CSCI 495/595 Mid Term Exam

What is the purpose of forward pass in backpropagation algorithm? **(1 point)**

The purpose of the forward pass is to calculate the outputs of each neuron in the network using their weights and associated inputs. The outputs of the hidden layer neurons are eventually fed into the output layer neurons, which give the final output(s) of the network. These values are then used in the backward pass to determine how much the weights should be adjusted to increase prediction accuracy.

1. How does backward pass reduce the computational complexity in the backpropagation algorithm? **(1 point)**

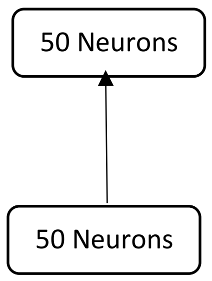
   It computes the gradients used to adjust the weights by propagating the error through the network backwards. This is done by applying the chain rule from calculus and working one layer at a time, which avoids repeated calculations.

   Also, the fact that the activation functions we use in our neural networks (like tanh and sigmoid) refer to themselves in their derivatives simplifies calculations, since we don't need to know the inputs to these functions to find the value of their derivatives.

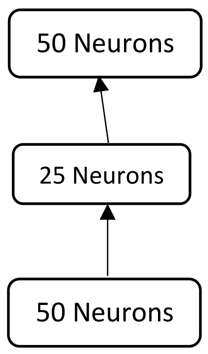
2. What situations are ideal to use tanh as an activation function in the output layer? **(2 points)**

   tanh may be useful in the output layer if you want values centered around 0 and bound between +/-1. The former is useful during training because it may speed up convergence at the beginning of training. The latter is useful if the labels are also bound between +/-1, since the function will output that naturally without any additional processing. It's also similar to the signum function, while being continuous. This makes it useful for single-layer perceptrons, which was discussed in lecture.

3. Consider the following two multilayer perceptron, where all of the layers use linear activation functions. **(2 points)**



Network A



Network B

a. Give one advantage of Network A over Network B

   Network A has fewer neurons overall and fewer layers than network B, which should make it faster and cheaper to train.

b. Give one advantage of Network B over Network A

   Network B has more layers and more neurons, which will make it better at capturing a hierarchy of details. This may allow it to better understand more complex patterns in data compared to network A.

4. Consider a list Var_X contains the entries [2, 5, 15, 9, 25] but the solution requires the output to be within the range of 0 and 1. One of the possible solutions is to use the softmax function to convert the Var_X values within the desired range. Show the steps on how would you convert the values of Var_X using softmax function. **(2 points)**

The softmax function is defined as follows:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

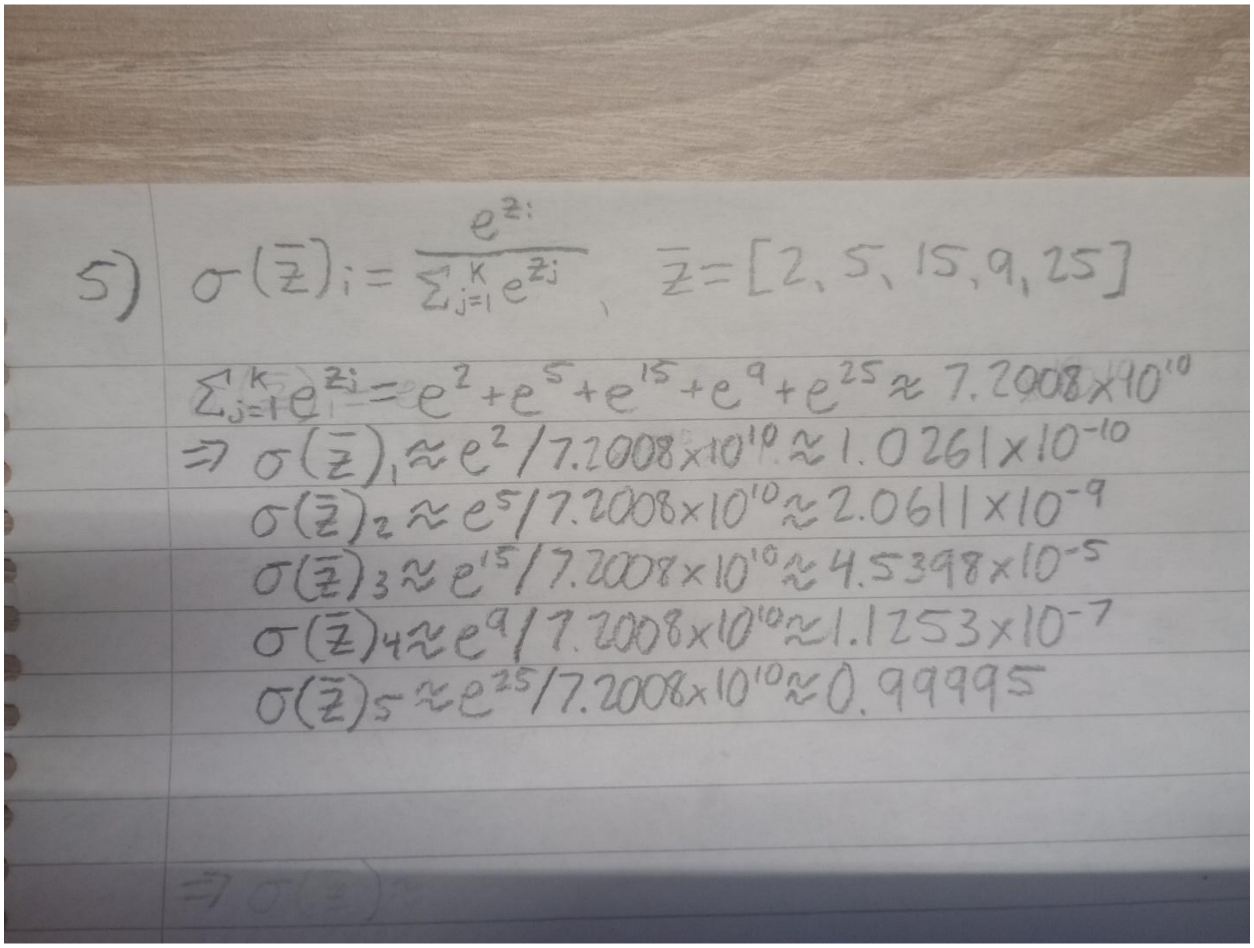
With this in mind, we know that the function takes a vector as input and outputs a new vector of equal size whose contents add up to 1 (barring any problems with underflow). We'll begin by computing the denominator as shown below:

$$d = e^2 + e^5 + e^{15} + e^9 + e^{25} \qquad = 7.20081766 \times 10^{10}$$

Next, we will compute e^z[i] divided by the denominator we just found for each element of the input vector. This results in five values that are seen below:

$\frac{e^2}{d}$	$= 1.02614126 \times 10^{-10}$
$\frac{e^5}{d}$	$= 2.06105981 \times 10^{-9}$
$\frac{e^{15}}{d}$	$= 4.53978635 \times 10^{-5}$
$\frac{e^9}{d}$	$= 1.12530053 \times 10^{-7}$
$\frac{e^{25}}{d}$	$= 0.9999544874$

My paper calculations can also be found below that illustrate the same process shown above:



5) $\sigma(\bar{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$, $\bar{z} = [2, 5, 15, 9, 25]$

$\sum_{j=1}^{K} e^{z_j} = e^2 + e^5 + e^{15} + e^9 + e^{25} \approx 7.2008 \times 10^{10}$

$\Rightarrow \sigma(\bar{z})_1 \approx e^2 / 7.2008 \times 10^{10} \approx 1.0261 \times 10^{-10}$

$\sigma(\bar{z})_2 \approx e^5 / 7.2008 \times 10^{10} \approx 2.0611 \times 10^{-9}$

$\sigma(\bar{z})_3 \approx e^{15} / 7.2008 \times 10^{10} \approx 4.5398 \times 10^{-5}$

$\sigma(\bar{z})_4 \approx e^9 / 7.2008 \times 10^{10} \approx 1.1253 \times 10^{-7}$

$\sigma(\bar{z})_5 \approx e^{25} / 7.2008 \times 10^{10} \approx 0.99995$

5. Give an example of an activation function (other than tanh and log. Sigmoid) which are used in deep learning architecture. You may have to do some research to identify this activation function, and answer the following questions. **(1 x 4 = 4 points)**

a. Mathematical representation and range of this activation function

   A common activation function used today is ReLU, which is defined like so: ReLU (x) = max (0, x). Its range is [0, inf), meaning it can be any non-negative number.

b. Advantages of this activation function

   One advantage of this function is that it and its derivative are incredibly cheap to compute compared to sigmoid or tanh. The fact that it doesn't have a maximum value can also reduce some problems during gradient descent. Finally, it is also good for networks that need to output non-negative values.

c. Derivative of this activation function

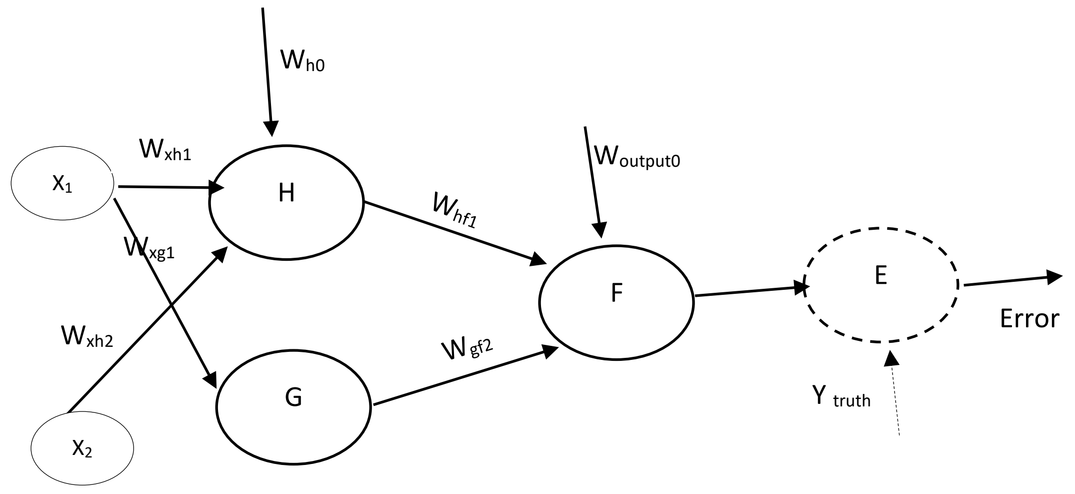
   The derivative of ReLU is 0 when x < 0 and 1 when x > 0. It doesn't technically have a derivative at x = 0, so it is often assumed to be either 0, 0.5, or 1.

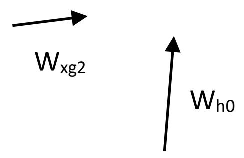
d. A brief description of this function (not more than 3-5 sentences).

   This function is extremely simple. It simply outputs x (the input) directly if it's positive and 0 otherwise.

6. For this question, you have an option to attempt any one of the two networks shown below. There is no bonus attached to attempting both but you can certainly do that for fun if you want.

A. Consider the following feed-forward network below (**Network A**) and answer the following questions. Refer to figure below for this question, X_i are inputs, W_{xyi} are weights from layer x to y for sequence number i, and $\mathbf{W_{h0}}$ **is bias weight** for hidden layer neurons G and H and their activation function is tanh. The neuron F in the output layer uses logistic sigmoid as activation function. Bias weight for output layer is $\mathbf{W_{output0}}$

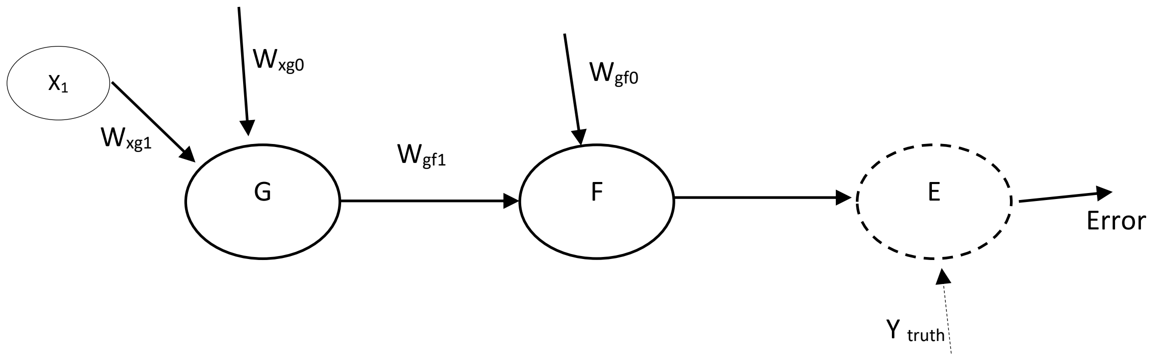




NETWORK A

Parameters for Network A		
Initial weights	Training examples	Learning rate
a. $W_{h0} = 0.4$ b. $W_{xh1} = 0.5$ c. $W_{xh2} = -0.2$ d. $W_{xg1} = 0.2$ e. $W_{xg2} = -0.1$ f. $W_{output0} = 0.1$ g. $W_{hf1} = -0.1$ h. $W_{gf2} = 0.45$	i. $X_1 = -0.5$ j. $X_2 = 0.3$ k. $Y_{truth} = 1.0$	**Learning rate** = 0.1

B. Consider the following feed-forward network below (**Network B**) and answer the following questions. Refer to figure below for this question, X_i are inputs, W_{xyi} are weights from layer x to y for sequence number i, and **W_{xg0} and W_{gf0} is bias weight** for hidden and output layer neurons G and F respectively, and both hidden layer and output layer uses logistic sigmoid as their activation function.



NETWORK B

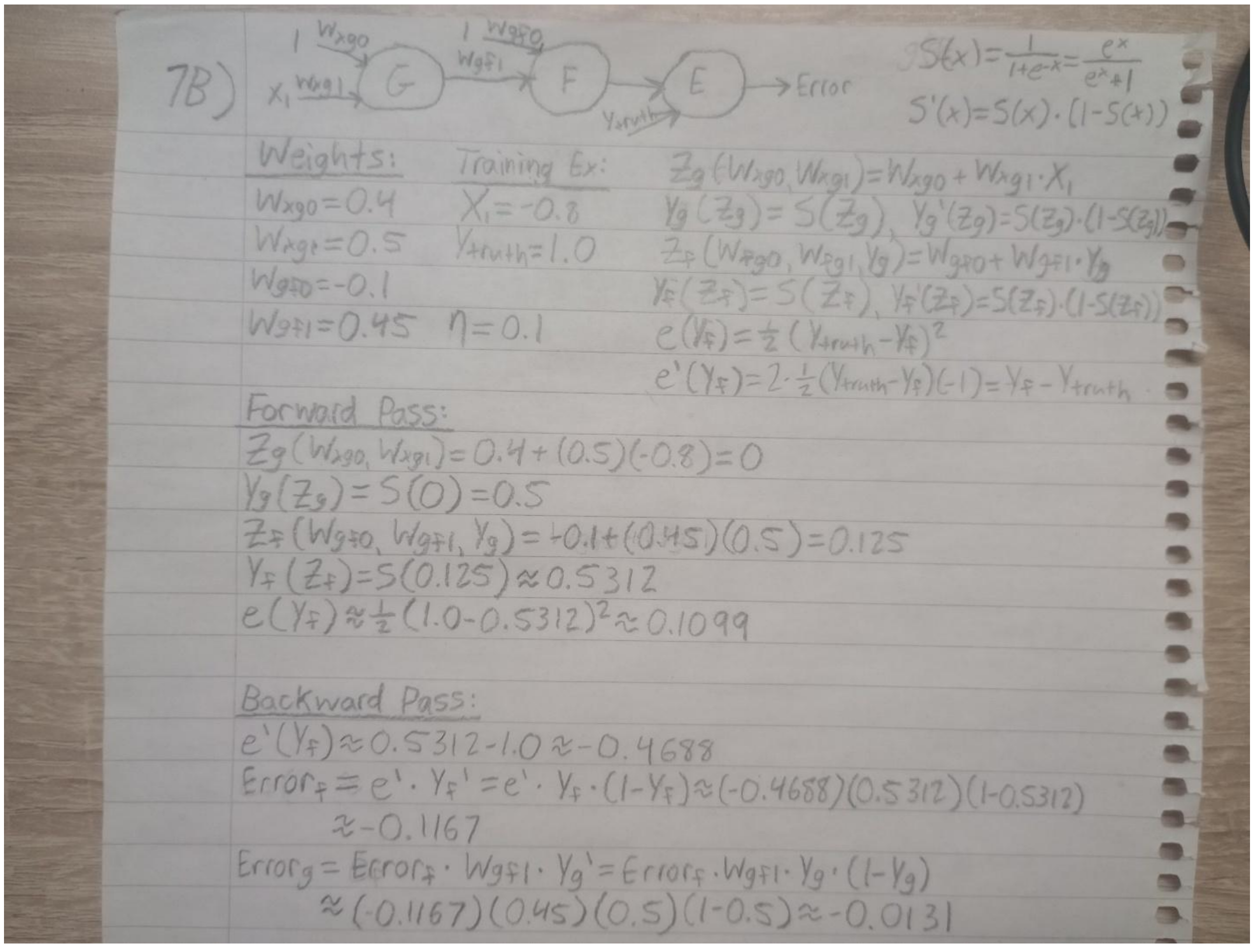
Parameter for NETWORK B

Initial weights	Training examples	Learning rate
a. $W_{xg0} = 0.4$	e. $X_1 = -0.8$	**Learning rate** = 0.1
b. $W_{xg1} = 0.5$	f. $Y_{truth} = 1.0$	
c. $W_{gf0} = -0.1$		
d. $W_{gf1} = 0.45$		

For the network of your choice, attempt the following questions:

a. Show steps and calculations for forward pass **(3 points)**
b. Show steps and calculations for backward pass **(5 points)**

My calculations for both passes can be found below:



7B)

$$\begin{tikzcd}
1 \arrow[dr, "W_{xg0}"] & & 1 \arrow[dr, "W_{gf0}"] & & & \\
X_1 \arrow[r, "W_{xg1}"] & G \arrow[rr, "W_{gf1}"] & & F \arrow[r] & E \arrow[r] & \text{Error} \\
& & & Y_{truth} \arrow[ur] & &
\end{tikzcd}$$

$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$

$S'(x) = S(x)\cdot(1-S(x))$

Weights:
$W_{xg0} = 0.4$
$W_{xg1} = 0.5$
$W_{gf0} = -0.1$
$W_{gf1} = 0.45$

Training Ex:
$X_1 = -0.8$
$Y_{truth} = 1.0$
$\eta = 0.1$

$Z_g(W_{xg0}, W_{xg1}) = W_{xg0} + W_{xg1}\cdot X_1$

$Y_g(Z_g) = S(Z_g)$, $Y_g'(Z_g) = S(Z_g)\cdot(1-S(Z_g))$

$Z_f(W_{gf0}, W_{gf1}, Y_g) = W_{gf0} + W_{gf1}\cdot Y_g$

$Y_f(Z_f) = S(Z_f)$, $Y_f'(Z_f) = S(Z_f)\cdot(1-S(Z_f))$

$e(Y_f) = \frac{1}{2}(Y_{truth} - Y_f)^2$

$e'(Y_f) = 2\cdot\frac{1}{2}(Y_{truth} - Y_f)(-1) = Y_f - Y_{truth}$

Forward Pass:

$Z_g(W_{xg0}, W_{xg1}) = 0.4 + (0.5)(-0.8) = 0$

$Y_g(Z_g) = S(0) = 0.5$

$Z_f(W_{gf0}, W_{gf1}, Y_g) = -0.1 + (0.45)(0.5) = 0.125$

$Y_f(Z_f) = S(0.125) \approx 0.5312$

$e(Y_f) \approx \frac{1}{2}(1.0 - 0.5312)^2 \approx 0.1099$

Backward Pass:

$e'(Y_f) \approx 0.5312 - 1.0 \approx -0.4688$

$Error_f = e'\cdot Y_f' = e'\cdot Y_f\cdot(1-Y_f) \approx (-0.4688)(0.5312)(1-0.5312)$
≈ -0.1167

$Error_g = Error_f\cdot W_{gf1}\cdot Y_g' = Error_f\cdot W_{gf1}\cdot Y_g\cdot(1-Y_g)$
$\approx (-0.1167)(0.45)(0.5)(1-0.5) \approx -0.0131$

You don’t need to include the weight adjustment for this problem. You will be awarded points based on correctness and the calculation steps for this question.