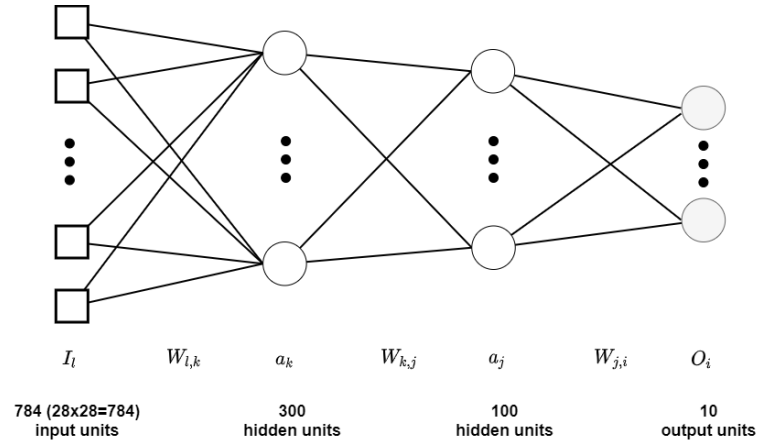
THE HONG KONG UNIVERSITY OF SCIENCE & TECHNOLOGY
Department of Computer Science and Engineering
COMP4331: Introduction to Data Mining
Fall 2021 Answer: Assignment 3
Due time and date: 11:59pm, November 27 (Sat), 2021.

Q1. In this assignment, you will use the MNIST dataset, which contains handwritten digits. Partial codes and TODO instructions for questions Q1 (a), (c), (d), (f) are provided in assignment3.ipynb. Follow the TODO instructions to complete these questions and show the required outputs in assignment3.ipynb.

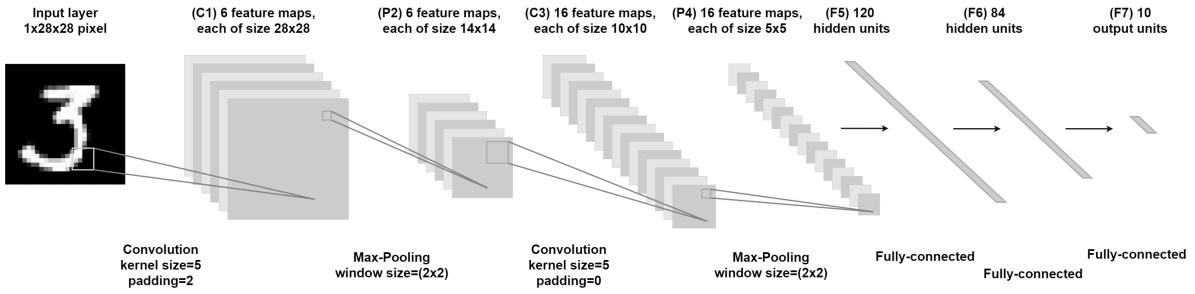
(a) Build the following multilayer perceptron (MLP) by adding codes for *__init__(self)* and *forward(self, x)*.



(b) How many parameters (including the bias) need to be learned in this MLP?

(c) Load training data with batch_size=25 and testing data with batch_size=1000. Train the MLP for 3 epochs. Report the trained MLP's accuracy on the training set and accuracy on the testing set.

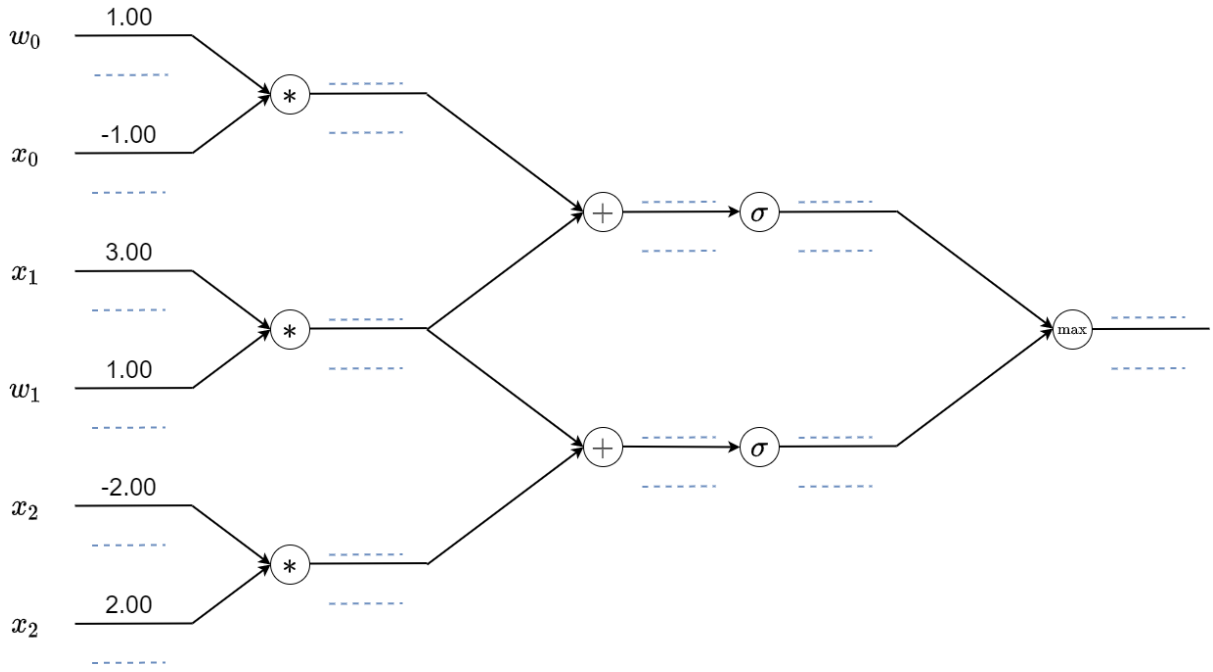
(d) Next, build the following convolutional neural network (CNN) by adding codes for the *__init__(self)* and *forward(self, x)*.



(e) How many parameters (including bias) need to be learned in this CNN?

(f) Load training data with batch_size=200 and testing data with batch_size=1000. Train the CNN for 3 epochs. Report the trained CNN's accuracy on the training set and accuracy on the testing set.

Q2. The following shows the computational graph for $f(\mathbf{w}, \mathbf{x}) = \max(\sigma(w_0 * x_0 + w_1 * x_1), \sigma(w_1 * x_1 + w_2 * x_2))$, where $\mathbf{w} = [w_0, w_1, w_2]$ and $\mathbf{x} = [x_0, x_1, x_2]$, $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function (with derivative is $\frac{d\sigma(x)}{dx} = (1 - \sigma(x))\sigma(x)$). As in the lecture notes, obtain the forward and backward propagation results (4 decimal places) for the given $\mathbf{w}$ and $\mathbf{x}$. Write your answer in the figure, and then scan it to the report.



Submission Guidelines

Please submit

(i) a report report.pdf containing your results for Q1 (b), (c), (e), (f), and Q2.

(ii) a python notebook assignment3.ipynb containing your code and outputs for Q1 (a), (c), (d), (f).

Zip all the files to YourStudentID_assignment3.zip (e.g., 12345678_assignment3.zip). Please submit the assignment by uploading the zip file to Canvas. Note that the assignment should be clearly legible, otherwise you may lose some points if the assignment is difficult to read. Plagiarism will lead to zero point on this assignment

Solution:

Q1.

(a) Please check Figure 1 for the added code.

```
# Build MLP
class MLP(nn.Module):
    def __init__(self):
        super(MLP, self).__init__()
        self.fc1 = nn.Linear(28*28, 300)
        self.fc2 = nn.Linear(300, 10)

    def forward(self, x):
        x = x.view(-1, 28*28)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

Figure 1: Added code for Q1(a)

(b)
- 1st fully-connected layer: $785 * 300 = 235,500$
- 2nd fully-connected layer: $301 * 100 = 30,100$
- 3rd fully-connected layer: $101 * 10 = 1,010$

So $235,500 + 30,100 + 1,010 = 266,610$ parameters.

(c) Final training accuracy should be roughly 96.68%, while final testing accuracy should be roughly 95.76%.

(d) Please check Figure 2 for the added code.

```
# Build Convolutional Neural Network (CNN)
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(in_channels=1, out_channels=6, kernel_size=5, padding=2)
        self.conv2 = nn.Conv2d(in_channels=6, out_channels=16, kernel_size=5)
        self.pool = nn.MaxPool2d(kernel_size=2, stride=2) # pool of square window of size=2, stride=2
        self.fc1 = nn.Linear(in_features=16*5*5, out_features=120)
        self.fc2 = nn.Linear(in_features=120, out_features=84)
        self.fc3 = nn.Linear(in_features=84, out_features=10)


    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = torch.flatten(x, 1) # flatten all dimensions except batch
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x
```

Figure 2: Added code for Q1(e)

(e) Formula to compute learnable parameters in each convolutional layer: ((filter width * filter height * number of feature maps in previous layer + 1) * number of feature maps in current layer)

- C1 layer: (5*5*1+1)*6 = 156

- P2 layer: Max-Pooling layer doesn't learn parameters
- C3 layer: (5*5*6+1)*16 = 2,416
- P4 layer: Max-Pooling layer doesn't learn parameters
- F5 layer: (5*5*16+1)*120 = 48,120
- F6 layer: (120+1)*84 = 10,164
- F7 layer: (84+1)*10 = 850

So $156 + 2416 + 48120 + 10164 + 850 = 61706$ parameters.

(f) Final training accuracy should be roughly 97.19%, while final testing accuracy should be roughly 96.93%.

Q2