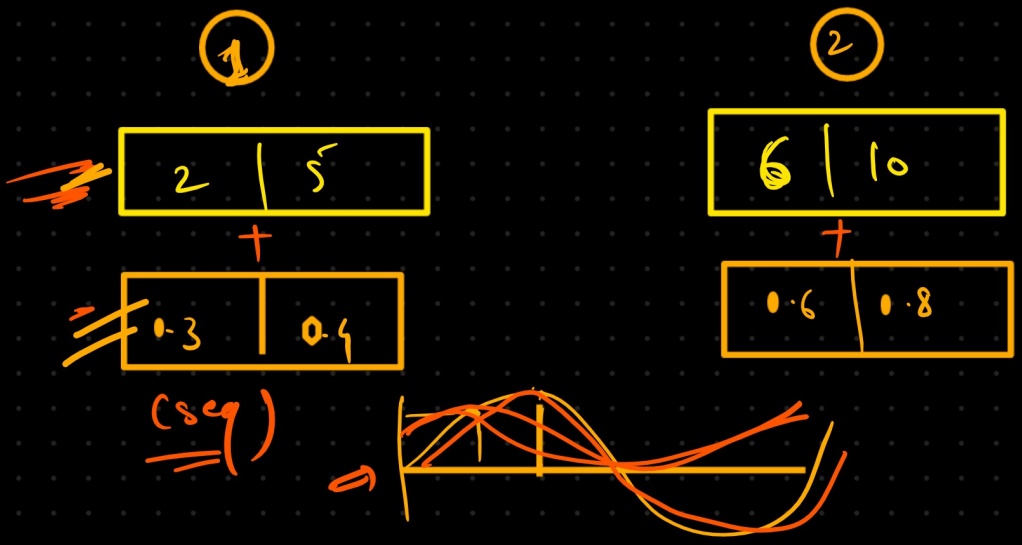
0

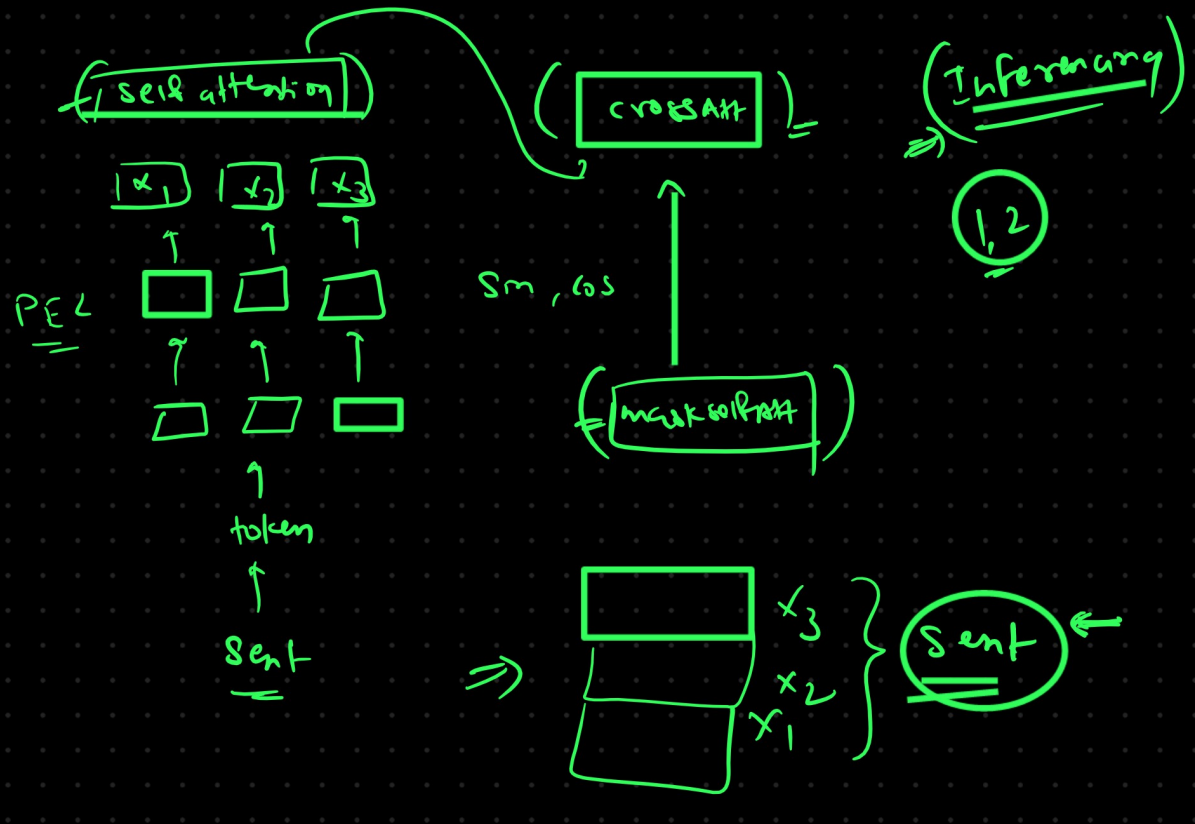
$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{2i/d_{model}}})$$
$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{2i/d_{model}}})$$

i = 0, 1, 2

PE(1,0)
PE(1,1)
PE(1,2)
PE(1,3)
PE(1,4)
PE(1,5)


6 dim
0 1 2 3 4 5
(0, 1, 2)
(i) = (dim)/2


1
2 | 5
0·3 | 0·4
2
6 | 10
0·6 | 0·8
(seq)


self attention
CROSSATT
Inferencing
1, 2
PE
Sm, cos
token
Sent
Sent
x3
x2
x1