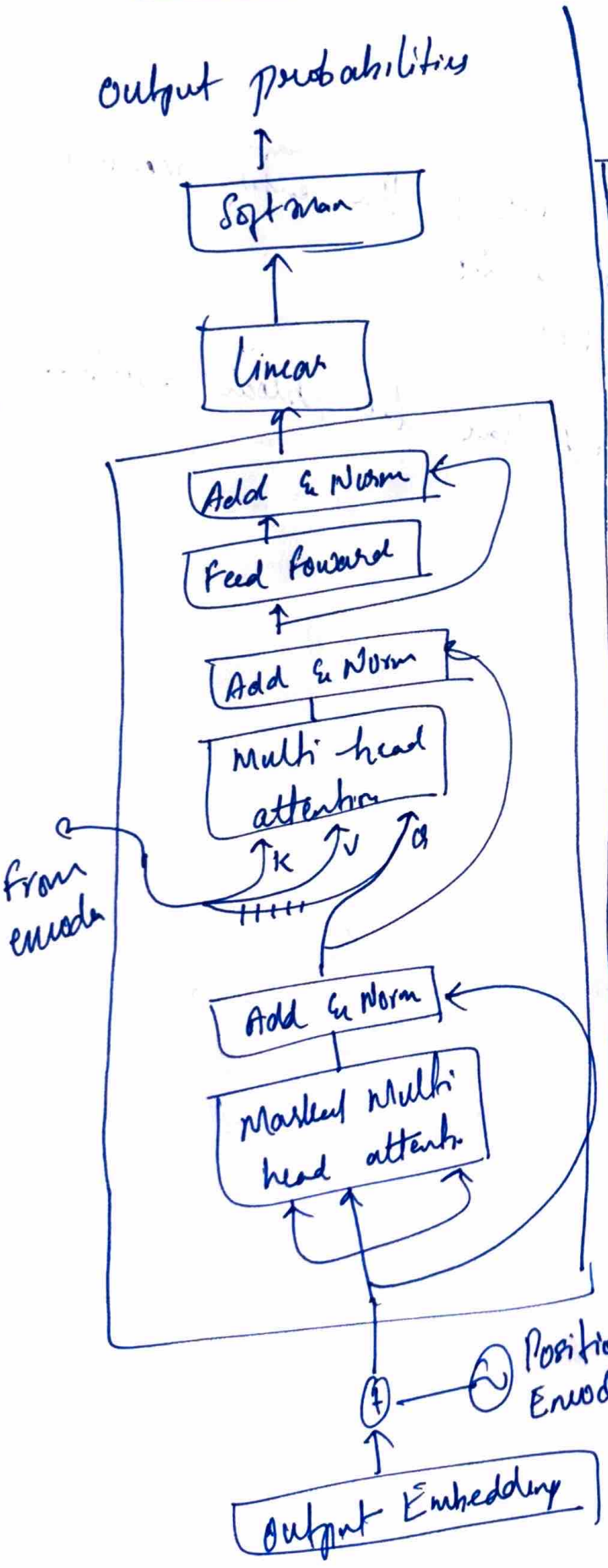
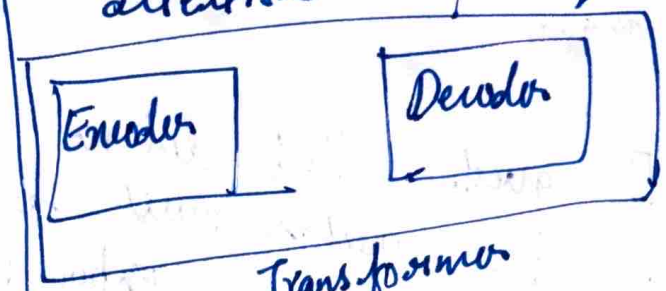
23/5/24 - Friday

Transformer Decoder Architecture



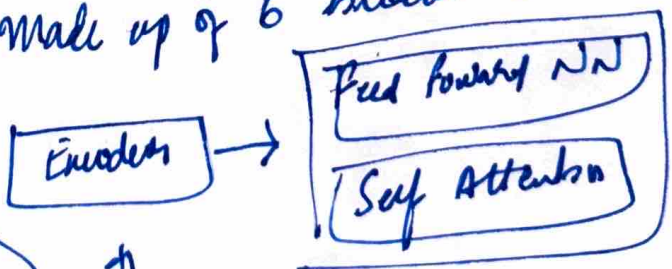
Self attention and cross attention → Previous



Transformer

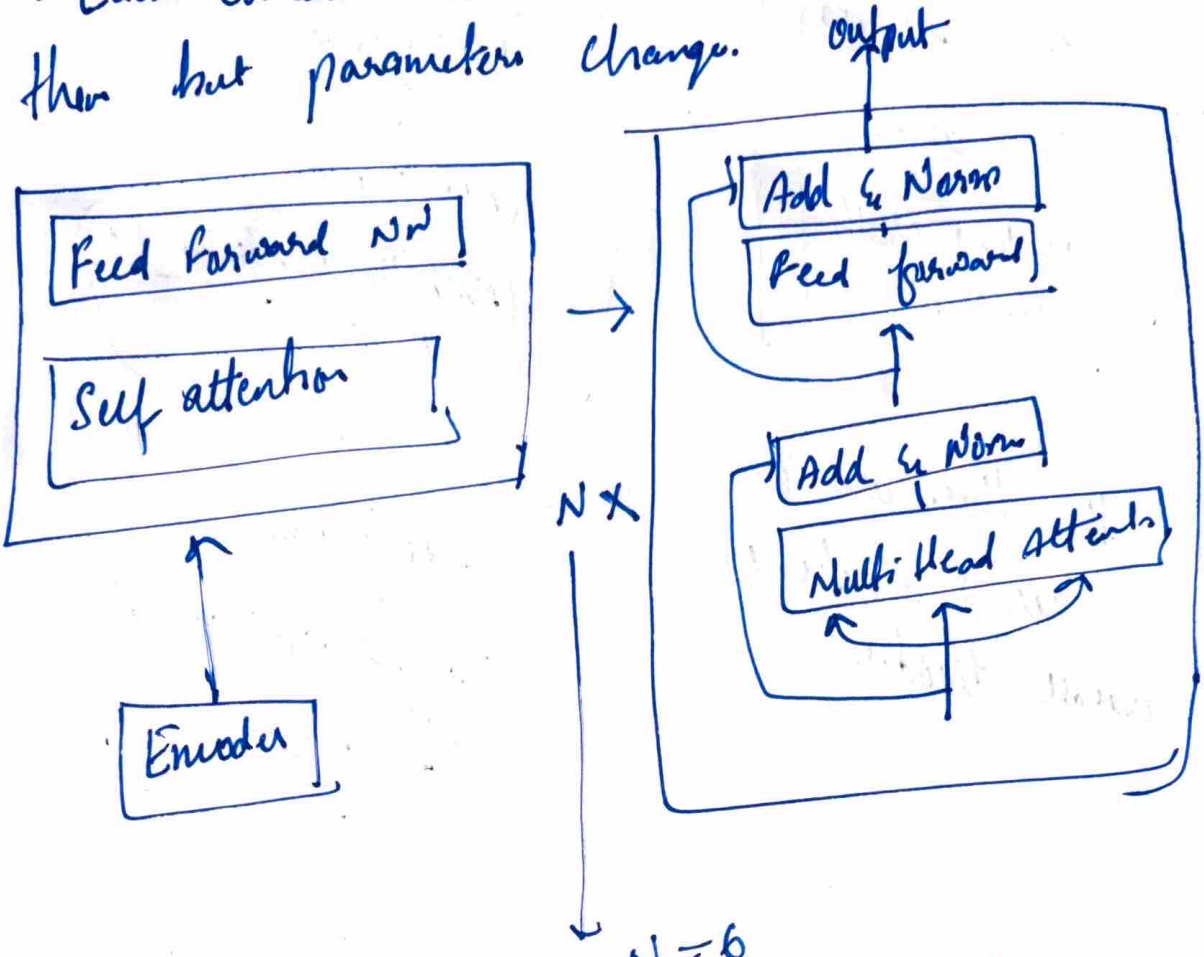
Encoder → encodes input
Decoder → decode the encoded input

Encoder and Decoder are made up of 6 blocks each



In each encoder, it's made up of these 2 blocks

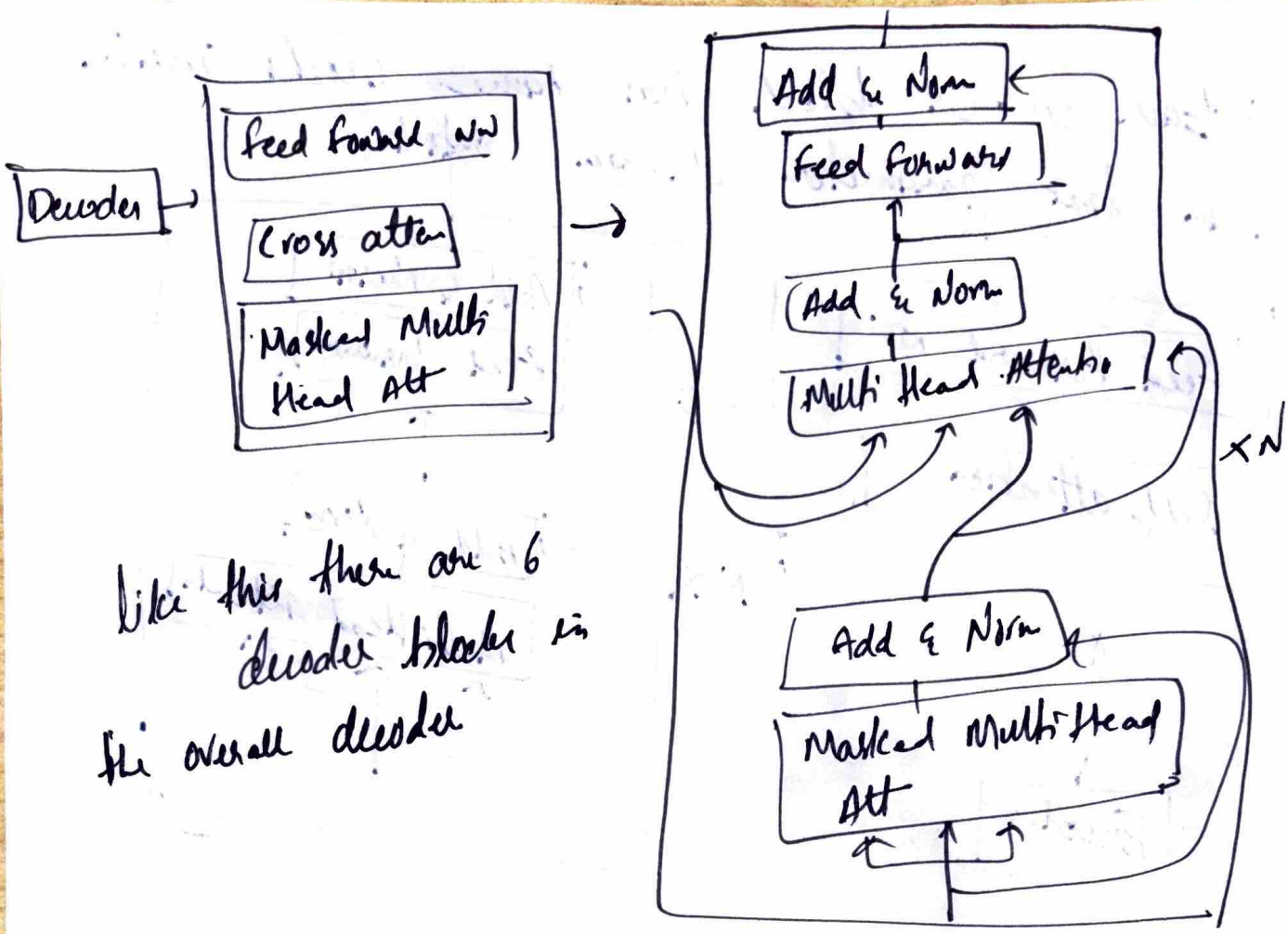
→ Each encoder block has same blocks inside them but parameters change.



N = 6

So there are 6 such blocks

the output from this Encoder will be given to decoder.

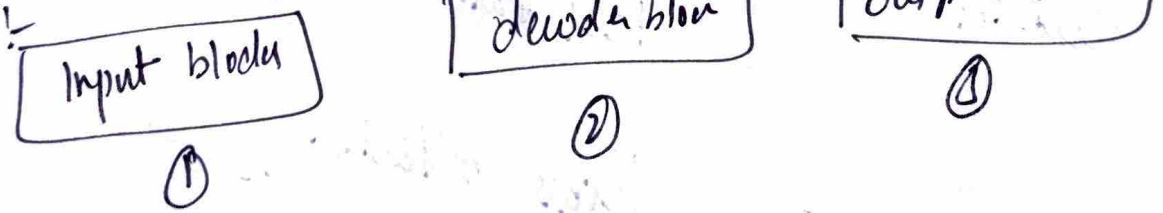


like this there are 6 decoder blocks in the overall decoder

(Q) what happens in a single decoder?

(A) As all the decoder blocks are same, so if you understand one decoder block, rest all are same.

Steps :-



Example : Machine Translation

English to Hindi

English = we are friends

Hindi = Hum dost hai

English Sentence goes into Encoder

↓

generates contextual Embeddings for decoder to work

① input block

there are 4 operations in this :-

1. Shifting
2. Tokenization
3. Embedding
4. Positional Embeddings.

→ the work of input block is to take output sentence (hindi sentence) and process it and gets in the form to send it to decoders first block.

Hum dost Hai

↓

① Right Shift → adds a token start before sentence

||

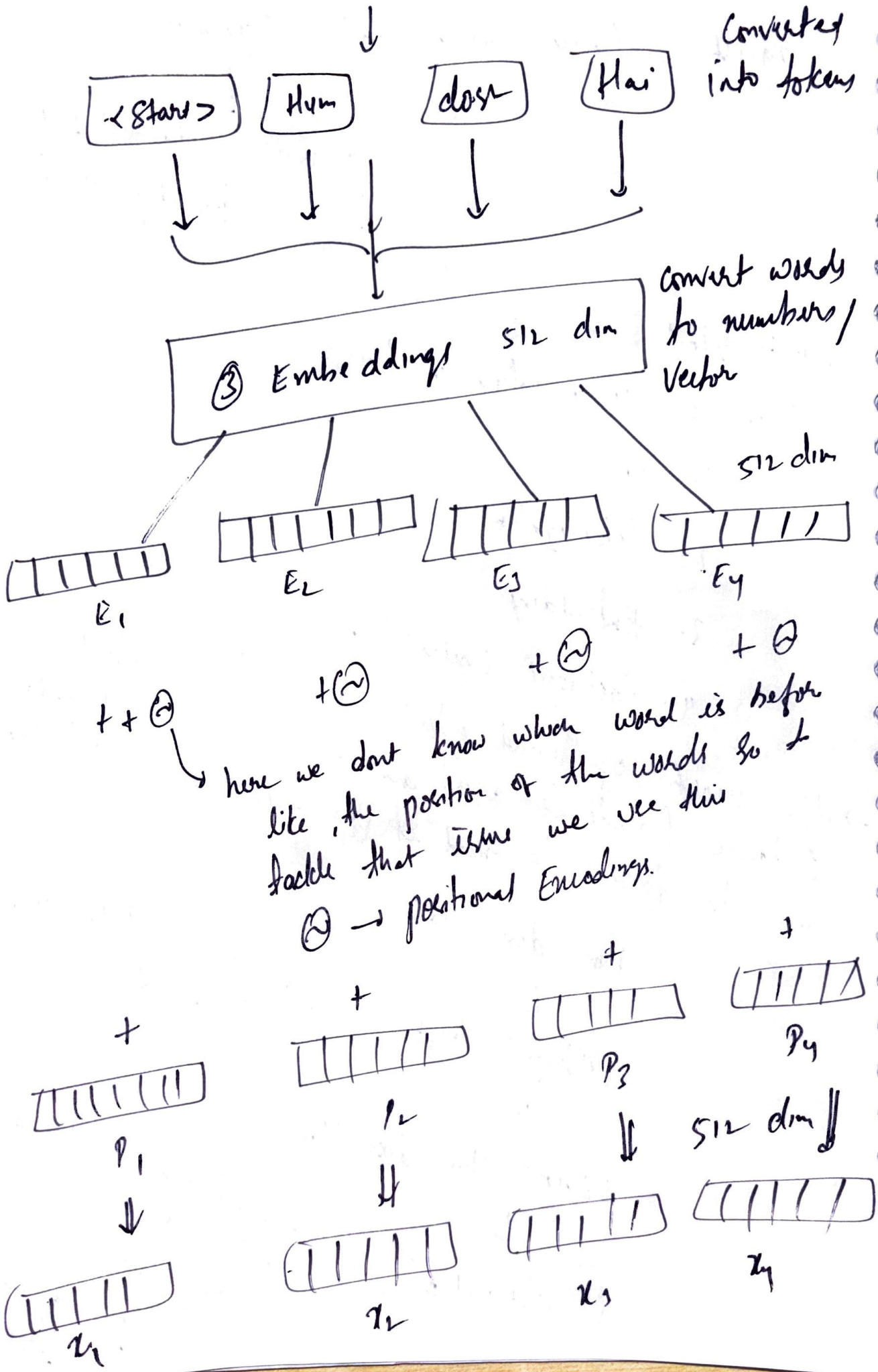
<Start> Hum dost Hai

↓

② Tokenization → break down sentences into tokens

any gram

eg: 1 gram token

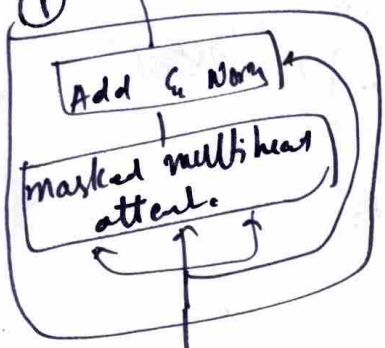

<Start>
Hum
closr
Hai
Converted into tokens
③ Embeddings 512 dim
convert words to numbers / Vector
512 dim
E1
E2
E3
E4
here we dont know when word is before like, the position of the words so to tackle that issue we use this
→ positional Encodings.
P1
P2
P3
P4
512 dim
x1
x2
x3
x4

Now we are done with input block. the final x_1, x_2, x_3, x_4 vectors are the inputs for decoder

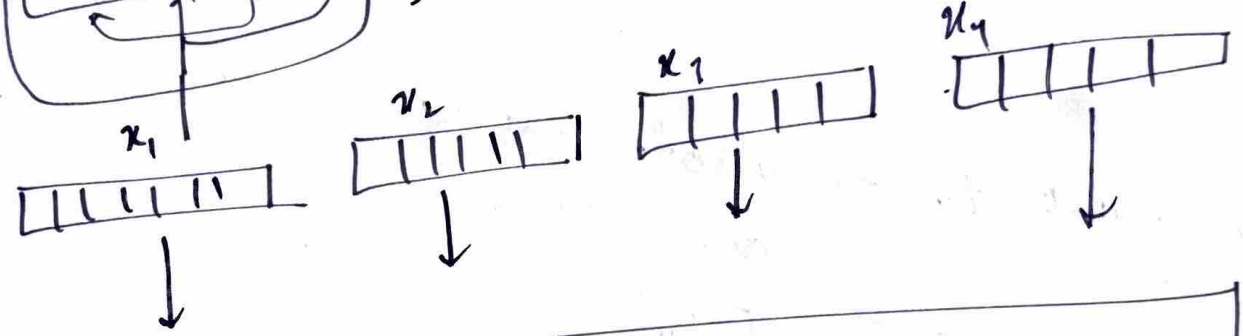
---

② Decoder Block.

① 1st decoder block has 3 tasks in it



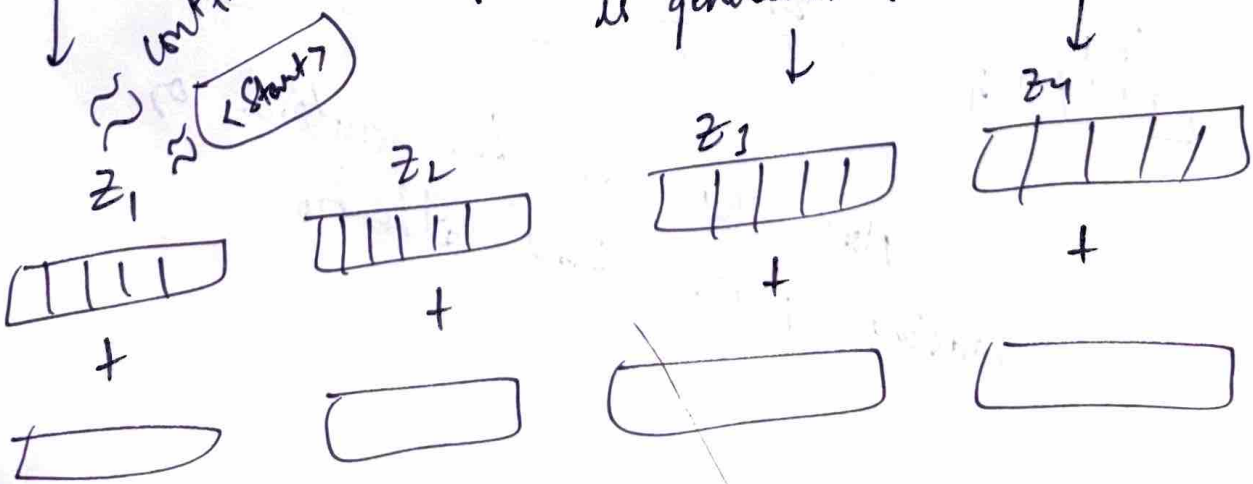
1. Masked Multi Head Attention
2. Multi head Attention (cross-attention)
3. Feed Forward Network



① Masked Multi Head Attention

here masking is implimented for every input vector, corresponds to that, a contextual vector is generated, z_1, z_2, z_3, z_4

contextual vector



to create this masked contextual vectors, we only consider z_1 vector, i.e., <Start> and we do not consider rest. $z_1 \approx x_1$

To generate z_2 → we only consider <Start> and Hum

$z_2 \approx x_1 \; x_2$

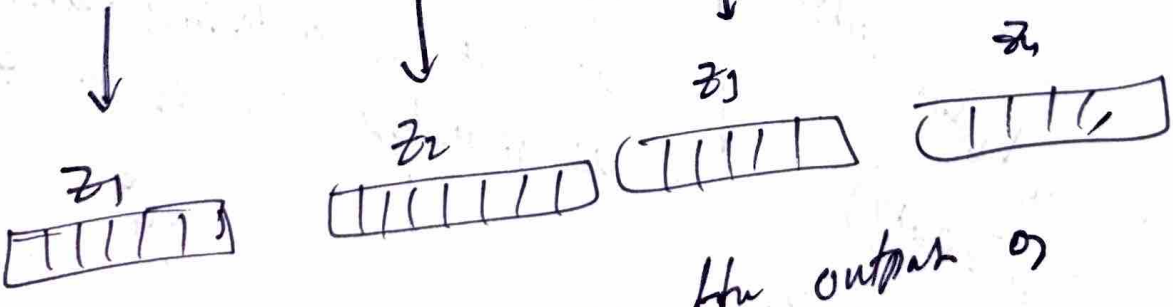
To generate z_3 → we only consider <Start> Hum doctor

$z_3 \approx x_1 \; x_2 \; x_3$

To generate z_4 → we only consider previous and current

Start Hum doctor Hai

$z_4 \approx x_1 \; x_2 \; x_3 \; x_4$



Now there are the output of masked Multi Head attention

Now perform **Add & Norm**

Add = [outputs of masked multi head attn] + [original inputs to the decoder]

z_1 z_2 z_3 z_4 x_1 x_2 x_3 x_4

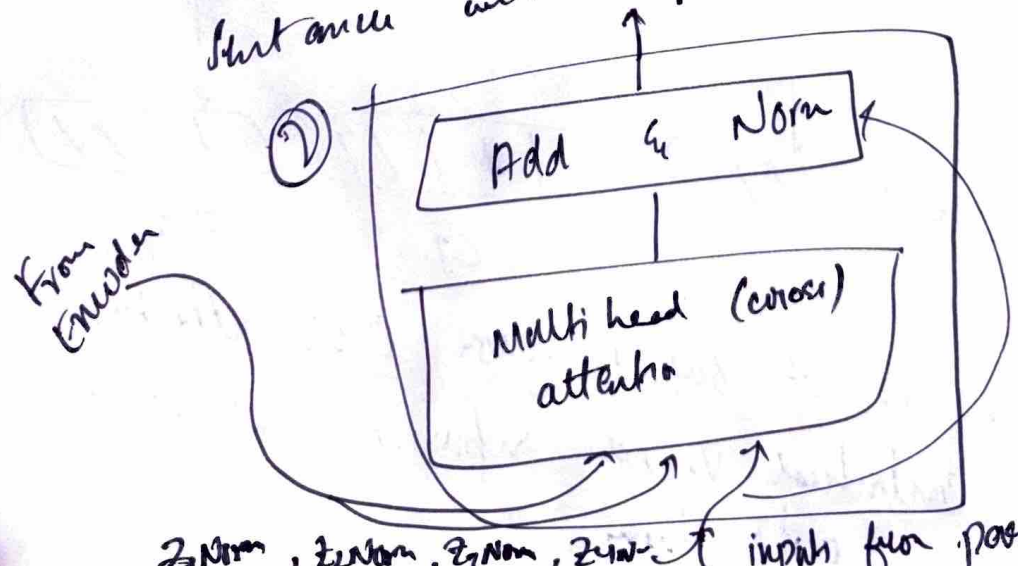
Now perform Normalization ≈ layer Norm

z_1 z_2 z_3 z_4

Layer Normalization

for each vector, this will calculate μ, σ and with that help it normalize, this is done to make training process stable

Now cross attention comes into field here, the connection between Hindi and English sentences will happen.



Cross attention

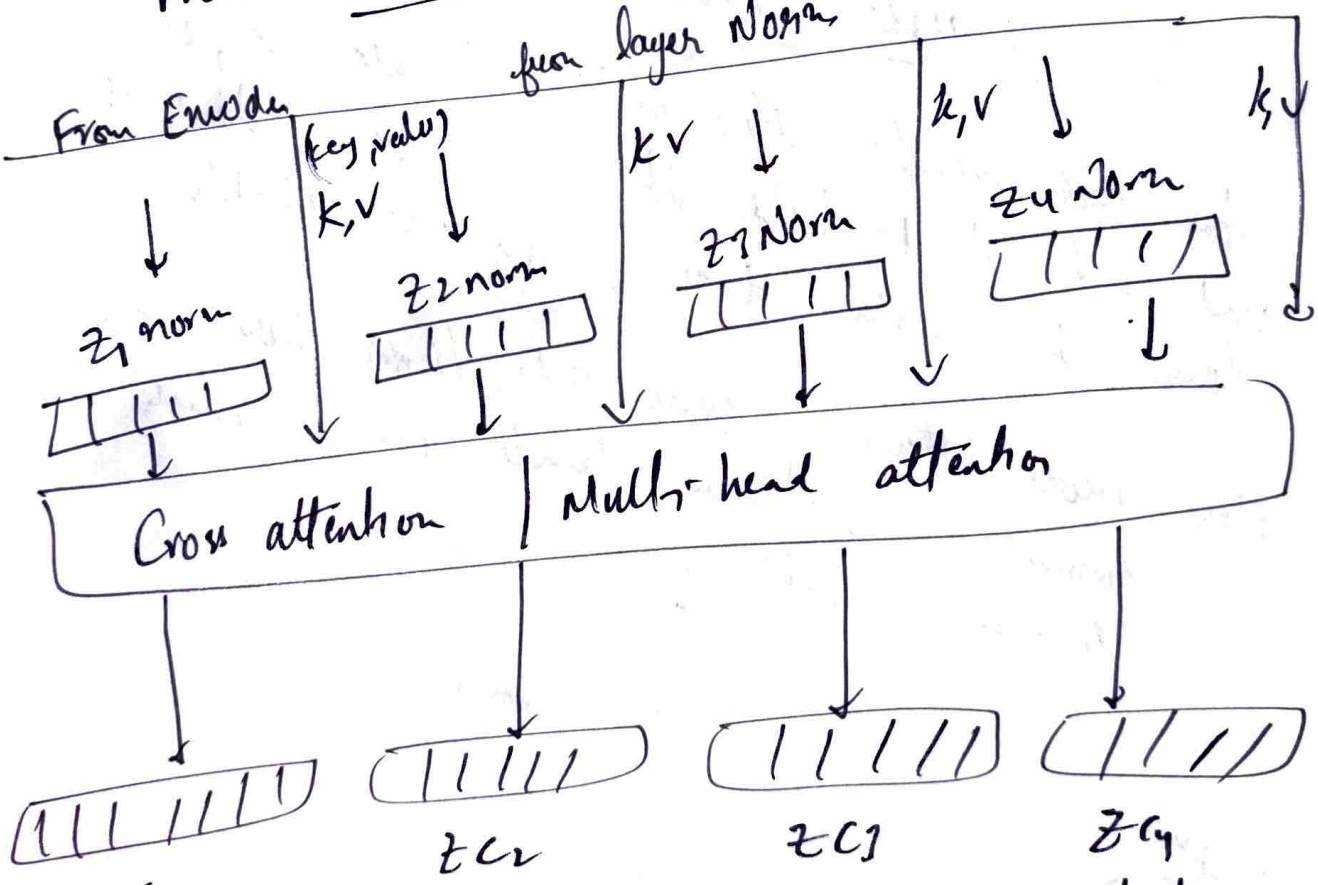
→ 1 Seq (English) from encoder

→ 2 Seq (Hindi) from previous step of decoder

→ this is done as we need 3 vector key, value and Query

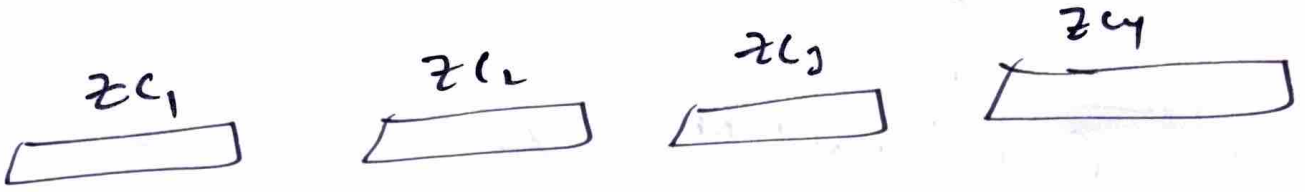
From Encoder → ~~Query~~ key and value is given

From Decoder → Query vector is given and



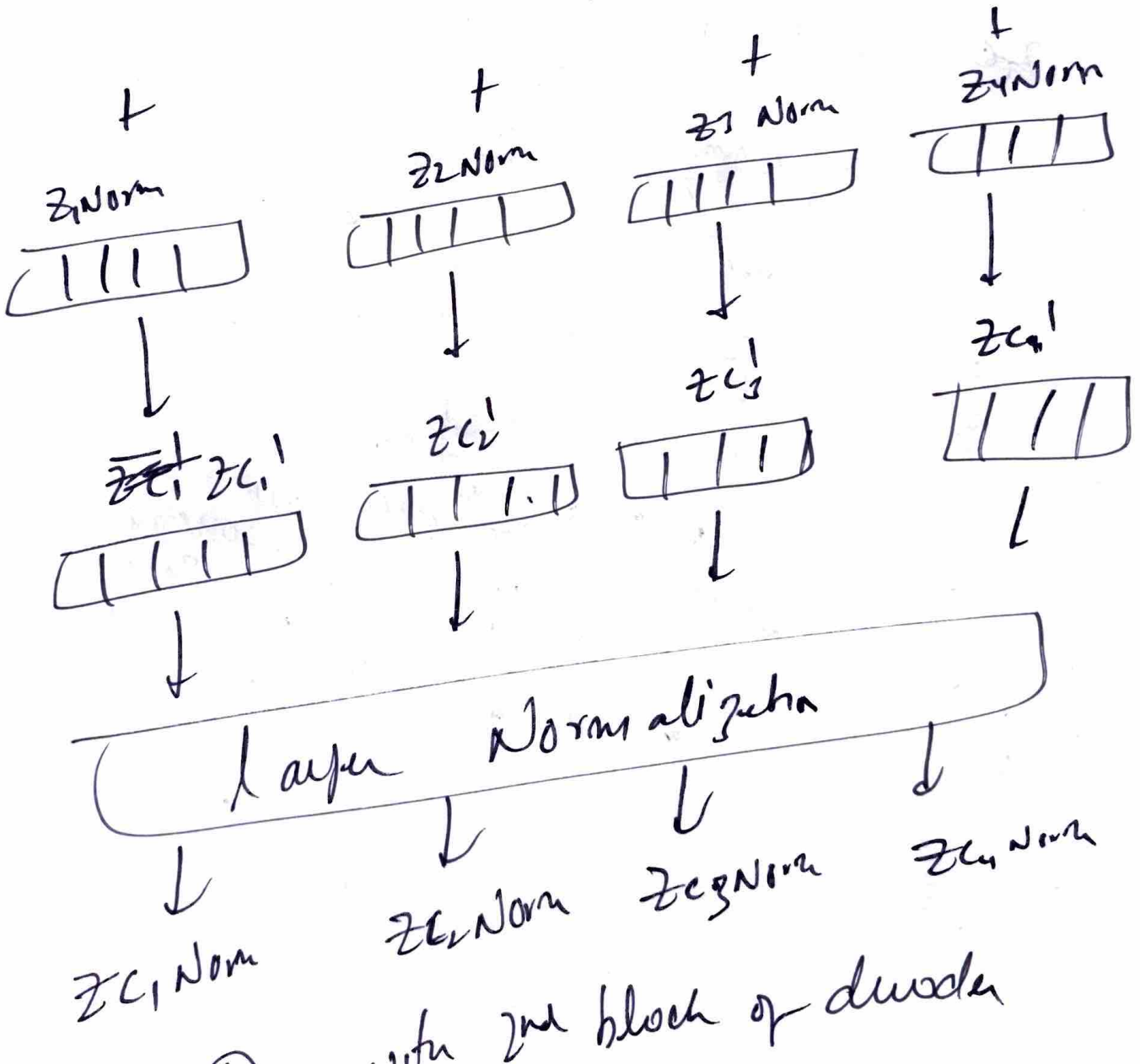
these are the outputs from cross attention where the contextual vector information is attached with them

for each token of the output vector (z_1, z_2, z_3, z_4) there will be a contextual embedding vector



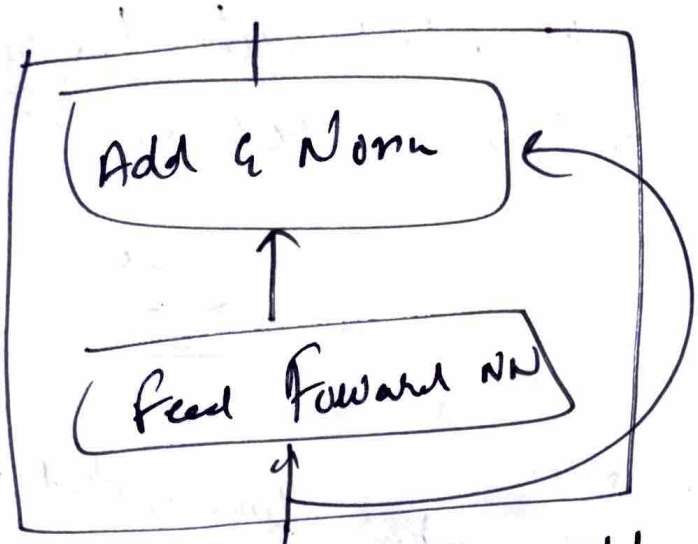
Add and Norm

↳ Previous step output from masked multi head attn (z_1 Norm, z_2 Norm, z_3 Norm z_4 Norm)



Done with 2nd block of decoder

(3)

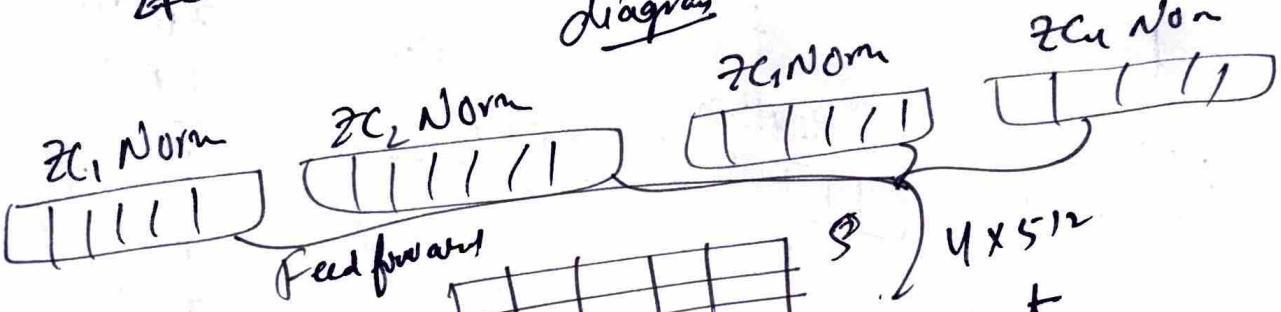


input from previous block

$z_1 Norm, z_2 Norm, z_3 Norm, z_4 Norm$

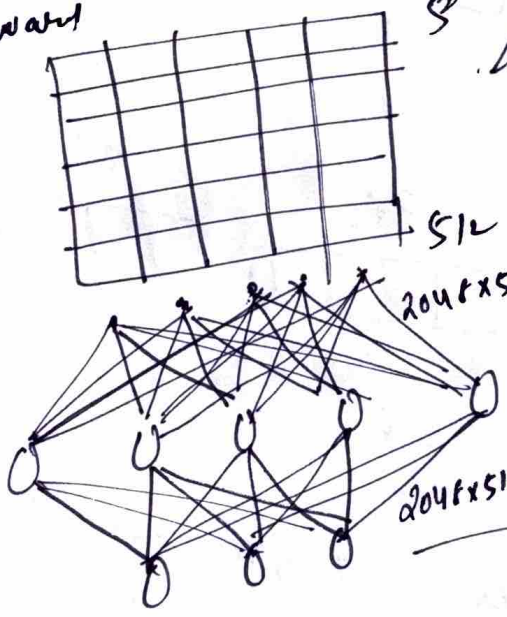
~~z_1~~

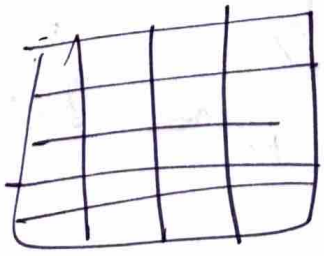
diagram



Same architecture as encoder

2 layers, 2048 neurons → relu





output

512

$(w_1 \cdot z + b_1)$

$[4 \times 512 \quad 512 \times 2048 \approx 4 \times 2048]$ dimension Check

$\text{ReLu}\,(w_1 \cdot z + b_1)$

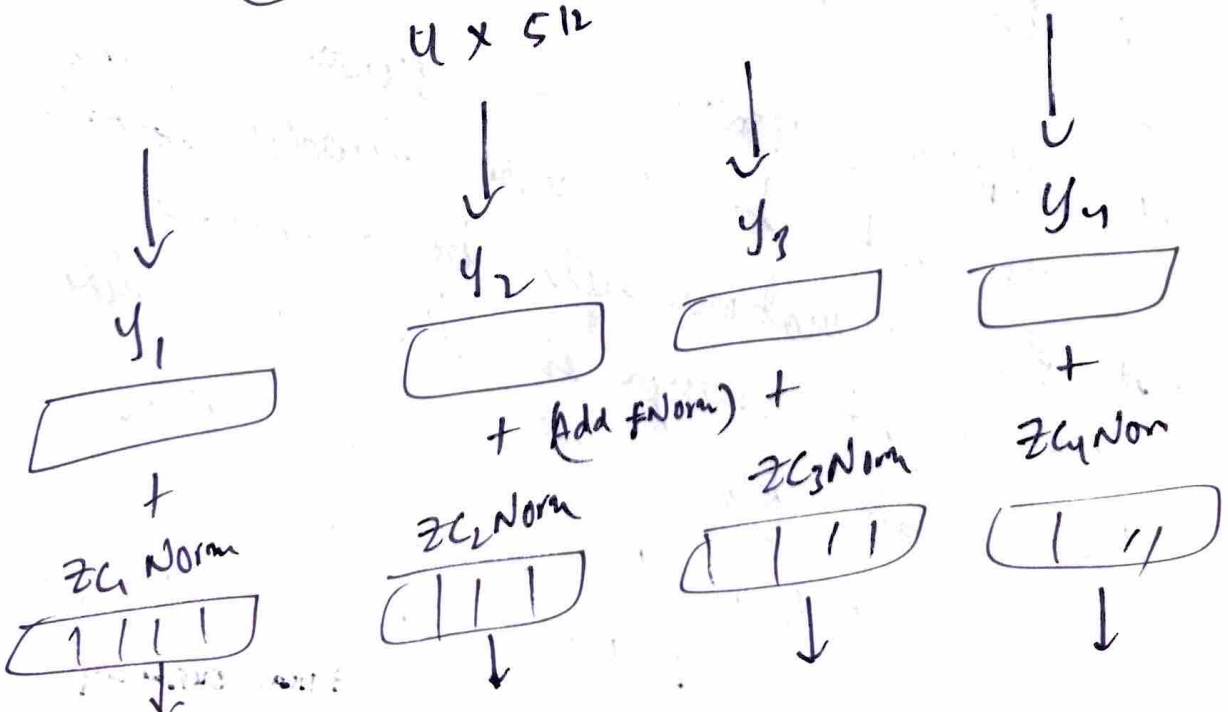
Shape $= 4 \times 2048$

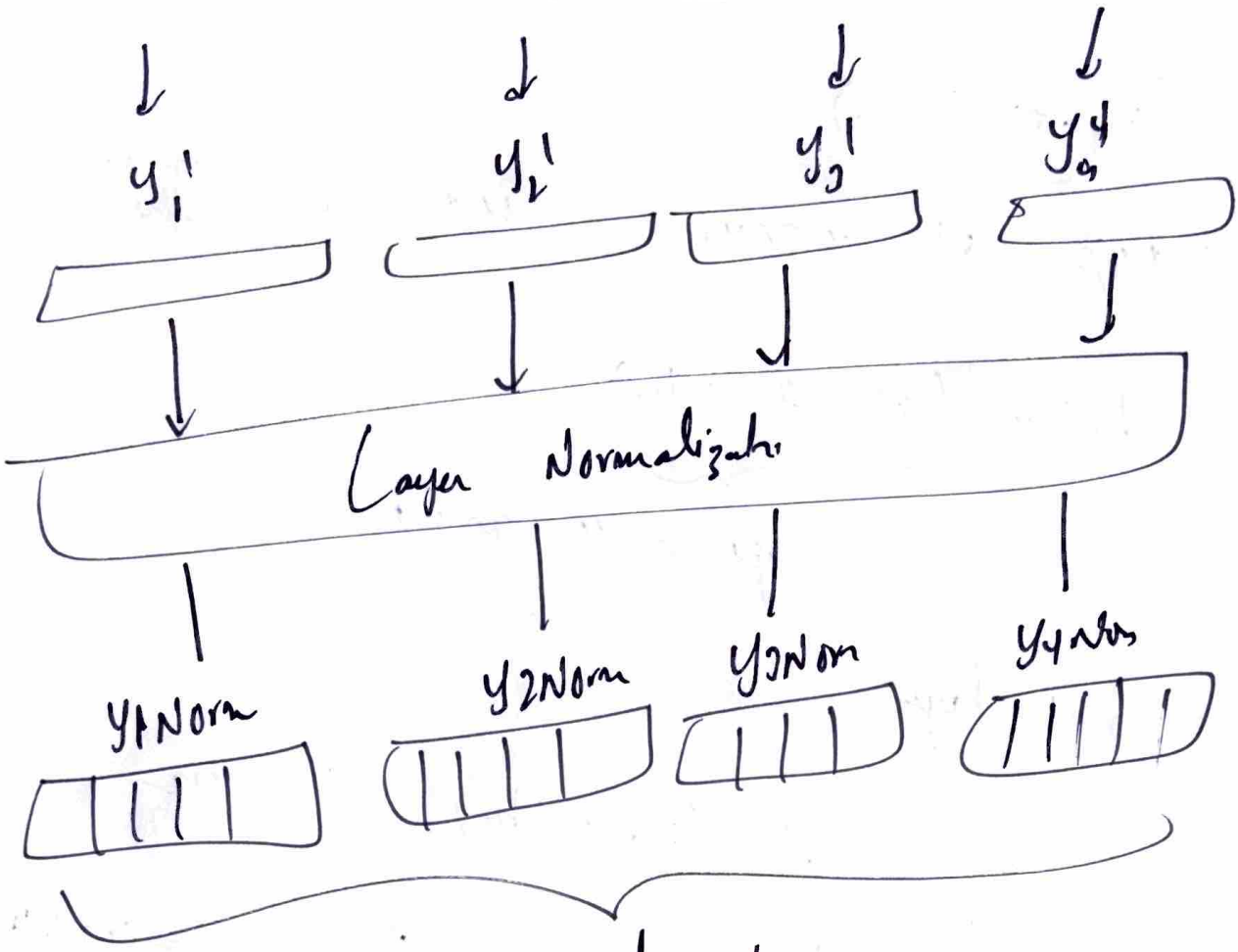
In next layer

$w_2 \left[\text{ReLu}\,(w_1 \cdot z + b_1)\right] + b_2$

dimension Check $(4 \times 2048 \cdot 2048 \cdot 512 = 4 \times 512)$

linear $\left[w_2 \left[\text{ReLu}\,(w_1 \cdot z + b_1)\right] + b_2\right]$





512 dimens

the 4 vectors are the ~~final~~ output of decoder 1

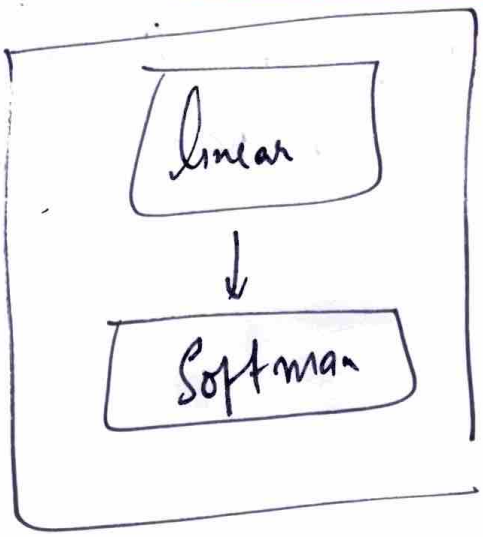
---

③ Output block

Then y1Norm, y2Norm ... y4Norm layers are sent to 5 more decoder ~~blocks~~ and same operations will be executed with slight change in the parameters the output from 6th decoder layer the output are called as

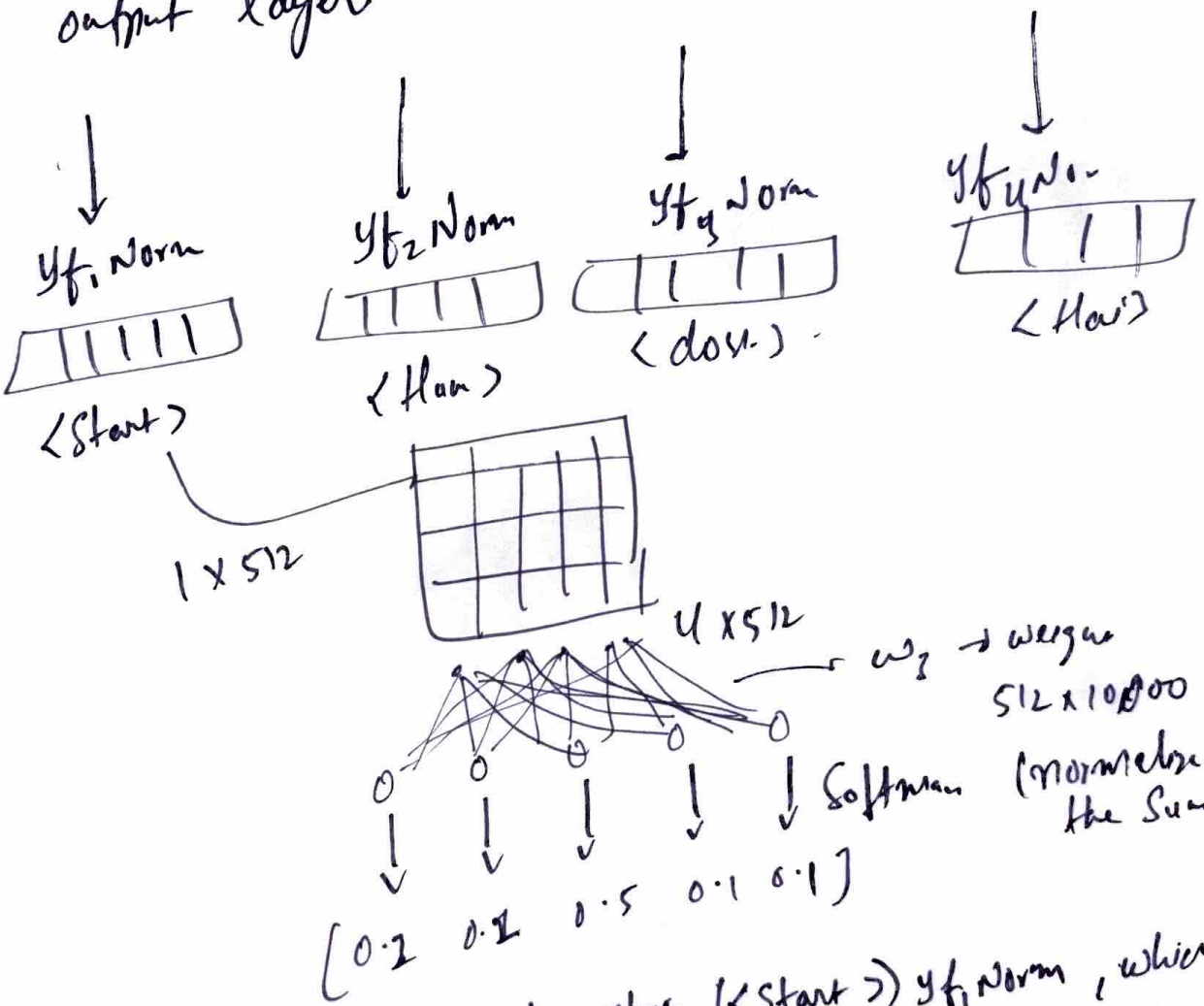
yf1Norm, yf2Norm .. yf4Norm

final output



Output block

made of 2 layers, linear and softmax.
Can be compared to feed forward NN
output layer



Now to corresponding to where (<Start>) yf_1 Norm, which word has high probability from the neural netwo
now send yf_2 Norm to the neural network and see where word

has high probability, that is the kind:
translation of that english word.