CSCI 4050U, Machine Learning
Test 2, 2021

- Answer all questions.
- **You are not permitted to communicate with others during the test.**
- "I don't know" policy is in effect. If **the only** answer given is "I don't know.", you will receive 20% of the question.

Total: 50

Q1. Convolution

Now, consider the kernel ***K*** of:

1	1
1	1

with the input ***x*** of:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

[5] (1a) Compute the convolution: K * x. Assume no padding is used.

```
Unset
Your answer goes here…
TL = 1+2+5+6 = 14
TM = 2+3+7+7 = 18
TR = 3+4+7+8 = 22
ML = 5+6+9+10 = 30
MM = 6+7+10+11 = 34
MR = 7+8+11+12 = 38
BL = 9+10+13+14 = 46
```

```
BM = 10+11+14+15 = 50
BR = 11+12+15+16 = 54

Therefore:
K*x = [14  18  22
       30  34  38
       46  50  54]
```

[5] (1b) With $y = \sum_i \sum_j (K * x)[i, j]$, compute the gradient of y with respect to K.

Unset

```
Your answer goes here...
[54  63
 90  99]
```

[5] (1c) If we want to minimize y by changing just **one** of the entries in K, which of *K*[*i, j*] would you choose to change? Explain your choice.

Unset

```
Your answer goes here…
We would change the last entry in K. If we look at the input x,
we can see that the largest number is always the last in the 2x2.
Which means if we were to change the last entry in K, we would
end up minimizing y. Another way we can see this is by looking at
the gradient with respect to K, we see that w4 is the heaviest
weight taken into account.
```

Q2. Recurrent Neural Networks

Consider a recurrent neural network defined by the following state update function:

$$h(s_{i-1}, x_i) = As_{i-1} + Bx_i$$

where: $s_i \in R^2$ and $x_i \in R^2$.

Suppose the input is a sequence of length 3, given by.

$x_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, x_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, x_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Let the model parameters be initialized to:

$A = \begin{bmatrix} 0.3 & 0.4 \\ -1 & 0.5 \end{bmatrix}$, and $B = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$

Let the initial condition be defined as $s_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

[10] (2a) Compute the sequence of states: s1, s2, s3. Show how you computed them.

```
Unset
Your answer goes here...
S1 = A*S_0 + B*x_1 =
[0.3  0.4 * [0  + [-1 0  * [1
-1    0.5]   0]    0 -1]    1]
S1 = [-1
      -1]

S2 = [0.3  0.4 * [-1  + [-1 0  * [1
     -1    0.5]   -1]    0 -1]    1]
S2 = [-1.7
      0.5]

S3 = [0.3  0.4 * [-1.7  + [-1 0  * [1
     -1    0.5]   -0.5]    0 -1]    1]
```

```
S3 = [-1.71
       0.45]
```

[10] (2b) Suppose the loss function is defined on the final state:

$loss = s_3[0]^2 + s_3[1]^2 \in R$

What is the gradient of $\nabla loss$ with respect to A and B? Show how you computed them.

Unset

```
Your answer goes here...
I dont know
```

Q3. Applied Machine Learning

Suppose that we are to design a mobile OS component to improve app loading speed while extending the battery life of the phone. We will use a neural network as part of the solution. The general strategy is to predict for the next 60 seconds if an application will be used. Using the prediction, the mobile OS will decide whether to load the app into the main memory or not. Our hope is that the neural network can learn the user patterns so that only apps that are used will be loaded.

[5] (3a) Describe the training data, and how it can be collected.

Unset

```
Your answer goes here…
The training data could be the phone's activity logs, and app
usage information. We will need the history of apps used over
time and how long they were used for. We could also collect data
such as the phone's location and area to check if an app is going
to be used for an extended period of time, for example if a phone
```

```
is detected at a movie theater during busy hours, we can assume
that the user is not really going to make a long phone call. The
targets that we will use for the loss function could be the
actual real time the user spent in the app. The input to the
model could be the location of the device, and time.
```

[5] (3b) Describe the neural network design.

Unset

```
Your answer goes here...
class MyModel(nn.Module):
      def __init__(self,input_dim, hidden_dim, output_dim):
            self.lstm = nn.LSTM(input_dim, hidden_dim)
            self.linear = nn.Linear(hidden_dim, output_dim)
            self.activation = nn.Sigmoid()
      def forward(self, x):
            y, (s, c) = self.lstm(x)
            out = self.linear(c[0])
            out = self.activation(out)
            return out
```

[5] (3c) Describe how the neural network should be trained and evaluated.

Unset

```
Your answer goes here...
The neural network will be trained using a standard training
loop, we can use an optimizer to optimize the model parameters
and use a loss function. The loss function will be used to
calculate the loss between the predicted value (whether to load
the app or not). We can evaluate the model by setting aside 20%
```

of the dataset as testing data, this data will not be seen by the model during training and will only be used to calculate the accuracy of the model.