Bahria University, Islamabad Campus

Department of Computer Science

Final Assessment

Class/Section: MS(DS/CS)-3A

(Spring 2020 Semester)

Course:	**Deep Learning**	Date Assigned:	**July 3, 2020**
Course Code:	DSC 707	Start Time:	**15:30**
Faculty's Name:	Dr. Imran Siddiqi	Submission Time:	**23:30**
Max Marks:	50		

INSTRUCTIONS:

I. The assessment is an individual effort and is assumed to be completed with academic honesty.

II. **Plagiarism** (copying) is not tolerable and will be considered equivalent to cheating in a regular exam. Plagiarized content will be awarded **zero credit** without any debate.

III. Submissions will only be accepted **through LMS** and not through any other medium.

IV. Use this file as an **answer sheet** and provide your solutions. Submit the solution as a **single PDF file**.

V. Clearly write your **Name and Enrolment No**. in the space provided below.

VI. Both typed and handwritten submissions are acceptable.

Name:	**Waqar kaleem khan**	**Enrolment No.**	**01-249191-013**

Question	Marks Obtained	Max. Marks
1		16
2		12
3		14
4		08

Name # waqar Kaleem Khan
Enrolment # 01-249191-013
Course # DL
Date # 03-07-2020
Final term
Teacher Imran Siddiqui

Q1 Part 1

Apply transposed convolution on the feature map with filter "f" using Stride = 2 and no padding

Input $\begin{bmatrix} 2 & 4 \\ 1 & 9 \end{bmatrix}$ f $\begin{bmatrix} 2 & 4 & 9 \\ 1 & 9 & 1 \\ 0 & 1 & 3 \end{bmatrix}$

Multiplying

$$\begin{bmatrix} 2 & 4 \\ 1 & 9 \end{bmatrix} * \begin{bmatrix} 2 & 4 & 9 \\ 1 & 9 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

$$2 \times \begin{bmatrix} 2 & 4 & 9 \\ 1 & 9 & 1 \\ 0 & 1 & 3 \end{bmatrix} = \begin{bmatrix} 4 & 8 & 0 \\ 2 & 18 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$4 * \begin{bmatrix} 2 & 4 & 9 \\ 1 & 9 & 1 \\ 0 & 1 & 3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 36 \\ 0 & 0 & 4 \\ 0 & 0 & 0 \end{bmatrix}$$

2

$$1 * \begin{bmatrix} 2 & 4 & 9 \\ 1 & 9 & 1 \\ 0 & 1 & 3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$9 * \begin{bmatrix} 2 & 4 & 9 \\ 1 & 9 & 1 \\ 0 & \cancel{1} & 3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 27 \end{bmatrix}$$

Now adding all the results

$$\begin{bmatrix} 4 & 8 & 0 \\ 2 & 18 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 36 \\ 0 & 0 & 4 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 27 \end{bmatrix} = \begin{bmatrix} 4 & 8 & 36 \\ 2 & 18 & 4 \\ 0 & 1 & 27 \end{bmatrix}$$

Q1 part 2

Total Multiplication	
standard convolution	$\left(\frac{N-7}{2}+1\right) \times \left(\frac{N-7}{2}+1\right) \times 7 \times 7 \times D$
Depthwise seperable convolution.	$\left(\frac{N-7}{2}+1\right) * \frac{N-7}{2}+1 \times 7 \times 7 \times D \times M$
standard convolution	$\left(D \times 7 \times 7 \times \left(\frac{N-7}{2}+1\right) * \left(\frac{N-7}{2}+1\right)\right) + M$
Depthwise seperable convolution	$\left(M \times D \times 7 \times 7 \times \left(\frac{N-7}{2}+1\right) \times \left(N-7/2\right)\right) + M$

Q1 part 3:- The above digrames given in question we can see that both fine tuning and feature extraction on different CNN models. Where we can see that accuracy is leading 90% for the AlexNet and LetNet as compared to the other network which is 70%. And the other networks GoogleNet and RestNet 50 have accuracy of 90% for feature extracting using SVM classifier. Where RestNet 50 have more than 90% accuracy and googleNet have also 90% accuracy also we can see that fine tunning in both of these networks are near to each other there is just a minute difference where both network show around accuracy of 85%.

The googleNet have 22 layers and RestNet 50 have 50 layers which help these network to give enough time to extract feature from image that's the reason that these two network are showing better performance as a feature extraction. And the other two network shown in digrame have 5 layer (LeNet) and 8 layers (AlexNet)

Q2 Part 1 $f_t = \delta(w_{xf} \cdot x_t + w_{hf} \cdot h_{t-1} + b_f)$

$$f_t = \delta \begin{bmatrix} 0.1 & 0.1 & 0.1 \\ 0.2 & 0.2 & 0.2 \end{bmatrix} \begin{bmatrix} 0.6 \\ 0.6 \\ 0.4 \end{bmatrix} + \begin{bmatrix} 0.1 & 0.2 \\ 0.2 & 0.4 \end{bmatrix} * \begin{bmatrix} 0 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$f_t = \delta \begin{bmatrix} 0.1 * 0.6 + 0.1 * 0.6 + .1 * 0.4 \\ 0.2 * 0.6 + 0.2 * 0.6 + 0.2 * 0.4 \end{bmatrix} + 0 + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$f_t = \delta \begin{bmatrix} 0.16 \\ 0.32 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$f_t = \delta \begin{bmatrix} 0.18 \\ 0.36 \end{bmatrix}$$

$$f_t = \delta \begin{bmatrix} 0.54 \\ 0.58904 \end{bmatrix}$$

Now solving it

$i_t = \delta(w_{xi} \cdot x_t + w_{hi} \cdot h_{t-1} + b_i$

$$i_t = \delta \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.1 & 0.1 \end{bmatrix} \begin{bmatrix} 0.6 \\ 0.6 \\ 0.4 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 0.5 & 0.5 \end{bmatrix} \times \begin{bmatrix} 0 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

P.T.O

$$it = \delta \begin{bmatrix} 0.5*0.6 + 0.5*0.6 + 0.5*0.4 \\ 0.1\times 0.6 + 0.1*0.6 + 0.1*0.7 \end{bmatrix} + 0 + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$it = \delta \begin{bmatrix} 0.3 + 0.3 + 0.2 \\ 0.06 + 0.06 + 0.04 \end{bmatrix} + 0 + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$it = \delta \begin{bmatrix} 0.8 \\ 0.16 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$it = \delta \begin{bmatrix} 0.10 \\ 0.20 \end{bmatrix}$$

$$it = \begin{bmatrix} 0.524 \\ 0.549 \end{bmatrix}$$

Now calculating ct

$$ct = \tanh(wxc \cdot xt + whc \cdot ht-1 + bc)$$

$$ct = \tanh \begin{bmatrix} 0.1 & 0.2 & 0.1 \\ 0.2 & 0.4 & 0.6 \end{bmatrix} \begin{bmatrix} 0.6 \\ 0.6 \\ 0.4 \end{bmatrix} + \begin{bmatrix} 0.5 & 0.1 \\ 0.1 & 0.5 \end{bmatrix}$$

$$\begin{bmatrix} 0 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$ct = \tanh \begin{bmatrix} 0.06 + 0.12 + 0.04 \\ 0.12 + 0.24 + 0.24 \end{bmatrix} + 0 + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$ct = \tanh \begin{bmatrix} 0.22 \\ 0.60 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$\tanh \begin{bmatrix} 0.24 \\ 0.62 \end{bmatrix} \Rightarrow ct = \begin{bmatrix} 0.2357 \\ 0.5648 \end{bmatrix}$$

~~Updating cell state~~

$Ct = ft * Ct-1 + it * Ct$

$$Ct = \begin{bmatrix} 0.54 \\ 0.589 \end{bmatrix} * 0 + \begin{bmatrix} 0.524 \\ 0.549 \end{bmatrix} * \begin{bmatrix} 0.2354 & 0.5648 \end{bmatrix}$$

$$Ct = 0 + \begin{bmatrix} 0.524 * 0.2354 & 0.524 * 0.5648 \\ 0.549 * 0.2354 & 0.549 * 0.5648 \end{bmatrix}$$

$$Ct = \begin{bmatrix} 0.1233 & 0.2959 \\ 0.12923 & 0.3100 \end{bmatrix}$$

Now calculating ot

$$ot = \delta\,(wo\,(ht-1, xt) + bo)$$

$$ot = \delta \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.5 & 0.1 \end{bmatrix} \begin{bmatrix} 0.6 \\ 0.6 \\ 0.4 \end{bmatrix} + \begin{bmatrix} 1 & 0.1 \\ 0.5 & 0.1 \end{bmatrix} * \begin{bmatrix} 0 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

$$ot = \delta \begin{bmatrix} 0.3 + 0.3 + 0.2 \\ 0.06 + 0.3 + 0.04 \end{bmatrix} + \begin{bmatrix} 1 & 0.1 \\ 0.5 & 0.1 \end{bmatrix} * \begin{bmatrix} 0 \end{bmatrix} + \begin{bmatrix} 1.2 \\ 0.4 \end{bmatrix}$$

$$\begin{bmatrix} 0.8 \\ 0.4 \end{bmatrix} + 0 + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix} \Rightarrow \delta \begin{bmatrix} 0.10 \\ 0.8 \end{bmatrix}$$

$$ot = \begin{bmatrix} 0.52 \\ 0.68 \end{bmatrix}$$

$ht = ot * \tanh(Ct)$

$$ht = \begin{bmatrix} 0.52 \\ 0.68 \end{bmatrix} * \begin{bmatrix} 0.2354 \\ 0.5648 \end{bmatrix}$$

$$ht = \begin{bmatrix} 0.52 * 0.2354 + 0.52 * 0.5648 \\ 0.68 * 0.2354 + 0.68 * 0.5648 \end{bmatrix}$$

$$ht = \begin{bmatrix} 0.122408 + 0.293696 \\ 0.160072 + 0.384067 \end{bmatrix}$$

$$ht = \begin{bmatrix} 0.416104 \\ 0.544136 \end{bmatrix}$$

Q2 Part 2

→

2	4	9
1	9	1
0	0	7

2/3	4/13	9/17
1/3	9/13	1/17
0	0	7/17

Filling The 3x3 table using normalize value.

	T=0	T=1	T=2
"B"	0.66	0.30	0.52
"y"	0.33	0.69	0.05
"-"	0	0	0.41

P.T.O

Loss calculation
There are three time-step in matrix t_0, t_1, t_2 and Three character

Calculating all possible Paths

"BY-", "BYB", "-BY" the path are find ~~by~~ Now calculate values

"BY-" = 0.66 * 0.69 * 0.41 = ~~0.186~~ = 0.186
"BYB" = 0.66 * 0.69 * 0.52 = 0.23
"-BY" = 0.41 * 0.69 * 0.66 = 0.186

Sum of all possible path = 0.59

threshold: 0.66 * 0.69 = 0.45

BY Paths is more probable then Threshold
(0.59 > 0.45)

①

Q3 part 1:- In order to predict the third word we will calculate H, Y
H = time state Y = Final time state
also calculate error and update weights through back propagation.

At time $t=0$

$H_0 = \tanh(w_{hx} X + b_h)$

$Y_0 = g(w_{yh} H_0 + b_y)$

At time $t=1$

$H_1 = \tanh(w_{hx} X + W_{hh} H_0 + b_h)$

$Y_1 = \text{softmax}(w_{yh} H + b_y)$

Error = Actual output - predicted output)

Calculation

exam = [1 0 0 0]

is = [0 1 0 0]

tough = [0 0 1 0]

easy = [0 0 0 1]

$$w_{hx} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$w_{yh} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

(2)

$$w_{hh} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$x_t = [1 \ 0 \ 0 \ 0]$$

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

$$w_{hh} \cdot h_{t-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$h_{t-1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$w_{hx} \cdot x_t = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$w_{hx} \cdot x_t = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$h_t = \tanh(w_{hh} \times h_{t-1} + w_{xh} x_t + b_h).$$

$$\tanh\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\right)$$

$$h_t = \tanh\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.76 \\ 0.96 \\ 0.76 \end{bmatrix}$$

(3)

Find second word "is"

$is = [0\ 1\ 0\ 0]$

$$whh \cdot ht\text{-}1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0.76 \\ 0.96 \\ 0.76 \end{bmatrix}$$

$$whx \cdot xt = \begin{bmatrix} 0.76+0+0 \\ 0+0.96+0 \\ 0+0+0.76 \end{bmatrix} = \begin{bmatrix} 0.76 \\ 0.96 \\ 0.76 \end{bmatrix}$$

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

$$ht = \tanh\left(\begin{bmatrix} 0.76 \\ 0.96 \\ 0.76 \end{bmatrix}\right) + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$= \left(\begin{bmatrix} 2.76 \\ 2.96 \\ 0.76 \end{bmatrix}\right)$$

$$ht = \begin{bmatrix} 0.992 \\ 0.997 \\ 0.992 \end{bmatrix}$$

$$why \times ht = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0.992 \\ 0.997 \\ 0.992 \end{bmatrix}$$

(4)

$$= \begin{bmatrix} 0.992 + 0 + 0 \\ 0.992 + 0.992 \\ 0 + 0 + 0 \\ 0.992 + 0.994 + 0.992 \end{bmatrix}$$

$$= \begin{bmatrix} 0.992 \\ 1.984 \\ 0 \\ 2.978 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.992 \\ 1.987 \\ 0 \\ 2.978 \end{bmatrix}$$

$$= \begin{bmatrix} 0.992 \\ 1.984 \\ 0 \\ 2.978 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 0.992 \\ 1.984 \\ 0 \\ 2.978 \end{bmatrix}$$

using softmax

$$\sum e^{zk} = 2.69 + 7.30 + 1 + 19.64$$
$$= 30.63$$

$$\begin{bmatrix} 0.08 \\ 0.23 \\ 0.03 \\ 0.64 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Q3 Part(iii) Difference b/w Fast R-CNN and Yolo

Fast R-CNN (*) Instead of feeding the region proposals to CNN we feed the input image to CNN

(*) Fast R-CNN based of VGG16

(*) Fast-R-CNN uses cross entropy for foreground and background loss and L_1 for cordinates

(*) It detect small object because 9 anchor boxes

YoLo

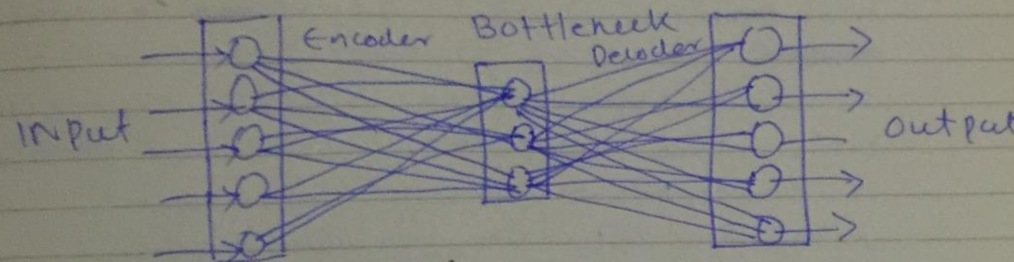
(*) Yolo does detection and classification at same time

(*) Yolo is fully end-to-end-training

(*) Yolo architecture base on GoogleNet

(*) Yolo used l_2 loss for bounding box regression, classification.

Q4 part 1 An auto encoder is a technig we by using neural network to encode something autmaticaly the auto encoders is able to learn how to decompose data (in our case image data) to a small bits of data and then reconstruct the orignal data as similar as it can be



steps follow to fill missing data in image:

(1) input images data set
(2) write function to convert raw matrix to image and change the color to RGB system
(3) load data set and adapt it to our needs
(4) Implementing Autoencoder
(5) define input size of image
(6) output size of image
(7) define hidden layers
(8) activation function
~~(9) Loss function.~~
(10) use encoder and decoder probabalistic concept $p(X/Y)$
(11) Convolution layers and Activation function and ~~Activation function~~ Batch normalization and Batch normalization
12) De-convolution and Batch normalization for Decoding
13) calculate loss by using loss function such as RMSE, crossentropy etc.

Q4 Part 2 In the rapid increase in research of the deeplearning or artifical intelligence these days. For simpler problem AI is now being catered as a Solution. Using deep learning for problems which can be solve easily by using traditional techniques is not good experiments or way because the first problem is that AI environments providing for every researcher is not easy because of it's expensive tools like (GPU's). The second problem is that ML or DL can't be happen without past data avalibality so it's hard for researchers to find data related to the problem They are solving or if the data is avalible then its not free.