CSE-5368 Neural Networks Fall 2025 Exam_01

Last Name	First Name	Student ID Number
Solution		

Prob #	**1**	**2**	**3**	**4**	**Total**
Points	**25**	**25**	**25**	**25**	

Time: 80 Minutes

Solution

CSE-5368 Neural Networks Fall 2025 Exam_01

Last Name	First Name	Student ID Number
Solution		

$$F(\mathbf{x}) = F(\mathbf{x}^*) + \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*) + \frac{1}{2}(\mathbf{x}-\mathbf{x}^*)^T\nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*) + \cdots$$

$$\frac{\mathbf{p}^T\nabla F(\mathbf{x})}{\|\mathbf{p}\|} \qquad \frac{\mathbf{p}^T\nabla^2 F(\mathbf{x})\mathbf{p}}{\|\mathbf{p}\|^2} \qquad \alpha_k = -\frac{\mathbf{g}_k^T\mathbf{p}_k}{\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k\mathbf{g}_k \qquad \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{p}_k$$

$$L_i = \sum_{j\neq i} max(0, y_j - y_i + \Delta)$$

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

$$H(p,q) = -\sum_x p(x)log(q(x))$$

$$L_i = -log(\frac{e^{y_i}}{\sum_j e^{y_j}})$$

CSE-5368 Neural Networks Fall 2025 Exam_01

Last Name	First Name	Student ID Number
Solution		

1. Consider the following performance surface

$$F(X) = 2x_1^2 - 3x_1x_2 + 5x_1 - 4x_2$$

Given the initial point $\begin{bmatrix}2\\1\end{bmatrix}$, take **two steps** of the **steepest descent algorithm**, minimizing along a line **at each step**.

You must show the resulting position after each step.

$$\nabla F(x)|_{X=X_0} = \begin{bmatrix}4x_1 - 3x_2 + 5\\ -3x_1 - 4\end{bmatrix} = \begin{bmatrix}10\\-10\end{bmatrix}$$

$$\nabla^2 F(x)|_{X=X_0} = \begin{bmatrix}4 & -3\\ -3 & 0\end{bmatrix}$$

$$p_0 = -g_0 = \begin{bmatrix}-10\\10\end{bmatrix}$$

$$\alpha_0 = -\frac{[10 \quad -10]*\begin{bmatrix}-10\\10\end{bmatrix}}{[-10 \quad 10]\begin{bmatrix}4 & -3\\-3 & 0\end{bmatrix}\begin{bmatrix}-10\\10\end{bmatrix}} = -\frac{-200}{[-10 \quad 10]\begin{bmatrix}-70\\30\end{bmatrix}}$$

$$= -\frac{-200}{1000} = 0.2$$

$$X_1 = \begin{bmatrix}2\\1\end{bmatrix} + 0.2\begin{bmatrix}-10\\10\end{bmatrix} = \begin{bmatrix}0\\3\end{bmatrix}$$

$$\nabla F(x)|_{X=X_0} = \begin{bmatrix}-4\\-4\end{bmatrix}$$

$$p_1 = -g_1 = \begin{bmatrix}4\\4\end{bmatrix}$$

$$\alpha_1 = -\frac{[-4 \quad -4]*\begin{bmatrix}4\\4\end{bmatrix}}{[4 \quad 4]\begin{bmatrix}4 & -3\\-3 & 0\end{bmatrix}\begin{bmatrix}4\\4\end{bmatrix}} = -\frac{-32}{[4 \quad 4]\begin{bmatrix}4\\-12\end{bmatrix}} = -\frac{-32}{-32} = -1$$

$$X_2 = \begin{bmatrix}0\\3\end{bmatrix} + (-1)\begin{bmatrix}4\\4\end{bmatrix} = \begin{bmatrix}-4\\-1\end{bmatrix}$$

Last Name	First Name	Student ID Number
Solution		

Problem 1 Continued

CSE-5368 Neural Networks Fall 2025 Exam_01

Last Name	First Name	Student ID Number
Solution		

2. After executing the program below, what exactly is printed? Show the result below..

```
import torch
import torch.nn as nn

# Input: (batch, channels, height, width)
x = torch.randn(100, 3, 96, 140)

model = nn.Sequential(
    nn.Conv2d(in_channels=3, out_channels=20, kernel_size=(7,5),stride=(3,5),
    padding=(2, 0), bias=False),
    nn.MaxPool2d(kernel_size=(2,2), stride=(2,2), padding=(0,0)),
    nn.Flatten(),
    nn.Linear(in_features=4480, out_features=60, bias=False))

with torch.no_grad():
  y0 = model[0](x)
  y1 = model[1](y0)
  y2 = model[2](y1)
  y3 = model[3](y2)

print("1:", tuple(model[0].weight.shape))
print("2:", tuple(y0.shape))
print("3:", tuple(y1.shape))
print("4:", tuple(y2.shape))
print("5:", tuple(model[3].weight.shape))
print("6:", tuple(y3.shape))
```

1: (20, 3, 7, 5)
2: (100, 20, 32, 28)
3: (100, 20, 16, 14)
4: (100, 4480)
5: (60, 4480)
6: (100, 60)

Last Name	First Name	Student ID Number
Solution		

Problem 2 Continued

CSE-5368 Neural Networks Fall 2025 Exam_01

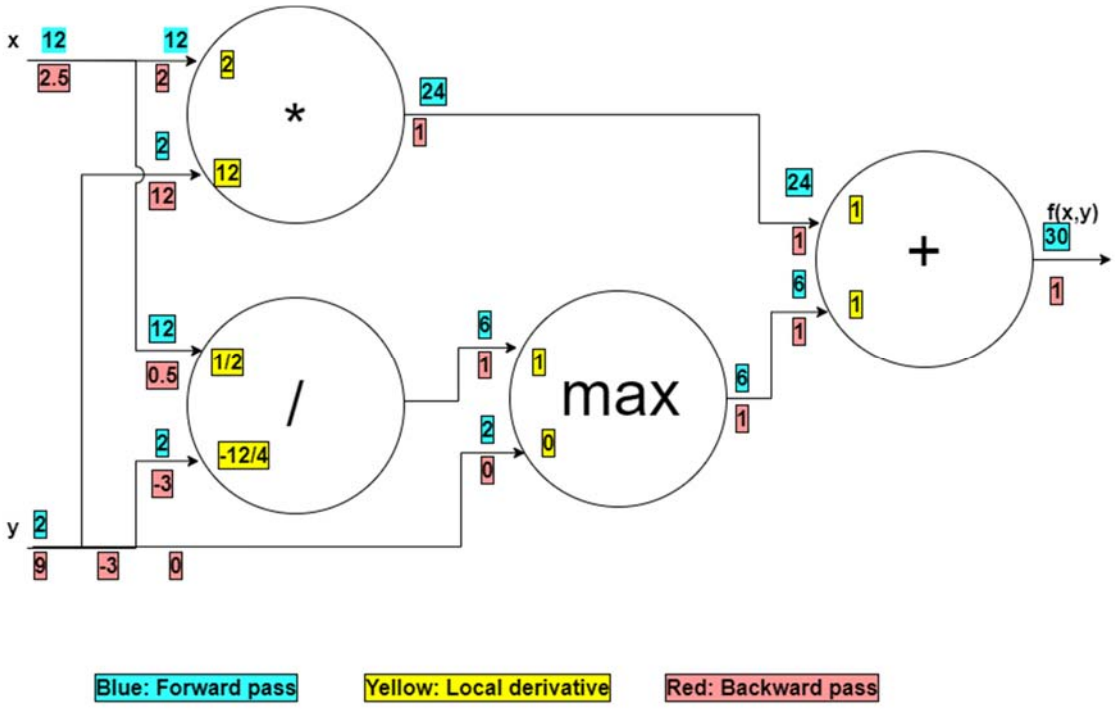
Last Name	First Name	Student ID Number
Solution		

3. Consider the expression: $\boldsymbol{f(x,y) = (x * y) + \max(\frac{x}{y}, y)}$

Given the inputs $\boldsymbol{x = 12,}$ $\boldsymbol{y = 2}$

Draw the computational graph and calculate the $\frac{\delta f(x,y)}{\delta x}$ and $\frac{\delta f(x,y)}{\delta y}$

Show all the numerical values of the forward and backward pass.
You MUST SHOW all the numerical values as they flow in the forward and backward path in the computational graph.



Last Name	First Name	Student ID Number
Solution		

Problem 3 Continued

Last Name	First Name	Student ID Number
Solution		

4. Complete the code for the following function.

This function is identical to the initialization step used in **Assignment 01**.

```
import numpy as np
def initialize_multi_layer_nn(X_train,Y_train,layers):
""" This function Initializes all the weights and returns weight
matrices in a list. You do not need to train the network.
bias should be included in the weight matrix.
X_train: numpy array of input shape=[input_dimension,num_train_samples]
Y_train: numpy array of targets
shape=[output_dimension,num_train_samples]
layers: array of integers representing number of nodes in each layer
return: This function should return a list of weight matrices.
Each element of the list should be a numpy array corresponds to the
weight matrix of the corresponding layer.
Initialize all the weights to zero """
    previous_dimension=X_train.shape[0]
    weight_matrices=[]
    for k in layers:
        temp_w=np.zeros((k,previous_dimension+1))
        weight_matrices.append(temp_w)
        previous_dimension=k
    return weight_matrices
```

#Another solution

```
    temp=np.append(X_train.shape[0],np.copy(layers))
    return [np.zeros((temp[k+1],temp[k]+1)) for k in range(len(temp)-1)]
```

Last Name	First Name	Student ID Number
Solution		

Problem 4 Continued