Birla Institute of Technology & Science, Pilani
Work Integrated Learning Programmes Division
Second Semester 2019-20
M.Tech. (Data Science and Engineering)
Comprehensive Examination (Regular)

Course No.	: DSECLZG524	No. of Pages = 4
Course Title	: DEEP LEARNING	*No. of Questions = 6*
Nature of Exam	: Open Book	
Weightage	: 50%	
Duration	: 2 Hours	
Date of Exam:	July 5, 2020	Time of Exam: 2:00 PM – 4:00 PM

Note: Assumptions made if any, should be stated clearly at the beginning of your answer. Show your rough work to get partial credit, when appropriate.

Question 1. **[2+3+2.5+2.5=10 marks]**

A. Refer to the partial code of an ANN implementation using Tensorflow and answer following questions:

```
import tensorflow as tf
data = pd.read_csv('train.csv')
a = tf.placeholder(tf.float32, shape=[None, 16])
b = tf.placeholder(tf.float32, shape=[None, 3])
phi = {
   'alpha': tf.Variable(tf.random_normal([16, 8])),
   'beta': tf.Variable(tf.random_normal([8, 3]))
            }
omega = {
   'alpha': tf.Variable(tf.random_normal([8])),
   'beta': tf.Variable(tf.random_normal([3]))
            }

tl = tf.add(tf.matmul(x, phi['alpha']), omega['alpha'])
tl = tf.nn.relu(tl)
fl = tf.matmul(tl, phi['beta']) + omega['beta'])
cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(fl, b))
optimizer = tf.train.AdamOptimizer(learning_rate=0.5).minimize(cost)
init = tf.initialize_all_variables()
```

a) Which activation function is used in the output layer? Why do you think this particular activation function was chosen? What difference would it make if we used sigmoid function instead?

Softmax is used in the output layer. The sum of output values from all the output nodes would be 1 so it is likely that the required output is probability distribution of a given variable. Sigmoid could also be used but the output values will not sum to 1.

b) Calculate the output values assuming the input vector to the output layer to be [2,0,1,0], weight matrix to be [[0.2,0.3,0.2,0.1],[0.1,0.2,0.5,0.1],[0.2,0.1,0.1,0.1]] and bias vector to be [0.3,0.1,0.1]
w.x+b = [0.9, 0.8, 0.6]; softmax values = [0.38, 0.34, 0.28]

B. Refer to the following code snippet.

```
>>> from tensorflow.keras.applications.vgg16 import VGG16
>>> from tensorflow.keras.models import Model
>>> model = VGG16()
>>> model.layers.pop()
>>> model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
```

a) If we add Batch Normalization after every convolution layer in the modified VGG16 model (popping last layer), what will be the total number of additional trainable parameters, beta's and gamma's?
8,448

Convolution Layer Channels	**Number of Layers**	**Trainable Parameters** No. of layers x no. of channels x 2 parameters (Beta and Gamma)	**Non-Trainable Parameters** No. of layers x no. of channels x 2 parameters (Mean and Variance)	**Total Parameters**
64	2	2*64*2 = 256	2*64*2 = 256	512
128	2	2*128*2 = 512	2*128*2 = 512	1,024
256	3	3*256*2 = 1,536	3*256*2 = 1,536	3,072
512	6	6*512*2 = 6,144	6*512*2 = 6,144	12,288
		8,448	**8,448**	**16,896**

b) What will be the total number of non-trainable parameters, i.e., means and variances in the first 3 Batch Normalization layers?
512 (2*64*2 + 128*2)

Question 2. [**2+3+2+2=9 Marks**]

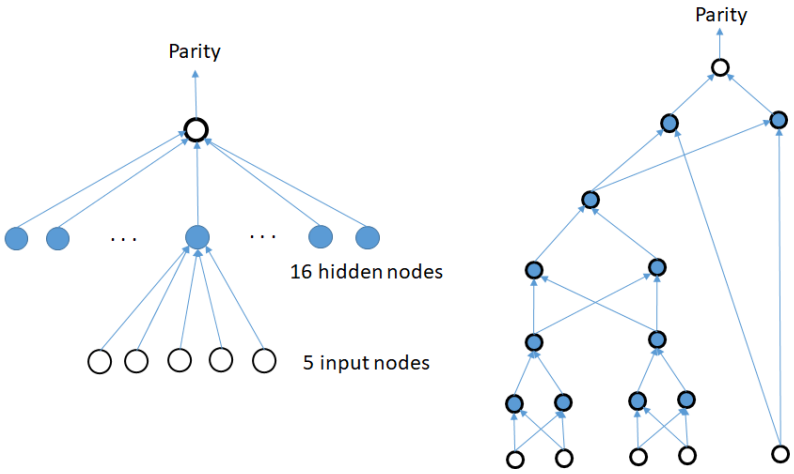
An even parity bit generator outputs 1 if number of '1's in a binary string is zero or even (otherwise, 0 output is generated). Consider an even parity bit generator for binary string length of 5. Activation functions are step threshold, i.e., output =1 if total input >= threshold, else 0.

A. What is the minimum number of hidden nodes in a single hidden layer perceptron network required for implementing this even parity generator?
Parity = x1 xnor x2 … xnor x5, given input string = (x5, x4, …, x1). So, number of hidden nodes for single layer implementation = $2^{(5-1)}$ = 16.

B. What is the minimum number of hidden nodes if multiple hidden layers are used? How many hidden layers will be required?
In binary tree-based implementation, number of hidden nodes = 3*(5-1) -1 = 11. Corresponding number of hidden layers = 5

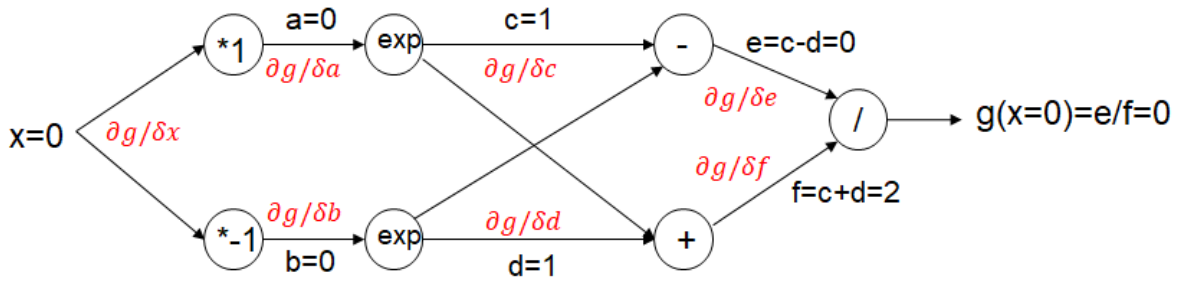
C. What is the total number of weights needed for single hidden layer based implementation?
Number of input-hidden layer weights = 16 * 5. Number of hidden-output layer weights = 5. So, total number of weights = 85.

D. What is the total number of weights needed for more than one hidden layer based implementation?
24



Question 3. [0.5+0.5+1+1+1+1+1+1.5+1.5=9 Marks]

A. Refer to the computational graph below. The values of the variables in forward pass is indicated in black. Calculate the partial derivatives in the backward pass δg/δe, δg/δf, δg/δc, δg/δd, δg/δa, δg/δb and δg/δx. You can use d/dx (1/x) = $-1/x^2$, d/dx exp(x) = exp(x).



[*Note: g(x) = tanh(x)*]

δg/δe = 1/f = 1/2 δg/δf = $-e/f^2$ = 0 δg/δc = 1/2*1 = 1/2 δg/δd = 1/2*(-1) = -1/2
δg/δa = 1/2*1=1/2 δg/δb = -1/2 *1 = -1/2 δg/δx = 1/2 - (-1/2) = 1

B. Is it necessary to randomize the order of training data in full batch gradient descent? Explain your answer.

It is not necessary. Each iteration of full batch gradient descent runs through the entire dataset and therefore order of the dataset does not matter.

C. When is Gradient Descent preferred over Stochastic Gradient Descent, and vice-versa? Explain your answer.

GD theoretically minimizes the error function better than SGD. However, SGD converges much faster once the dataset becomes large. That means GD is preferable for small datasets while SGD is preferable for larger ones. In practice, however, SGD is used for most applications because it minimizes the error function well enough while being much faster and more memory efficient for large datasets.

Question 4. [1+3+1+2+1+2 = 10 Marks]

A. Consider a 1x3 convolution (as per Deep Learning terminology) operator o =[o_1, o_2, o_3]= [-1 2 -1]. Output is generated by stacking two such operators on the 1-D input data (i.e., the operator is applied twice on the input).

a) What is the size of the equivalent operator O that will give the same output, when applied only once on the input data?

Height of O = 1 Width of O = 2*3-1 = 5

b) Show the elements O(i,j) of equivalent operator O.

O can be obtained by cross-correlation of o with o.

O = [1 -4 6 -4 1]

$$O(1,j) = \sum_{k=1}^{j} o_k o_{4-k}$$

B. Consider a 3x3 operator and a 7x7 input, as shown below. Point (i, j) corresponds to the i-th row (ranging 1, 2, …) from top to bottom and j-th column (ranging 1,2, …) from left to right.

-1	0	-1
0	4	0
-1	0	-1

0.5	1	1	1	1	1	1
1	0.5	**0**	0	**0**	0	0
0	0	0.5	1	1	1	1
1	1	**1**	0.5	**0**	0	0
0	0	0	0	0.5	1	1
1	1	**1**	1	**1**	0.5	0
0	0	0	0	0	0	0.5

a) What will be the output size for 2-dilated convolution with padding 0 and stride 1?

i = 7, k =3, p=0, s=1. So, output size = (7+2*0-3-2*1) + 1= 3

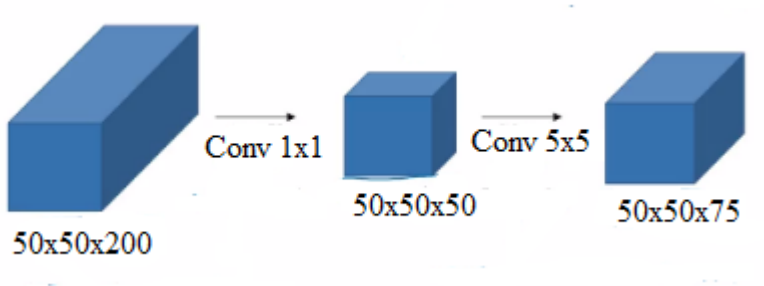
b) What will be the value at (2, 1) in the output plane?
-1x1+0x0-1x0+0x1+4x1+0x0-1x1+0x1-1x1=1

C. In the following figure, 1x1 operators are used first to process the same 50x50 images of depth 200, and first output 50x50 images of depth 50, and then 5x5 operators are used to output 50x50 images of depth 75.

a) What is the padding size used in the first step and padding size in the last step?
For 1x1 convolution, padding size = 0. For 5x5 convolution, padding size = 2.

b) How many multiplication operations are needed here?

ii.



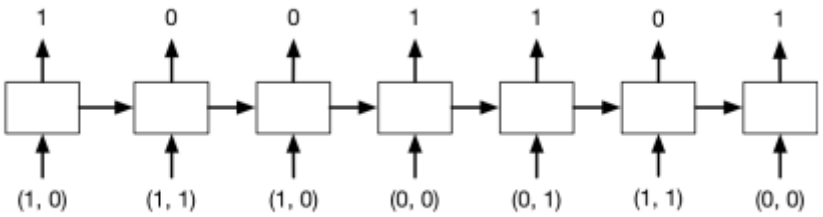
For 1x1 convolution, # of multiplication operations = 50x50x50x200=2.5x10^7
For 5x5 convolution, # of multiplication operations = 50x50x5x5x50x75=234375000
So, total # of operations = 236,875,000

Question 5. [1+1+1+3+2+1 = 9 marks]

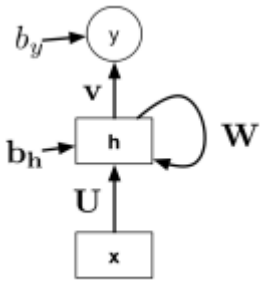
Design a recurrent neural network that adds binary numbers of arbitrary length. The inputs are given as binary sequences, starting with the least significant binary digit. The sequences will be padded with at least one zero on the end. For instance, the problem 100111 + 110010 = 1011001 would be represented as:

- Input 1: 1, 1, 1, 0, 0, 1, 0
- Input 2: 0, 1, 0, 0, 1, 1, 0
- Correct output: 1, 0, 0, 1, 1, 0, 1

There are two input units corresponding to the two inputs, and one output unit. Therefore, the pattern of inputs and outputs for this example would be:

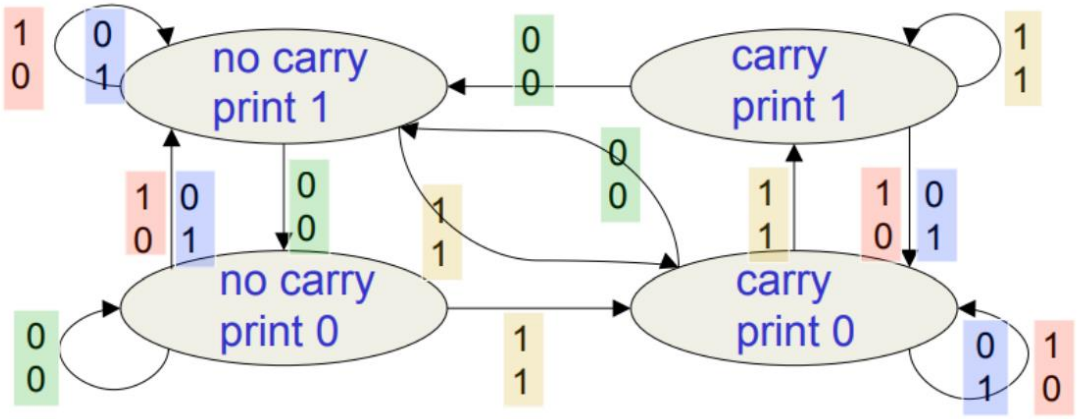


The RNN has two input units **x**, three hidden units **h**, and one output unit y. All of the units use the hard threshold activation function, i.e., output is 1 if total weighted input is >=bias, else 0.



Note, at time t, $\mathbf{h}_t = \text{step}(\mathbf{W}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x} + \mathbf{b}_h)$ and $y_t = \text{step}(\mathbf{v}\mathbf{h}_t + b_y)$

A. Choose the state representations for the underlying finite state machine in terms of outputs of 3 hidden nodes.



Refer to the above state transition diagram from session 16 lecture slides. We can choose, the states as [h1,h2,h3]=[000], [100], [110], and [111], corresponding to "no carry print 0", "no carry print 1", "carry print 0" and "carry print 1" states, respectively.

B. For the chosen representations, specify weight matrices **U** (size 3x2), **v** (size 1x3), and **W** (size 3x3), bias vector $\mathbf{b}_\mathbf{h}$ (3x1) and scalar bias b_y. Elements of **U**, **v** and **W** can be only -1, 1 or 0.

$h1_t$	$h2_t$	$h3_t$	y_t
0	0	0	0
1	0	0	1
1	1	0	0
1	1	1	1

$y_t = h1 - h2 + h3$. So, v = [1 -1 1] and $b_y = 0.5$.

From the above state transition diagram and chosen state representation,

$h1_t$ is 0 only when $h2_{t-1} = 0$, $h3_{t-1} = 0$, x1=0, x2=0.
$h1_t' = h2_{t-1}'h3_{t-1}'x1'x2'$
or, h1(t) = h2(t-1)+h3(t-1)+x1+x2 corresponding, b = 0.5

Similarly,
h3(t) is 1 when h2(t-1)=1, x1=1, x2=1
or, h3(t) = h2(t-1)x1x2
or, h3(t) = h2(t-1) + x1 + x2 corresponding, b=2.5

h2(t) is 1 when h2(t-1)=0, x1=1 x2=1 or h2(t-1)=1, x1+x2=1
or, h2(t) = h2(t-1)+x1+x2 corresponding, b=1.5.

Thus,

$$W=\begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$U=\begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$b_h^T = [\,0.5\ 1.5\ 2.5]$$

[Caution: There may be multiple solutions]

Question 6. Answer the following questions [3 Marks]

A. For what choice of activation functions in hidden and output nodes, autoencoder weights represent the principal component vectors?
Linear (i.e., output = input)

B. For reconstruction of real inputs, what type of loss function is used in autoencoders?
Sum of squared error (i.e., SSE)

C. Gradient ascent is used in training the generative network in GAN. True / False
False

D. If reconstructed input are binary (0 or 1), what type of activation function are needed in the output layer of autoencoders?
Sigmoid (i.e., 1/(1+exp(-x))

E. For reconstruction of binary (0 or 1) inputs, what type of loss function used in autoencoders?
Log-loss / binary cross-entropy

F. What is the regularization functional used in contractive autoencoder?
Jacobian