(2) Suppose we are Training a simple RNN for a binary action classification task using image inputs & one time-step. The Primary eqns of RNN are given as below;

$$h_t = \tanh(W_{hx}x_t + W_{hh}h_{t-1})$$

$$y = \text{Sigmoid}(W_{hy}h_t)$$

W_{hx}, W_{hh} & W_{hy} denote parameter Matrices, x denotes The input the input image feature vector, h_0 is same as x, h_t is Intermediate RNN state vector, & y denotes output. Suppose The Sum of squared errors functions is used as the loss function. write the expression for the total loss L by Accumulating losses over all the Training samples, & Calculate the 1ST derivate of L w.r.to to the Three parameter matrices.

Total loss(L)

→ Loss function for a single Training example

for This we use sum of squared Error function

n Training samples

loss function for single sample is $L_i = (y_i - t_i)^2$

L_i = loss for i^{th} Training sample

y_i = predicted output for i^{th} sample

t_i = target output for i^{th} sample

calculating total loss (L):
Σ (summation) of all individual losses over all Training samples

$$L = L_1 + L_2 + \dots L_N$$

derivatives of L w.r.t parameter matrices (W_{hh}, W_{xh} & W_{hy})

(i) $\frac{dL}{dW_{hh}}$ 1st derivative

$h_1 = \tanh(W_{hhh0} + W_{xhx})$ { Chain rule to calculate this derivative }

$$\frac{dL}{dW_{hh}} = \frac{dL}{dh_1} \times \frac{dh_1}{dW_{hh}} \quad -(1)$$

$\frac{dL}{dh_1}$ calculate this using chain rule

$$\frac{dL}{dh_1} = \frac{dL}{dy} \times \frac{dy}{dh_1} \quad -(2)$$

$y = \text{sigmoid}(W_{hy}h_1)$.

$$\frac{dy}{dh_1} = \text{sigmoid}(W_{hy}h_1) \times W_{hy}$$

$$\frac{dh_1}{dW_{hh}} = \tanh(W_{hhh0} + W_{xhx}) \times W_{hh0} \quad -(3)$$

from 2 & 3

$$\frac{dL}{dW_{hh}} = \frac{dL}{dy} \times \frac{dy}{dh_1} \times \frac{dh_1}{dW_{hh}} \quad -(4)$$

$$\frac{dL}{dW_{hh}} = \frac{dL}{dy} \times \text{sigmoid}(W_{hy}h_1) \times W_{hy} \times \tanh(W_{hhh0} + W_{xhx}) \times W_{hh0} \quad -(4)$$

IInd derivative '' L w.r.to W_{xh} ($\frac{dL}{dW_{xh}}$) using chain rule

$$\frac{dL}{dW_{xh}} = \frac{dL}{dh_1} \times \frac{dh_1}{dW_{xh}} \quad -(5)$$

$\left(\frac{dh_1}{dW_{xh}}\right) \rightarrow \frac{d}{dx}$ with tanh

$$\frac{dh_1}{dW_{xh}} = \tanh(W_{hh}h_0 + W_{xh}x) \times W_{xh} \quad -(6)$$

5 & (6) → 6 in (5) we

$$\frac{dL}{dW_{xh}} = \frac{dL}{dh_1} \times \tanh(W_{hh}h_0 + W_{xh}x) \times W_{xh} \quad -(7)$$

IIIrd derivative '' L. w.r.to W_{hy} ($\frac{dL}{dW_{hy}}$) using chain rule

$$\frac{dL}{dW_{hy}} = \frac{dL}{dy} \times \frac{dy}{dW_{hy}} \quad -(8)$$

Calculating $\frac{dy}{dW_{hy}}$ using differentiate the sigmoid function

$$\frac{dy}{dW_{hy}} = \text{sigmoid}(W_{hy}h_1) \times h_1 \quad -(9)$$

from 9 in (8)

$$\frac{dL}{dW_{hy}} = \frac{dL}{dy} \times \text{sigmoid}(W_{hy}h_1) \times h_1$$

finally Loss functions (L) are: w.r.to parameters (W_{hh}, W_{xh}, W_{hy})

$$\frac{dL}{dW_{hh}} = \frac{dL}{dy} \times \text{sigmoid}(W_{hy}h_1) \times W_{hy} \times \tanh(W_{hh}h_0 + W_{xh}x) \times W_{hh}h_0$$

$$\frac{dL}{dW_{xh}} = \frac{dL}{dh_1} \times \tanh(W_{hh}h_0 + W_{xh}x) \times W_{xh}$$

$$\frac{dL}{dW_{hy}} = \frac{dL}{dy} \times \text{sigmoid}(W_{hy}h_1) \times h_1$$

(3) (5) five Things from (1) & (2) s&l's. (Assignment (1))

(1) Residual Learning: (Resnet 50) introduced concept of Residual connection, enabling the Training of very deep n/w's by mitigating the vanish gradient problem

I learnt and understood Pretrained CNN models are trained on large scale datasets, so we can use them for feature representation to new tasks, it saves time and computational resources by avoiding need to Train model from scratch

(2) Extracting features and class labels from images with corresponding Annotations [PASCAL VOC 2007 Dataset] and getting all the class labels (objects)

(3) Learnt about normalization & (support vector Mechanism) used to separate data into Two Classes based on their features, maximizing ~~data~~ the margining b/w classes & Handling both linearly & nonLinearly separable data.

(4) finally Confusion Matrix & accuracy used to Evaluate The Performance & measure Accuracy of classification model.

(5) RNN for Binary classification! -

By using sum-of-squared errors as the loss function, we can minimize discrepancy b/w Predicted output (y) & actual Target values.

from This method I learnt How This approach enables us to train The RNN model & optimize Parameters metrices to improve classification Performance.