Date : ....
MON TUE WED THU FRI SAT SUN

To generate these 3 different vectors, we take 3 matrices W_q, W_k, W_v

$$e_{bank} \times W^q = q_{bank}$$

These matrices are ~~b~~ trained with data using backpropagation.

Stanford Lec 8:

Self-attention Problem 1:

Sequence order:

adding P_i to our inputs.

~~Prob. 2: seq~~

Positional embedding:

$$\tilde{x}_i = x_i + P_i$$

→ Sinusoidal representation:

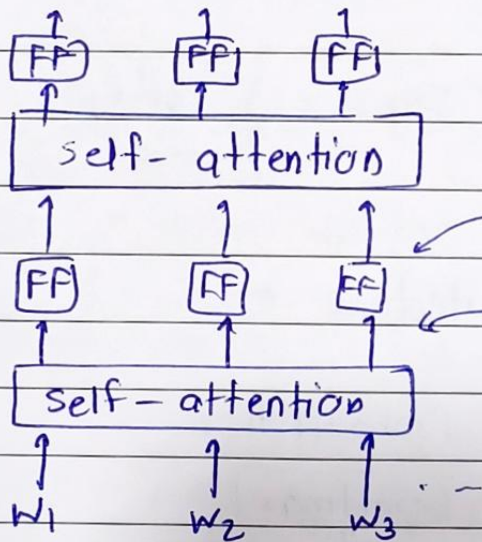
$$P_i = \begin{bmatrix} \sin(i/10000^{2i/d}) \\ \cos(i/10000^{2i/d}) \\ \vdots \\ \sin(i/10000^{2 \cdot d/2/d}) \\ \cos \end{bmatrix}$$

Cons:
Not learnable

Other way : make P_i = learnable param

Problem 2: adding nonlinearities.
Add a feed forward network to ~~possess~~ post-process each O/P vector.

$$m_i = MLP(O/P)$$
$$= W_2 \times ReLU(W_1 \, output_1 + b_1) + b_2$$



O/P passed to FF

Here the O/P is the contextual word embedding.

FFN enhances the embeddings for tasks like transl^{tn}, summarization, etc. NLP tasks.

Problem 3 - Masking the future words
In text translation / generation, In DEC part, we want the model to look only

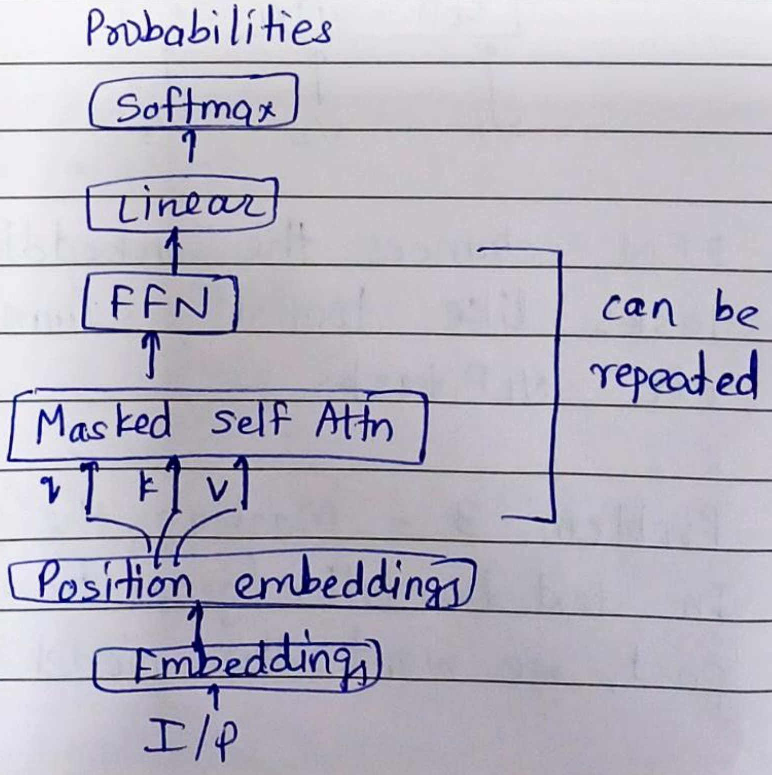
at words generated previously, not future.

Inefficient soln:
- Change the keys & query vectors @ every timestep.

Efficient soln: Mask by assigning $-\infty$ so that $e^{-\infty}$ in softmax $= 0$

$$e_{ij} \text{ or } s_{ij} = \begin{cases} q_i^T k_j, & j \le i \\ -\infty, & j > i \end{cases}$$

→ Minimal SA architecture ←



Date :................................
MON TUE WED THU FRI SAT SUN

Multi-Head Attention:

eg. I went to Stanford CS 224n & learned.

Attn Head 1 might focus on ENTITIES like names, places.

Attn Head 2 might focus on SYNTACTICALLY RELEVANT WORDS

These 2 heads will have different wts.

eg. The cat sleeps.

$$\underbrace{[1\ 0.5\ 0\ 0.1] \quad [0.3\ 1.2\ 0.7\ 0] \quad [0.8\ 0\ 0.9\ 0.4]}_{\text{Word Embeddings}}$$

$$\underset{(3\times4)}{X} = \begin{bmatrix} \text{Emb(The)} \\ \text{Emb (cat)} \\ \text{Emb (sleeps)} \end{bmatrix} = \begin{bmatrix} 1 & 0.5 & 0 & 0.1 \\ - & - & - & - \\ - & - & - & \end{bmatrix}$$

Let $Q \in R^{4\times2}$ $\quad K \in \mathbb{R}^{4\times2}$ $\quad V \in \mathbb{R}^{4\times2}$

$$XQ = \begin{bmatrix} \text{Query for "The"} \\ \text{Query for "Cat"} \\ \dots \end{bmatrix}$$

$XK = - \text{—}$ $\qquad XV = [\text{—}]$

$(3\times4)\times(4\times2) = (3\times2)$

$XQ\cdot(XK)^T \in \mathbb{R}^{3\times 2} \times \mathbb{R}^{2\times 3} \in \mathbb{R}^{3\times 3}$

or $\mathbb{R}^{n\times n}$

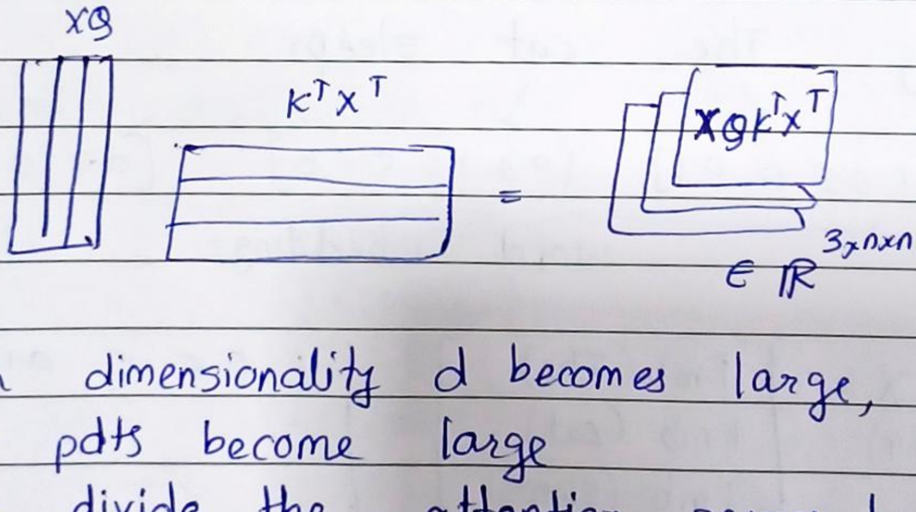
If # heads = 2

$Q_\ell, K_\ell, V_\ell \in \mathbb{R}^{4\times \frac{d}{h}}$

$d=4$ & $h=2$ $\therefore Q_\ell, K_\ell, V_\ell \in \mathbb{R}^{4\times 2}$

Each head will contribute 3×2 mtx

Thus concatenating, final o/p: 3×4 mtx.



$\in \mathbb{R}^{3\times n\times n}$

when dimensionality d becomes large, dot pdts become large

$\therefore$ We divide the attention scores by $\sqrt{d/h}$

$$\text{output}_\ell = \text{softmax}\left(\frac{XQ_\ell K_\ell^T X^T}{\sqrt{d/h}}\right) \cdot XV_\ell$$

→ Residual Connections ←

General: $x^{i-1} \longrightarrow \boxed{\text{Layer}} \longrightarrow x^{i}$

some transformation

Res con: $x^{i-1} \longrightarrow \boxed{\text{Layer}} \longrightarrow \oplus \longrightarrow x^{i}$

$$x^{i} = x^{i-1} + \text{Layer}(x^{i-1})$$

Each layer is responsible for learning a "residual" rather than entire transf^mathn.

→ Adv → faster convergence (optimization)

→ Layer Normalization ←

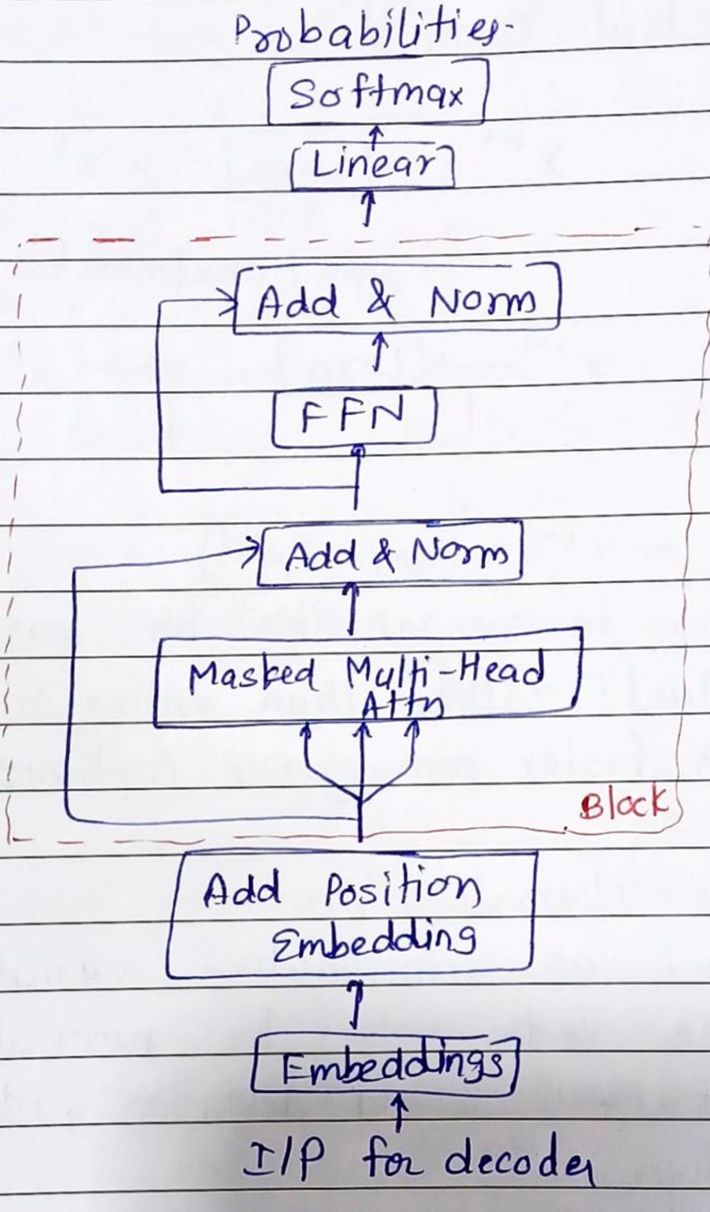
Cut down on uninformative variation in hidden vector values by normalizing to unit mean & std deviation within each layer.

$$x \in \mathbb{R}^{d} \qquad \mu = \frac{1}{d}\sum_{j=1}^{d} x_j$$

$$\sigma = \sqrt{\frac{1}{d}\sum_{j=1}^{d}(x_j-\mu)^2} \qquad \gamma \in \mathbb{R}^{d} \quad \beta \in \mathbb{R}^{d}$$

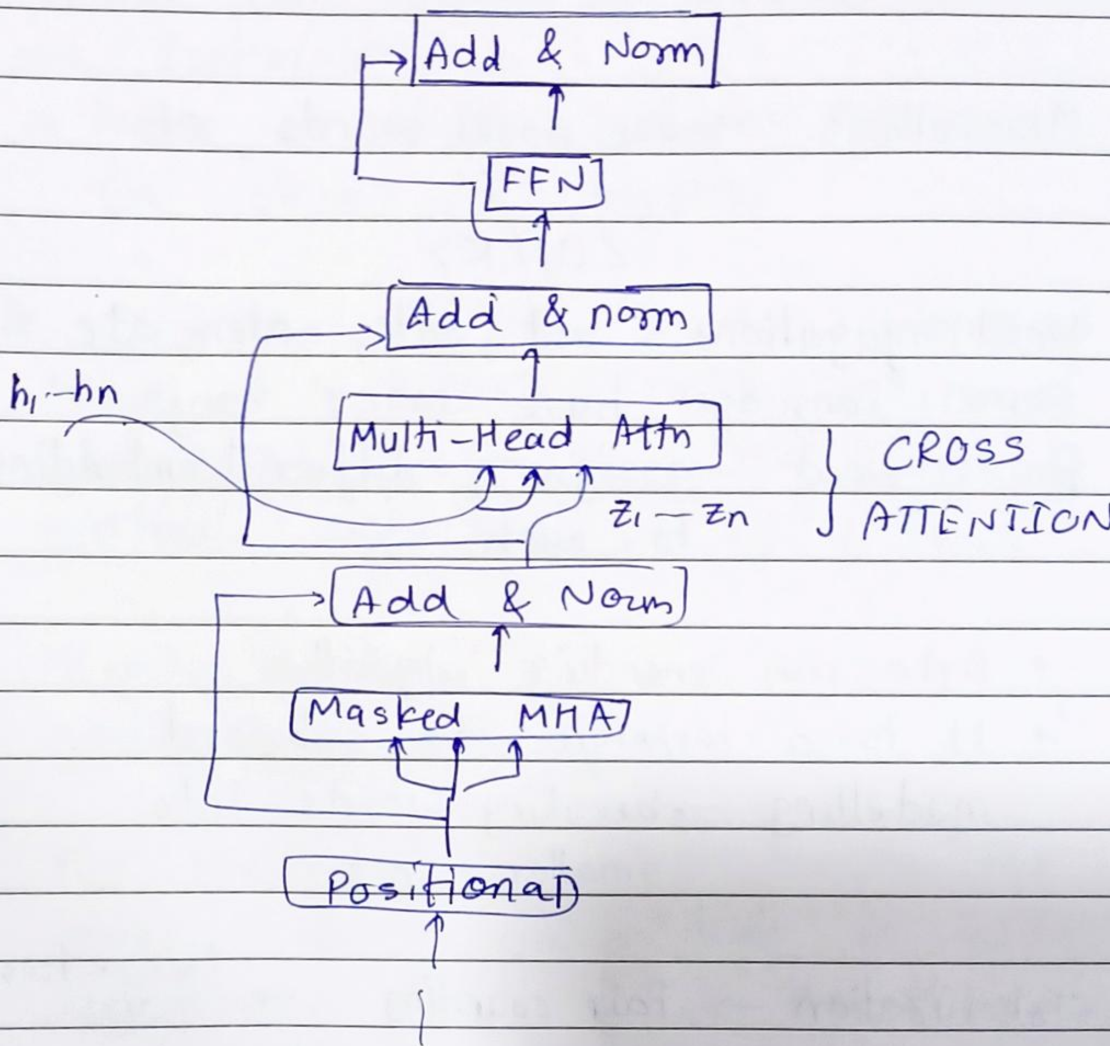
$$o/p = \frac{x-\mu}{\sqrt{\sigma}+\varepsilon} * \gamma + \beta$$

Date : ....
MON TUE WED THU FRI SAT SUN



ENC same just used Multi-Head Attn instead of Masked.

Date : ....
MON TUE WED THU FRI SAT SUN



Transf. ENC-DEC

$k_i = K \cdot h_i$, $v_i = V h_i$ | keys & values from ENCODER

$q_i = Q z_i$ | Queries from DECODER

Disadv: • Quadratic compute in SA.
For RNN → grows linearly.

• Position repr. → simple absolute indices X

Date :..................................
MON TUE WED THU FRI SAT SUN

LECTURE 9 - Pretraining

Misspellings, Newer novel words, etc
↓
<UNK>

Word conjugations - eat, eats, eaten, ate --
Some languages have many conjs
for a word - assigning different embedding
to each X

→ Byte-pair encoding algorithm
↳ It is a technique for subword
modelling. → breaking words into
smaller units.

Tokenization → Pair counting → Merge freq pairs
Repeat ↵

low lower lowest
↳ 'l', 'o', 'w' ———
↳ 'lo', 'w', 'lo', 'w', 'e', 'r', . ———
↳ 'low', 'low', 'er', 'low', 'est'.

GPT uses BPE.
Other options - WordPiece

Date :..................................
MON TUE WED THU FRI SAT SUN

Modern NLP: almost all params initialized via Pretraining.
→ helps in param. initialization for strong NLP models.

Step1 - Pretraining (on lang. modelling)
Model is trained on large corpus of text data to learn general language patterns, representations & structures.

Step 2 - Finetune on your task
Adapt to the task.

For finetuning, we don't specify the model - ki ye wale data ko jiyada importance do - but as we finetune on already set parameters, it understands himself.

→ Encoders
- Get bidirectional context
- Can't do language modelling.
- Replace some words with [MASK]
- Predict these [MASK]s.

Date :..................................
MON TUE WED THU FRI SAT SUN

#BERT - Bidirectional Encoder Representations from Transformers

- Replace I/P word with [MASK] 80% of time
- " " " with random token 10%
- Leave I/P word unchanged - 10%

↑
All this bcz there won't be [M] tokens in fine-tuning

- Also by using segment Embeddings, BERT was trained to predict whether 1 chunk follows other.
  (next sentence prediction)

* BERT-base - 110 mn params. - 10 attn heads
  BERT-large - 340 mn params. - 16 attn heads
  ↳ trained ~~on~~ with 64 TPUs for 4 days

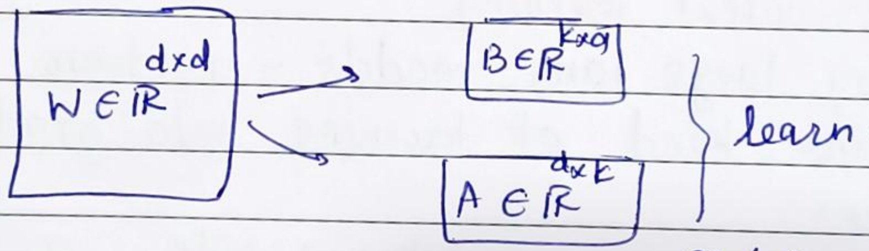
* BERT can be used for fill in the blanks type tasks, give topic labels, sentiment anal.
  But don't use it to generate a sequence of texts

* Use RoBERTa instead of BERT

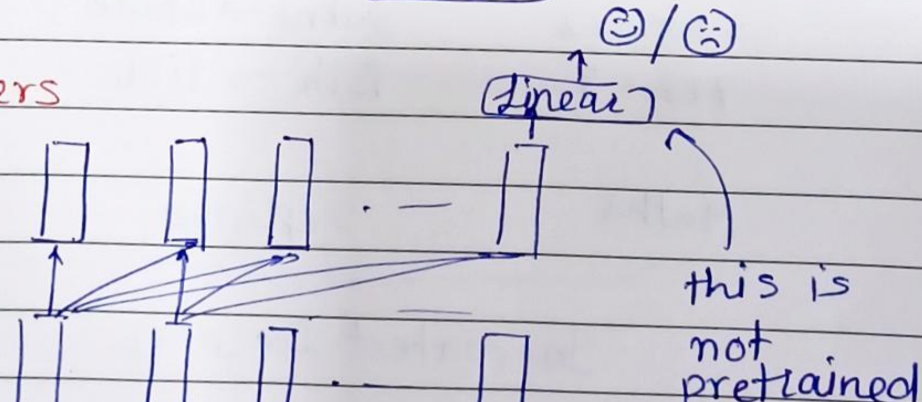
→ Full finetuning - adapt ALL params.
Lightweight " - train a few existing or new params.
- less overfitting.

PREFIX TUNING -
add a prefix of params. - only train these.

Low Rank Adaption - easier than ↑
- learns a low-rank "diff"



→ Decoders



$P_\theta(w_t \mid w_{1:t-1})$

T5 – 11 bn params

Date :..............................
MON TUE WED THU FRI SAT SUN

Generative Pretrained Transformer (GPT) 2018

- 117 mn params.
- 12 layers, 768-dim hidden states
- 3072 dim. FFN hidden layers.
- BPE with 40,000 merges.
- 7000 books

GPT-3 – 175 bn params
↓ – 300 bn tokens of text

In-context learning:

Very large lang. models – perform some kind of learning w/o gradient steps.

5+8=13 sakne → snake thanks → merci
7+2=9 fsih → fish

Maths Seplings Translatn

In-context Learning

★ Chain-of-thought prompting
- Describe steps