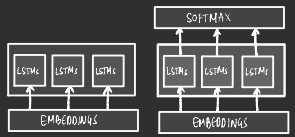
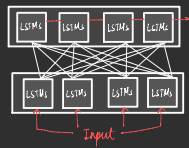
Origin Story

2014 → Seq 2 Seq learning with NN
→ Concept of Encoder and Decoder



→ Problem → Long Inputs
↳ Inputs are stored in Context vectors.

2015 → Neural Machine Translation
→ Concept of attention was introduced



→ Problem: Cannot work in parallel
↳ use of LSTMs.

2017 → Attention is all you need.
→ LSTMs & RNNs → ✗
→ Transformer Arc & Self Att → ✓
→ Very stable on huge datasets

Self Attention

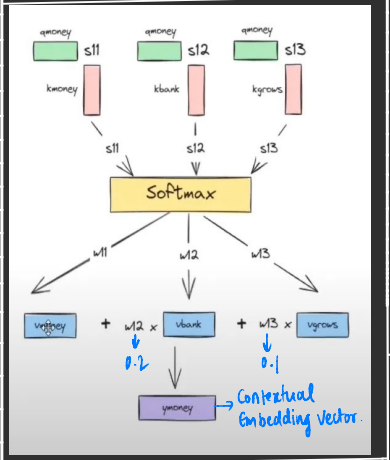
NLP → Convert Words → Numbers
• Word Embedding

Cricketer → [0.2, 0.2, 0.3, ...]
Queen → [0.7, 0.6, 0.1, ...]
King → [0.7, 0.1, 0.3]
↑ Royalty ↑ Feminity

Problem → Average Meaning
apple → Company ; apple → Fruit

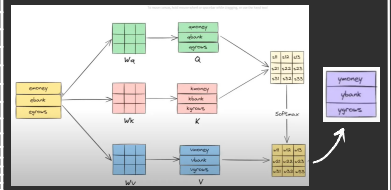
Sol^n → Contextual Embedding
e.g) Money Bank Grows
River Bank Flows → [Bank]

CE bank¹ = 0.7 Money + 0.2 Bank + 0.1 Grows
CE bank² = 0.5 River + 0.3 Bank + 0.2 Flows



Problem → Task Specific CE
• Piece of Cake
actual piece of Cake — Very Easy Task

Sol^n → Making 3 different vectors from Main CE Vector.
Query →
Key →
Value →
HOW? → Linear Transform.
WHY? → Accuracy and Efficiency.



Attention (Q, K, V)

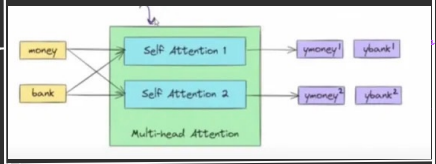
$$= \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$S_{11}, S_{12}, \ldots, S_{33} \rightarrow Q \cdot K^T \rightarrow$ Numbers
↳ If the diff ↑ = Variance ↑ hence it will cause problem after Softmax is applied.

Sol^n → Divide by $\sqrt{d_k}$ → dimensionality of K-Vector.
→ Variance ↑ ∝ d_k ↑ ↳ [x → var(x), y = cx, var(y) = c²var(x)]

Problem → Self Attention captures only 1 perspective.
e.g.) The Man Saw the Boy with Camera.

Multi-Head Att



Masked Self Att

Prev. Input

Decoder → Inference time → AutoReggress
→ Training time → Non-AutoRegress (Parallel)

Problem → Training → AR → Slow ✗
→ Training → NAR → cheating ✗

Teacher forcing → We can use NAR while training because we have data-set available.

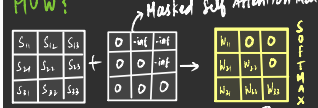
Why cheating → Inference cannot be done in NAR and hence we cannot do Training in NAR.

Money bank grows.

$Money_{CE} = 0.7\,Money + 0.2\,bank + 0.1\,grows$
$Bank_{CE} = 0.2\,Money + 0.5\,Bank + 0.3\,grows$
$grows_{CE} = 0.1\,Money + 0.3\,bank + 0.6\,grows$

→ Values of 'bank' and 'grows' are anonymous for $Money_{CE}$ as they have not been made.
→ Hence we have to make them "0".

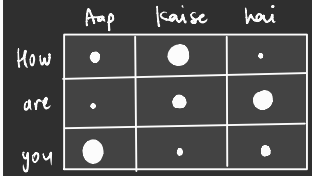
HOW?



• Softmax makes [-inf to "0"]

Cross-Attention

Find Similarity in one Sentence → SA
Find Similarity in different Sentence
↳ Cross Attn.



SA → (Q, K, V) → same Vector
CA → (Q) → Decoder Vector
(K, V) → Encoder Vector.

Positional Encoding

→ Self Attention works in Ilal.
Problem → works in Ilal and hence position of words become ambigious.

→ Out of the 512 dimension we can add one more Embedding to End and make it 513 dimension and add [1,2,3,4...] to it.

PROBLEM

- unbounded → no upper limit
  - BP hates huge numbers as it creates instability.
- Discreate Nos → we need continous numbers.
- Relative Position
  - subject can be 5th and Verb be 6th or it can be 105th and Verb can be 106th.
  - This relationship of verb always following the subject makes the reading part easy for model.

Trignometric functions →

- Instead of adding discreate nos [1,2,3...] we add the sine values of it.
- Repetetion will occur hence we add the cosine values also [sin(1) cos(1)]

WE = [| | | | ..... |]

- To reduce more Repetetion we add more components of Sine & Cosine function.

e.g. [sin(1), cos(1), sin(1/2), cos(1/2)..]

- Do decide how many Sine and cosine component will come.

→ make a PE(vector) [$d_{PE} = d_{WE}$]

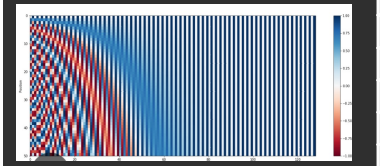
e.g.) Money Bank Grows

WE money = [1|2|3|4|5]

\+

PE money = [1|2|3|4|5]

[We add this vector and do not concanate]



- As the number of words increase I value of (pos) increases and hence the value of numerator in PE formula changes which brings more difference in the PE vector.
- As observed from the graph we can see that for the 1st words the PE vector is just 0s & 1s but as the no. of words increase the values get more diverse.
- This is similar to Binary Encoding.

e.g.) 0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100

[1) one place changes Alternatively. 2) tens place changes after 2 iterations....]

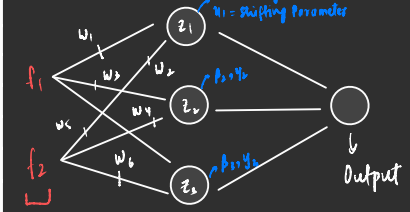
Normalization

- process of transforming the data or the output in a way where $\mu = 1$ and $\sigma^2 = 0$.
- Why? → Improves Training stability
  - → Faster Convergence
  - → Reduces the covariate Shift.

* Batch Normalization

Restore Flexibility Post Normalize..

β_1 = Scaling Parameter
γ_1 = Shifting Parameter



Input $w_1 f_1$ $w_2 f_2$ w_3

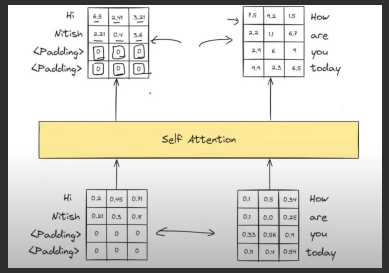
f_1	f_2	Z_1	Z_2	Z_3
2	3	7	5	4
1	1	2	3	4
3	6	1	2	3
4	1	7	5	6

$z_1 = 2w_1 + 2w_2 + b_1 = 7$
$z_2 = 2w_4 + 3w_5 + b_2 = 5$
$z_3 = 2w_6 + 3w_6 + b_3 = 4$

BN calculation → $\frac{7-\mu_1}{\sigma_1} = 0.36$ → $0.36\gamma_1 + \beta_1$

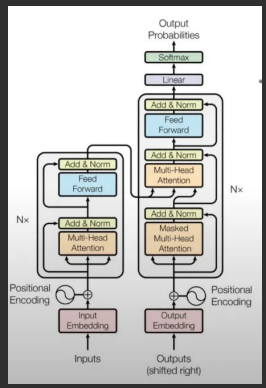
$\frac{6-\mu_3}{\sigma_3} = 0.80$ → $0.80\gamma_3 + \beta_3$

PROBLEM → does not work well with Self Attention.



- Now as the GCC are stacked up BN will cause issue because of the zeros.
- Hence, we use Layer normalization (Horizontally). Hence instead of calculating mean Vertically we do them horizontally.

Transformer Architecture



- There are multiple Encoders and Decoders in Transformers.

"Attention is all you need" specifically used 6 of them.

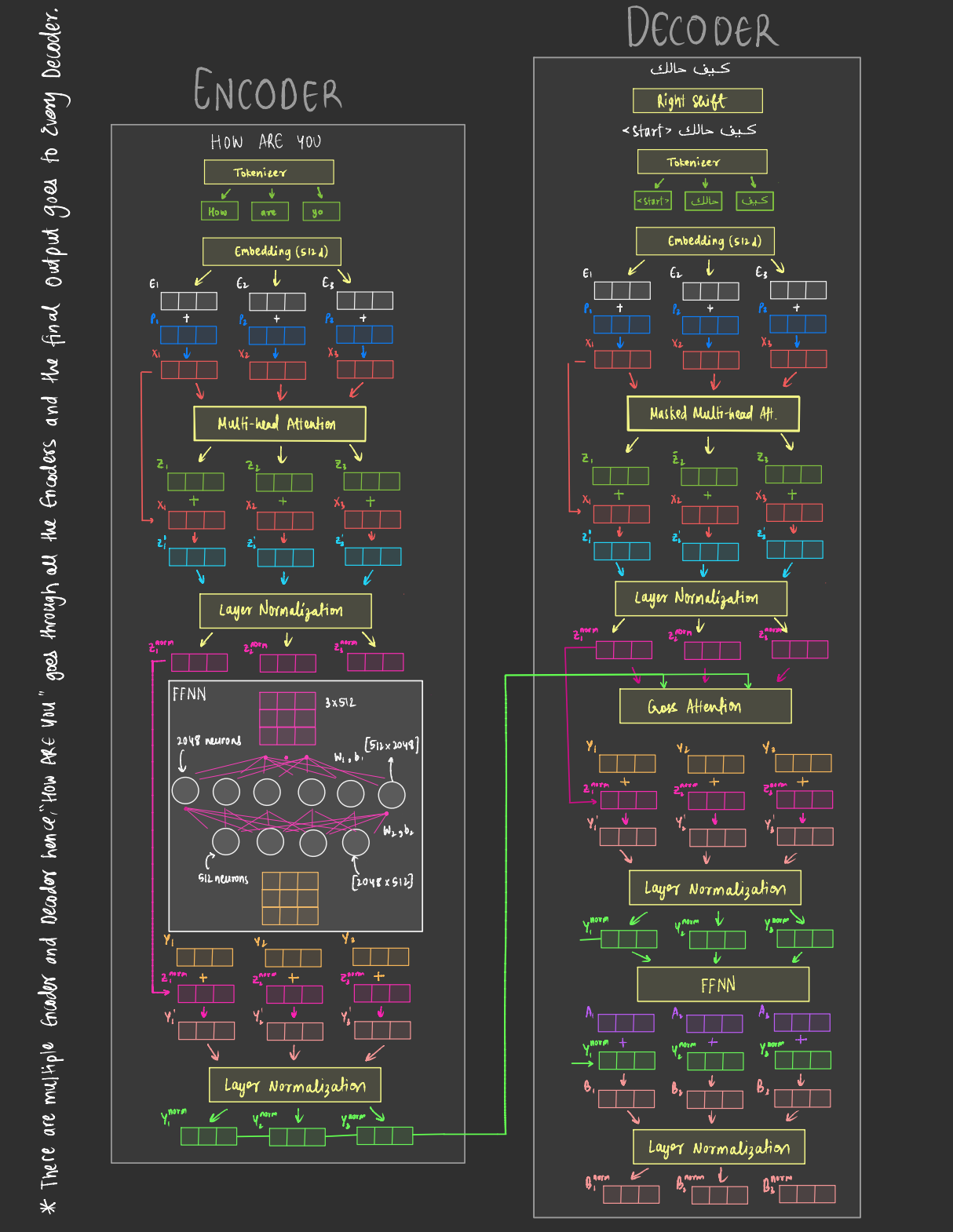
Feed Forward Neural Network

- "Feed-Forward" = No regression or Loops.
- Why? → Adds Non-Linearity.
  - Helps the model learn more complex patterns

How? → Makes the representation more Expressive.

→ Each token is processed individually but the weight remain the same. [parallizable]

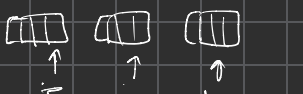
$[\text{relu}(Z_{norm} W_1 + b_1)] W_2 + b_2$


* There are multiple Encoder and Decoder hence, "How are you" goes through all the Encoders and the final Output goes to Every Decoder.
ENCODER
HOW ARE YOU
Tokenizer
How
are
yo
Embedding (512 d)
Multi-head Attention
Layer Normalization
FFNN
3x512
2048 neurons
[512x2048]
512 neurons
[2048x512]
Layer Normalization
DECODER
كيف حالك
Right Shift
<start> كيف حالك
Tokenizer
<start>
حالك
كيف
Embedding (512 d)
Masked Multi-head Att.
Layer Normalization
Cross Attention
Layer Normalization
FFNN
Layer Normalization

Calc. of PE vector Using Formula

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

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



$\Rightarrow \boxed{dm = 4}$

I am Nauman

pos → (0) (1) (2)

I → i=0

$PE_{(0,0)} = \sin(0) = 0$

$PE_{(0,1)} = \cos(0) = 1$

i=1

$PE_{(0,2)} = \sin(0) = 0$

$PE_{(0,3)} = \cos(0) = 1$

i=2

$PE_{(0,4)} = \sin(0) = 0$

$PE_{(0,5)} = \cos(0) = 1$

Am → i=0

$PE_{(1,0)} = \sin\left(\frac{1}{10000^{0}}\right) = 0.841$

$PE_{(1,1)} = \cos\left(\frac{1}{10000^{0}}\right) = 0.540$

i=1

$PE_{(1,2)} = \sin\left(\frac{1}{10000^{2/4}}\right) = 0.009$

$PE_{(1,3)} = \cos\left(\frac{1}{10000^{2/4}}\right) = 0.995$

i=2

$PE_{(1,4)} = \sin\left(\frac{1}{10000^{4/4}}\right) = 0.00001$

$PE_{(1,5)} = \cos\left(\frac{1}{10000^{4/4}}\right) = 0.99999$

Nauman → i=0

$PE_{(2,0)} = \sin\left(\frac{2}{10000^{0}}\right)$

$PE_{(2,1)} = \cos\left(\frac{2}{10000^{0}}\right)$

i=1

$PE_{(2,2)} = \sin\left(\frac{2}{10000^{2/4}}\right) = 0.002$

$PE_{(2,3)} = \cos\left(\frac{2}{10000^{4/4}}\right) = 0.9999$

i=2

$PE_{(2,4)} = \sin\left(\frac{2}{10000^{4/4}}\right) = 0.00002$

$PE_{(2,5)} = \cos\left(\frac{2}{10000^{4/4}}\right) = 0.9999999$