RNN - Recurrent Neural Network

RNN is a type of Neural Network that works on Sequential data.

RNN captures the Semantic meaning of the data.

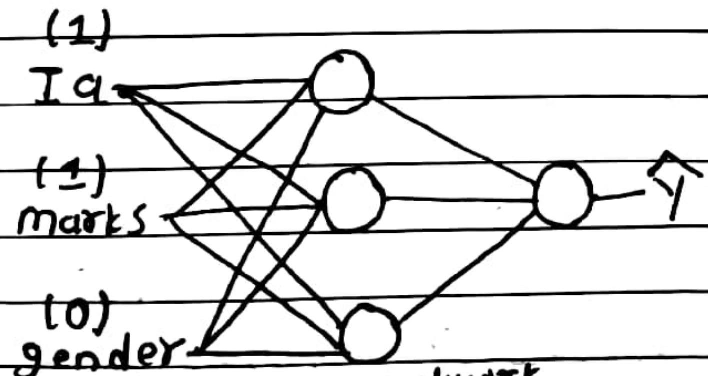
eg- Hi my name is miraj It can't be miraj is name my Hi.

Why Need of RNN?

- ANN ma hamle data ko semantic meaning capture garna Sakedam. because yesma sabai input ekai choti Neural network ma Janxa ra prediction hunxa

eg-

Iq	marks	gender	Place
1	1	0	0
2	8	0	1



yo neural network ANN ho yesma (1, 1, 0) ekai choti layer 1 ma Janxa ani further process vayera calculate hunxa. So, kun paila gayo, kun pari kei tha hunna no capture of semantic meaning. tara RNN ma yesto hunna kina hunna hami yesko feed forwarding ma bhuixau.

RNN Feed forwarding

Review	Sentiment
movie was good	1
movie was bad	0
movie was not good	0

Machine don't under English words let's
we simple vectorizing technique one hot endoding

Vocabulary = movie was good bad not = 5 words
Now, let's give each word 5 number representation

Vectors representation
movie = [1, 0, 0, 0, 0]
was = [0, 1, 0, 0, 0]
good = [0, 0, 1, 0, 0]
bad = [0, 0, 0, 1, 0]
not = [0, 0, 0, 0, 1]

Now, let's write data set again

Review	Sentiment
[1,0,0,0,0] [0,1,0,0,0] [0,0,1,0,0]	1
[1,0,0,0,0] [0,1,0,0,0] [0,0,0,1,0]	0
[1,0,0,0,0] [0,1,0,0,0] [0,0,0,0,1] [0,0,1,0,0]	0

RNN ma hami yo format ma input dinxam
i.e (timesteps, Input word size)
for example:
[1,0,0,0,0] [0,1,0,0,0] [0,0,1,0,0]
shape = (3, 5)

time step (no of words)

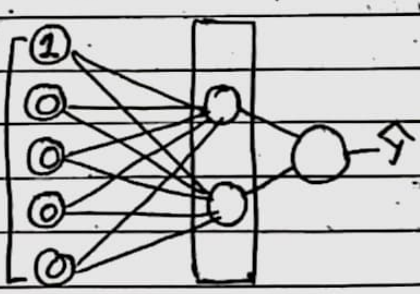
euta word lai kati num ley represent garekoxa / input size for NN

t=1 t=2 t=3 [∵ t = timestep]
[1,0,0,0,0] [0,1,0,0,0] [0,0,1,0,0]

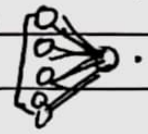
[1, 0, 0, 0, 0] →

5

Input size for NN



euta word lai 5 ota number ley represent garera
Input size for NN 5 vayo i.e



NOW, let's see the shape of all features
[1,0,0,0,0] [0,1,0,0,0] [0,0,1,0,0] → (3,5)
[1,0,0,0,0] [0,1,0,0,0] [0,0,0,1,0] → (3,5)
[1,0,0,0,0] [0,1,0,0,0] [0,0,0,0,1] → (4,5)

In keras
simple RNN ma yo format ma value dinxam
Simple RNN → batchsize, timesteps, input feature
(input size for NN or euta word lai kati num ko representation ta)

for our dataset the value will be
Batchsize, timesteps, input feature
(1 , 4, 5)
say

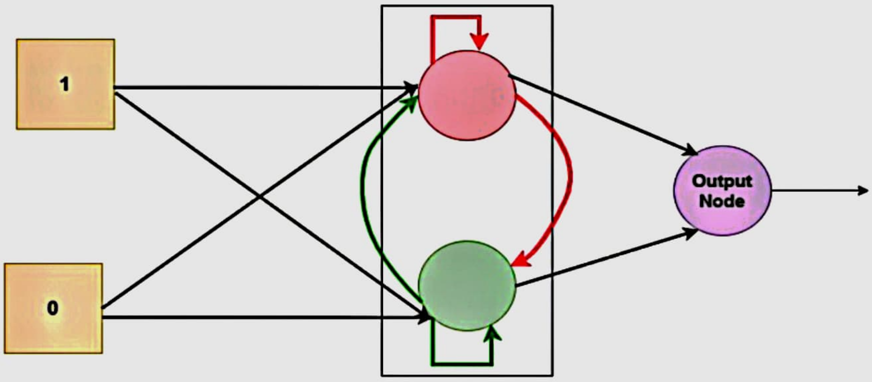
longest Sentence have 4 words

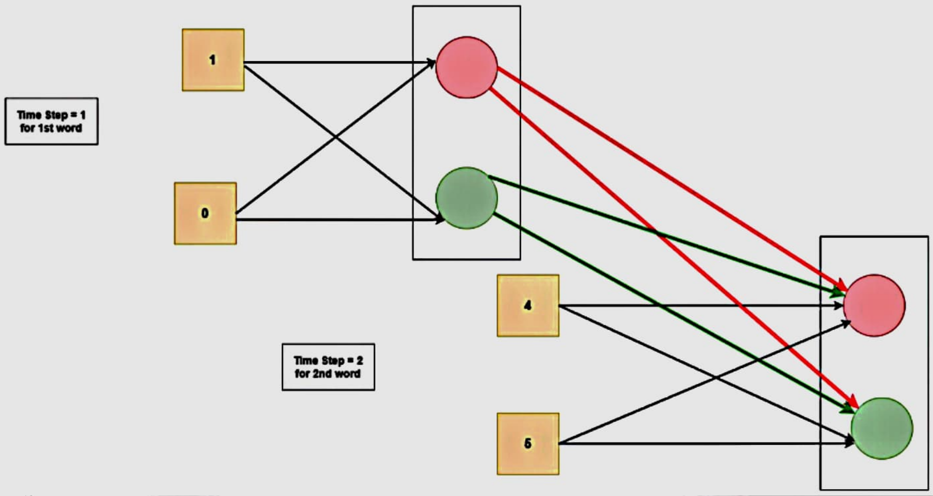
euta word lai 5 ota num ko representation ta

How does RNN feed forwarding works?

⇒ Suppose we have this dataset

Ram	good	1	(suppose in vector form
[1,0]	[4,5]	1	Ram = [1,0] good = [4,5])





At time Step = 1

first word Ram i.e [1,0] Neural Network Ma gayo, weight Sanga multiply vayo ra hidden layer ko nodes ley some output calculate gayo. Aba: arko word Janxa.

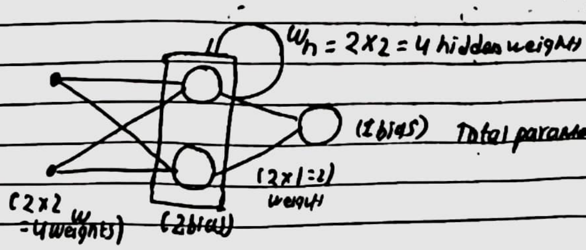
At time step = 2

good [4,5] Neural Network ma gayo + previous time step ko output pani yo time step as a input gayo ra aba yes bata ne output niskinxa Jun chai further arko output layer ma Janxa ra prediction hunxa.

(Simple: At timestep = 1

1 Euta word Janxa, some output niskinxa. At timestep = 2 arko word + timestep 1 ko output as a input NN ma Janxa ra prediction hunxa)

Now, let's see parameters



Total parameter = 4 + 2 + 2 + 1 + 4

= 13 parameters

let's See Detailly feed forwarding:

we have previously this data set

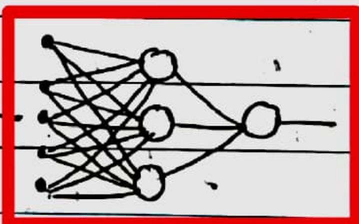
Review	Sentiment
[1,0,0,0,0] [0,1,0,0,0] [0,0,1,0,0]	1
[1,0,0,0,0] [0,1,0,0,0] [0,0,0,1,0]	0
[1,0,0,0,0] [0,1,0,0,0] [0,0,0,0,1] [0,0,1,0,0]	0

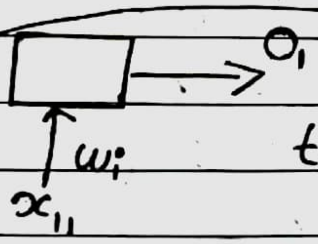
let's Say

x_{11}	x_{12}	x_{13}		1
x_{21}	x_{22}	x_{23}		0
x_{31}	x_{32}	x_{33}	x_{34}	0

Now,

let's See It





$\tanh(x_{11} \cdot w_i + b) \Rightarrow O_1$

$\boxed{t=1}$

x_{11} = first word Matrix

w_i = weights matrix

O_1 = output from lay one Matrix

b = bias matrix

Now, let's see

$x_{11} = [1,0,0,0,0] \Rightarrow (1,5)$

$$w_i = (5,3) \Rightarrow \begin{bmatrix} 0 & 0.1 & 0.2 \\ 0.3 & 0.4 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.5 \end{bmatrix}$$ say

$b = (1;3) \Rightarrow [0.1, 0.2, 0.3]$ say

NOW,

Doing $x_{11} \cdot wi + b$ we get

$$= \left([1, 0, 0, 0, 0] \begin{bmatrix} 0 & 0.1 & 0.2 \\ 0.3 & 0.4 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.5 \end{bmatrix} \right) + [0.1, 0.2, 0.3]$$

$$= (1, \boxed{5}) \times (\boxed{5}, 3) + (1,3)$$

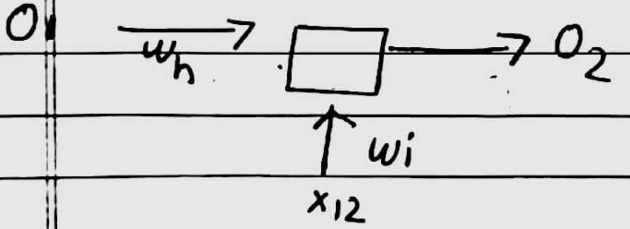
$$= (1,3) + (1,3)$$

$$= (1,3)$$

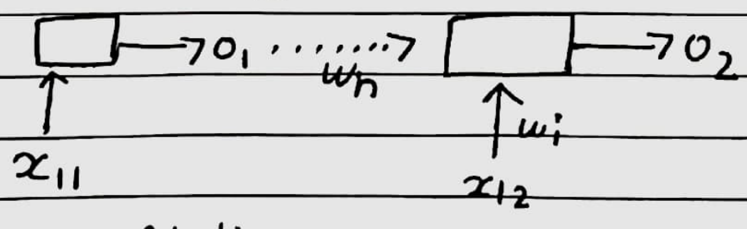
$\therefore O_1 = [0.1, 0.2, 0.8]$ let's say $(1,3)$

NOW,

$\boxed{t = 2}$



$\tanh(x_{12} \cdot wi + O_1 w_h + b)$



NOW, At time step 2 x_{12} next word goes to same hidden layer with same weights but this time previous step output also comes. so, each node of hidden layer calculate $\tanh(x_{12} \cdot wi + O_1 \cdot wh + b)$ and gives O_2 output.

NOW,

$x_{12} = [0, 1, 0, 0, 0] \Rightarrow$ second word $(1,5)$

$wi = (5,3) = \begin{bmatrix} 0 & 0.1 & 0.2 \\ 0.3 & 0.4 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.5 \end{bmatrix} (5,3)$ $\quad b = [0.1, 0.2, 0.3] = (1,3)$

3 uta Nodexa
3otai lai output Tanxa
So, weight (3,3)

$$W_h = (3,3) \Rightarrow \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.1 & 0.2 & 0.3 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} \text{(say)}$$

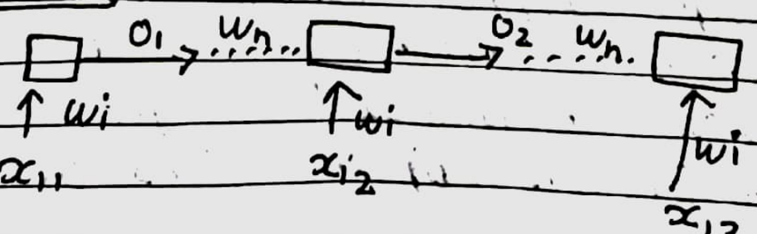
now, doing $(x_{12} \cdot w_i + O_1 \cdot w_h + b)$

$$= \left(\underbrace{[0,1,0,0,0]}_{x_{12}} \underbrace{\begin{bmatrix} 0 & 0.1 & 0.2 \\ 0.3 & 0.4 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.5 \end{bmatrix}}_{w_i} \right) + \left(\underbrace{[0.1, 0.2, 0.8]}_{O_1} \times \underbrace{\begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.1 & 0.2 & 0.3 \\ 0.1 & 0.2 & 0.3 \end{bmatrix}}_{w_h} \right) + \underbrace{[0.1, 0.2, 0.3]}_{b}$$

$$= (1,\boxed{5}) \times (\boxed{5},3) + (1,\boxed{3}) \times (\boxed{3},3) + (1,3)$$
$$= (1,3) + (1,3) + (1,3)$$
$$= (1,3)$$
$$= O_2$$

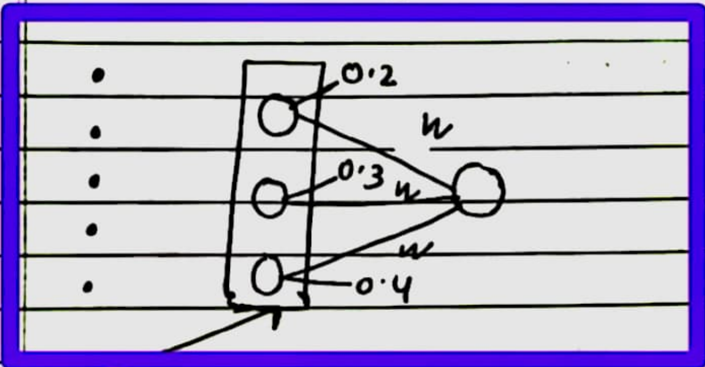
Now,

$\boxed{t=3}$

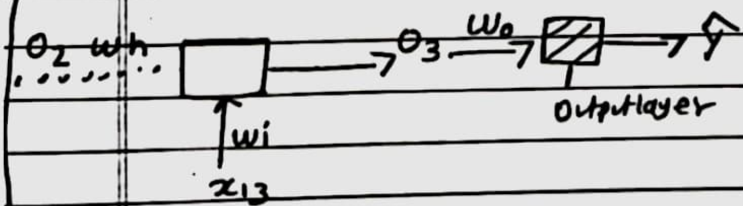


Here similar process happen x_{13} goes to NN with output of timestep2 and each node calculate

$\tanh(x_{13} \cdot w_i + O_2 W_h + b)$ and gives
$\Rightarrow O_2$ of $(1,3)$ shape
Now, How O_2 looks like $[0.2, 0.3, 0.4]$ say



Now, we again have to do dot product to calculate the output of output layer



Now,
$f(O_3 \cdot w_o + b_{of\,output})$ will get calculated into output layer
$O_3 = [0.2, 0.3, 0.4]$
$\Rightarrow w_o =$ weights matrix **going** to output layer $(3,1)$
let $w_o = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix}$

$b =$ bias of output $\Rightarrow [1,1] \Rightarrow [2]$
Now, calculating $O_3 \cdot w_o + b_{of\,output}$

$$\Rightarrow [0.2, 0.3, 0.4] \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix} + [2]$$

$$= (1,3) \times (3 \times 1) + (1,1)$$
$$= (1,1) + (1,1)$$
$$= (1,1)$$
i.e $\hat{y}$ prediction.

So, this is how forward propagation in RNN works.