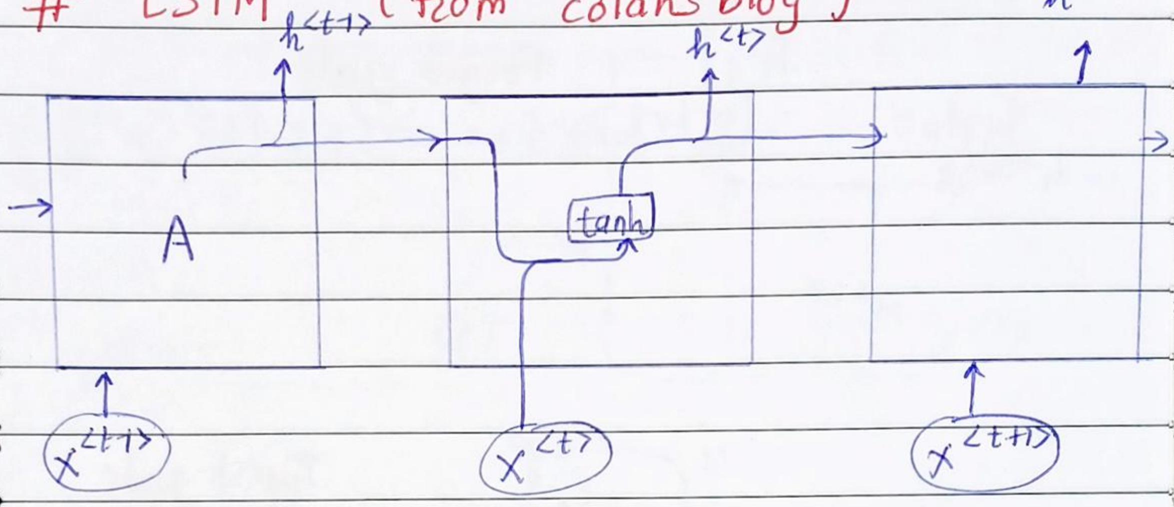
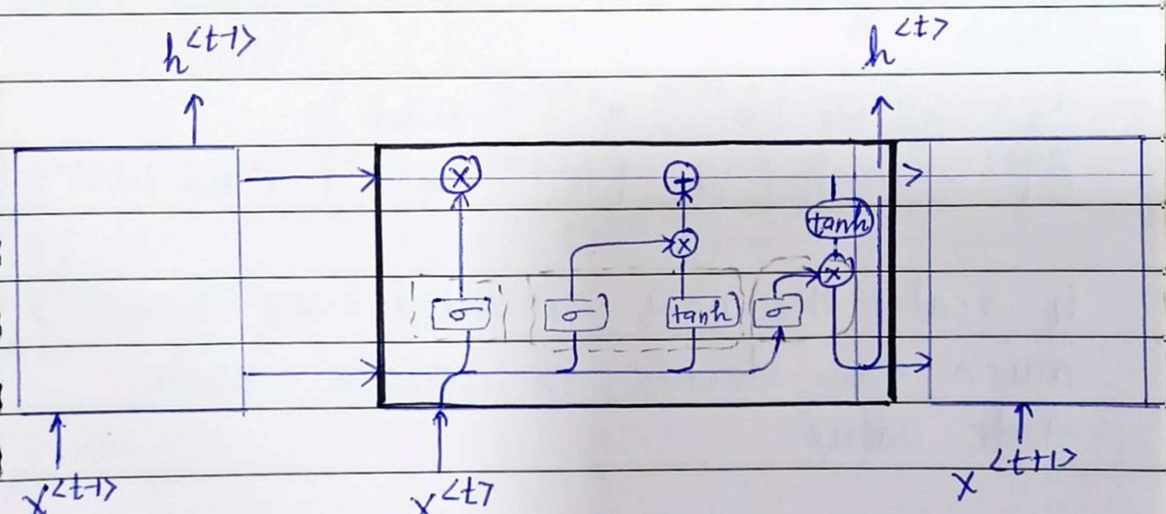
Basic RNN suffers from short-term memory problem

Date : 12/8/24
MON TUE WED THU FRI SAT SUN

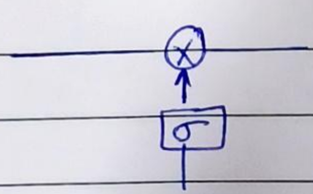
LSTM (from colah's blog)



Repeating module in a standard RNN contains single layer



each $\square$ = neural net layer

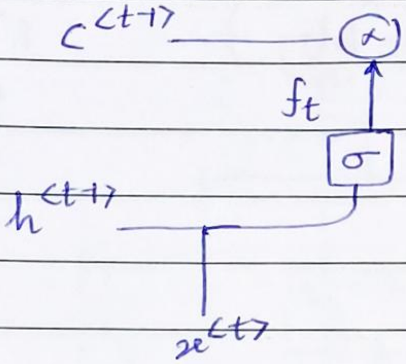


= gate

sig → how much of each component shd be let thru.

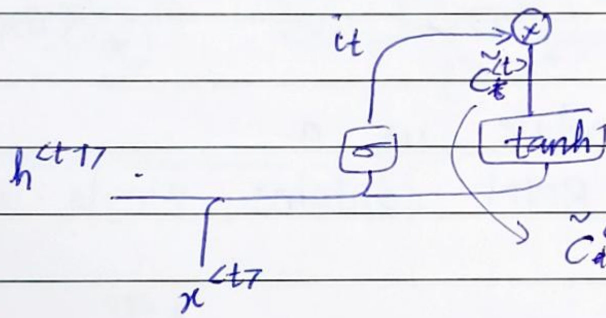
Date :

MON TUE WED THU FRI SAT SUN



Forget gate

$$f_t = \sigma(W_f \cdot [h^{<t-1>}, x^{<t>}] + b_f)$$



Input gate

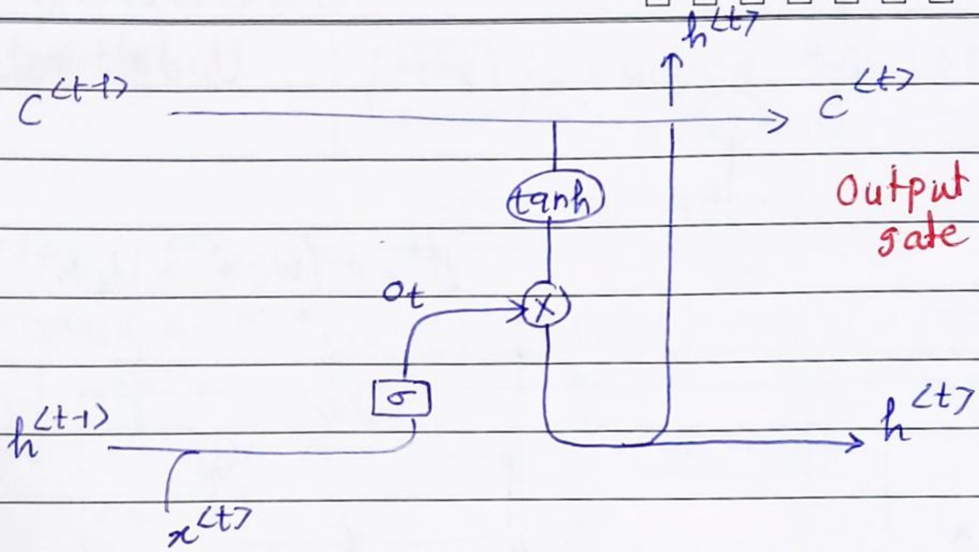
$\tilde{C}_t^{<t>}$ = vector of new candidate values

$$i_t = \sigma(W_i \cdot [h^{<t-1>}, x^{<t>}] + b_i)$$

$$\tilde{C}_t^{<t>} = \tanh(W_C \cdot [h^{<t-1>}, x^{<t>}] + b_C)$$

it scales the new candidate by how much we decide to update each state value.

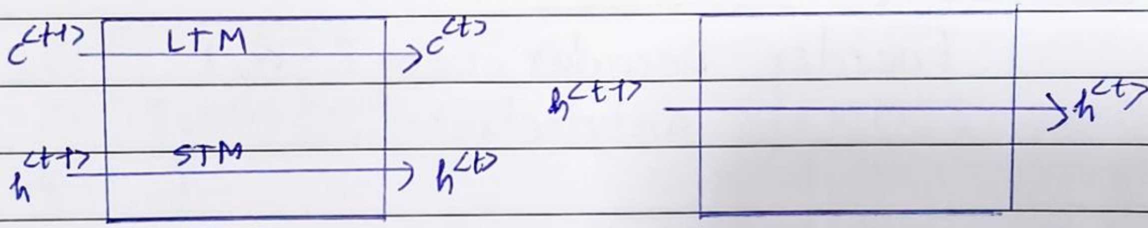
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$



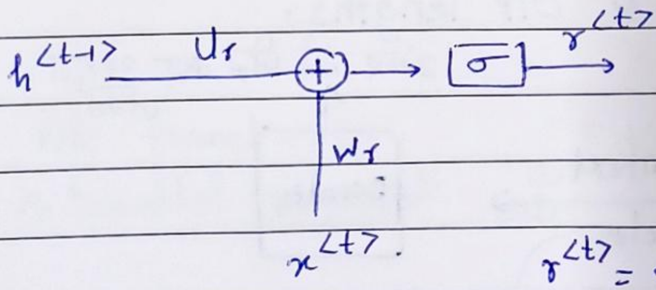
Output gate

$$o_t = \sigma\left(W_o\left[h^{<t-1>}, x^{<t>}\right] + b_o\right)$$
$$h^{<t>} = o_t * \tanh(C_t)$$

GRU



LSTM GRU

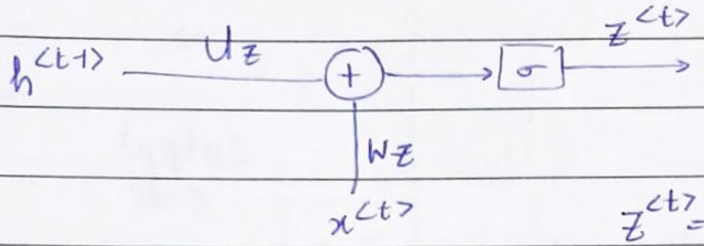


Reset gate

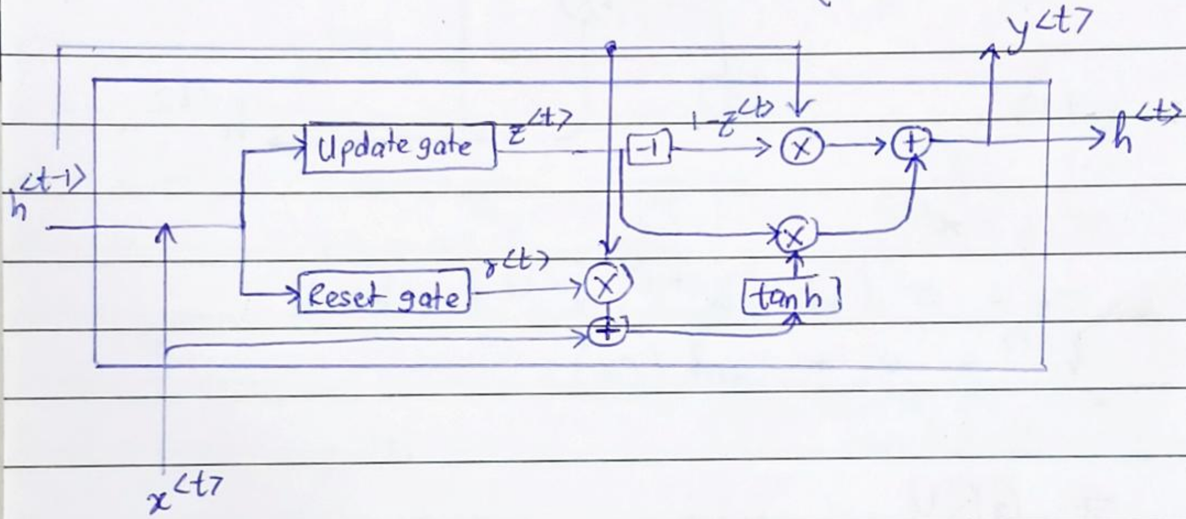
$$r^{<t>} = \sigma\left(W_r x^{<t>} + U_r h^{<t-1>}\right)$$

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

Update gate



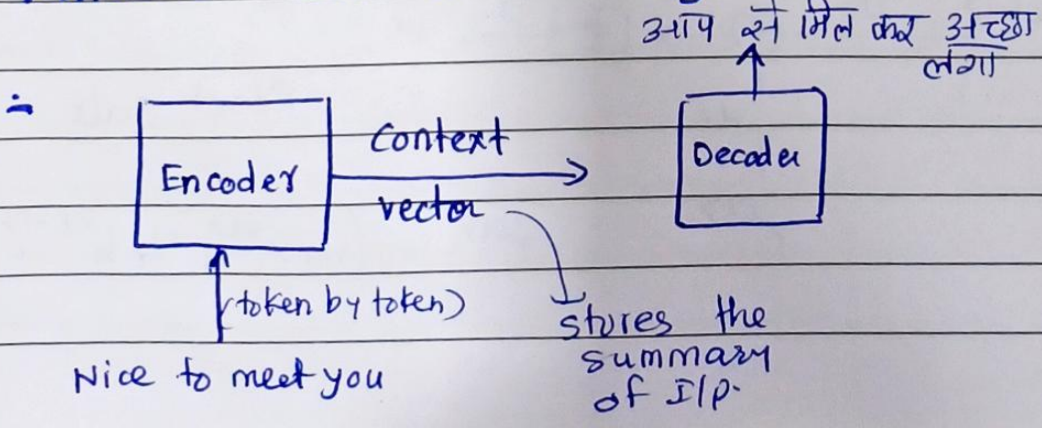
$$z^{<t>} = \sigma\left(W_z x^{<t>} + U_z h^{<t-1>}\right)$$



Encoder-Decoder (seq2seq) architecture

Just LSTMs could not do Machine translation bcz they cannot handle variable I/P & O/P lengths.

∴

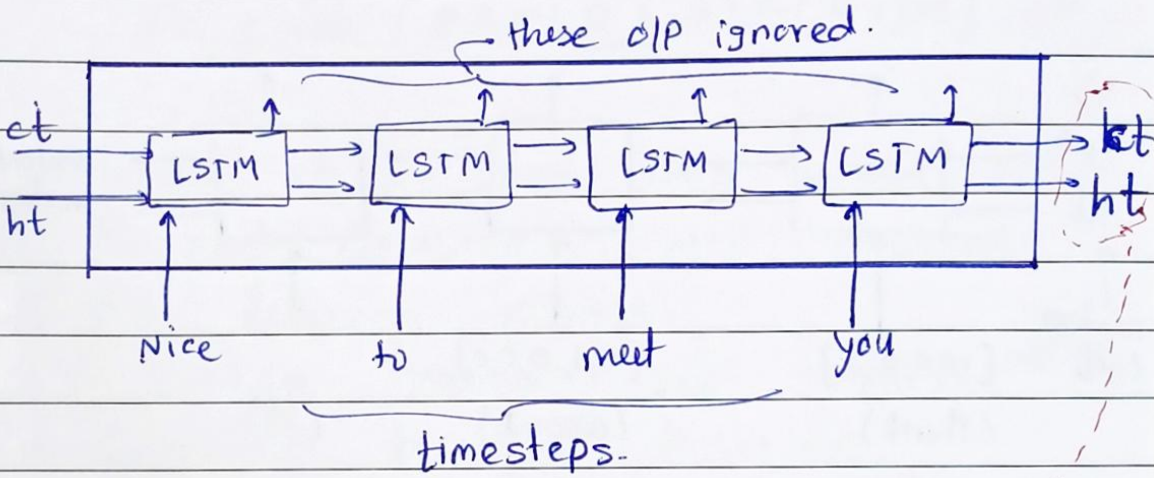


Date: 19/8/24

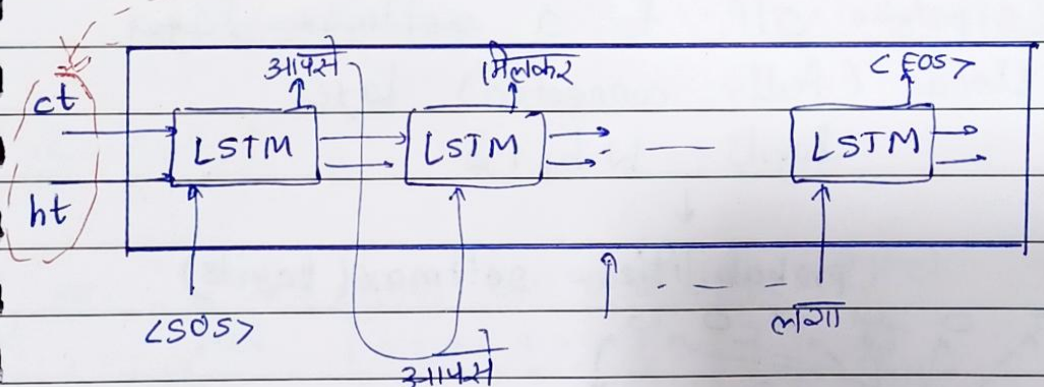
MON

Encoder:

→ unfold this LSTM over time



Decoder



M/c transl:

1) Parallel Dataset

	ENG	HIN
①	—	—
②	—	—

2) tokenization.

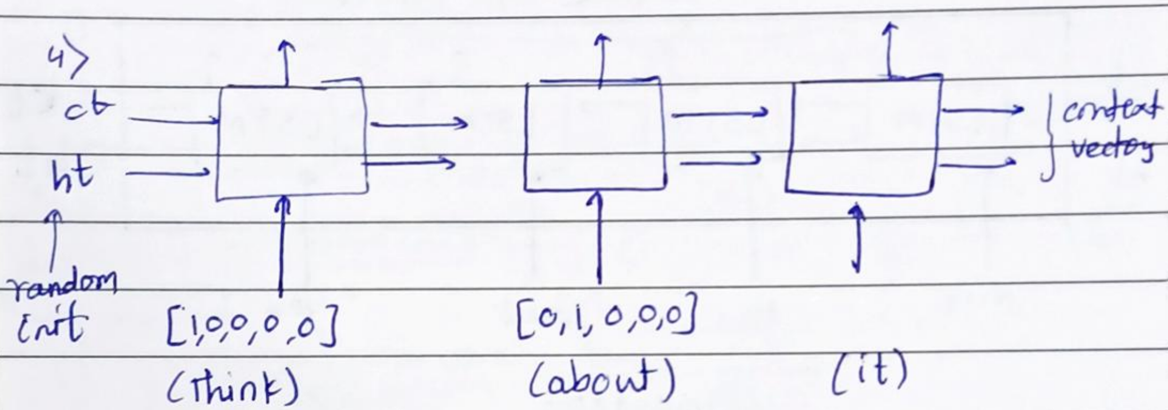
3) One-hot enc.

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

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

For Hindi vocab: add <SOS> & <EOS> tokens too.

4)

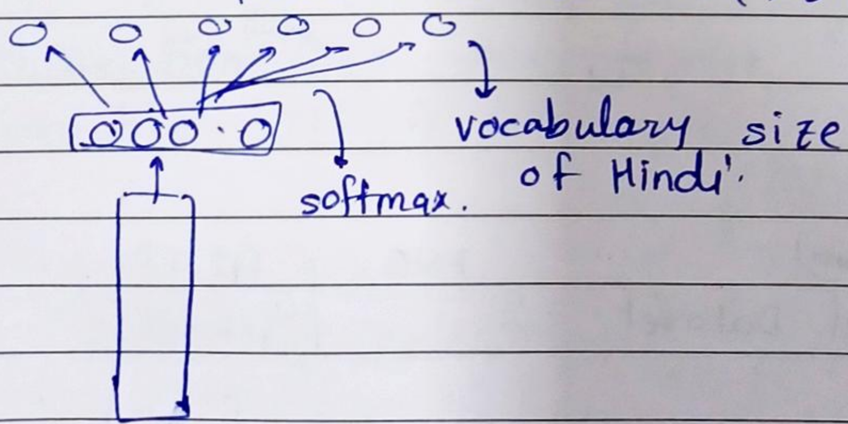


5) To output vectors (1-hot wale) from the decoder, we pass the LSTM's O/P to a ~~softmax layer~~ Linear (Fully connected) layer.

$$\text{logits} = W \cdot h_t + b$$

$\downarrow$

$$\text{probabilities} = \text{softmax}(\text{logits})$$



y_true [0,1,0,0,0,0,0] सोच

$\hat{y}$ or y_pred [0.2, 0.1, <u>0.3</u>, 0.2, 0.1, 0.1, 0.0] लो

6> Now, if we send this wrong O/P लो to next timestep, it will decrease the learning process.

∴ During Training, we supply सोच (correct) to next timestep.

↳ this is called "Teacher Forcing"

Now, suppose O/Ps from decoder were:

लो जाओ <EOS> → Forward Prop

7> **Loss**

eg. y^{true} [0,1,0,0,0,0,0] सोच

y^{pred} [0.2, 0.1, <u>0.3</u>, 0.2, ...] लो

Loss = Categorical Cross Entropy.

7 ← Hind vocab size

$$L = -\sum_{i=1}^{7} y_i^{true} \log(y_i^{pred})$$

$$L_{t=1} = -1 \times \log(0.1) = 1$$

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

simly calc. losses for all timesteps

8) $\boxed{\text{Gradient}}$

Calc. the gradient of loss wrt each of the trainable parameters (LSTM, softmax, linear layer)

$$\frac{\partial loss}{\partial logits_t} = \hat{y}_t - y_t$$

For each hidden state h_t,

$$\frac{\partial loss_t}{\partial h_t} = \frac{\partial loss_t}{\partial logits_t} \cdot \frac{\partial logits_t}{\partial h_t}$$

$$\frac{\partial loss}{\partial W_h} = \sum_{t=1}^{T} \frac{\partial loss_t}{\partial h_t} \cdot \frac{\partial h_t}{\partial W_h}$$

Backprop thru encoder

$$\frac{\partial loss}{\partial h_{enc}} = \sum_{t=1}^{T_{dec}} \frac{\partial Loss_t}{\partial h_{dec,t}} \cdot \frac{\partial h_{dec,t}}{\partial h_{enc}}$$

9) $\boxed{\text{Update params}}$

$$W_{new} = W_{old} - \alpha \frac{\partial loss}{\partial W}$$

Date :..................................

MON TUE WED THU FRI SAT SUN

init wts → Forward prop → O/P + true → Losses ↓ gradient → wt update → learning rate → new wts (new training ↻)

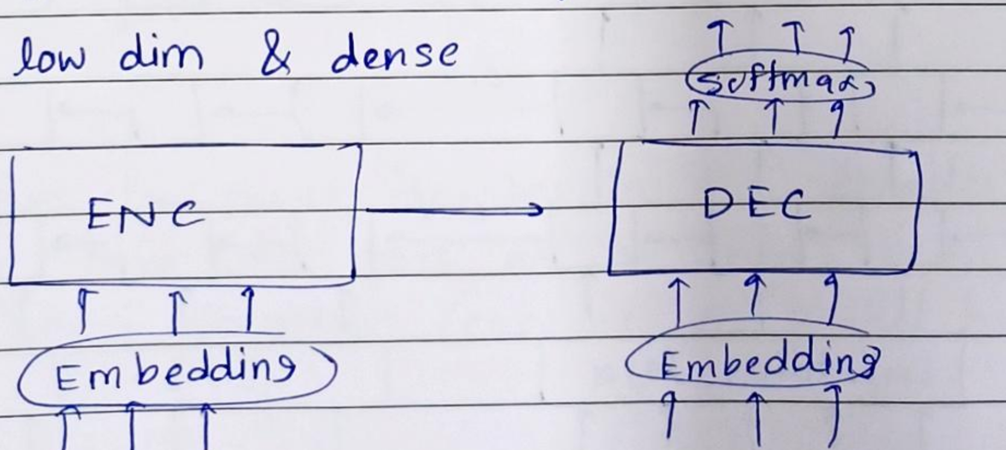
10) Prediction

When the wts are freezed, we don't do teacher forcing & supply decoder LSTM o/p as I/P for next timestep

Improvement 1

Embedding Layer → for both ENC & DEC
↓
it contains a summary for each word

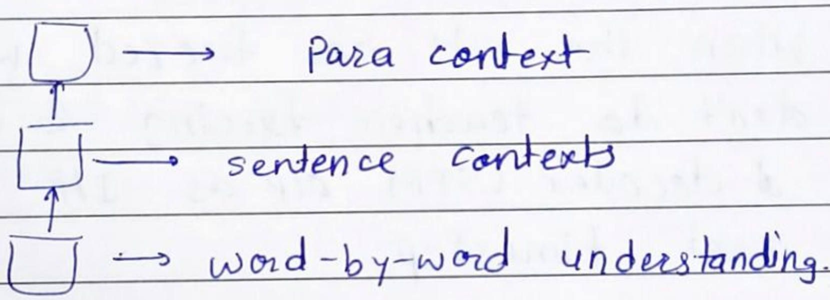
- low dim & dense



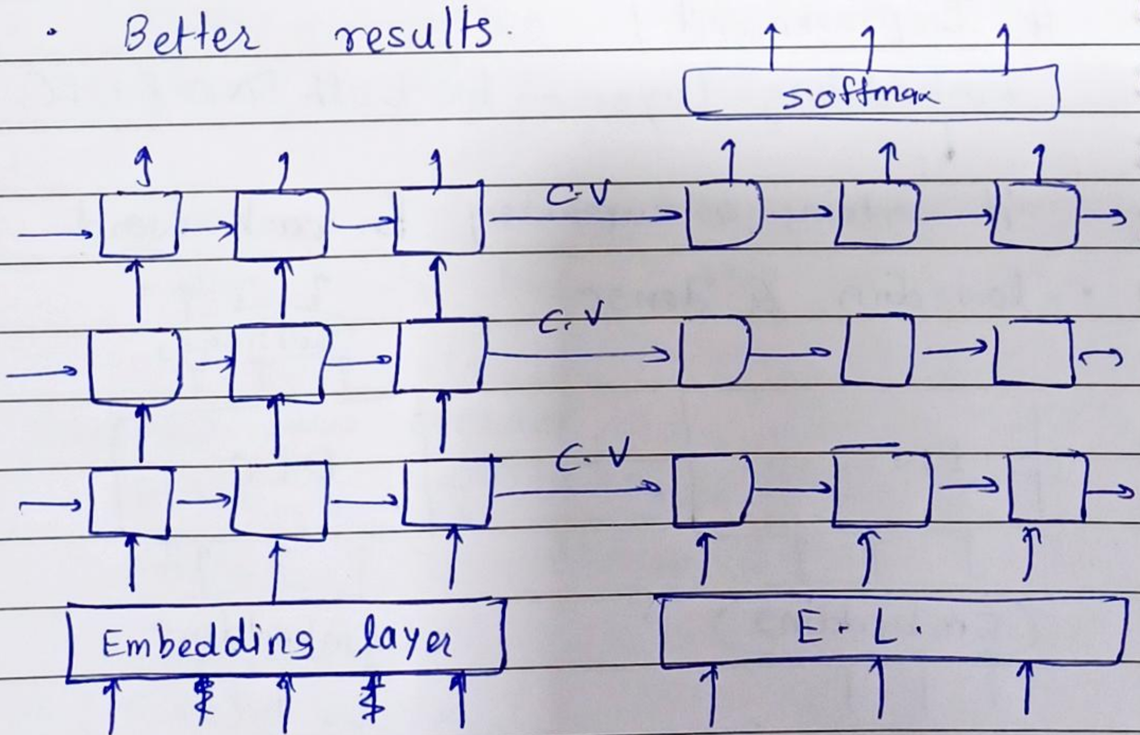
Date :..............................
MON TUE WED THU FRI SAT SUN

Improvement 2 — Deep LSTMs

- Can handle Long Term Dependencies well.
- More context vectors to store summaries.
- Can understand hierarchical levels.



- Better results.



3 layer LSTM

Improvement 3 – Reversing the I/P.

- Works for languages where the starting words hold much of context

Think about it
it about think

} <sos> सोच लो

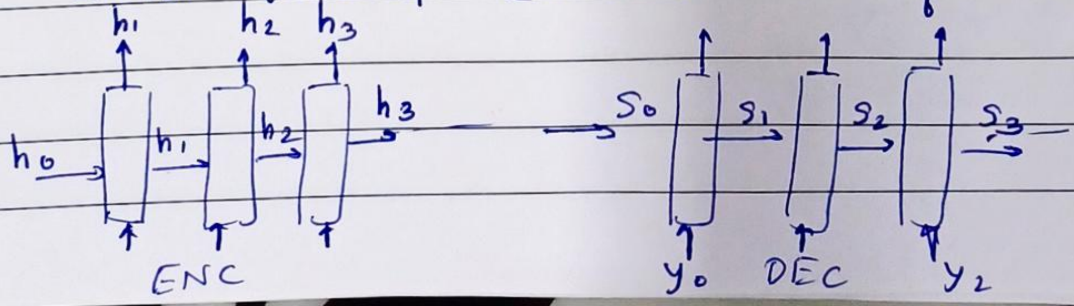
ATTENTION

Problems with Encoder-Decoder:

1) Cannot summarize sentences with ~> 25 words.
2) Turn off the lights
   ↳ <SOS> लाइट बंद करो <EOS>

We do not need the whole I/P sentence to generate only लाइट

Thus for every timestep in the decoder, we have to pass on an info as to 'konsi timesteps (encoder ki) useful hai'.



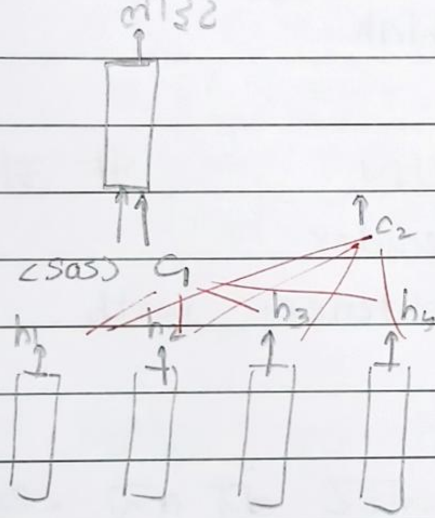
Vanilla enc-dec

↓

for every timestep of DEC → 2 I/Ps
[y_{i-1} & s_{i-1}]

Attention → 3 I/Ps [y_{i-1}, s_{i-1}, c_i]

attention I/P.



For each timestep in DEC, a wt(α_i) is assigned for each of the ENC timestep.

$c_1 = \alpha_1 h_1 + \alpha_2 h_2 + \alpha_3 h_3 + \alpha_4 h_4$

scalars vectors (4 dim here)

$$\boxed{c_i = \Sigma \alpha_{ij} h_j}$$

i , j

timestep of DEC

" " ENC

Date :......................................

MON TUE WED THU FRI SAT SUN

α_{21} depends on h_1 as well as $\boxed{s_1}$

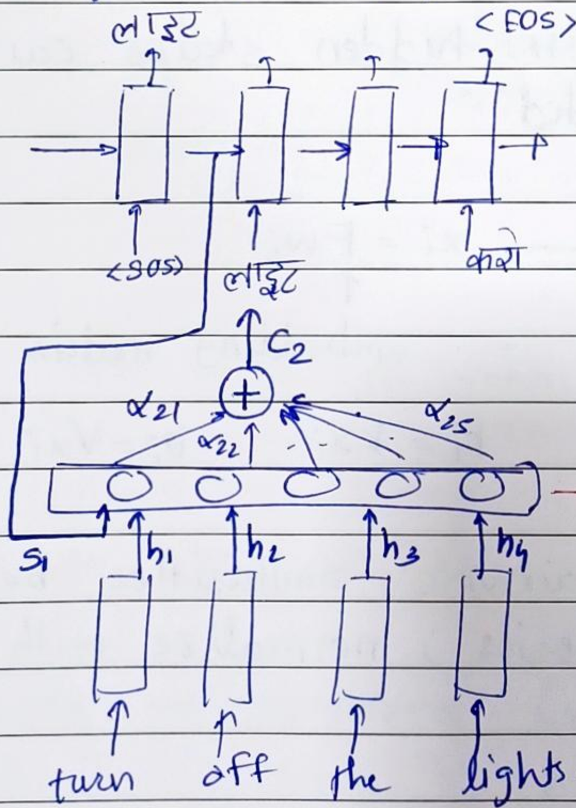
$\shortparallel$

prev hidden state of DEC.

this is bcz:
already kitna transl^n
ho chuka hai is IMP.

$$\therefore \boxed{\alpha_{ij} = f(h_j, s_{i-1})}$$

instead of finding any function here, we use ANN



→ Time distributed dense layer

Backprop
~~ke upar bhi~~
me ANN wts
bhi adjust hote
hai

Date : 19/8/24
MON TUE WED THU FRI SAT SUN

→ Lecture 8 – Self Attn & Transf ←

* Lack of parallelizability: in recurrent models:

- For & back prop have O(seq. length) unparallelizable ops.
  - GPU can perform independent comps.
  - Future RNN hidden states cannot be computed

For each w_i — $x_i = E w_i$
↑
embedding matrix

1) $q_i = Q x_i$ $\quad k_i = K x_i$ $\quad v_i = V x_i$

2) Compute pairwise similarities betⁿ keys & queries ; normalize with softmax.

$$e_{ij} = q_i^T k_j$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{j'} \exp(e_{ij'})}$$

3) Compute o/p for each word as weighted sum of values.

$$o_i = \sum_j \alpha_{ij} v_j$$

Transformers

- Revolution in NLP
- Democratising AI
- Multimodal capabilities
- Acceleration of Gen AI (GANs)
- Unification of DL.

Self - Attention

words → numbers (vectorization)

1) One Hot Encoding

2) Bag of words

	mat	rat	cat
S1	2	0	1
S2	0	2	1

Date :
MON TUE WED THU FRI SAT SUN

3) word embeddings - semantic meaning

this does not represent the "meaning" but "avg meaning" of the word

- eg. [X , Y]
repr: taste tech.

If there are 9000 ~~words~~ sentences relating to Apple as phone company the overall word emb will be tilted towards Y. And these embeddings are static

∴ We need Contextual embeddings (smart)
This is Self-Attention.

eg. Money bank grows
River bank flows

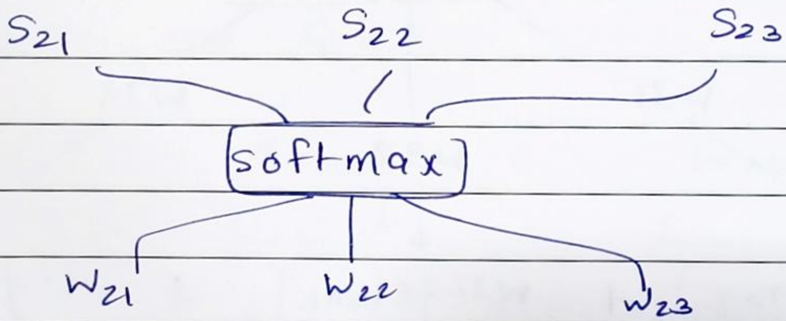
money = 0.7 money + 0.2 bank + 0.1 grows
bank = 0.25 money + 0.7 bank + 0.05 grows

these are individual old embedding (n dim)

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

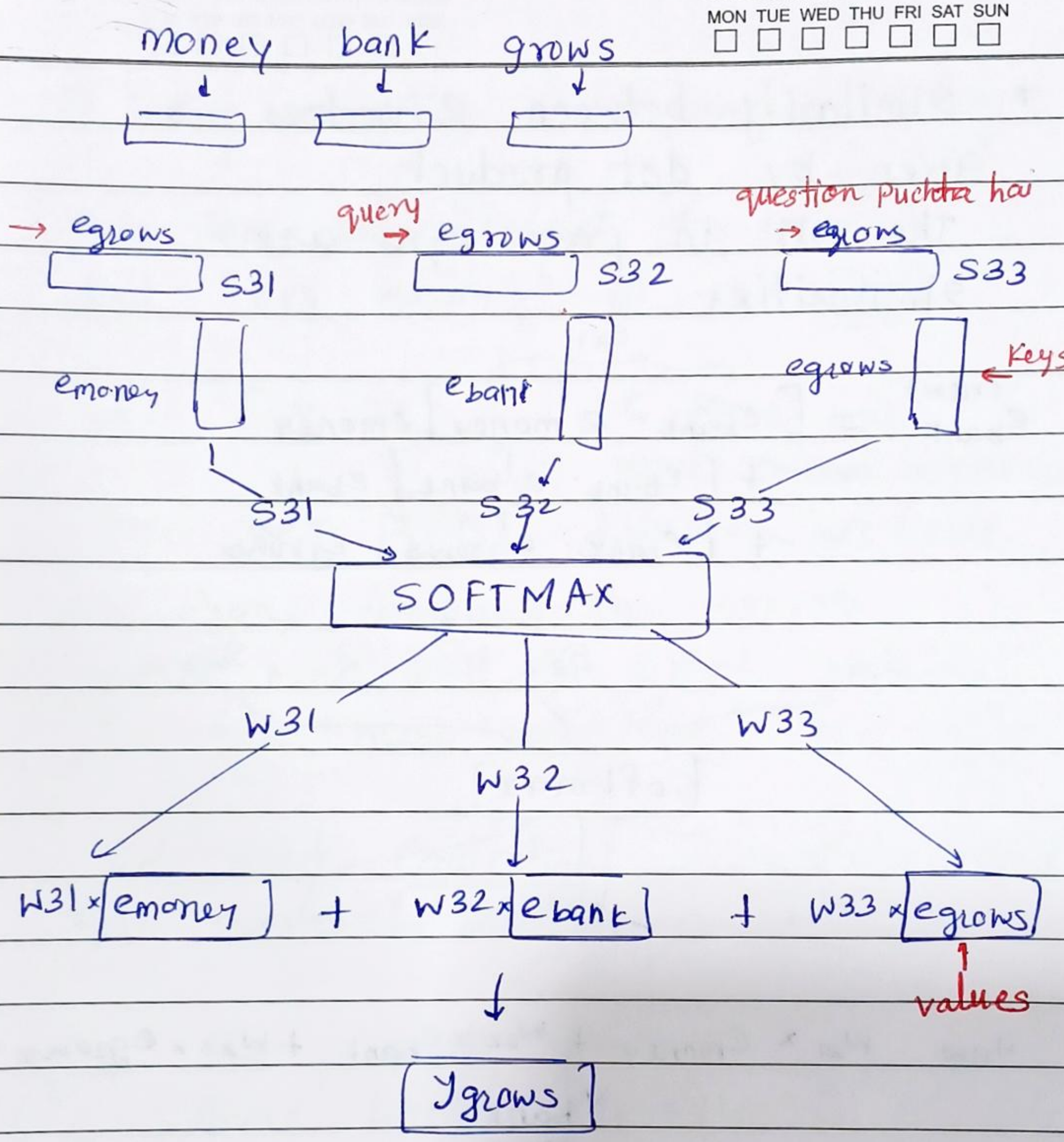
* similarity between 2 vectors is given by dot product
∴ The WTS in prev eqns are similarities.

$$
\begin{aligned}
e_{bank}^{(new)} &= \overbrace{[e_{bank} \cdot e_{money}^T]}^{s_{21}} e_{money} \\
s_{22} \quad &+ [e_{bank} \cdot e_{bank}^T] e_{bank} \\
s_{23} \quad &+ [e_{bank} \cdot e_{grows}^T] e_{grows}
\end{aligned}
$$



Now $W_{21} \times e_{money} + W_{22} \times e_{bank} + W_{23} \times e_{grows}$
$= Y_{bank}$.
contextual embedding.

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



→ All are parallel operations. (for GPU)

→ No Learning Parameters involved

· Model ko data se pharak nahi padta. Just depends on individual sentence.

· eg. 'piece of cake' → केक का टुकडा

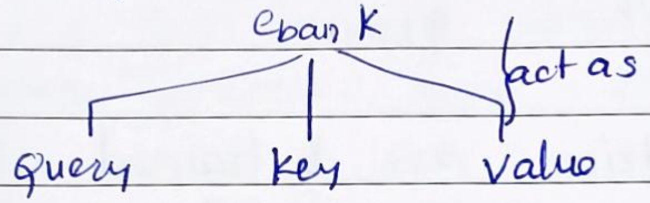
ypred

बहुत आसान काम

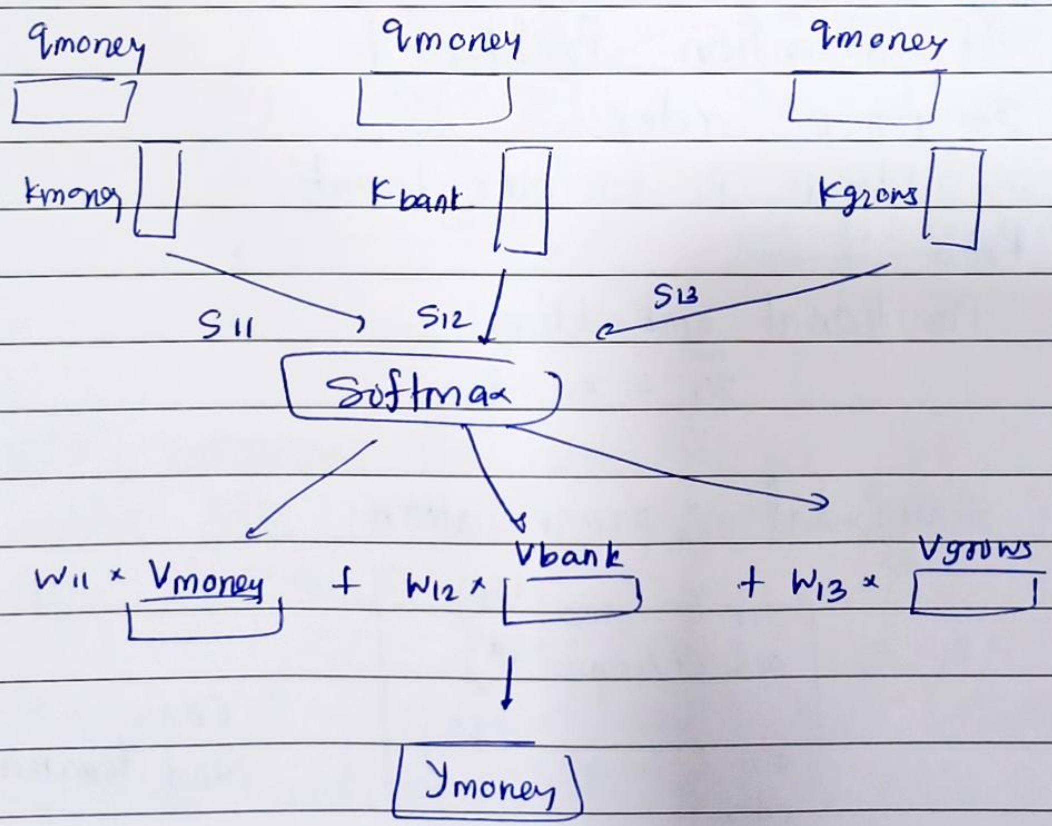
ytrue

Thus we need Task-specific contextual emb. and not general con. emb.

Query, key, Value



Now, we want to make out 3 diff vectors from single e_{bank} to function as Q, k, V



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