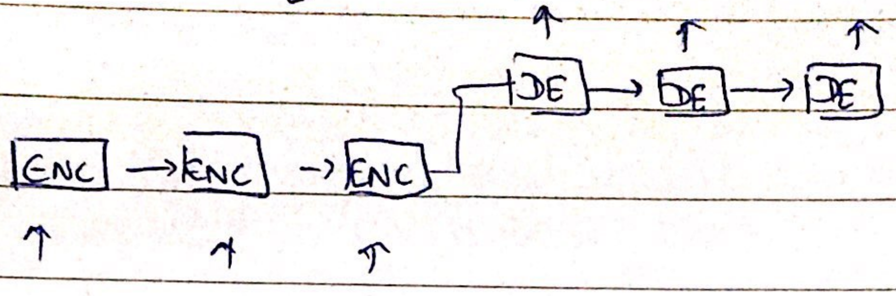
Encoder/Decoder

Seq 2 Seq:



- The → arrow is context vector
- If attention is added then it's weighted sum of context vector

Problems:

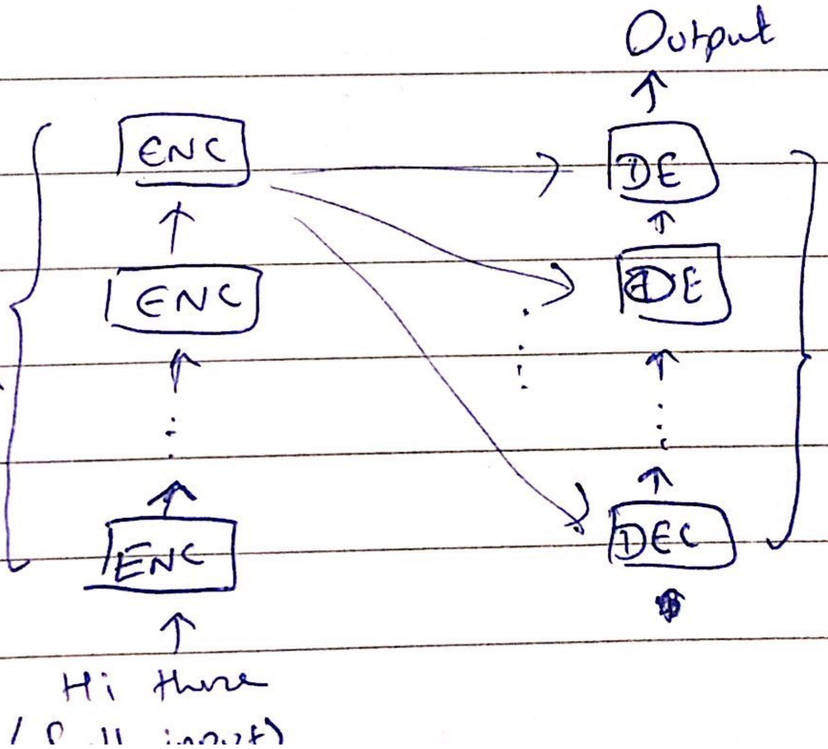
1) Cant do parallelization in this Structure
2) long range dependencies not possible.

Soln

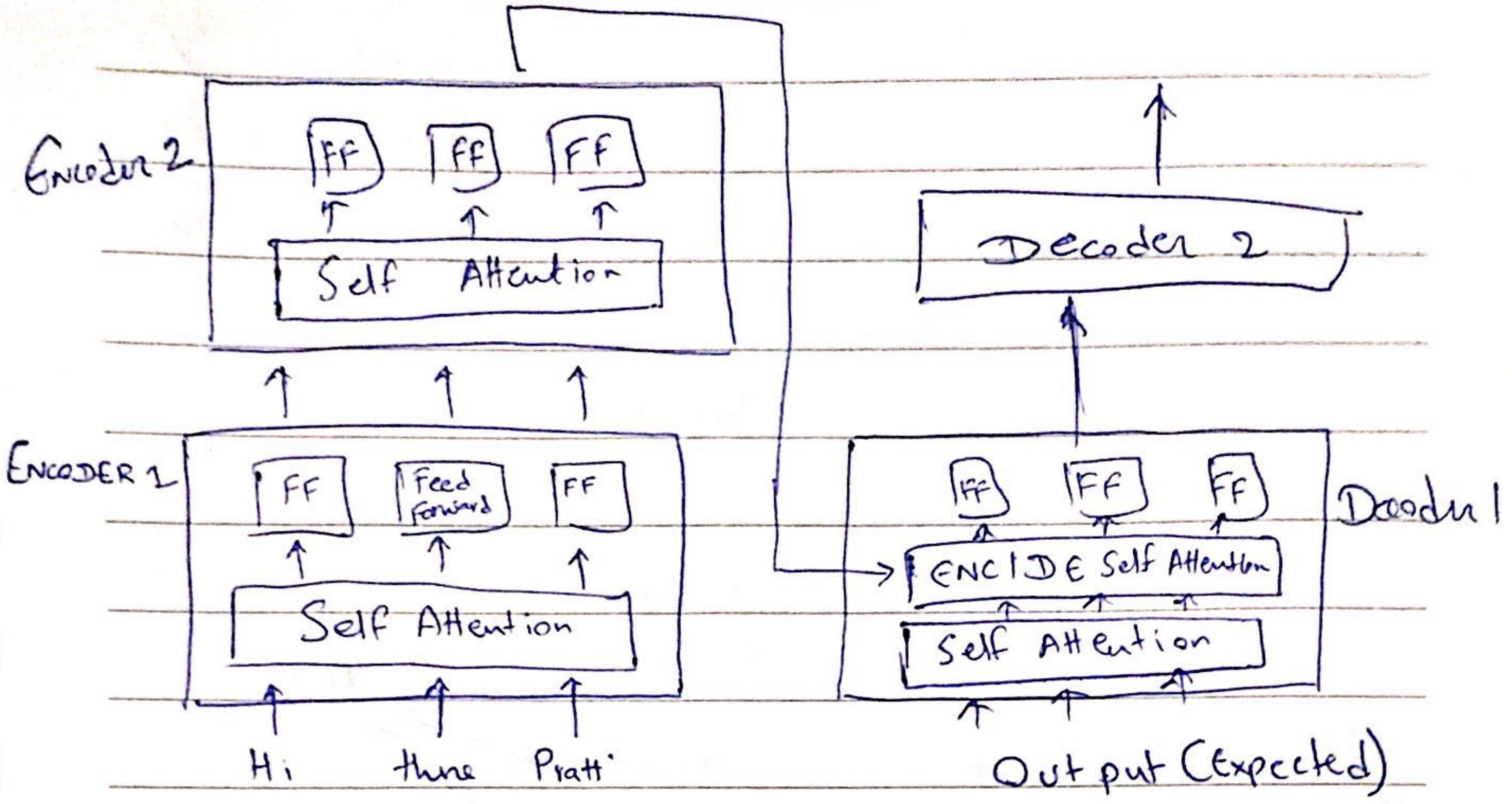
1) Transformers!

- Transformers.

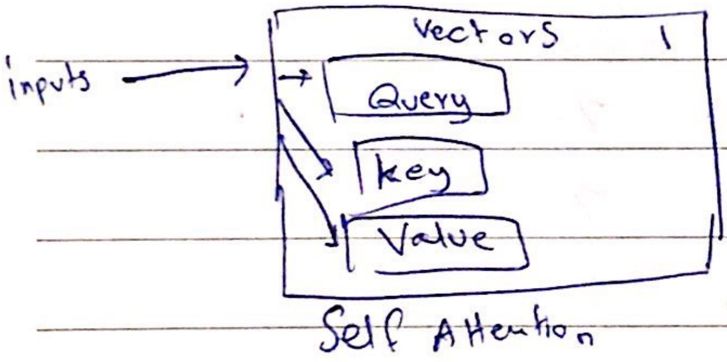
No of Encoders is a hyperparameter



No of Decoders = No of Encoders



- What is Self Attention?



Calc. relation to other words.

Word	q vector	k vector	v vector	score
Hi	q_1	k_1	V_1	$q_1 k_1$
there		k_2	V_2	$q_1 k_2$
Pratt		k_3	V_3	$q_1 k_3$

Word	q vector	kvector	vvector	score	score/8	Softmax
"	"	"	"	"	$q_1 k_1/8$	x_{11}
					$q_1 k_2/8$	x_{12}
					$q_1 k_3/8$	x_{13}

Mult. by value vectors
↑

Score/8	Softmax	Softmax * V	Sum
"	x_{11}	$x_{11} * V_1$	$\rightarrow z_1$
	x_{12}	$x_{12} * V_2$	↗
	x_{13}	$x_{13} * V_3$	↗

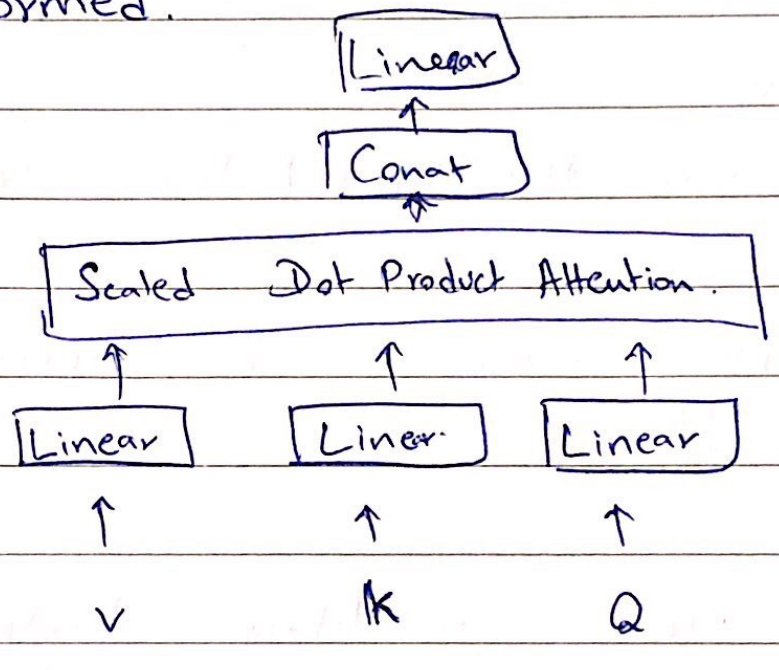
∴

Word		Sum
Hi	⟶	z_1
there	⟶	z_2
Pratt	⟶	z_3

Similarly we get Sum vector for all words z_1 z_2 z_3

In this way it can be calculated parallely.

The outputs are concatenated & linearly transformed.



[MULTI HEAD ATTENTION]

Challenges:

1) Fixed Seq. length

2) Thus sentence needs to be cut in the middle leading to loss of context from the other half.

Solution:

1) Transformer XL

Transformer XL

The Hidden state calc. from previous State is used as additional context for current segment.

Problem:

1) Increases the computation speed manifold.

→ A new Opponent has appeared! BERT!!!

BERT uses multilayer ~~bider~~ bidirectional Transformer encoder. It's self attention is in both directions