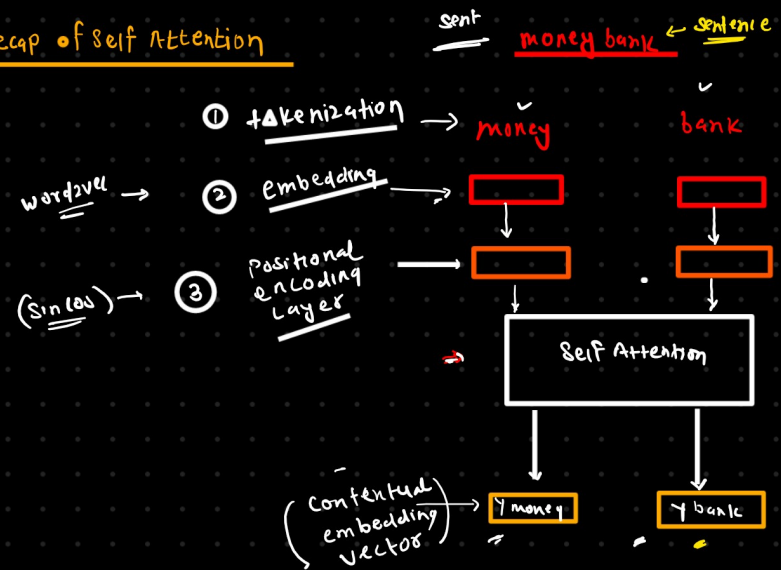
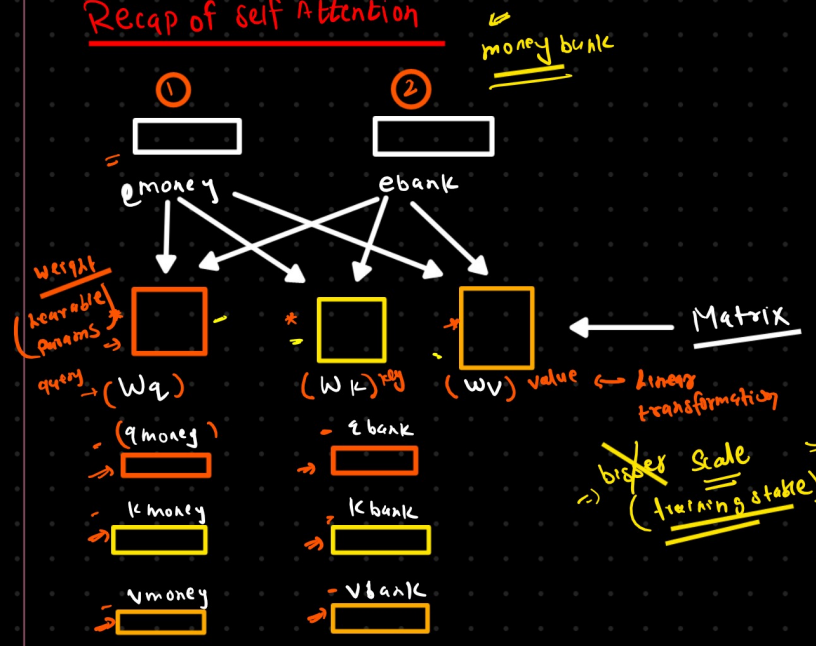
(Multihead, Cross & masked self Attention)

Recap of self Attention



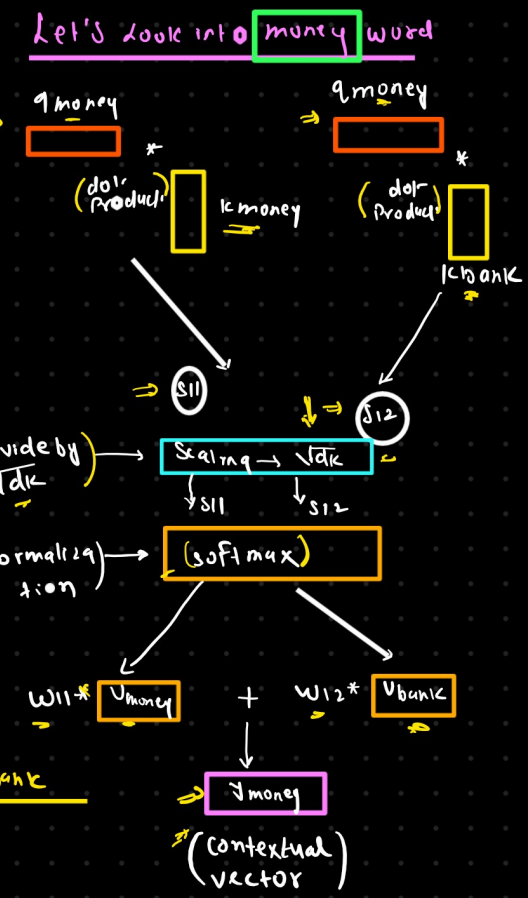
(self attention)

Recap of self Attention



bigger scale → (divide by √dk) (training stable)

Let's look into money word



ambiguity, 1 seal → interpretation → multiple way

Now why we understand multihead self Attention required

1. we use is for (ambiguous) sentence NLP
2. self attention with single head only can capture one perspective of sentence
3. in NLP one sentence have multiple Perspective.

For ex.

1. (Summarization) → seal! → multiple → (decode)
2. Few other sentence
   - The chicken is ready to eat
   - man saw the astronomer with a telescope

In electrical engineering, a transformer is a passive component that transfers electrical energy from one electrical circuit to another circuit, or multiple circuits. A varying current in any coil of the transformer produces a varying magnetic flux in the transformer's core, which induces a varying electromotive force (EMF) across any other coils wound around the same core. Electrical energy can be transferred between separate coils without a metallic (conductive) connection between the two circuits. Faraday's law of induction, discovered in 1831, describes the induced voltage effect in any coil due to

transformer → multiple → (Single Attention) → entire context of data ✗

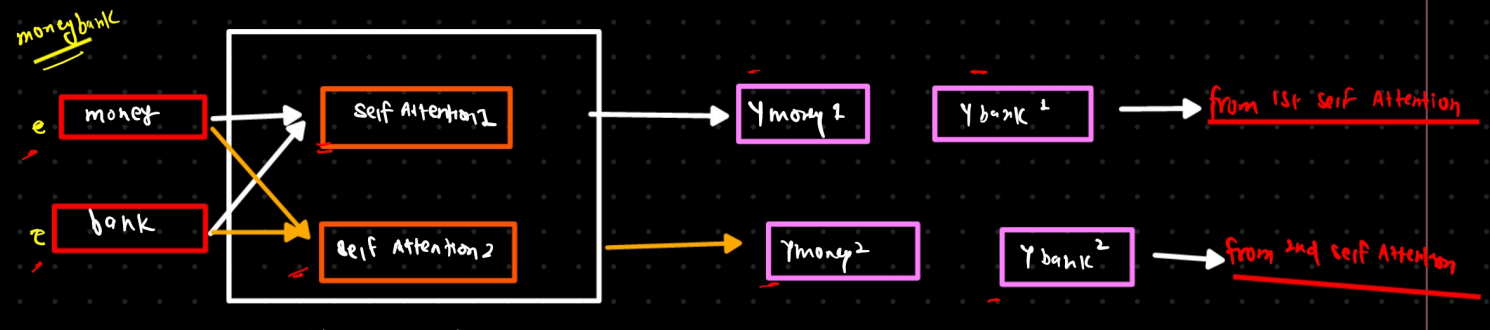
multihead

(Summary 1) person1

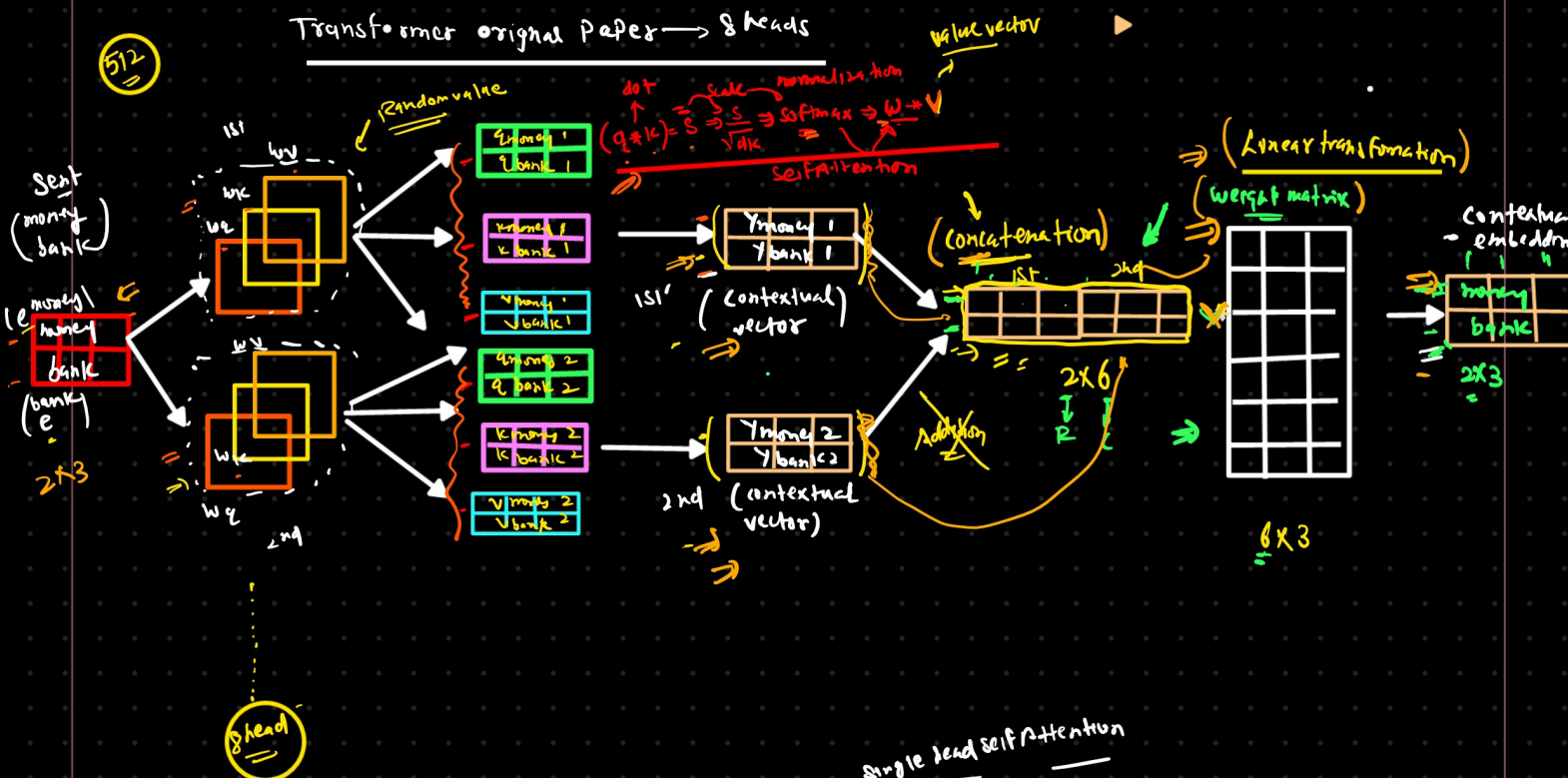
A transformer is a passive electrical component that transfers energy between circuits via electromagnetic induction. A varying current in one coil induces a magnetic flux in the core, generating an electromotive force (EMF) in other coils. Faraday's law of induction explains this phenomenon, enabling energy transfer without direct conductive connection.

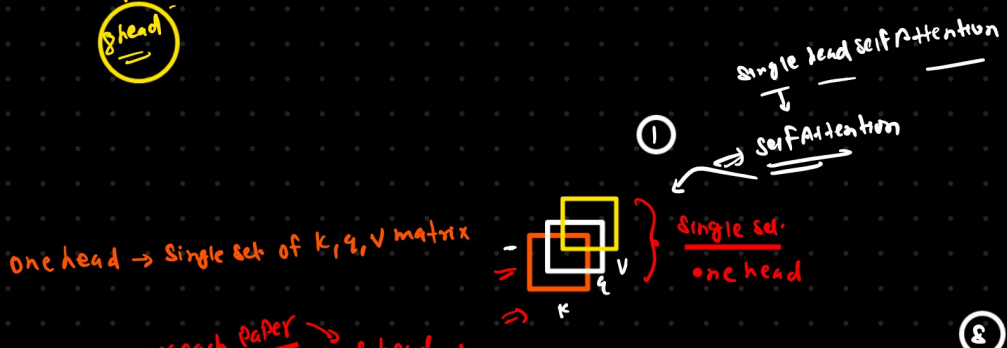
Summary 2 person2

Transformers use electromagnetic induction to transfer electrical energy between circuits. A changing current in one coil creates a magnetic flux, inducing voltage in another coil. This process, based on Faraday's law, allows energy transfer without physical connections.



Multihead self Attention.





Lets understand about the cross encoder

1. Transformer ← mathmatics (encoder & decoder)
   - Self Attention
   - Positional encoding
   - end to end working of encoder

   Previous session

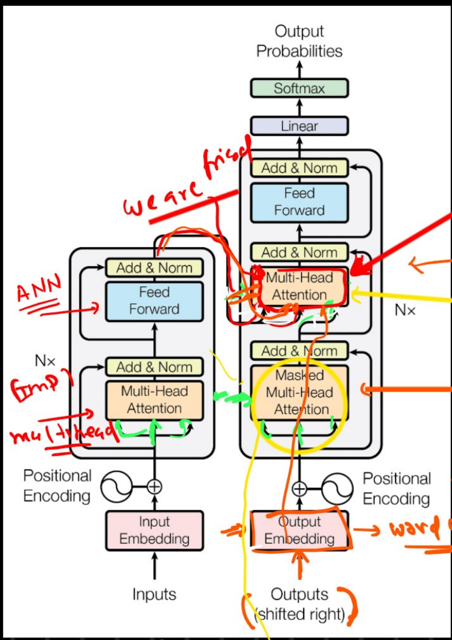
2. Transformer based model. (multimodel) → (Architecture)

(token) ← input

3. APIs ⇒ model → embedding / LLM (overview)
   - openAI
   - huggingface
   - cloud
   - gemini (groq.)
   - ollama, lm studio

4. Finetune - (5-6)

2. Cross Attention -
3. masked attention -



(w w w
K, Q, V)

(Decoder)

multihead Attention → special type of self attention
⇒ cross attention

(Decoder)

masked self attention ⇐

positional layer

word embedding

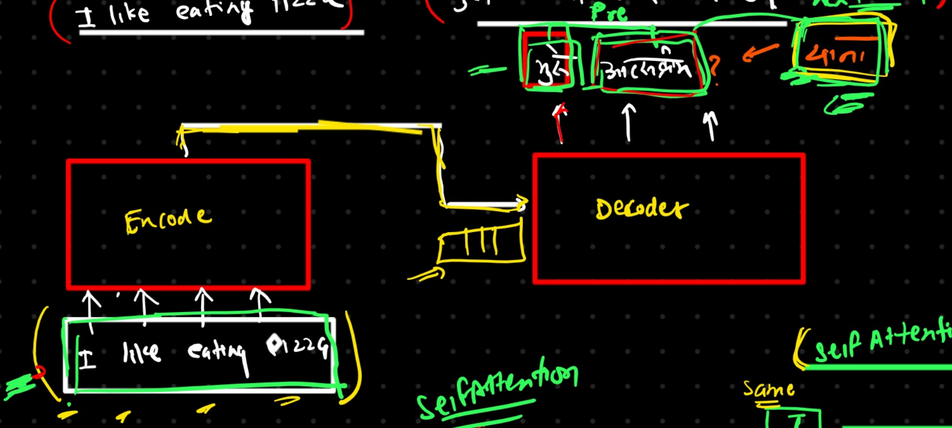
we are friend → हम दोस्त हैं

disease

(Transformer) → training —
→ (Inferencing) }

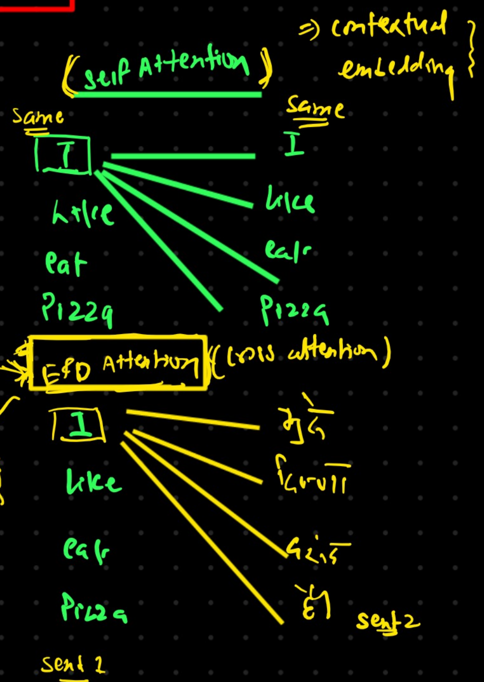
Transformer ⇒ (translation) (eng) → (hindi)

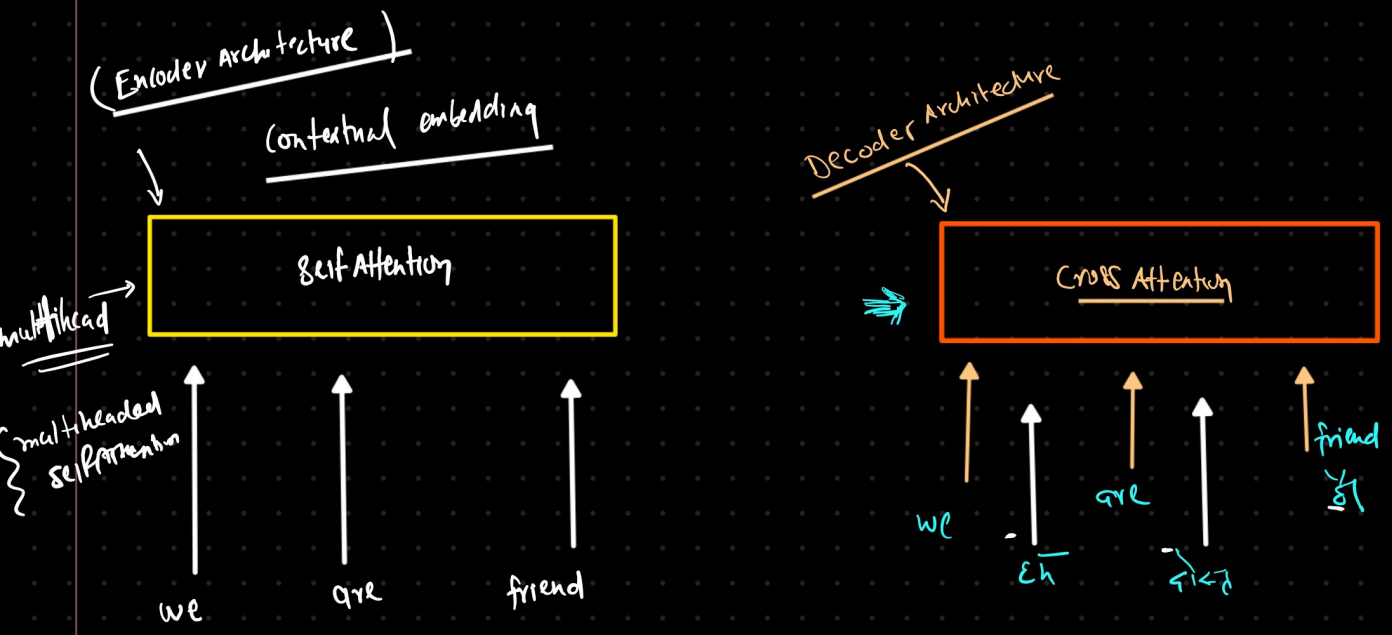
(I like eating pizza) → (मुझे पिज़्ज़ा खाना पसंद है।) next word



Self Attention

→ 1) what generated till now
→ 2) input sentence
→ (cross attention) transformer





SelfAttention

ce_we = $w_{11} * e\_we + w_{12} * e\_are + w_{13} * e\_friend$

ce_are = $w_{21} * e\_we + w_{22} * e\_are + w_{23} * e\_friend$

ce_friend = $w_{31} * e\_we + w_{32} * e\_are + w_{33} * e\_friend$

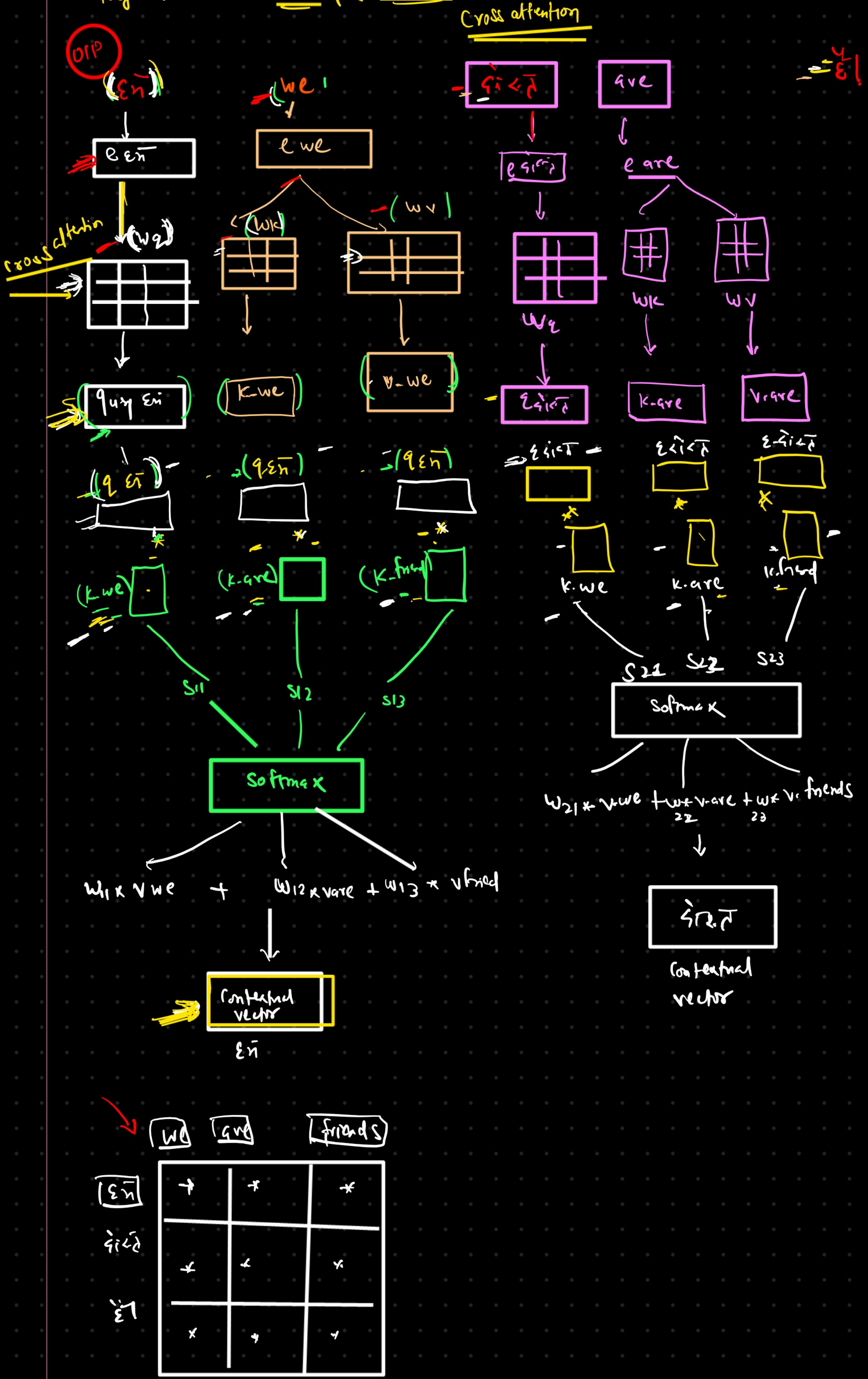
selfAttention	we	are	friend
we	*	*	*
are	*	*	*
friend	*	*	*

3 input → query, key, value

⇒ Q, K, V → matrix (weight matrix)

query → (output seq)(हम दोस्त है)

key & value → input seq (we are friends)

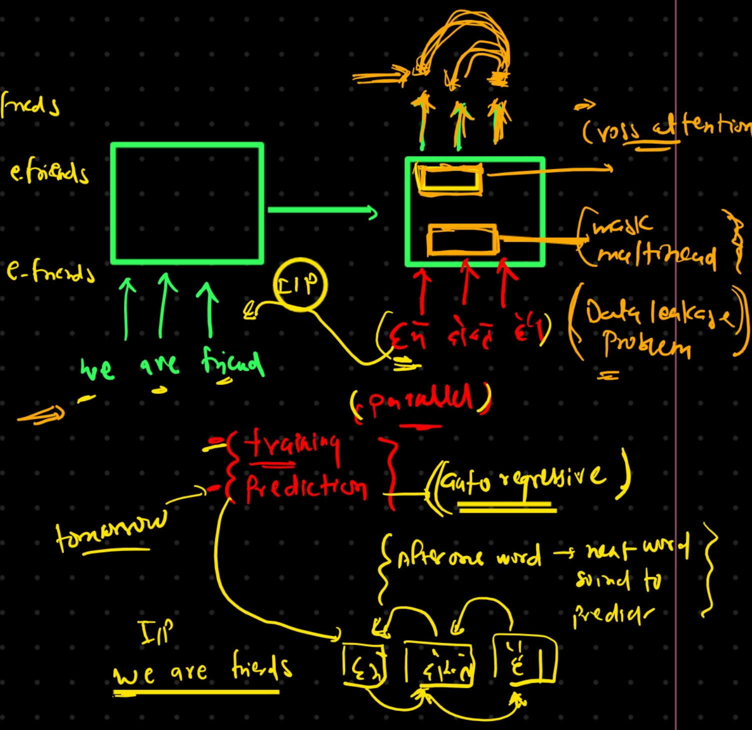


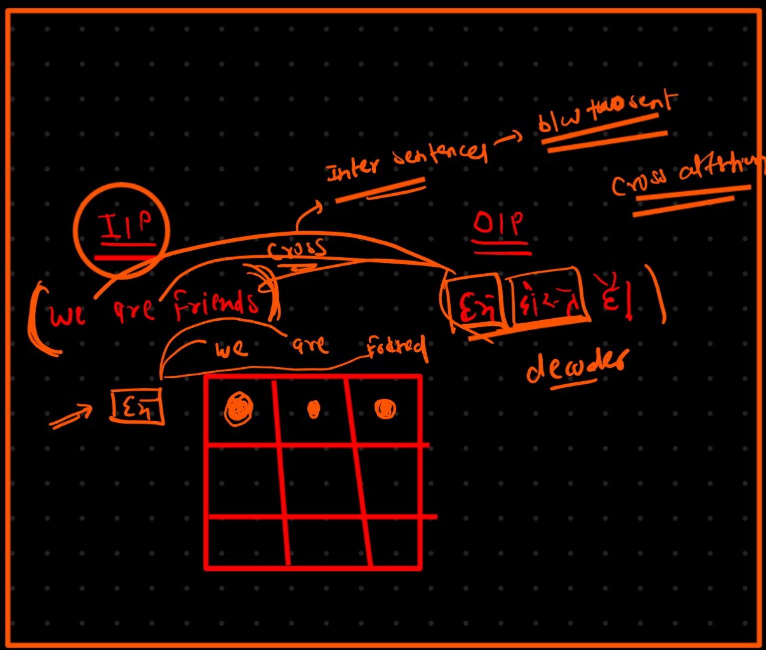
~~Cap~~ Complex

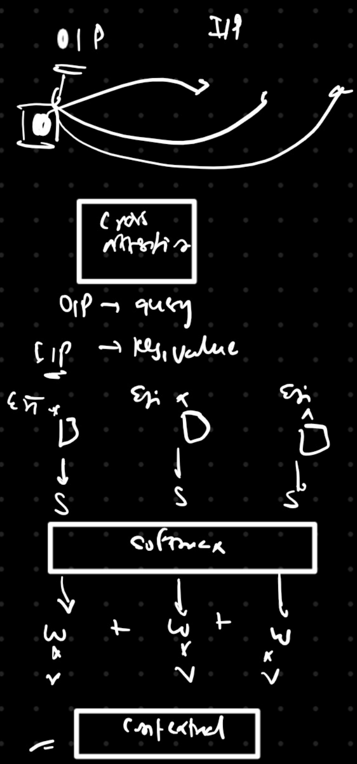
$ce\text{-}हम = w_{11} * e\_we + w_{12} * e\_are + w_{13} * e\_friends$

$ce\text{-}दोस्त = w_{21} * e\_we + w_{22} * e\_are + w_{23} * e\_friends$

$ce\text{-}हैं = w_{31} * e\_we + w_{32} * e\_are + w_{33} * e\_friends$

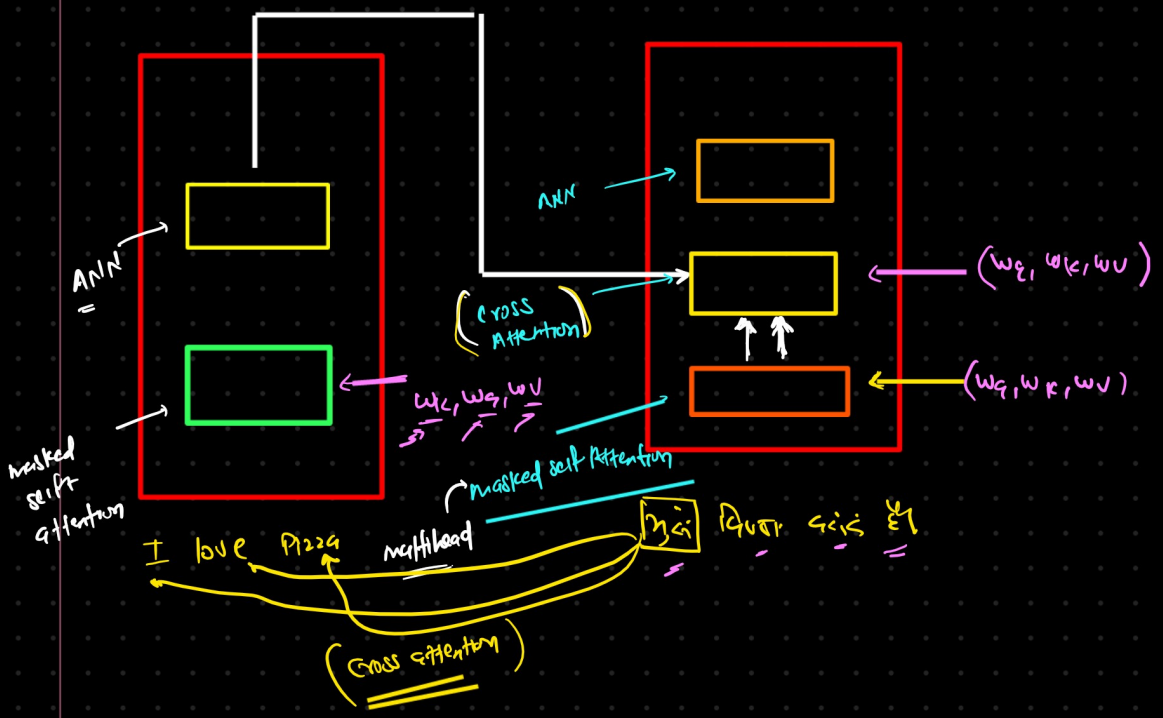




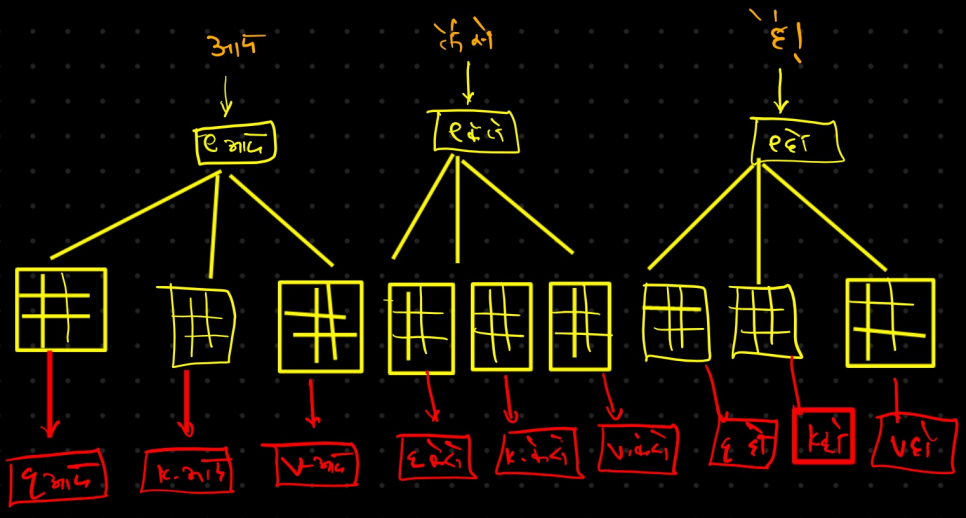


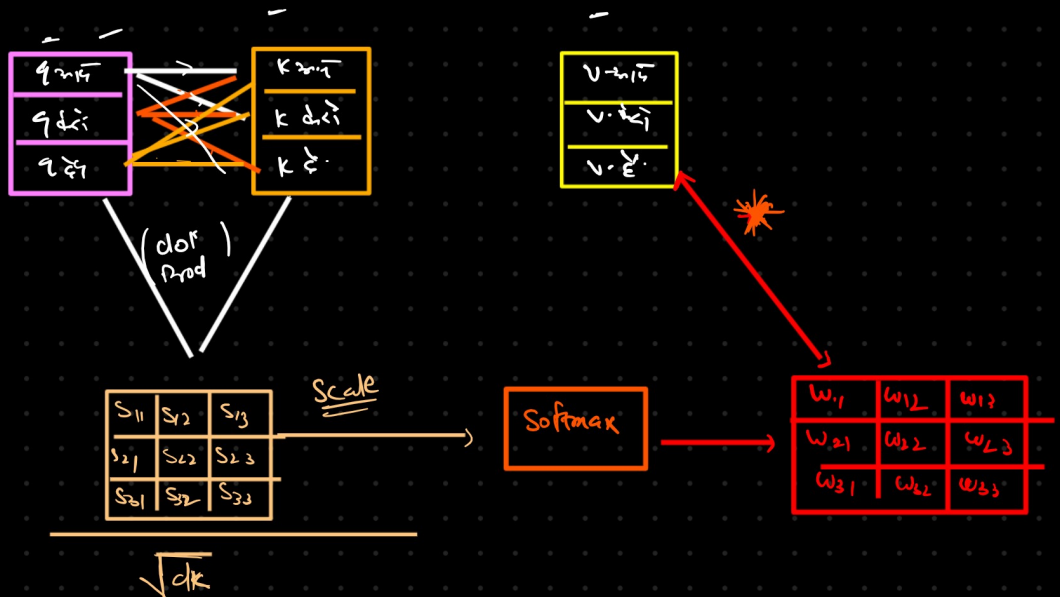
1. self attention -
2. multihead -
3. cross -
4. masked attention

	मुझे	समोसा	पसंद	है।
I	-	-	-	-
Like	-	-	-	-
Samosa	-	-	-	-



How are you
आप कैसे हैं!





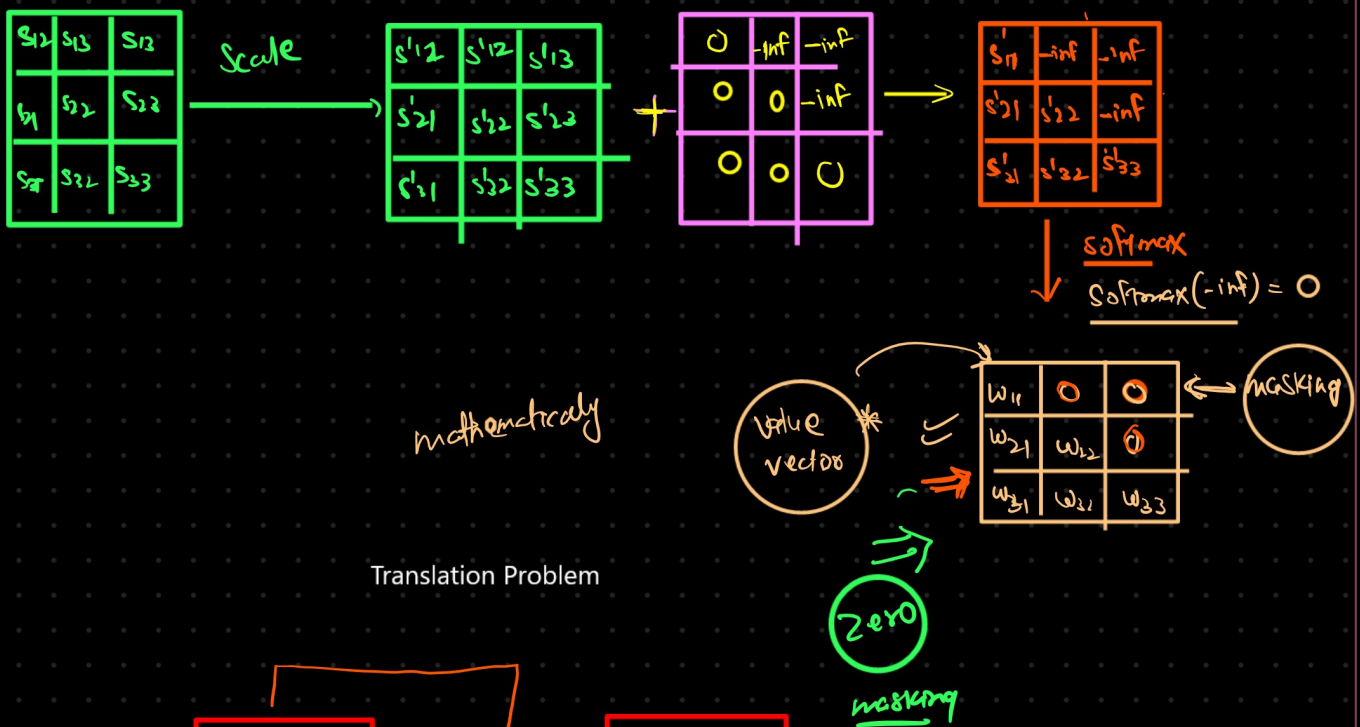
आप_ce = $w_{11} * v_{आप}$ + $w_{12} * v_{कैसे}$ + $w_{13} * v_{हो}$ — masking

कैसे_ce = $w_{21} * v_{आप}$ + $w_{22} * v_{कैसे}$ + $w_{23} * v_{हो}$

हो_ce = $w_{31} * v_{आप}$ + $w_{32} * v_{कैसे}$ + $w_{33} * v_{हो}$

MASKED Self Attention

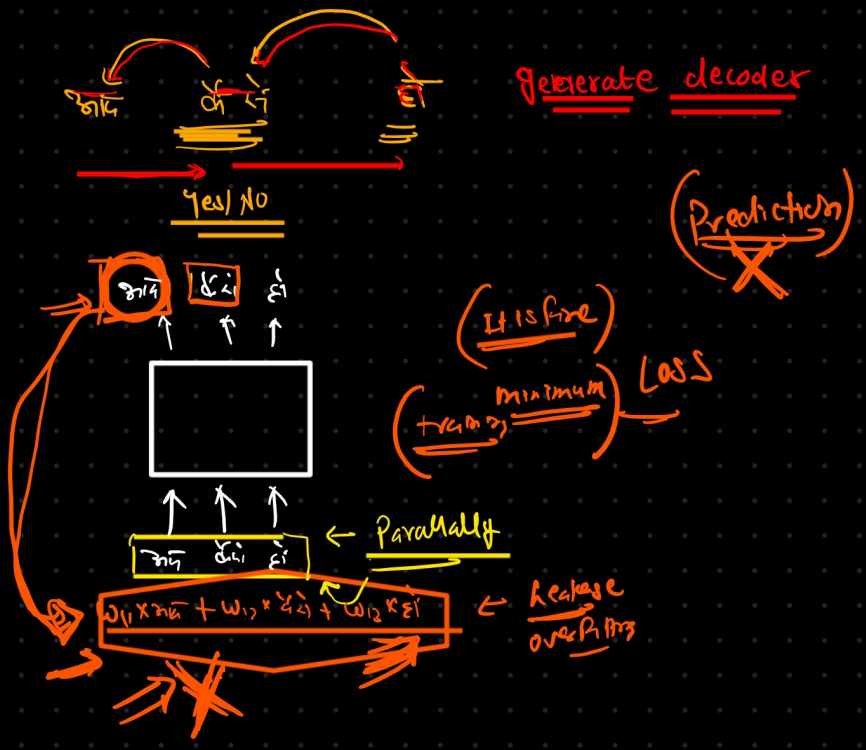
Data leakage ⇒ overfitting ⇒ (cheating)



$$\text{आप}_{ce} = w_{11} * \text{आप}_e + w_{12} * \text{दें}_e + w_{13} * \text{हो}_e$$

$$\text{दें}_{ce} = w_{21} * \text{आप}_e + w_{22} * \text{दें}_e + w_{23} * \text{हो}_e$$

$$\text{हो}_{ce} = w_{31} * \text{आप}_e + w_{32} * \text{दें}_e + w_{33} * \text{हो}_e$$



Generate decoder

Prediction

If is fine

Training minimum Loss

Leakage overfitting

Current token at the time of training

You revealed the future token

Current token

(आप)

$\Rightarrow (w_{11} * \text{आप} + w_{12} * \text{दें} + w_{13} * \text{हो})$

issue data leakage

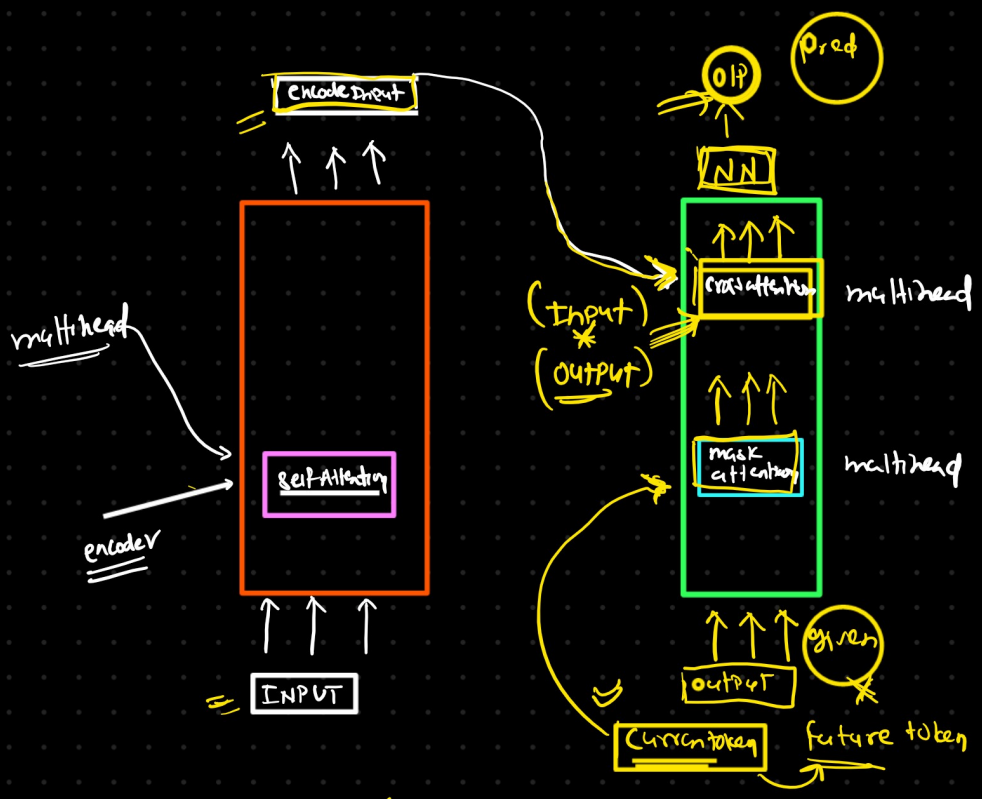
(training it is fine but when prediction it will fail

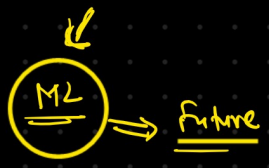
आप, दें

$w_{11} * \text{आप} + w_{22} * \text{दें} + w_{23} * \text{हो}$

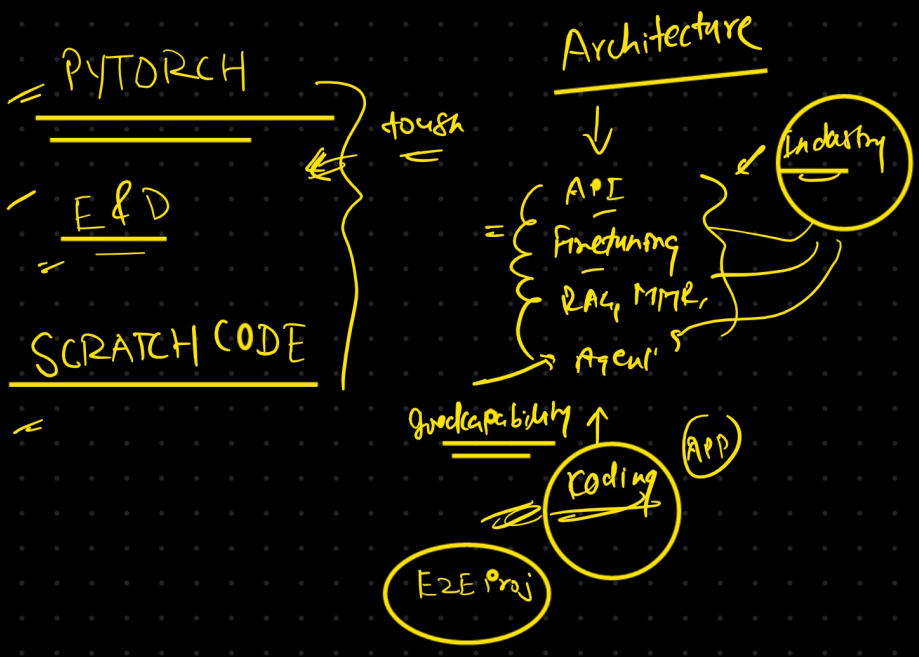
(आप दें हो)

x10 $\Rightarrow$ $w_{31} * \text{आप} + w_{32} * \text{दें} + w_{33} * \text{हो}$


Pred
O/P
NN
Encode Input
multihead
(Input)
*
(output)
multihead
mask attention
multihead
encoder
given
output
INPUT
Current token
future token


M2
future

=> Prediction


PYTORCH
touch
E&D
SCRATCH CODE
Architecture
API
Finetuning
RAG, MMR,
Agent
Industry
good capability
Coding
APP
E2E proj