Day 7, 03 Jan 2024

NLP - by mentor Dr. Ryan Ahmed, IIT Delhi + Harvard Alumni

Recap ⇒ ⇒ embedding, standardization, Normalization

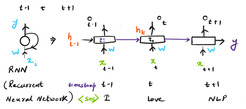
⇒ SZS, RNN

Topic - RNN → Theoretical, Practical implementation

Sequence 2 Sequence

I love NLP

t-1 t t+1



(Recurrent remembering t-1 t t+1

Neural Network) <SOS> I love NLP <EOS>

EOS ⇒ end of Sequence

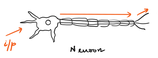
current state ⇒ hidden state

Q) what is difference b/w RNN & ANN

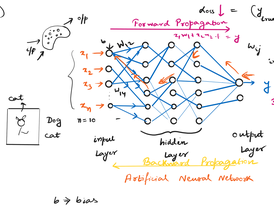
RNN	ANN
⇒ # inputs is not pre defined	⇒ no of inputs are fixed
⇒ weights are same across all layers	⇒ weights are different
⇒ no of parameters are higher	⇒ ... less

ANN (Artificial Neural Network)

human brain ⇒ Neurons



Human perceptron theory



$loss\ L = (y_{true} - y_{pred})$

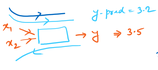
input ⇒ $x_1w_1 + x_2w_2 \dots + x_nw_n + b = y$

$y = \sum x_iw_i + b$

$y = \sum x_iw_i + b$

b ⇒ bias

weight updation



y.pred = 3.2

⇒ 3.5

$x_1w_1' + x_2w_2' \dots = y.pred$

↳ 3.4

y.true ⇒ 3.5

$loss = (y.pred - y.true)$

$= -(3.5 - 3.2)$

$= |-0.3| = 0.3$

$loss = (y.true - y.pred) = |(3.5 - 3.4)| = 0.1$

minimization of loss

$loss = 0 \quad y.true = y.pred$

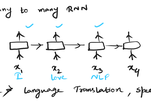
$y = x$

$x_1w_1 + x_2w_2 + b = y$

$y = \sum x_iw_i + b$

* Types of RNN

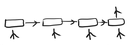
① many to many RNN



example ⇒ Language Translation, speech Recognition

② Many to One RNN

+ve

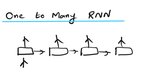


Burgers tasted good

example : Sentiment analysis

Text Classification

③ One to Many RNN

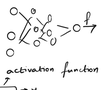


example : Image Captioning

④ One to One RNN

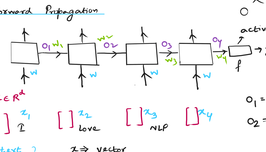
example : image Classification

$y = f(\sum x_iw^i + b)$



activation function

Forward Propagation



$x \in R^n$

[ ] x_1 I [ ] x_2 love [ ] x_3 NLP [ ] x_4

x ⇒ text ? x ⇒ vector

Computer (text) ✗ Number ✓

$loss = (y - \hat{y})$

y = true/ expected value

$\hat{y}$ - predicted value

$o_1 = f(x_1w)$

$o_2 = f_1(x_2w + o_1w_1) = f_1(x_2w + f(x_1w)w_1)$

$o_3 = f(x_3w + o_2w_2)$

$= f(x_3w + o_2w_2) = \dots$

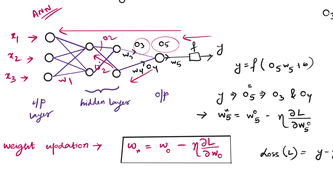
$o_1 \Rightarrow f(x_1)$

$o_2 \Rightarrow f(x_1, x_2, o_1, \dots)$

$o_3 \Rightarrow f(x_1\ x_2\ x_3\ \ o_1\ o_2 \dots)$

Backward Propagation

ANN



$y = f(o_5w_5 + b)$

$y \Rightarrow o_5^0 \Rightarrow o_3$ & o_4

$\rightarrow w_5^n = w_5^0 - \eta \frac{\partial L}{\partial w_5^0}$

chain Rule

$$\boxed{\frac{\partial L}{\partial w_5^0} = \frac{\partial L}{\partial o_5} \cdot \frac{\partial o_5}{\partial w_5^0}}$$

dot product

weight updation → $\boxed{w_n = w_0 - \eta \frac{\partial L}{\partial w_0}}$

$loss(L) = y - \hat{y}$

η = Learning Rate

w_n = modified wt/ new wt

w_0 = old wt

L = Loss function

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial o_5} \cdot \frac{\partial o_5}{\partial o_4} \cdot \frac{\partial o_4}{\partial o_2} \cdot \frac{\partial o_2}{\partial w_2}$$

$\eta = 0.01$

$\eta = 0.1$

$$\frac{\partial L}{\partial w_4} = \frac{\partial L}{\partial o_5} \cdot \frac{\partial o_5}{\partial o_4} \cdot \frac{\partial o_4}{\partial w_4}$$

$$\boxed{w_4^n = w_4^0 - \eta \frac{\partial L}{\partial w_4}}$$

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial o_5} \cdot \frac{\partial o_5}{\partial o_4} \cdot \frac{\partial o_4}{\partial o_2'} \cdot \frac{\partial o_2'}{\partial o_1} \cdot \frac{\partial o_1}{\partial w_1}$$

$$w_1^n = w_1^0 - \eta \frac{\partial L}{\partial w_1}$$

$$\boxed{w_1^n = w_1^0 - \frac{\partial L}{\partial o_5} \cdot \frac{\partial o_5}{\partial o_4} \cdot \frac{\partial o_4}{\partial o_2'} \cdot \frac{\partial o_2'}{\partial o_1} \cdot \frac{\partial o_1}{\partial w_1}}$$

weight updation ⇒ Chain Rule

Backpropagation