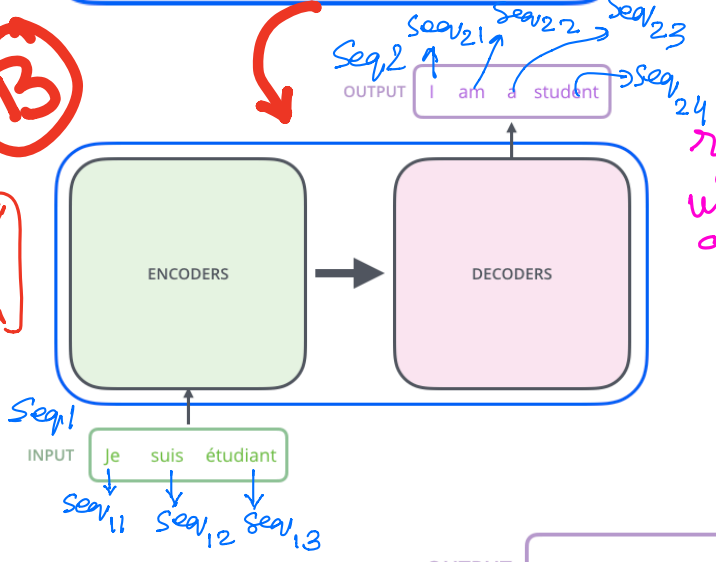
This material is taken from Jay Alammars Blog

"The illustrated Transformer"

As aliens entered our planet and began to colonize earth a certain group of extraterrestrial

(A)

(B)



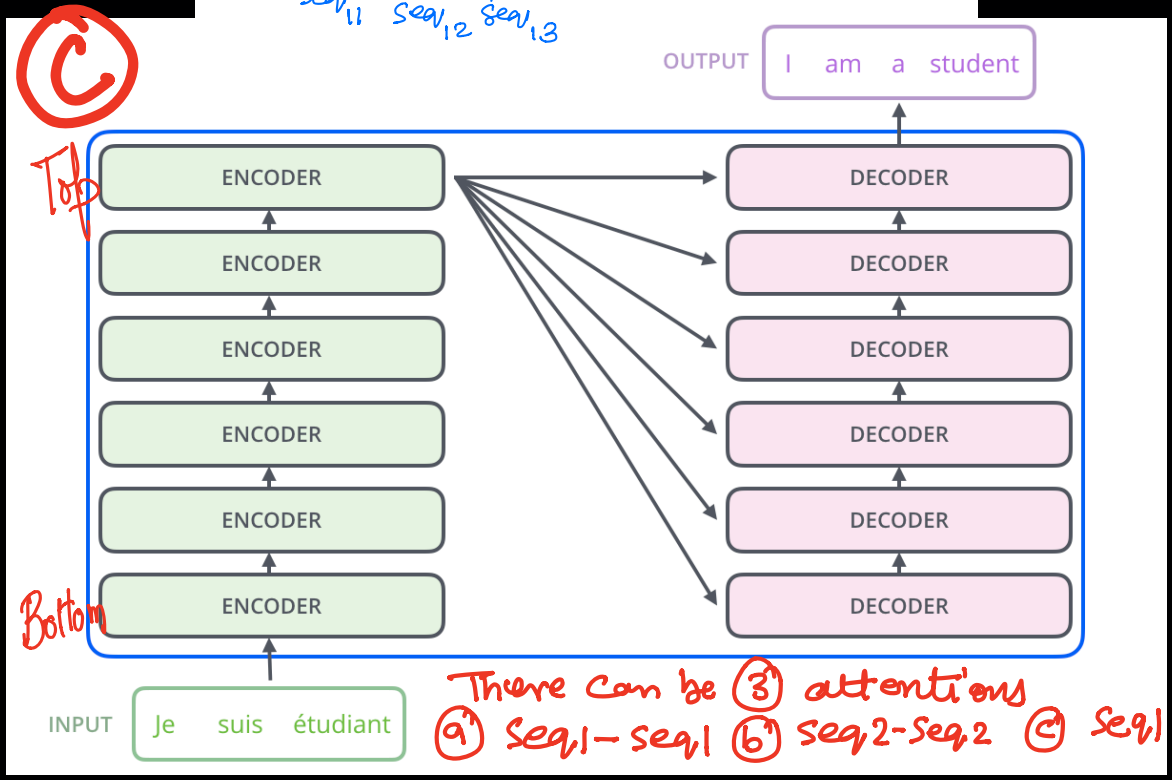
Encoder learns info for the entire Sequence [some Common intermediate representation]

Abstract & Continous representation

It takes that representation along with the previously generated O/P word

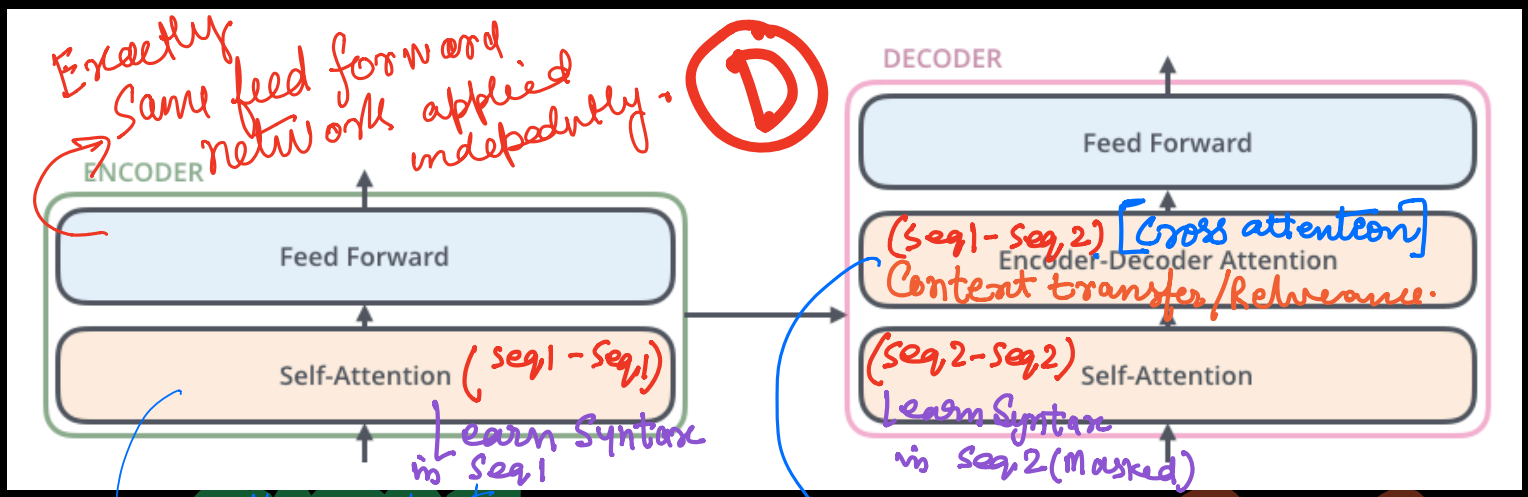
Generate the O/P sequence step by step.

(C)



All encoders are identical But do not share weights

There can be (3) attentions (a) Seq1-Seq1 (b) Seq2-Seq2 (c) Seq1-Seq2



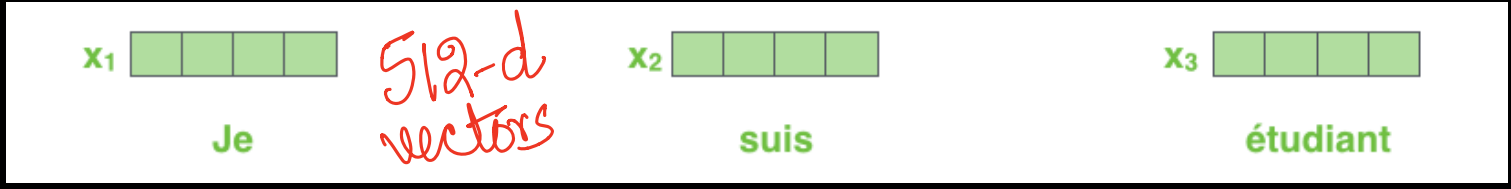
Helps encoder to consider other input words as it encodes a specific word

Additional layer in Decoder. helps to focus on relevant input words as it decodes a specific word.

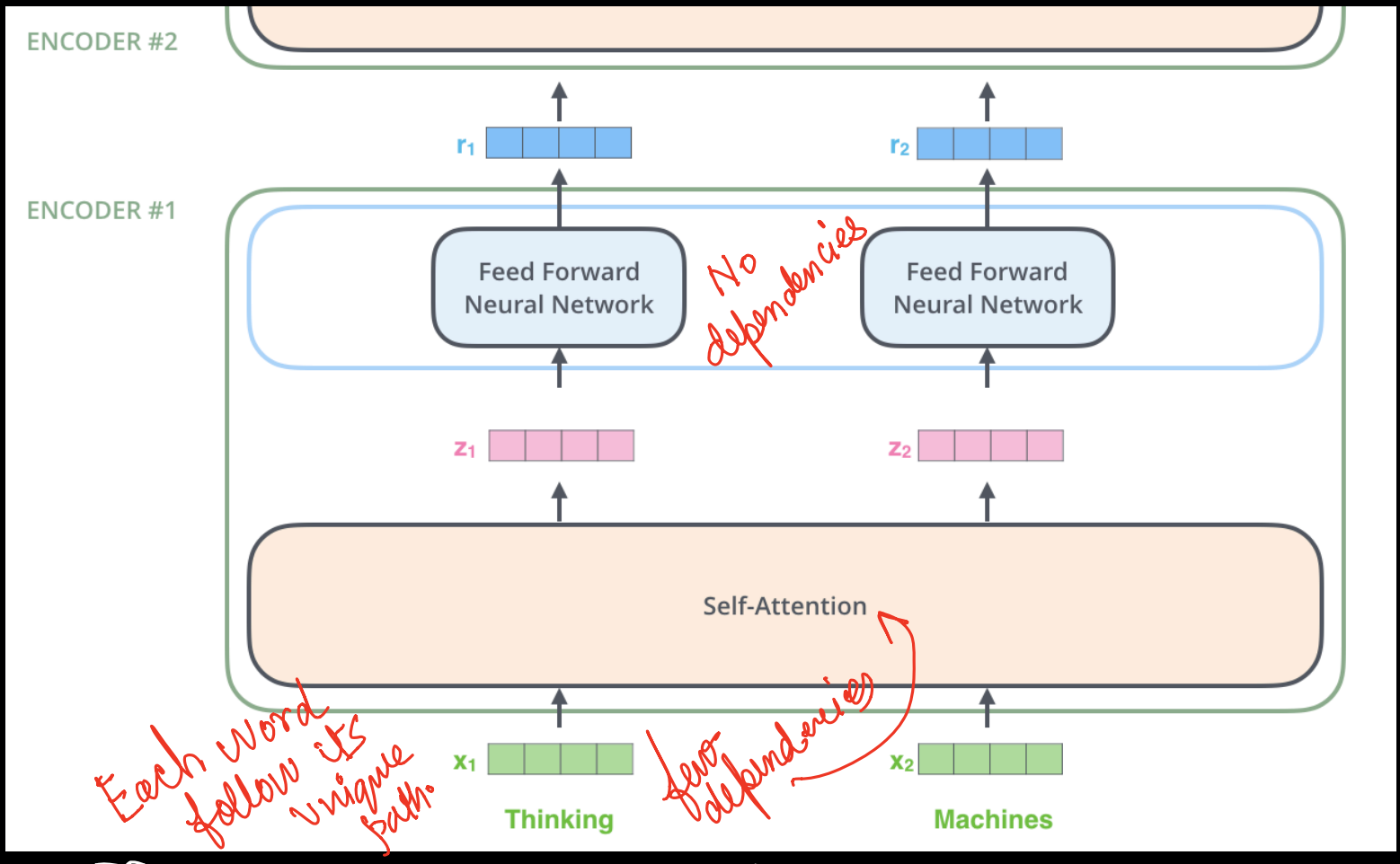
Sequence ⇒ Syntax + Semantic

Encoder learns the semantic dependence ⇒ Attention

I Word2Vec encoding



* Bottom encoder gets word embedding.
* Others get the o/p of previous encoders.
* Encoder receives a list of embedding vectors of input sentences.



II Self-Attention (SA)

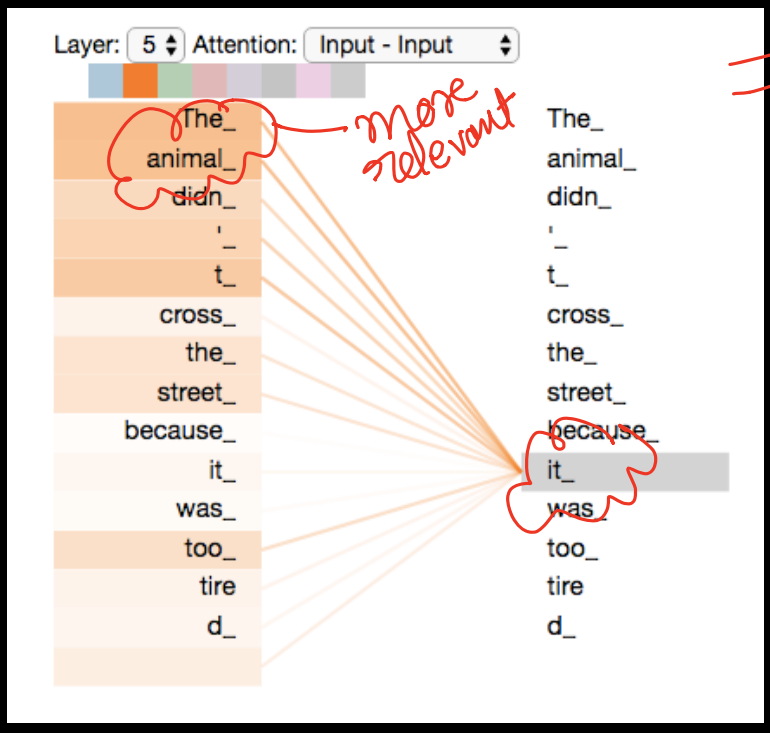
Not Referencing

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

Referencing

Hence while processing "it", SA allows to associate with animal.

SA mostly discovers such references and use them as clues to encode the word in terms of other related words

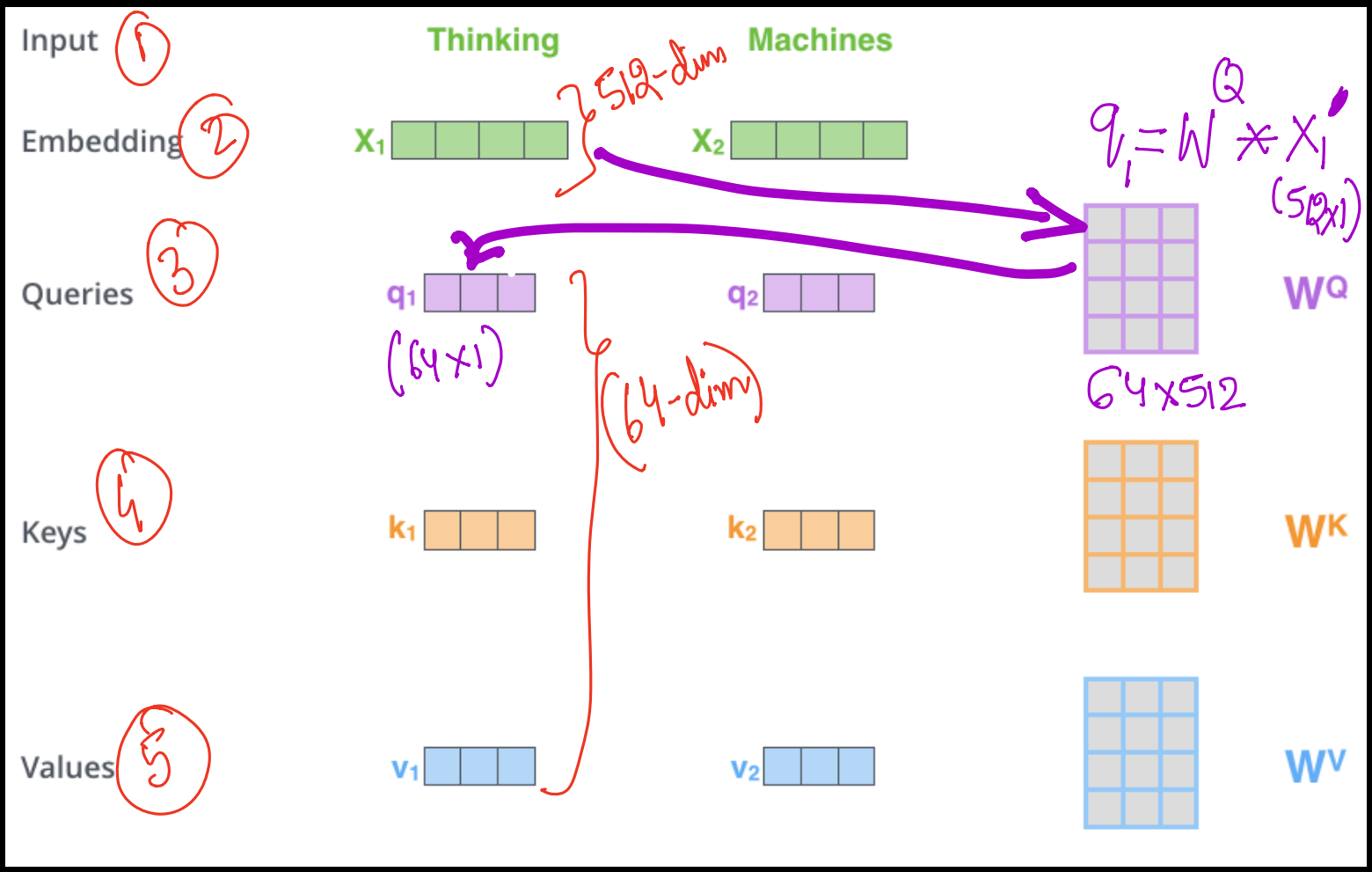


Showing the top encoders learned attention.

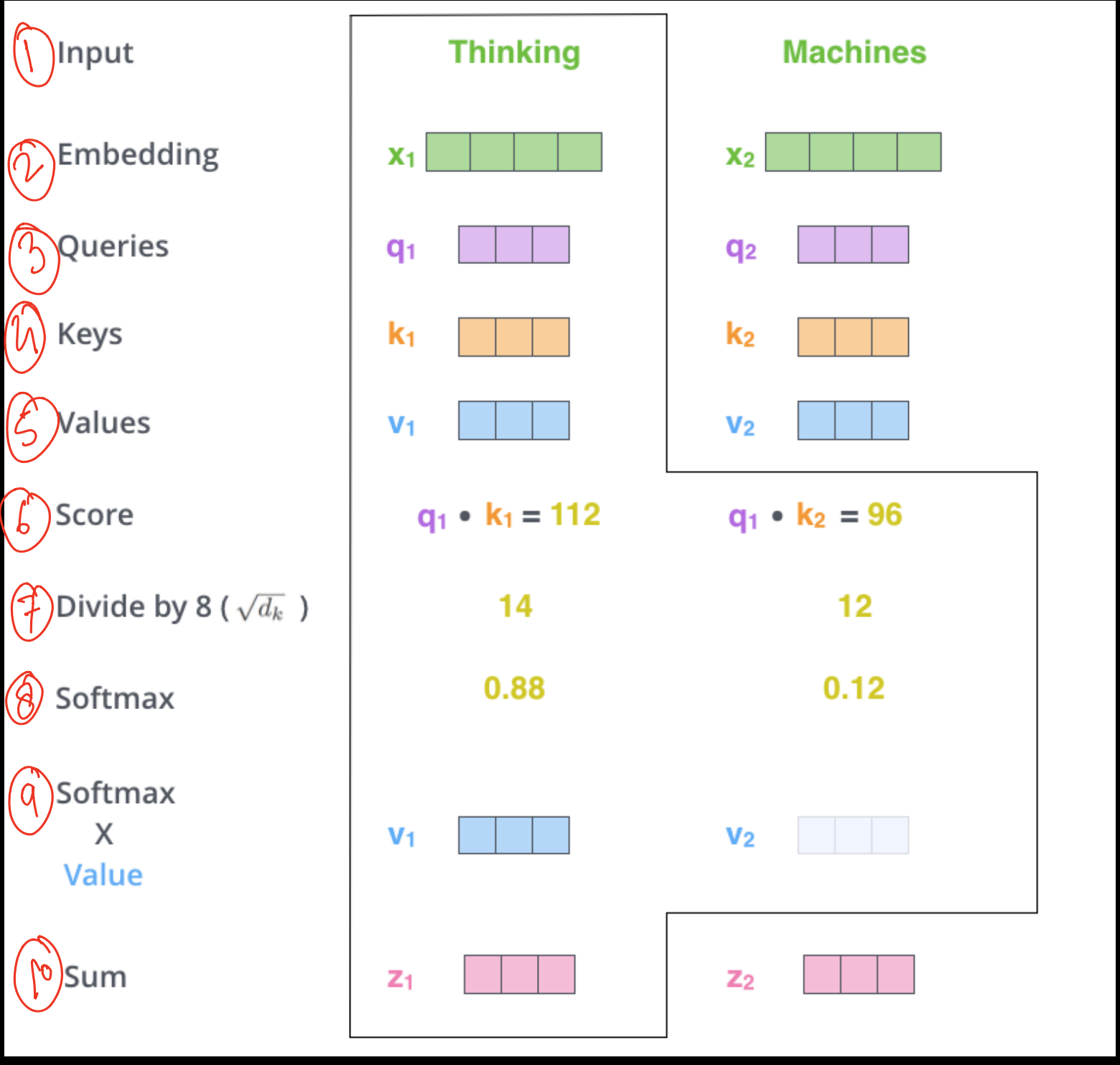
SA gives more attention to "The" & "Animal" while encoding "It"

III How SA is Calculated :

Basically Matrices need to be learned.

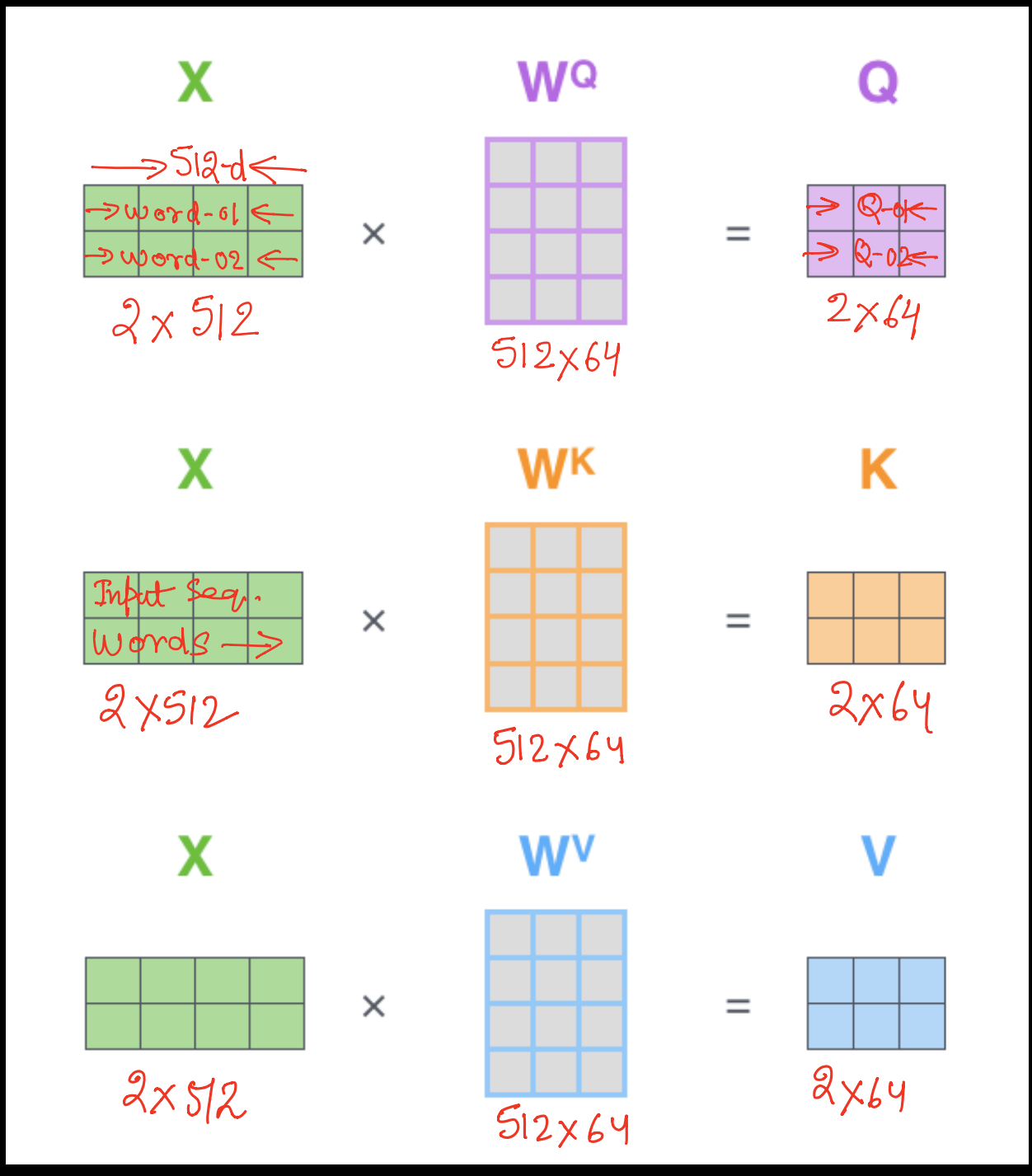


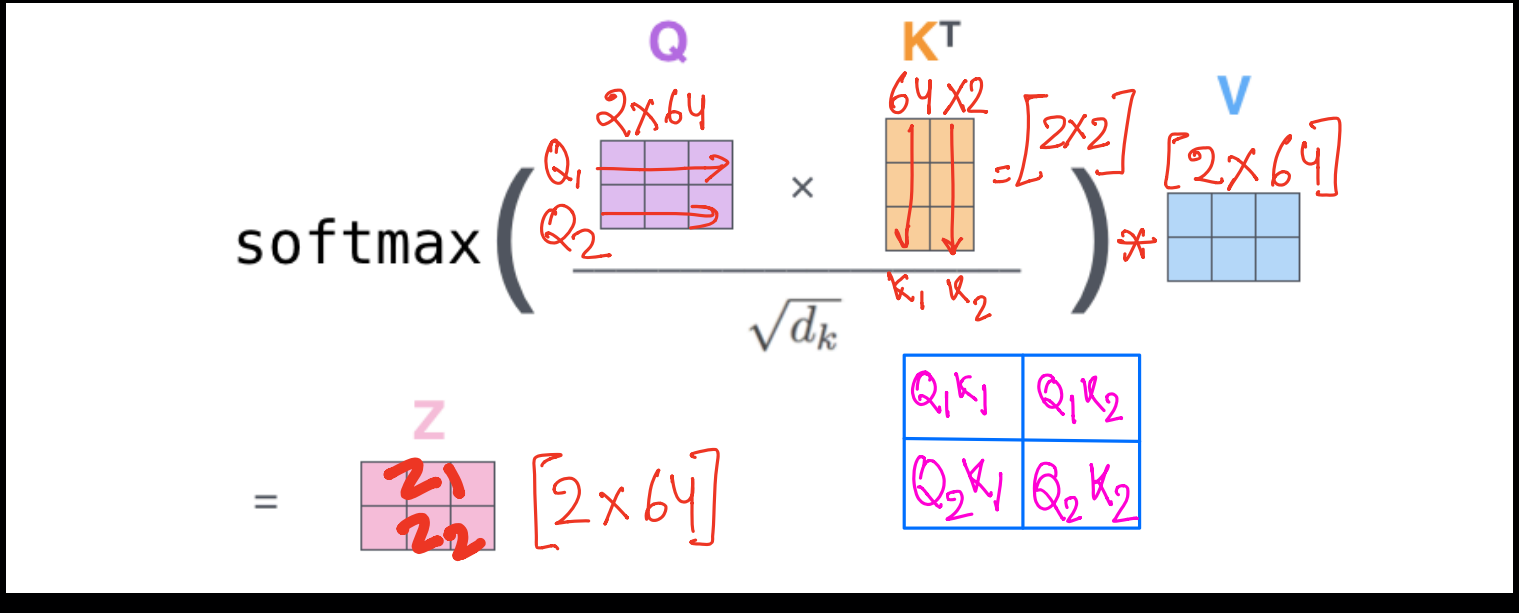
Role of Query / Key / Value need to be understood.



(*) Z_i's can be send to feed forward MLP network.

(*) These operations are done in matrix form for faster processing.


X
WQ
Q
512-d
word-01
word-02
2 x 512
512x64
Q-01
Q-02
2x64
X
WK
K
Input Seq.
Words
2X512
512x64
2x64
X
WV
V
2x512
512x64
2x64


Q
KT
V
2x64
64x2
[2x2]
[2x64]
Q1
Q2
K1 K2
softmax
$\sqrt{d_k}$
Z
[2x64]
Q1K1 Q1K2
Q2K1 Q2K2

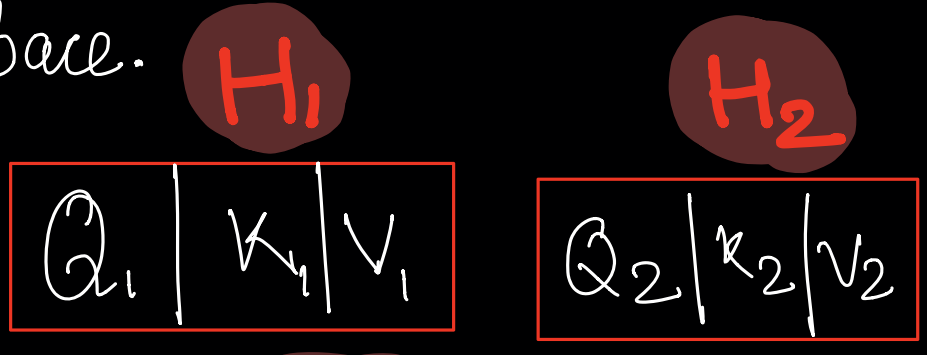
IV Multi-head attention :

* Attention allows model to focus on different positions. (Z_1 : weighted sum of all encodings)

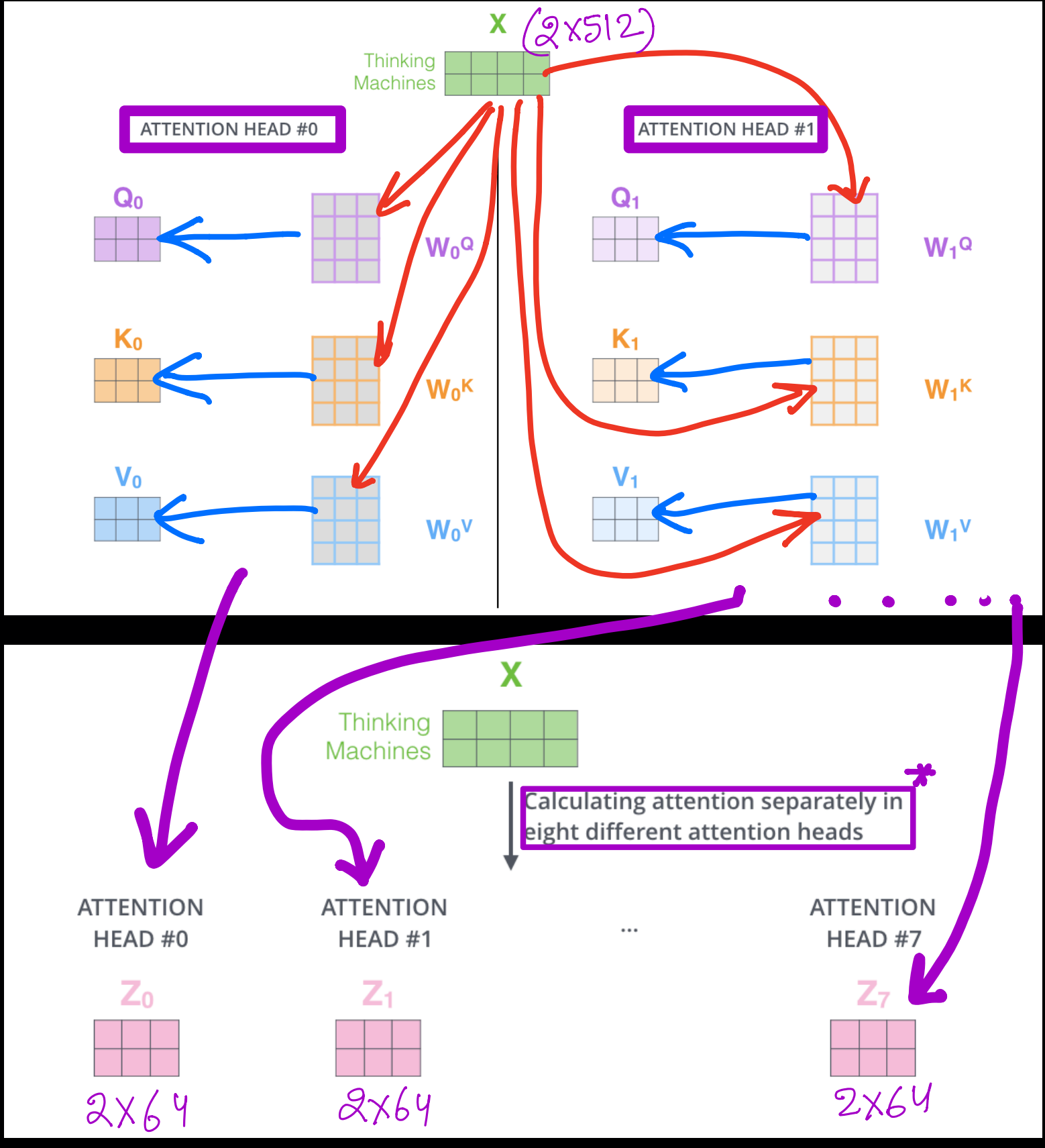
But dominated by its own embedding

* MHA enables attention layer to learn multiple representation Subspace.

Usually they have 8 heads.

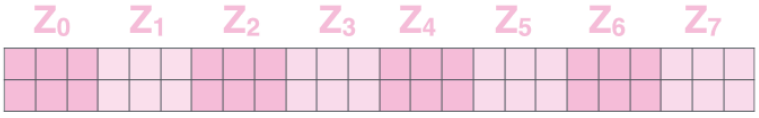


Each H_i projects its input into a different representation Subspace.

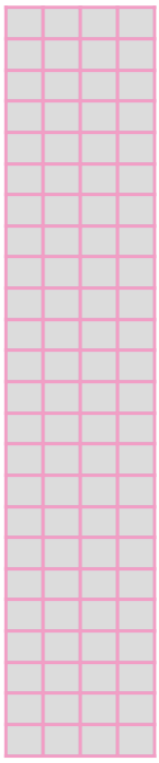


Instead of one [Z] vector of size (2x64) we have got (8) such [2x64] representation of (Z) vector.

1) Concatenate all the attention heads



2) Multiply with a weight matrix W^O that was trained jointly with the model



$Z_{Cat} = \underset{i=0-7}{\text{Concat}} [Z_i]$ $[2 \times (64 \times 8)]$ 2×512

x W_0 $[512 \times 64]$

3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

= Z

$[2 \times 64]$

$\therefore Z = Z_{Cat} * W_0$

$[2 \times 64]$ (2×512) (512×64)

Recap till now.

1) This is our input sentence*

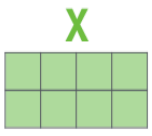
2) We embed each word*

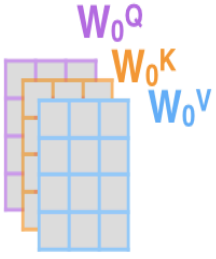
3) Split into 8 heads. We multiply X or R with weight matrices

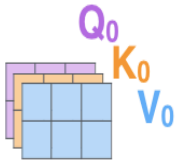
4) Calculate attention using the resulting Q/K/V matrices

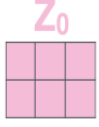
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

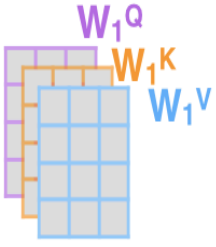
Thinking Machines

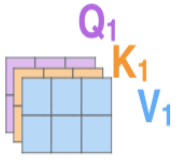


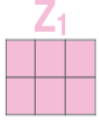


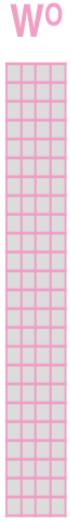


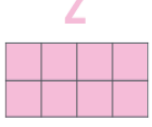




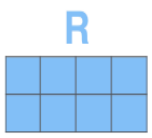


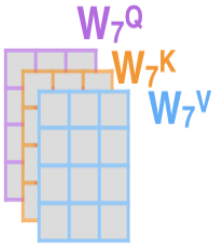


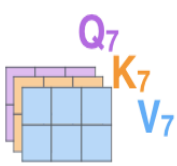


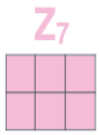


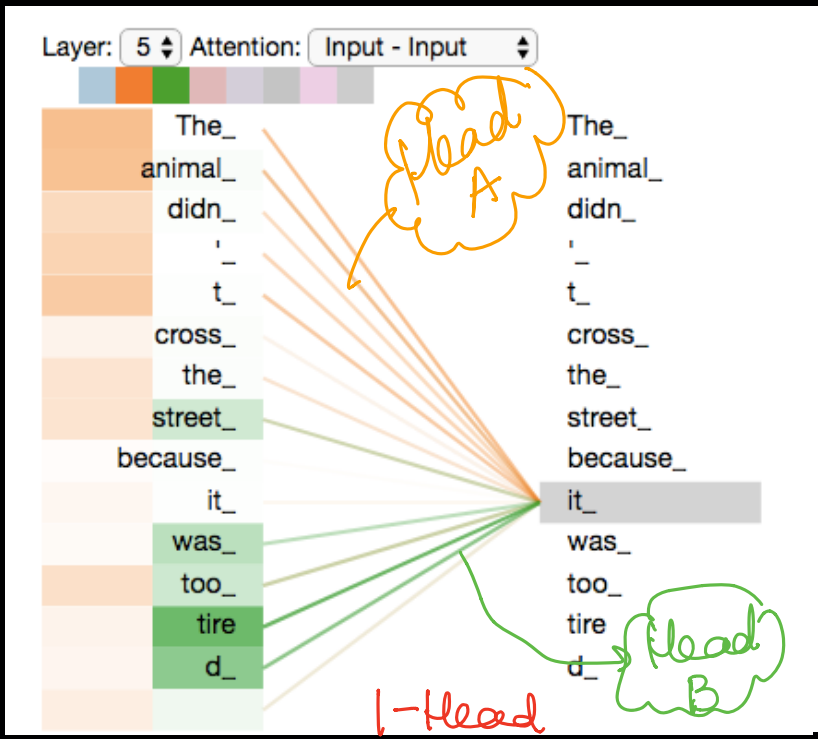
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

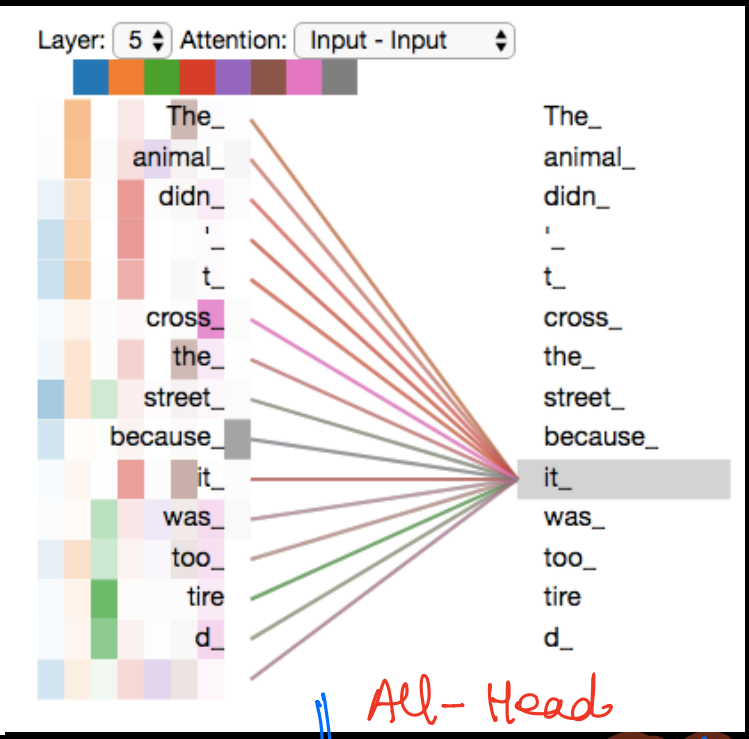












While encoding "it"

One head is very focusing on "Animal" or "The" (orange)

Other head may focus on "tired"

Accumulating the attention learned by all the heads!

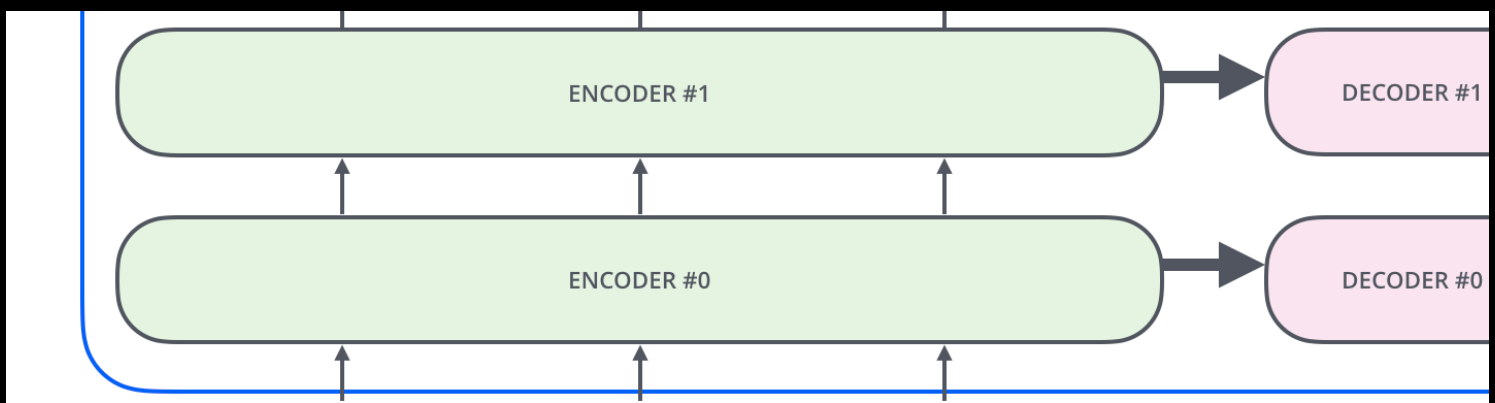
together "it" is being getting contributions from "Animal", "The", "Tired"

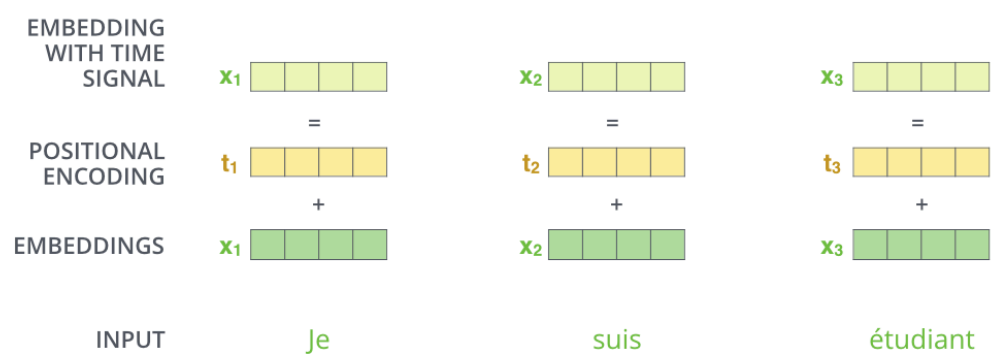
* Adding more heads may be helpful but later start to learn redundant attention or tend to overfit the data.

Positional Embedding

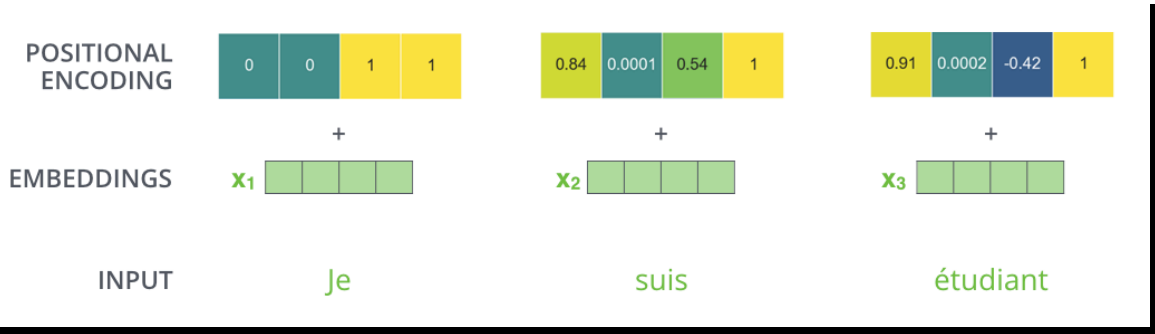
- Transformers are permutation equivariant.
- Hence alongwith word embeddings we need to pass positional embeddings.

$$I/P = W.E + P.E$$



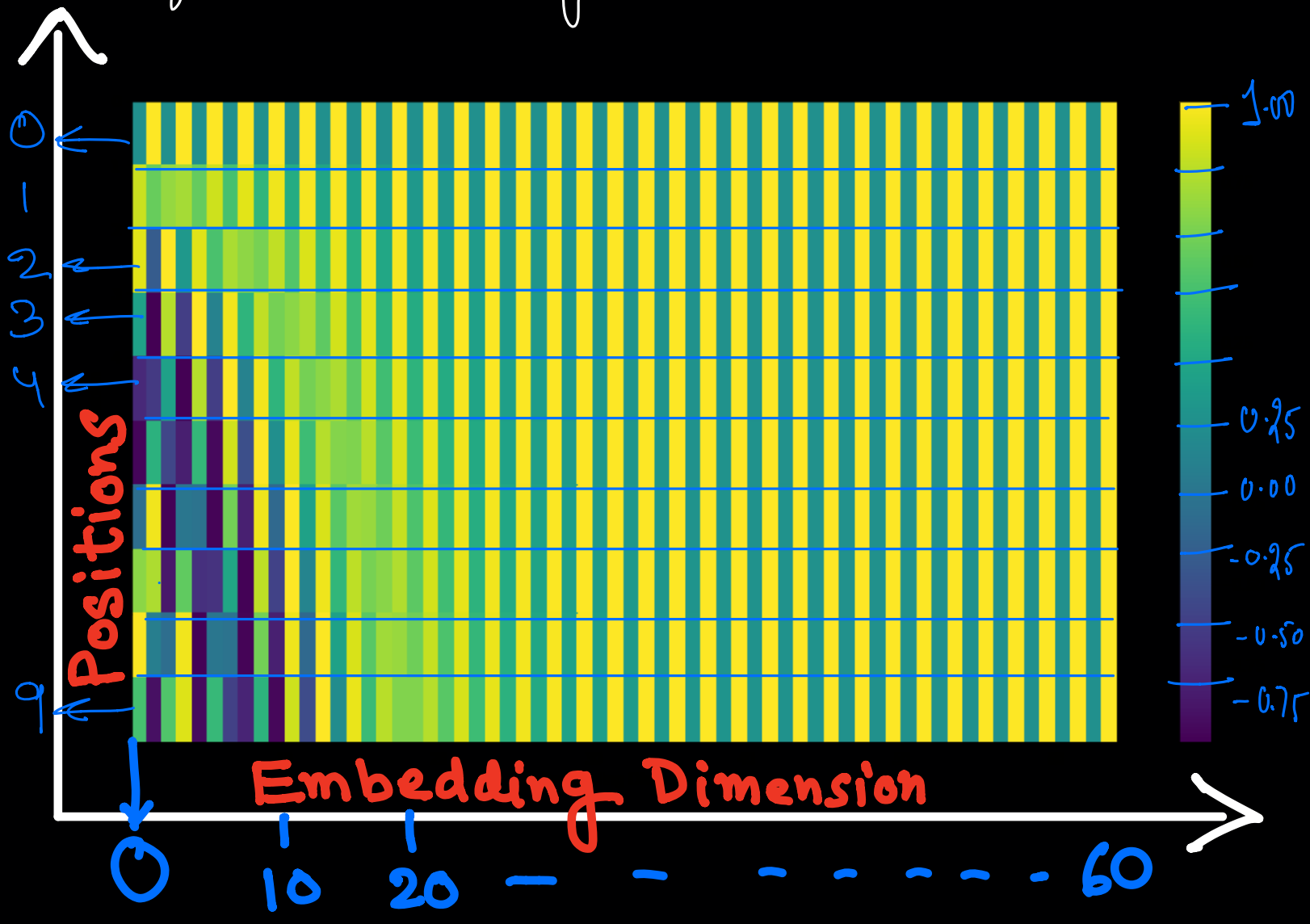


Helps N/W to understand the distance between the words in the I/P sequence.



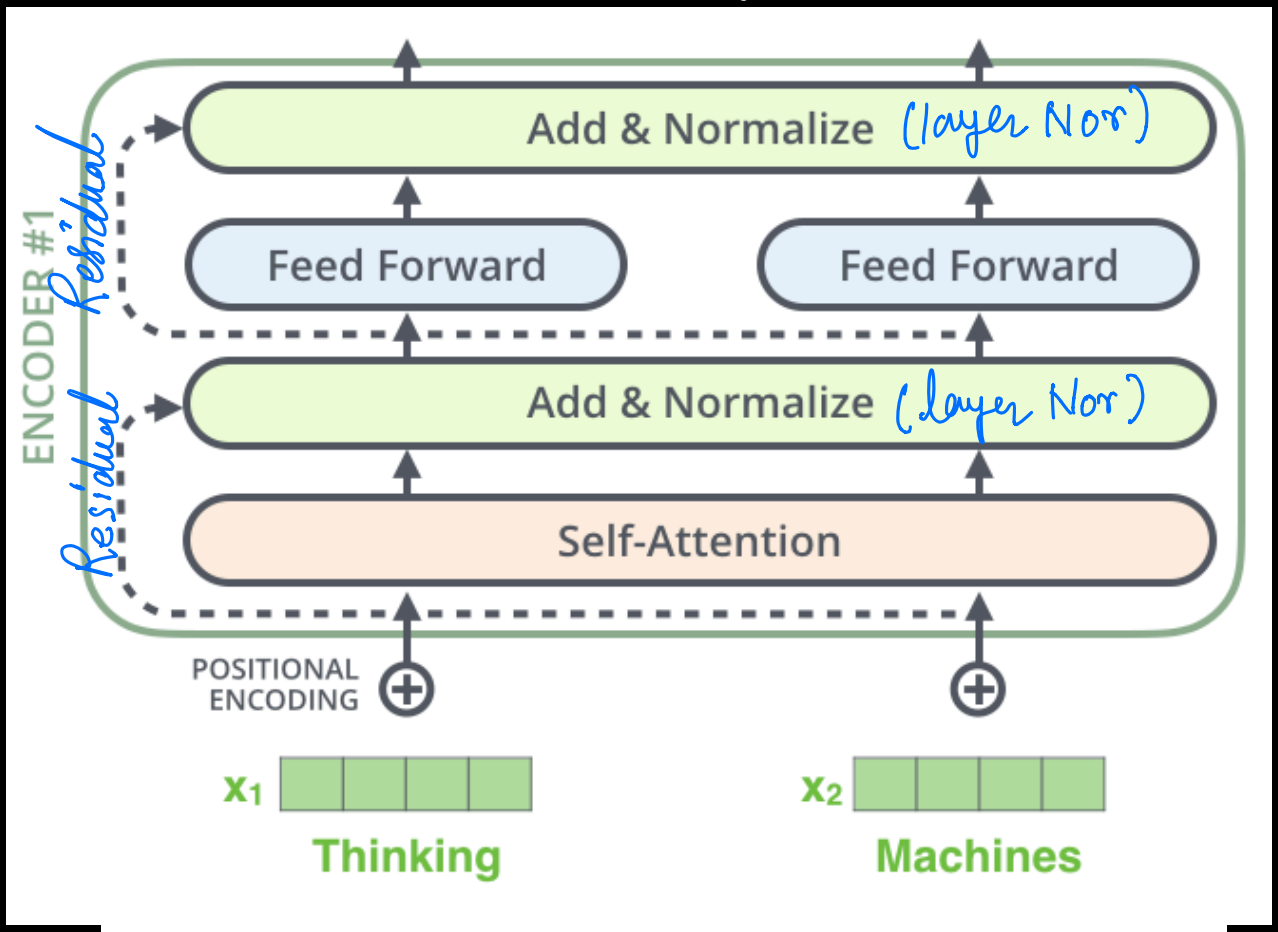
4-dim Embedding.

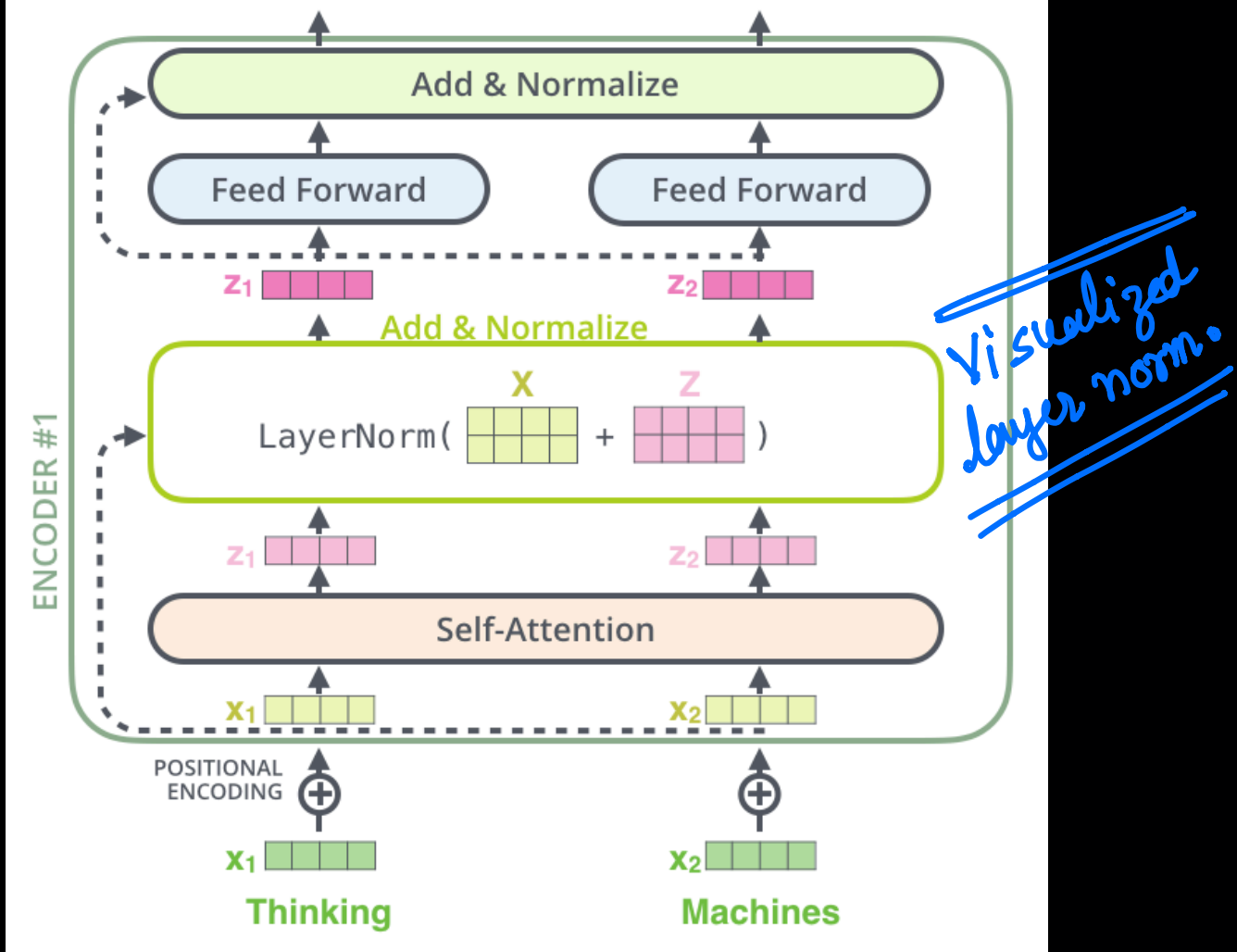
PE also helps to scale to unseen lengths of the testing sequences.

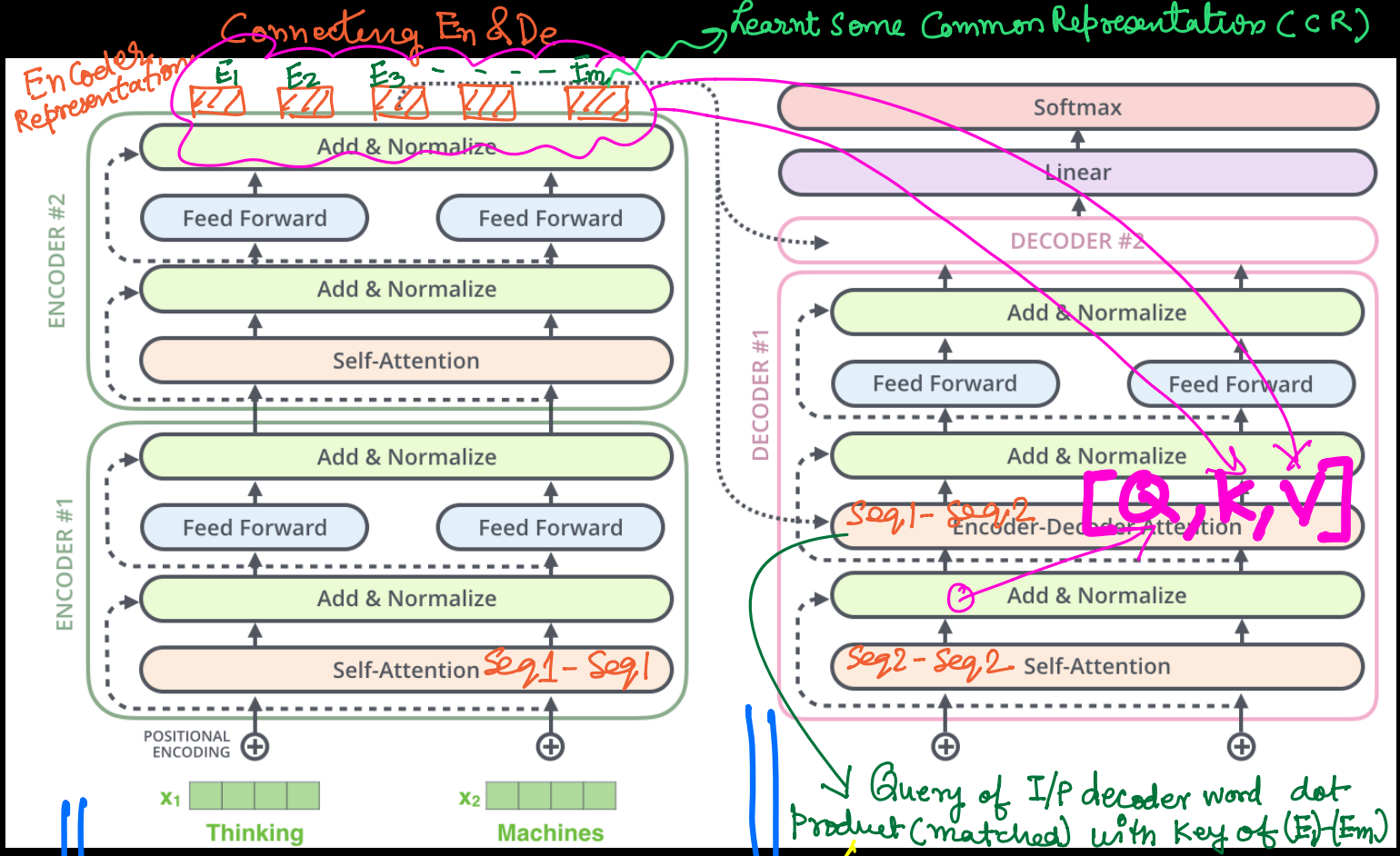


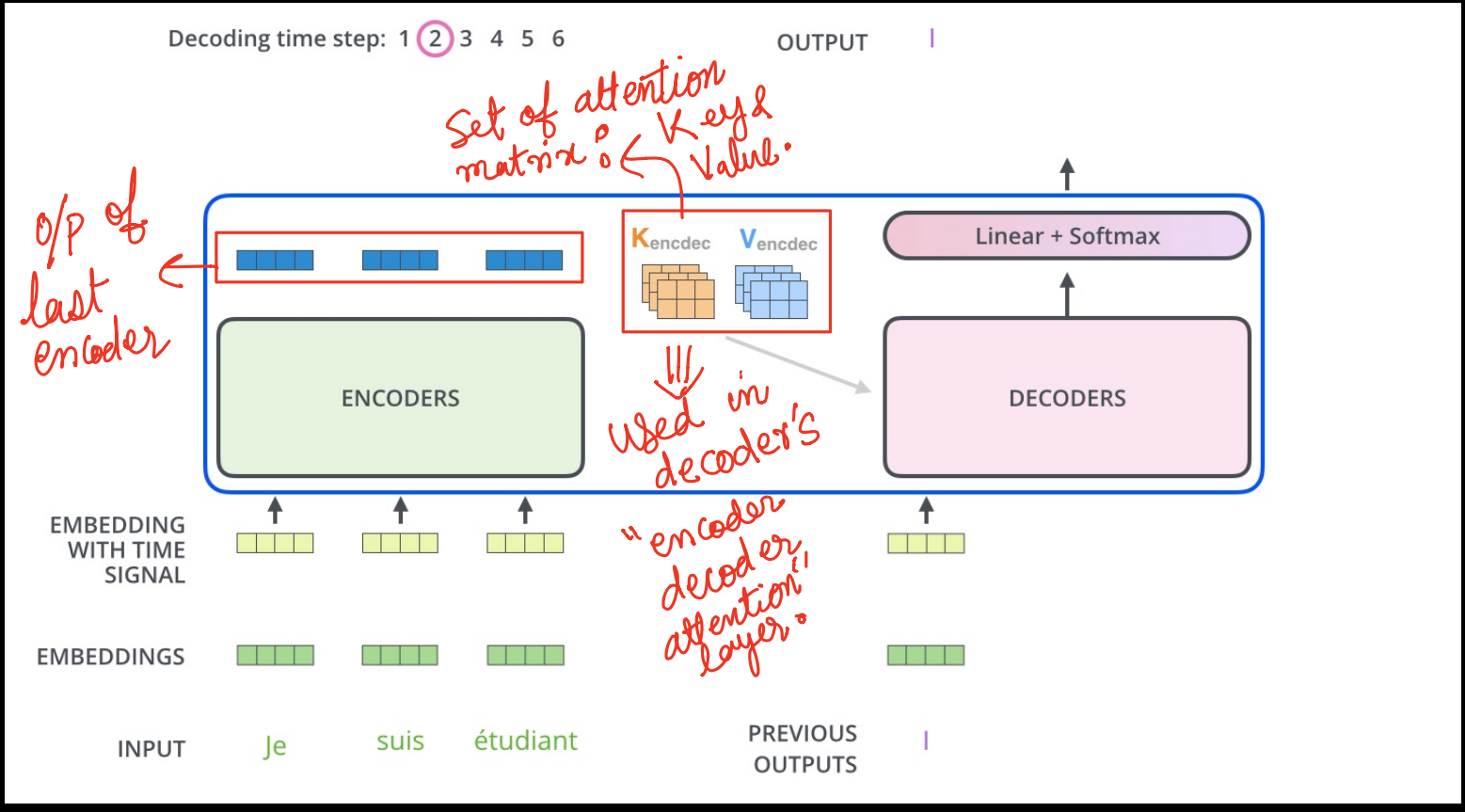
Both Sin & Cos functions are interleaved.

In encoder each sublayer has a residual connection, followed by layer normalization.









Decoder uses only previously shown input.

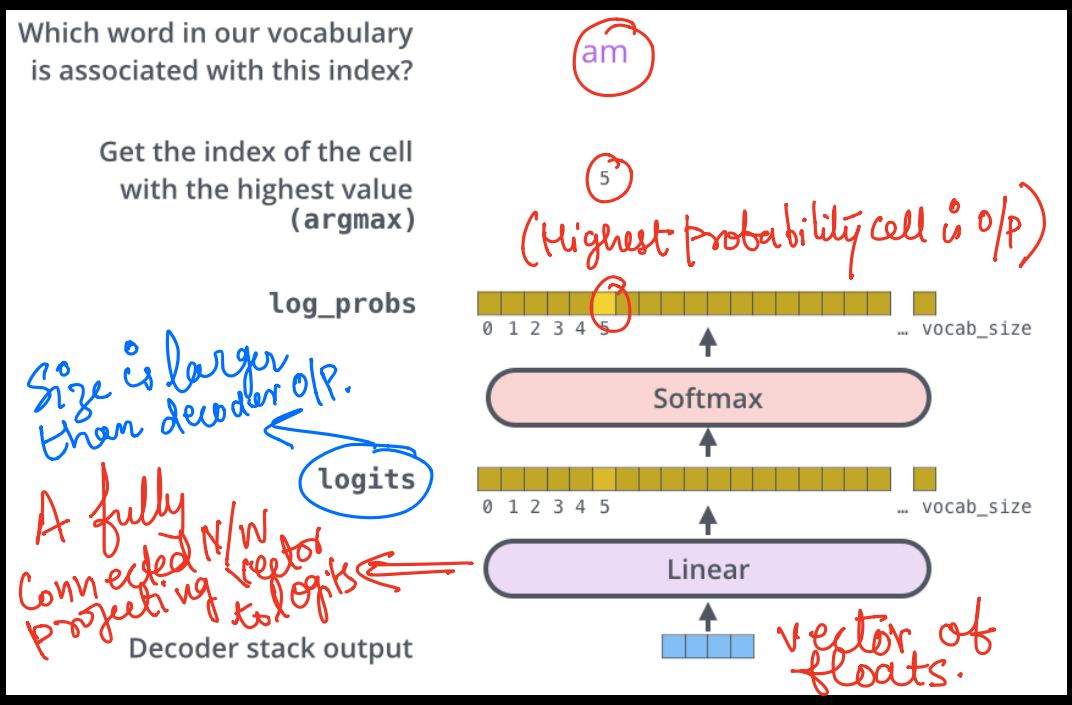
This is done by masking future positions by setting them to ($-\infty$) not (0) before Softmax step in self attention calculation.

(*) Encoder Decoder Attention layer works similar to [MHSA] but only creates Query matrix & takes Key & Value matrix from o/p of encoder stack.

(from Previous decoder layer) [Q] (from encoder o/p layer) [K] [V]

Final linear & Softmax Layer

(*) Decoder o/p : vector of floats ⟹ Vector 2 is Word is required for getting o/p word.

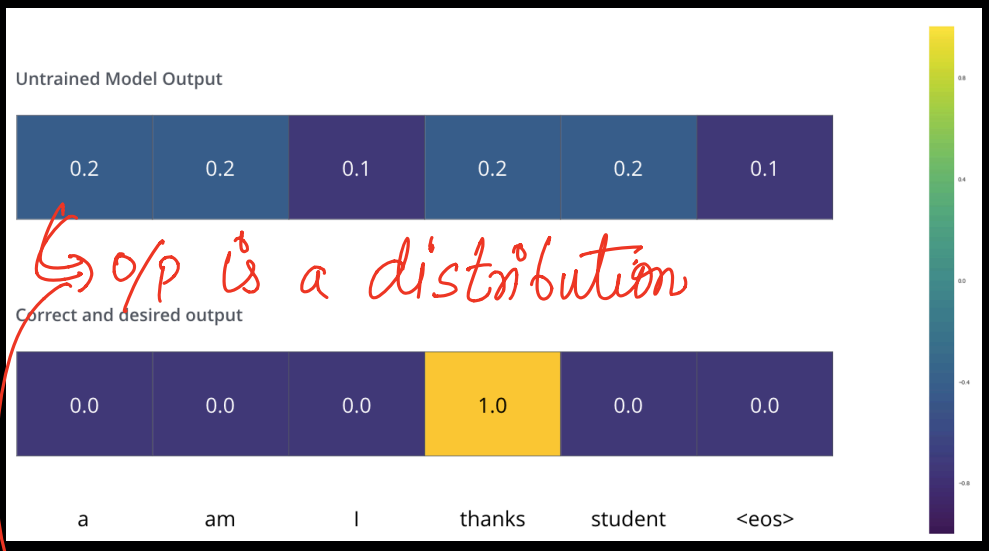


(*) Assuming that model has learned 1000 words as "o/p vocabulary".

⟹ logits's size is also (1K).

⊛ Finally Softmax converts raw logits scores into probabilities. Highest probability cell ⇒ O/P word.

The loss fn :



↳ O/P is a distribution

↳ Distributions can be compared using cross entropy / KL-Div.

Untrained model initially produce random / incorrect O/P.

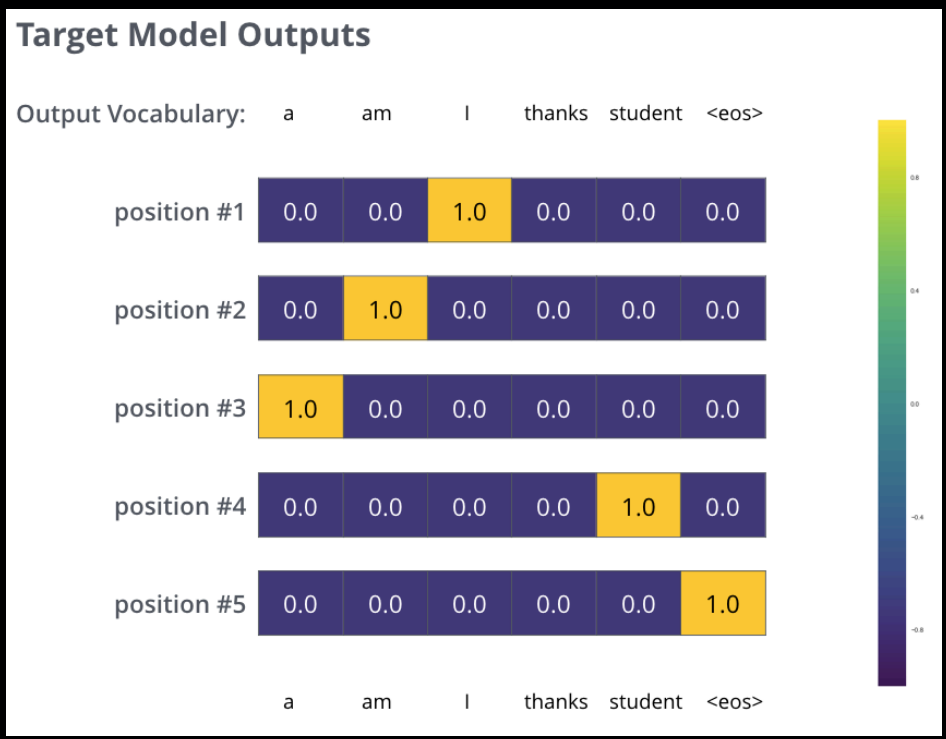
assuming O/P vocab: a / am / I / thanks / student / <eos>

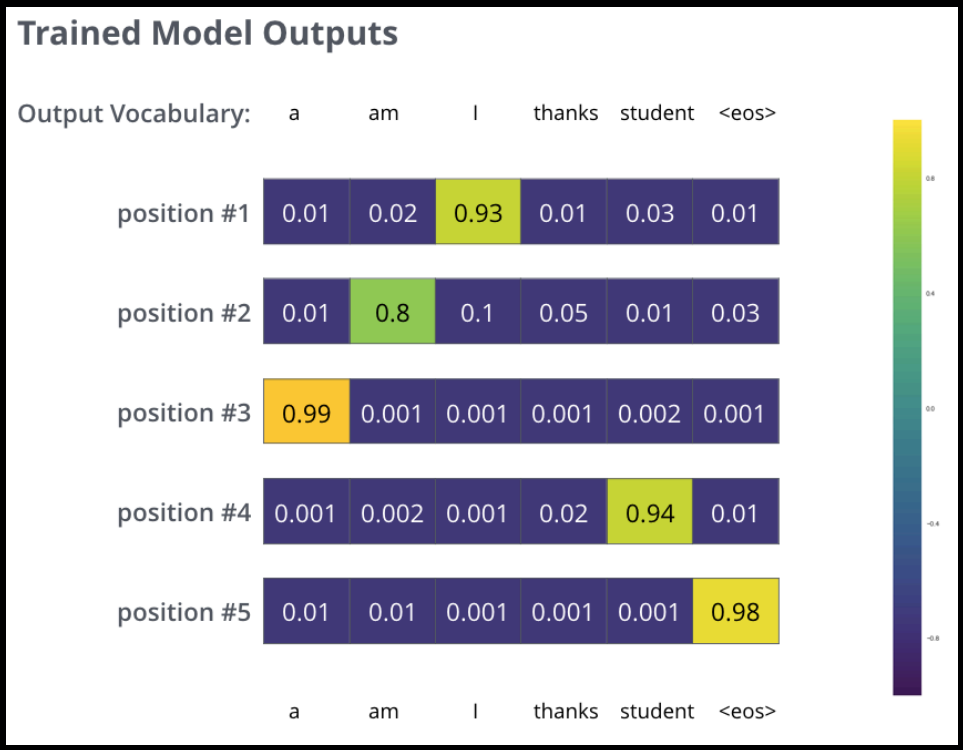
Assuming input as: je suis étudiant

⇓

expected O/P:

I am a student





After some initial training model may start to converge to the target distribution.

Model is selecting the "best match" highest probable word.

Beam Search

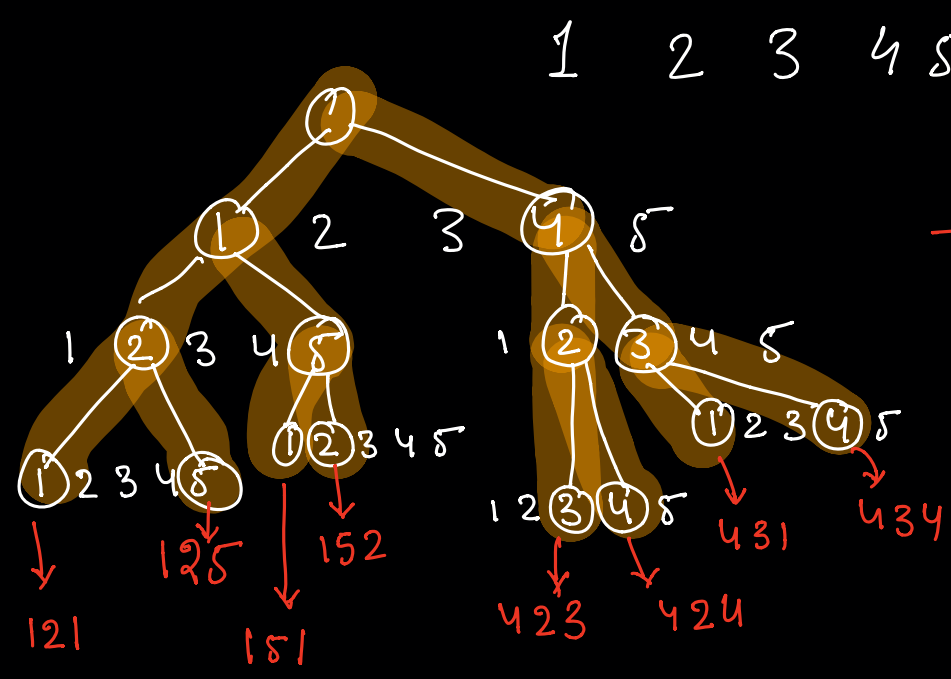
Instead of choosing Top 1 (say "I") ⇒ Considering the beam_size = 2

One can consider Top (2) words instead of just (1) (say "I" & "am")

(for the first o/p position) we need to run model twice once assuming "I" as the next word and once assuming "am" as the next.

⇒ whichever version produces less error

At any time (2) partial hypothesis (unfinished translation) is kept. are kept in memory.



beam size ≥ 2

max o/p length = 3

⇒ It will Consider all of them and Choose the one giving minimum error.