**LAB9: Basic of Deep Learning**

Q1: Please compute the output size of the convolutional layers using following configuration and verified with keras or pytorch convolutional layer:

(i) Input\_image=4, kernel\_size=3, stride=1 and padding, p=0

(ii) Input\_image=5, kernel\_size=4, stride=1 and padding, p=2

(iii) Input\_image=5, kernel\_size=3, stride=1 and padding, p=1

(iv) Input\_image=5, kernel\_size=3, stride=1 and padding, p=2

Q2: What the output size of 2D transpose convolutional layer if we have the following configuration:

(i) Input\_image=6, kernel\_size=3, stride=1 , padding=0, out\_padding=0

(ii) Input\_image=5, kernel\_size=4, stride=2 , padding=0, out\_padding=0

(iii) Input\_image=6, kernel\_size=3, stride=2 , padding=0, out\_padding=0

Q3: What the output size of 2D transpose convolutional layer if we have the following configuration and verify the results manually and with keras or pytorch:

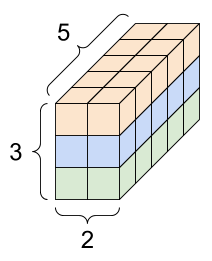
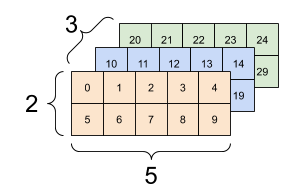
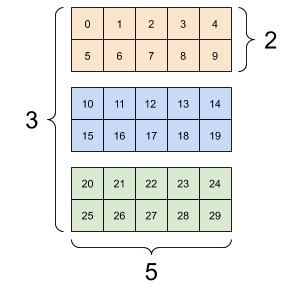
(i) Input\_image=6, kernel\_size=3, stride=1 , padding=0, out\_padding=0

(ii) Input\_image=5, kernel\_size=4, stride=2 , padding=0, out\_padding=0

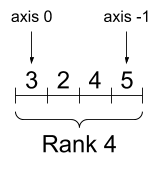
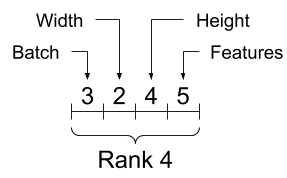
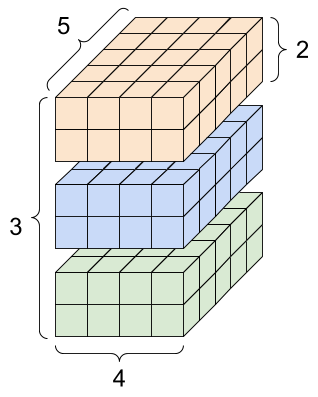
(iii) Input\_image=6, kernel\_size=3, stride=2 , padding=0, out\_padding=0

Various ways to give shape of give tensor is shown in the following figure.

A 3-axis tensor, shape: [3, 2, 5]



The deep learning layers required the dataset in tensor format as shown below.



Q3: Please apply the 2D-Upsampling in keras or Pytorch for the following tensors.

(i) 2Dupsampling using factor( 2,1)

(i) input\_shape = (2, 2, 1, 3)

x = np.arange(np.prod(input\_shape)).reshape(input\_shape)

(ii) 2Dupsampling using factor( 2,4)

(ii) input\_shape = (4, 4, 10, 13)

x = np.arange(np.prod(input\_shape)).reshape(input\_shape)

(iii) 2Dupsampling using factor( 6,6)

(iii) input\_shape = (4, 4, 10, 13)

x = np.arange(np.prod(input\_shape)).reshape(input\_shape)

Q4: We have the following network, please compute the size of each layer and trainable parameters for each layer. Build the network in keras and verify your manual results with model summary.

C1 layer (kernek\_size=5x5,

No.filters=11)

FC1 layer

No.Units=50

P1 layer

(Pool\_size=2x2,

strides=2x2)

C2 layer

(kernek\_size=3x3,

No.filters=20)

Output layer(classes=3)

P2 layer

(Pool\_size=3x3,

strides=3x3)

FC1 layer

No.Units=60

Input\_size

(200x200x1)

C3 layer

(kernek\_size=5x5,

No.filters=25)

P3 layer

(Pool\_size=2x2,

strides=2x2)

C4 layer

(kernek\_size=5x5,

No.filters=17)

Q5:Analyze the above model using the following command:

Model.get\_config()

Model.count\_params()

Check parameters of convolutional and fully connected layers

print(model.layers[0].count\_params())

print(model.layers[1].count\_params())

… so on

Check the size of input and output\_layers

print(model.layers[0].input\_shape)

print(model.layers[0].output\_shape)

…so on

Check weight and biases of the layers.

first\_layer\_weights = model.layers[1].get\_weights()

print(first\_layer\_weights[0].shape) # weight matrix

print(first\_layer\_weights[1].shape) # bias vector

Q6: Compute the trainable parameters for the following network and verify with the keras model summary().

