College of Engineering
Computer Science & Eng. Dept.
Course: COE 49412 Neural Networks & Deep Learning

Course Instructor: Dr. Imran Zualkernan
Email: izualkernan@aus.edu
Office: ESB-2066
Lab Instructor: Ms Hend ElGhazaly
Email: helghazaly@aus.edu
Office: ESB-1043 B

Homework 2 - Exploring Neural Network Forward and Backward Propagations

COE49412 - Spring 2020

Total Marks: 10
Due Date: Monday, 23-Mar-2020, 11.59pm

Please enter your Student ID & Name below:

In [ ]:
```
## Student ID:
## Student Name:
```

Exercise 1 [2 marks]

*Consider the following simple 2-layer neural network architecture containing one input, one hidden layer with a single node, and one output. By using a step-by-step approach by hand, calculate the results of a **single iteration** of **Forward and Backward propagation** and print **1)** the predicted value, a_2, of the network after the first forward pass, and **2)** the updated values of the parameters **weights** and **biases** after the first backward pass. You may refer to your lecture notebook for examples.*

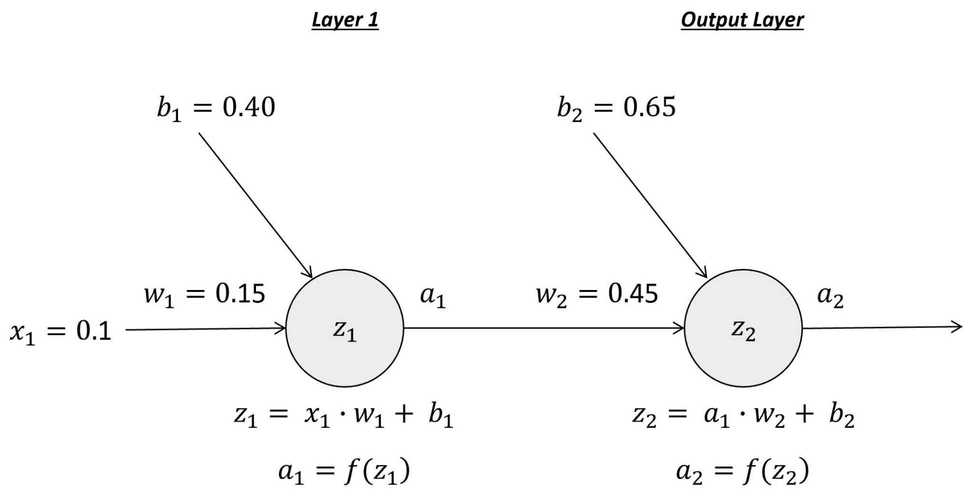

***Figure 1**: 2-layer neural network. Note: notations shown differ from your lecture notes.*

***Instructions**:*

- *For this exercise, use the **activation** function $f(z)$ as the **sigmoid** function:* $\sigma(z) = \frac{1}{1+e^{(-z)}}$
  - *Note the derivative of the sigmoid function* $\sigma'(z) = \sigma(z) \cdot (1 - \sigma(z))$
- *Assume the model is fed with only **one input data point** with the actual ground truth value* $y = 0.25$ *for the given input* $x = 0.1$
- *Use the **Loss** function as:* $Loss = \frac{1}{2}(y - a_2)^2$
  - *Note the partial derivative of the* $Loss$ *function* $\frac{\partial Loss}{\partial a_2} = -(y - a_2)$
- *Use the **learning rate*** $\eta = 0.4$
- *The initial **weights**, w, are given as:* $[\, w_1 = 0.15 \quad w_2 = 0.45 \,]$
- *The initial **biases**, b, are given as:* $[\, b_1 = 0.40 \quad b_2 = 0.65 \,]$

***Hints**:*

For backward pass, you may want to recall that using Gradient descent (refer your lecture notes):

- $w_{1\ updated} = w_1 - \eta \frac{\partial Loss}{\partial w_1}$
- $w_{2\ updated} = w_2 - \eta \frac{\partial Loss}{\partial w_2}$
- $b_{1\ updated} = b_1 - \eta \frac{\partial Loss}{\partial b_1}$
- $b_{2\ updated} = b_2 - \eta \frac{\partial Loss}{\partial b_2}$
- *And using the chain rule:*
  - $\frac{\partial Loss}{\partial w_2} = \frac{\partial Loss}{\partial a_2} \cdot \frac{\partial a_2}{\partial z_2} \cdot \frac{\partial z_2}{\partial w_2} = (-(y - a_2)) \cdot (a_2(1 - a_2)) \cdot a_1$
  - $\frac{\partial Loss}{\partial b_2} = \frac{\partial Loss}{\partial a_2} \cdot \frac{\partial a_2}{\partial z_2} \cdot \frac{\partial z_2}{\partial b_2} = (-(y - a_2)) \cdot (a_2(1 - a_2)) \cdot 1$
  - $\frac{\partial Loss}{\partial w_1} = \frac{\partial Loss}{\partial a_2} \cdot \frac{\partial a_2}{\partial z_2} \cdot \frac{\partial z_2}{\partial a_1} \cdot \frac{\partial a_1}{\partial z_1} \cdot \frac{\partial z_1}{\partial w_1} = (-(y - a_2)) \cdot (a_2(1 - a_2)) \cdot (w_2) \cdot (a_1(1 - a_1)) \cdot x_1$
  - $\frac{\partial Loss}{\partial b_1} = \frac{\partial Loss}{\partial a_2} \cdot \frac{\partial a_2}{\partial z_2} \cdot \frac{\partial z_2}{\partial a_1} \cdot \frac{\partial a_1}{\partial z_1} \cdot \frac{\partial z_1}{\partial b_1} = (-(y - a_2)) \cdot (a_2(1 - a_2)) \cdot (w_2) \cdot (a_1(1 - a_1)) \cdot 1$

In [ ]:
```
### ENTER YOUR CODE HERE ###
```

Expected Output(s):

a_2	*0.7153*
$w_{1\,updated}$	*0.1496*
$w_{2\,updated}$	*0.4272*
$b_{1\,updated}$	*0.3959*
$b_{2\,updated}$	*0.6121*

Exercise 2 [3 marks]

*Consider now a slightly larger variation of the above neural network architecture containing two inputs, one hidden layer with three nodes, and one output. By using a similar step-by-step approach by hand, calculate the results of a **single iteration** of **Forward and Backward propagation** and print **1)** the predicted value, a_4, of the network after the first forward pass, and **2)** the updated values of the parameters **weights** and **biases** after the first backward pass. You may refer to your lecture notebook for examples.*

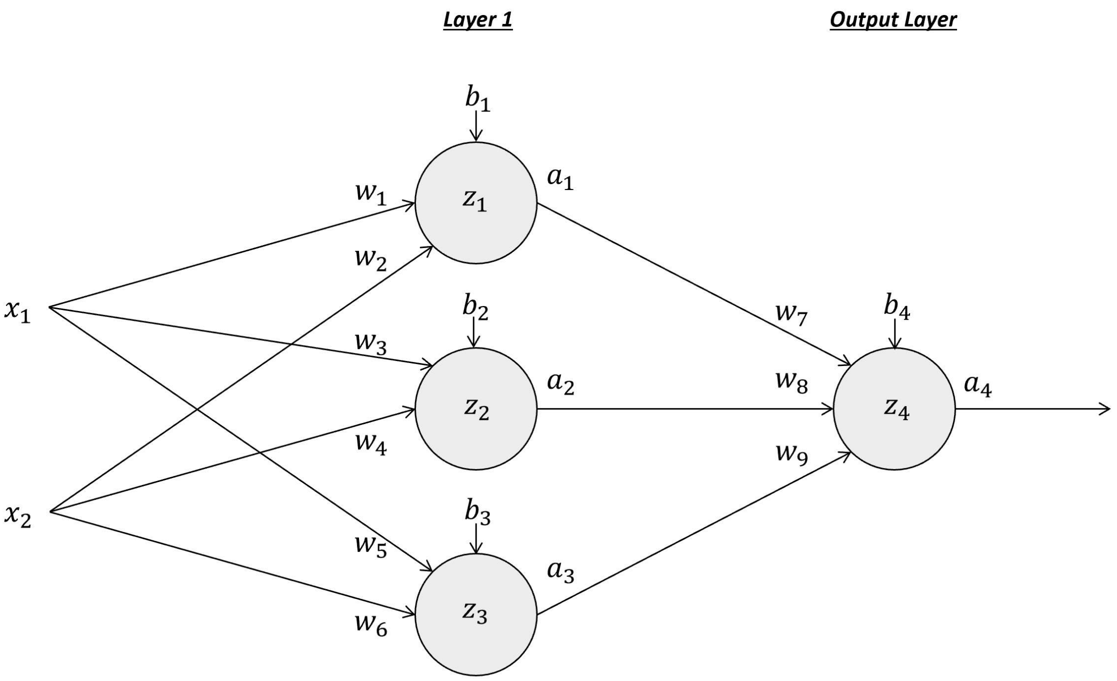

***Figure 2**: Neural network with 2 inputs, 1 hidden layer with 3 nodes, and 1 output.*
Note: notations shown differ from your lecture notes.

***Instructions**:*

- *For this exercise, use the **activation** function $f(z)$ as the **sigmoid** function:* $\sigma(z) = \frac{1}{1+e^{(-z)}}$
  - *Note the derivative of the sigmoid function* $\sigma'(z) = \sigma(z) \cdot (1 - \sigma(z))$
- *Assume the model is fed with only **one input data point** with the actual ground truth value* $y = 0.25$ *for the given inputs* $x_1 = 0.1$ *and* $x_2 = 0.35$
- *Use the **Loss** function as:* $Loss = \frac{1}{2}(y - a_4)^2$
  - *Note the partial derivative of the* $Loss$ *function* $\frac{\partial Loss}{\partial a_4} = -(y - a_4)$
- *Use the **learning rate*** $\eta = 0.4$
- *The initial **weights**,* w*, are given as:*
  $[\, w_1 = 0.15 \quad w_2 = 0.45 \quad w_3 = -0.35 \quad w_4 = -0.61 \quad w_5 = -0.52 \quad w_6 = 0.22 \quad w_7 = -0.72 \quad w_8 = 1$
- *The initial **biases**,* b*, are given as:* $[\, b_1 = 0.40 \quad b_2 = 0.65 \quad b_3 = 0.23 \quad b_4 = 0.15\,]$

In [ ]:
```
### ENTER YOUR CODE HERE ###
```

Expected Output(s):

a_4	*0.5625*

$$W_{Layer\,1\,updated} = \begin{bmatrix} 0.1505 \\ 0.4518 \\ \text{-}0.3509 \\ \text{-}0.6131 \\ \text{-}0.5198 \\ 0.2208 \end{bmatrix} \quad W_{OutputLayer\,updated} = \begin{bmatrix} \text{-}0.7397 \\ 1.202 \\ \text{-}0.3173 \end{bmatrix}$$

$$b_{Layer\,1\,updated} = \begin{bmatrix} 0.4051 \\ 0.6410 \\ 0.2323 \end{bmatrix} \quad b_{OutputLayer\,updated} = \begin{bmatrix} 0.1192 \end{bmatrix}$$

Exercise 3 [3 marks]

Consider now a more general version of the above neural network architecture containing two inputs, two hidden layers, and two outputs. By using a generalized approach, calculate the results of a ***single iteration*** *of* ***Forward and Backward propagation*** *and print* ***1)*** *the predicted values, a_8 and a_9, of the network after the first forward pass, and* ***2)*** *the updated values of the parameters* ***weights*** *and* ***biases*** *after the first backward pass. You may refer to your lecture notebook for examples.*

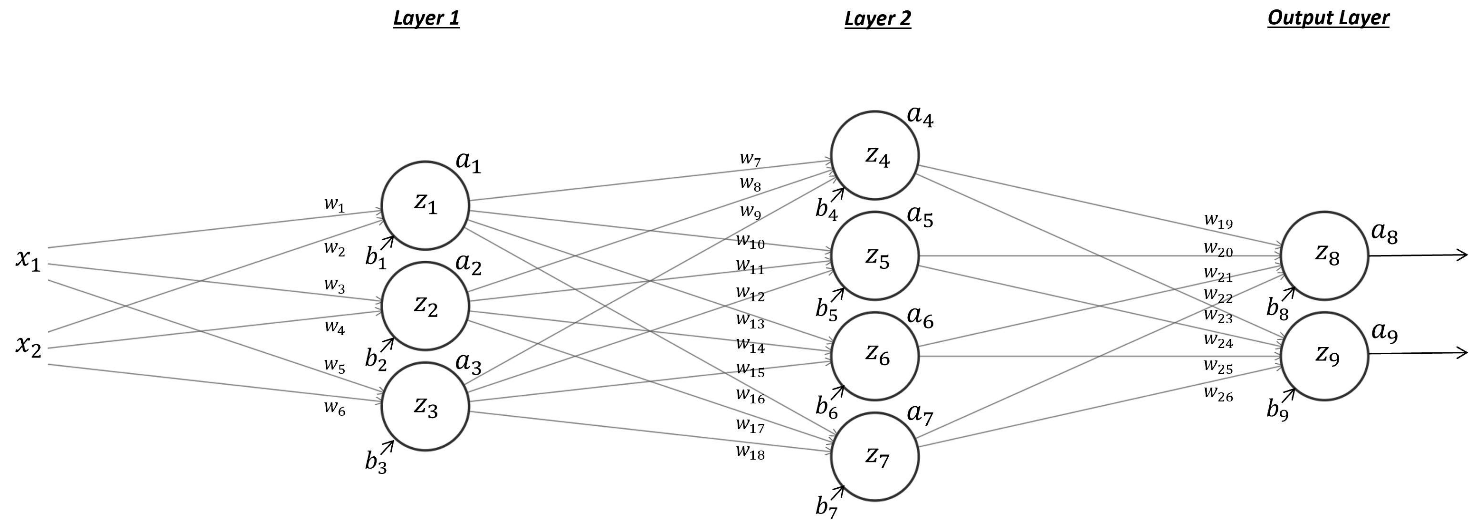

Figure 3*: Neural network with 2 inputs, 2 hidden layers, and 2 outputs.*
Note: notations shown differ from your lecture notes.

Instructions*:*

- *For this exercise, use the* ***activation*** *function $f(z)$ as the* ***sigmoid*** *function: $\sigma(z) = \frac{1}{1+e^{(-z)}}$*
  - *Note the derivative of the sigmoid function $\sigma'(z) = \sigma(z) \cdot (1 - \sigma(z))$*
- *Assume the model is fed with only* ***one input data point*** *with the actual ground truth values $y_1 = 0.25$ and $y_2 = 0.4$ for the given inputs $x_1 = 0.1$ and $x_2 = 0.35$*
- *Use the* ***Loss*** *function as: $Loss = \frac{1}{2}(y - a_{output})^2$*
  - *Note the partial derivative of the $Loss$ function $\frac{\partial Loss}{\partial a_{output}} = -(y - a_{output})$*
- *Use the* ***learning rate*** *$\eta = 0.4$*
- *The initial* ***weights****, w, are given as:*
  - $W_{Layer\ 1:\ w_1 \text{ to } w_6} = [0.45 \quad 0.83 \quad 0.62 \quad 0.84 \quad 0.22 \quad 0.86]^T$
  - $W_{Layer\ 2:\ w_7 \text{ to } w_{18}} = [0.78 \quad 1.03 \quad 0.43 \quad 0.98 \quad 0.39 \quad 0.42 \quad -0.05 \quad 1.11 \quad 0.99 \quad 0.79 \quad 0.52 \quad 1.08]$
  - $W_{Output\ Layer:\ w_{19} \text{ to } w_{26}} = [0.79 \quad 0.62 \quad -0.32 \quad 0.65 \quad 0.52 \quad -0.78 \quad 0.41 \quad 0.91]^T$
- *The initial* ***biases****, b, are given as:*
  - $b_{Layer\ 1:\ b_1 \text{ to } b_3} = [0.1 \quad 1.03 \quad 0.59]^T$
  - $b_{Layer\ 2:\ b_4 \text{ to } b_7} = [0.47 \quad 0.29 \quad -0.51 \quad 1.15]^T$
  - $b_{Output\ Layer:\ b_8 \text{ to } b_9} = [0.92 \quad -0.6]^T$

In [3]:
```
### ENTER YOUR CODE HERE ###
```

Expected Output(s):

a_8	0.9244
a_9	0.5956

$$W_{Layer\,1\,updated} = \begin{bmatrix} 0.4499 \\ 0.8297 \\ 0.6199 \\ 0.8398 \\ 0.2199 \\ 0.8597 \end{bmatrix} \quad W_{Layer\,2\,updated} = \begin{bmatrix} 0.7785 \\ 1.0280 \\ 0.4282 \\ 0.9802 \\ 0.3903 \\ 0.4203 \\ -0.0501 \\ 1.1097 \\ 0.9897 \\ 0.7890 \\ 0.5187 \\ 1.0788 \end{bmatrix} \quad W_{OutputLayer\,updated} = \begin{bmatrix} 0.7732 \\ 0.6046 \\ -0.3339 \\ 0.6322 \\ 0.5032 \\ -0.7954 \\ 0.3960 \\ 0.8922 \end{bmatrix}$$

$$b_{Layer\,1\,updated} = \begin{bmatrix} 0.0993 \\ 1.0294 \\ 0.5894 \end{bmatrix} \quad b_{Layer\,2\,updated} = \begin{bmatrix} 0.4675 \\ 0.2904 \\ -0.5103 \\ 1.1484 \end{bmatrix} \quad b_{OutputLayer\,updated} = \begin{bmatrix} 0.9011 \\ -0.6188 \end{bmatrix}$$

Exercise 4 [2 marks]

*Consider now replacing the **sigmoid activation function** in Exercise 3 with the **softmax activation function**: $f(z) = \frac{e^z}{\sum_i e^z}$ in the **output layer**, and the **ReLU activation function**: $f(z) = max(0, z)$ in the two **hidden layers**. Run your code from Exercise 3 for **50 iterations** and plot the $Loss$ function against the number of iterations. Compare your results with the **sigmoid** activation based hidden layers architecture in Exercise 3 and comment on your observations. You may refer to your lecture notebook for examples.*

Hints:

- *Account for the vanishing gradient problem*
- *Recall that the derivative of the **softmax** function $f(z) = \frac{e^z}{\sum_i e^z}$ is $f'(z) = \frac{e^z}{\sum_i e^z} - \frac{(e^z)^2}{(\sum_i e^z)^2}$*
- *Recall that the derivative of the **ReLU** function $f(z) = max(0, z)$ is $f(z) = \begin{cases} 0 & \text{if } z < 0 \\ 1 & \text{if } z > 0 \end{cases}$*
- *Sample indicative output plot shown below (your values may differ)*

In [2]:
```
### ENTER YOUR CODE HERE ###
```

Expected Output(s):

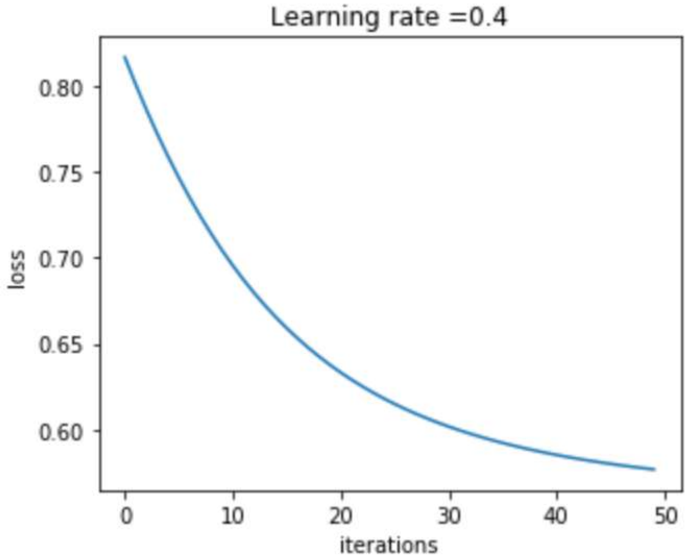

Figure 4: *Sample indicative plot of $Loss$ function vs. number of iterations (your values may differ)*