Q1.

[2] (a) An LSTM's W_c matrix has dimensions 50 by 7.

(i) How many inputs are going in per time slot?

(ii) What is the shape of the input gating vector?

(i) 7

(ii) 50

[2](b) This question builds on the notation we used in Week One Slides. Inputs are column vectors. Outputs from any layer is also a column vector. We will use the same notation in the exam.

Output of a particular 2-layer ANN is given by $yy = WW[\sigma(Wx + b)] + bb$, where σ is the rectilinear activating function acting on the column vector $Wx + b$ point-wise to produce a column vector of the same shape as $Wx + b$. yy represents the pre-activation outputs or Logits. There are 7 input variables and 4 classes. The first layer has 1000 neurons.

(i) What would be the shape of W?

(ii) At a moment when all the biases are zero, and all matrices contain 0.1s for their elements, find the vector yy, if all the inputs are ones.

(i) (1000, 7)

(ii) $Wx = \begin{bmatrix} 0.7 \\ \vdots \\ 0.7 \end{bmatrix} \updownarrow$ 1000 rows

$\sigma(Wx+b) = \begin{bmatrix} 0.7 \\ \vdots \\ 0.7 \end{bmatrix} \updownarrow$ 1000 rows

$yy = \underbrace{\begin{bmatrix} 0.1 & \cdots & 0.1 \\ 0.1 & \ddots & \\ 0.1 & & \ddots \\ 0.1 & & \end{bmatrix}}_{\text{1000 columns}} \updownarrow \text{4 rows} \cdot \begin{bmatrix} 0.7 \\ \vdots \\ 0.7 \end{bmatrix} \updownarrow \text{1000 rows} = \begin{bmatrix} 70 \\ 70 \\ 70 \\ 70 \end{bmatrix}$

[2](c) CBOW method is being used to obtain word embeddings, where output Neuron-*n* represents the *n*th word in the dictionary. The output of the ANN is given by $yy = Wxx$, where, W has shape 12345 by 234567, and xx is the output vector from the previous layer after activation. At the end of the training, W is given by $w(i,j) = (0.1)(2i+3j)$, with Python indexing.

(i) How many words were there in the dictionary?

(ii) If *Apple* is the 3rd (human language) word in the dictionary, what would be the word embedding vector $x(k)$ for *Apple*?

(i) 12345

(ii) $x(k) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow$ only 3^{rd} row $=1$, otherwise $=0$.

[1] (d) Suppose the sentence "word-0 word-1 word-2 word-3 word-4" is at the input to a Transformer. If the position-encoded vector for word-3 is given by p, find $p[3]$.

$k=3$

$p[3] = \cos\left(\frac{k}{10}\right) = \cos\left(\frac{3}{10}\right) = 0.955$

[2] (e) You and your friend work for the Canadian military and there is a war on. Every day you are responsible for transmitting a message about a recurring event whose probability distribution is $[0.6, 0.4]$. Your friend is responsible for transmitting a message about a different recurring event whose probability distribution is $[0.1, 0.1, 0.8]$. The war drags on. Over a long run, other things being equal, who will end up using more bandwidth?

Entropy of my event: $-0.6\log_{10}(0.6) - 0.4\log_{10}(0.4) = 0.292$

Entropy of my friend's event: $2\times[-0.1\cdot\log_{10}(0.1)] - 0.8\cdot\log_{10}(0.8) = 0.278$

my entropy > my friend's entropy

$\Rightarrow$ I transmit more information

$\Rightarrow$ I use more bandwidth.

[2] (f) A 9/100 - 1 ANN designed for function approximation with rectilinear activations is being trained. At the moment, all the inputs, weights, and biases have the same value 1, except for one bias on the first layer. If the output of the network is 991, find the highest possible value of that bias.

first layer common result: $\text{ReLu}(9\times(1\times1)+1) = 10$ # 99 of those.

"outlier" result: $\text{ReLu}(9\times(1\times1)+b) = \begin{cases} 9+b & \# \ b>-9 \\ 0 & \# \ b\leq -9. \end{cases}$

$y = 99\times 10 + 1\times\text{"outlier"} + 1$

$991 = \text{ReLu}(991 + 1\times\text{"outlier"})$

$\Rightarrow$ outlier result $= 0 \Rightarrow b\leq -9$

$\Rightarrow \max(b) = -9.$

[3] (g) Consider the condominium example in the GNN slides, but simplify the example and assume there were only 4 people in the condo and each is passionate about only 2 issues.

(i) What would be the shape of the state matrix H?

(ii) What would be the shape of the weight matrix W?

(iii) At the beginning of a particular iteration during training, the state matrix H is given by $h(i,j) = i-(0.1)j$, and the weight matrix W is given by $w(i,j) = (0.1)(i+j)$. The indexing starts at 1.

Find the state matrix at the end of this iteration, if the adjacency matrix A is given by:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Do not normalize, but remember to use A + I.

(i) $(4,2)$

(ii) $(2,2)$

(iii) $H = \underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}}_{(A+I)} \begin{bmatrix} 0.9 & 0.8 \\ 1.9 & 1.8 \\ 2.9 & 2.8 \\ 3.9 & 3.8 \end{bmatrix} \begin{bmatrix} 0.2 & 0.3 \\ 0.3 & 0.4 \end{bmatrix} = \begin{bmatrix} 2.76 & 3.87 \\ 2.76 & 3.87 \\ 4.68 & 6.56 \\ 3.34 & 4.68 \end{bmatrix}$

[2.7] (h) Featuremap X is given by $x_{i,j} = i + j$. A 7 by 7 transpose convolutional filter W, given by $w_{i,j} = -i + 2j$, operates on X to generate up-sampled map Y with stride 2. Find $y_{1,1} + y_{3,1}$.

As always, in image related questions, the indexing starts at 1.

$$y_{1,1} = x_{1,1} \cdot w_{1,1} = 2 \cdot 1 = 2$$

$$y_{3,1} = x_{1,1} \cdot W_{3,1} + X_{2,1} \cdot W_{1,1} = 2 \cdot (-1) + 3 \cdot 1 = 1$$

$$y_{1,1} + y_{3,1} = 2 + 1 = 3$$

Q2.

[14] (a) Consider a 2/2 - 1 ANN designed for regression with rectilinear activating functions that is ready for inference. A look at the parameters reveals that:

$$W = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \qquad WW = \begin{bmatrix} -1 & 2 \end{bmatrix} \qquad bb = 3$$

We know that this ANN is an implementation of a certain piece-wise-linear multivariable function $f(x_1, x_2)$. Find the algebraic expression that defines this function. That is, express $f(x_1, x_2)$ in terms of x_1 and x_2.

$$y = WW\left(\sigma(Wx + b)\right) + bb$$

$$= [-1 \quad 2]\,\sigma\left(\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \end{bmatrix}\right) + 3$$

$$= [-1 \quad 2]\,\sigma\left(\begin{bmatrix} x_1 - x_2 + 1 \\ x_1 + x_2 - 2 \end{bmatrix}\right) + 3$$

A: $x_1 - x_2 + 1 \geq 0$, $x_1 + x_2 - 2 \geq 0$

$\Rightarrow$ $x_1 \geq x_2 - 1$, $x_1 \geq 2 - x_2$,

$y = -(x_1 - x_2 + 1) + 2(x_1 + x_2 - 2) + 3 = x_1 + 3x_2 - 2$

B: $x_1 - x_2 + 1 \geq 0$, $x_1 + x_2 - 2 < 0$

$\Rightarrow$ $x_1 \geq x_2 - 1$, $x_1 < 2 - x_2$

$y = -(x_1 - x_2 + 1) + 3 = -x_1 + x_2 + 2$

C: $x_1 - x_2 + 1 < 0$, $x_1 + x_2 - 2 \geq 0$

$\Rightarrow$ $x_1 < x_2 - 1$, $x_1 \geq 2 - x_2$

$y = 2(x_1 + x_2 - 2) + 3 = 2x_1 + 2x_2 - 1$

D: $x_1 - x_2 + 1 < 0$, $x_1 + x_2 - 2 < 0$

$\Rightarrow$ $x_1 < x_2 - 1$, $x_1 < 2 - x_2$

$y = 3$

[2.7] (b) Suppose $y(i+2) = ay(i+1) + by(i) + 1,\ \ i = 1, 2, 3, \ldots.$

Write down a Python script that will find $\dfrac{\partial y(1000)}{\partial b}$. Make the necessary assumptions (as little as possible) about some of the initial values, before writing your script.

```
# Assumptions

a = 1

b = 1

y = [0] * 1000

y[0] = 1          # y(1) = 1

y[1] = 1          # y(2) = 1


# loop.

result = [0] * 1000

for i in range (2, 1000):

    y[i] = a*y[i-1] + b*y[i-2] + 1

    result[i] = a * result[i-1] + y[i-2]

print (result[999])
```