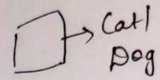
S. No.	Date	Title	Page No.	Teacher's Sign / Remarks

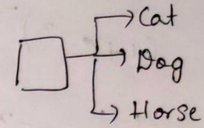
12/10

Binary class

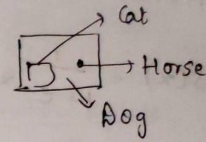


2variables

Multi-class



Multi-label



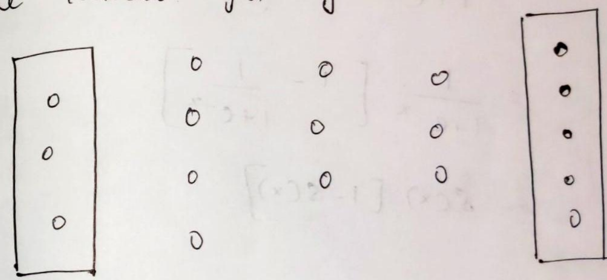
Single image, identifying different labels

Multi-task learning

Multi-label classification

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

Instead of training seperate neural networks for cat, Dog and Horse, we use a single Neural Network to learn all the labels for generalised representation



Input Image

$i = 1-m$ - no. of Images

$j = 1-n$ - no. of labels

$$Loss = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{n} L(\hat{y}_i - y_i)$$

$$= -y_i \log \hat{y}_i - (1-y_i) \log(1-\hat{y}_i)$$

Differentiation of sigmoid Activation function

$$S(x) = \frac{1}{1+e^{-x}} = (1+e^{-x})^{-1}$$

$$= -1\,(1+e^{-x})^{-1-1} \cdot e^{-x} \cdot -1$$

$$= \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1-1+e^{-x}}{(1+e^{-x})^2}$$

$$= \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2}$$

$$= \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2}$$

$$= \frac{1}{1+e^{-x}}\left[1 - \frac{1}{1+e^{-x}}\right]$$

$$= S(x)\,[1 - S(x)]$$

Classify the ~~network~~ factor i & ii using Hebbian Learning

13/10/24 RNN - Recurrent Neural Network

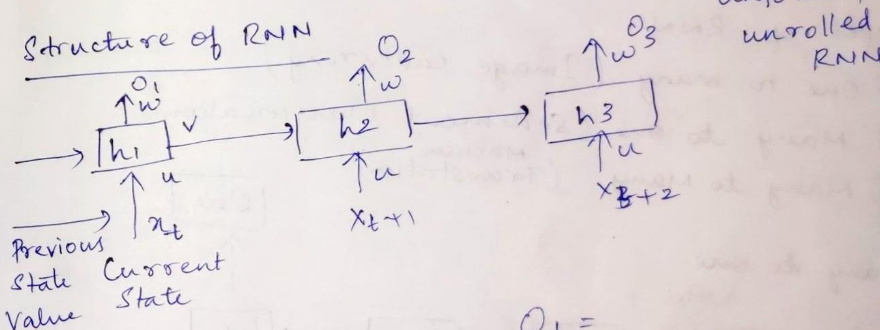
ANN vs RNN
↳ RNN
Structure of RNN
unfolded / unrolled RNN
Rolled RNN
(BPTT)
Applications of RNN
Types of RNN
Drawbacks of RNN

RNN
Autocomplete
↳ Ordering of words is important
↳ Different varying length

Ex: Stock Market Prediction
Predict tomorrow stock value →
Sequential data i/p from 1 to 12/10

RNN
Time series Application
Sequence of Data

Structure of RNN



$O_1 =$

h - hidden state
u - weight of x_t / weight to the i/p value
v - weight of i/p to h2

Notation of RNN

$ux_t + V h_{t-1} + \text{bias}$

↗ weights

Ordering is important

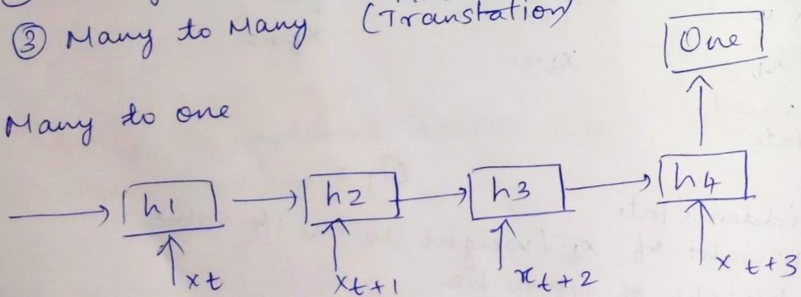
Applications of RNN

i) Speech Recognition
ii) Music translation
iii) Image captioning
iv) Sentiment classification
v) Machine translation
vi) DNA sequencing
vii) Stock market prediction

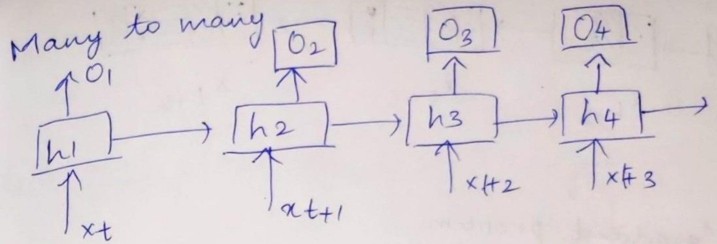
Types of RNN

① One to many (Image capturing)
② Many to one (Sentiment classification)
③ Many to Many (Machine Translation)

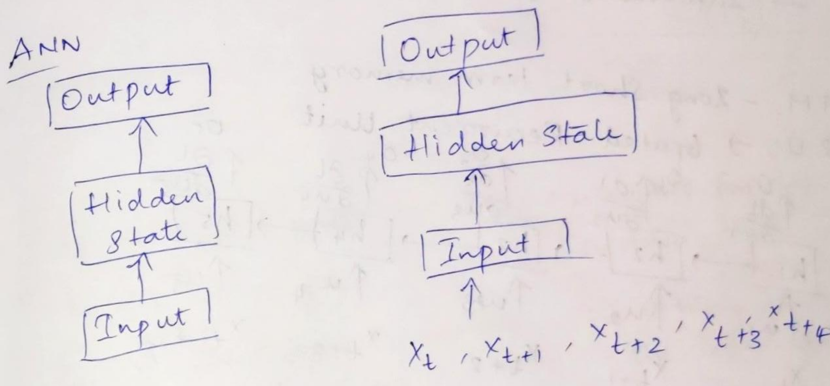
Many to one



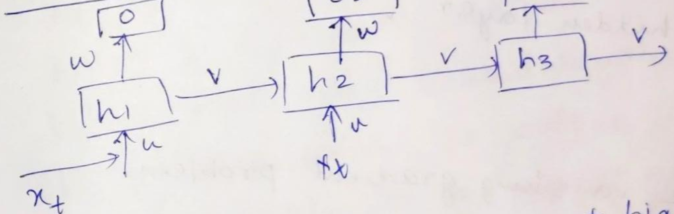
Many to many



ANN

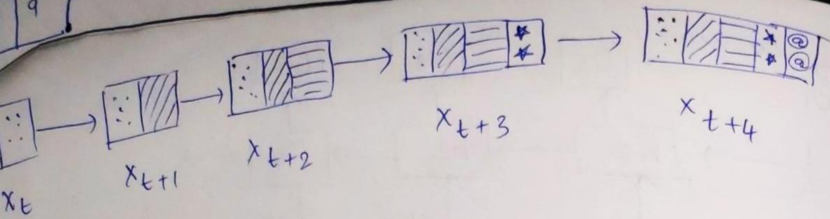


Structure



$h_t = ux_t + V h_{t-1} + \text{bias}$

$O_t = \text{sigmoid}(w h_t + \text{bias})$



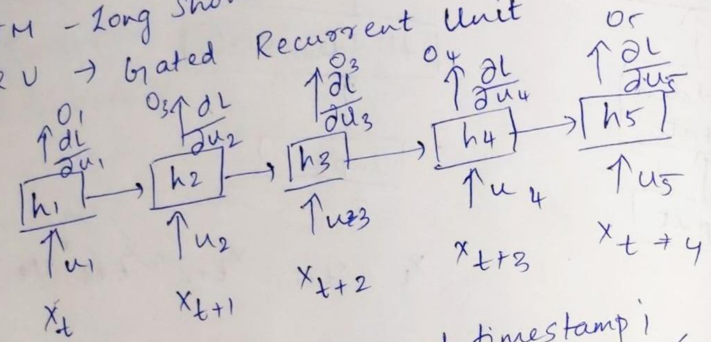
Drawback

Vanishing gradient problem

→ Information flow is very minimal

LSTM - Long Short term memory

GRU → Gated Recurrent Unit



$h_1, h_2,$ → hidden layer at timestamp i,

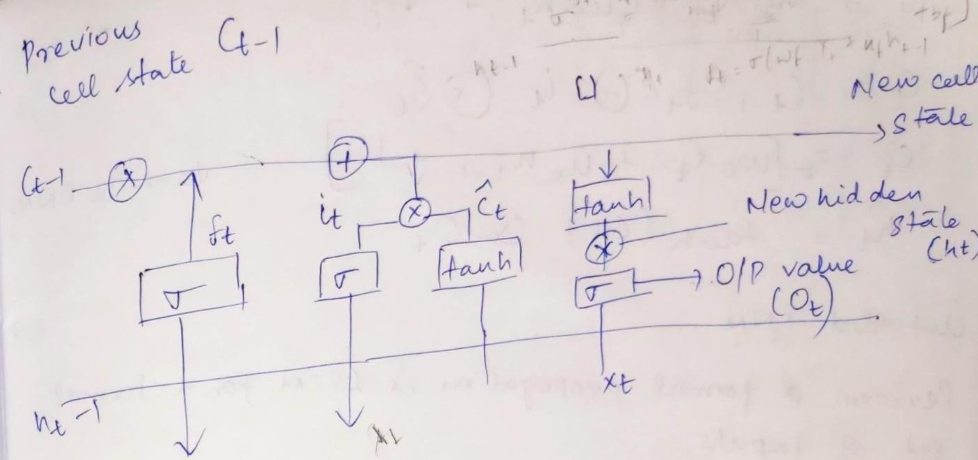
only one hidden layer.

LSTM

To solve vanishing gradient problem

3 Gates →

Forget Gate	Input gate	Output gate
f_t	i_t	$\hat{c}_t$



f_t ⇒ forget gate

i_t ⇒ input gate

$\hat{C}_t$ ⇒ candidate function

O_t ⇒ Output gate

b_f ⇒ bias value forget gate

b_i ⇒ bias value input gate

b_c ⇒ bias value candidate gate

b_o ⇒ bias value for output gate

W_f ⇒ weights for the forget gate

U_f ⇒ weights for the hidden state [forget gate]

U_i = weights for the hidden state [input gate]

W_i ⇒ weights for the input gate

W_o ⇒ weights for the output gate

U_o ⇒ weights for the hidden state

$$f_t = \sigma[W_f x_t + U_f h_{t-1} + b_f]$$

$$i_t = \sigma[W_i x_t + U_i h_{t-1} + b_i]$$

$\hat{C}_t \Rightarrow \tanh(w_c x_t + u_{\hat{c}} h_{t-1} + b_{\hat{c}})$

$C_t = C_{t-1} \otimes f_t \oplus i_t \otimes \hat{C}_t$

$O_t = \sigma[w_o x_t + u_o h_{t-1} + b_o]$

$h_t = \tanh(C_t) \otimes O_t$

Vannilla RNN

Unfolded RNN

Perform a formal propogation is LSTM for a formal set of inputs

weight for forget gate $[0.7 \ 0.45]$

weight for i/p gate $[0.95 \ 8]$

weight for o/p $\begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}$ Hidden o/p $[0.15]$

" candidate fn $\begin{bmatrix} 0.45 \\ 0.25 \end{bmatrix}$

bias forget gate $[0.15]$

" Input gate $[0.2]$

" Candidate gate $[0.65]$

" output gate $[0.1]$

weight for Hidden forget $[0.25]$

" I/P $[0.8]$

Candidate $[0.15]$

	x_1	x_2	Class Label
	1	2	0
t_0	1	2	
t_1	0.5	3	1

$w_f = \begin{bmatrix} 0.7 \\ 0.45 \end{bmatrix}$ $w_i = \begin{bmatrix} 0.95 \\ 8 \end{bmatrix}$

$w_c = \begin{bmatrix} 0.45 \\ 0.25 \end{bmatrix}$ $w_o = \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}$

Input $V_i = 0.8$

Output $V_o = 0.25$

Forget $V_f = 0.1$

Candidate $V_c = 0.15$

Bias: $\hat{C} = 0.2$ $h_{t-1} = 0$

I/P $= 0.65$

Forget $= 0.15$

O $= 1$

$f_t = \sigma\left[\begin{bmatrix} 0.7 \\ 0.45 \end{bmatrix}[1 \ 2] + [0.1 \times 0] + 0.15\right]$

$= \sigma[0.7 + 0.9 + 0.15]$

$= \sigma[\cancel{1.75}] = \underline{0.852}$

0.16
0.15
0.31

input $= \sigma[0.95 \ 8]\begin{bmatrix} 1 \\ 2 \end{bmatrix} + [0.8 \times 0] + 0.65]$

$= \sigma[16.95 + 0.65]$

$= \sigma[17.60]$ $\underline{0.96}$

Candidate $\hat{C}_t \Rightarrow \tanh[[0.45 \ 0.25]\begin{bmatrix} 1 \\ 2 \end{bmatrix} + [0.15 \times 0] + 0.2$

$= [0.95 + 0.2] = 0.15$ $\underline{0.818}$

8	9

$$\text{Output gate} = \sigma\left[0.6 \;\; 0.4\right]\begin{bmatrix}1\\2\end{bmatrix} + [0.25 \times 0] + 0.1$$

$$= \sigma[1.5]$$

$$= 0.817$$

$$C_t = f_t \otimes C_{t-1} \oplus i_t \otimes \hat{C}_t$$

$$C_t = 0.786$$

$$h_t = \tanh\left(C_t \times \text{Output gate}\right)$$
$$0.537$$

II: $$\text{Input} = \sigma\left[\left[0.95 \;\; 8\right]\begin{bmatrix}0.5\\3\end{bmatrix} + [0.8 \times 0.537] + 0.65\right]$$

$$= \sigma[25.55] = 1$$

$$f =$$