Homework 2

David A. Halloran

January 23, 2022

Theory

1. Compute gradients when using the following matrix as input $h = \begin{bmatrix} 1 & 2 & 3 & 4 \end{bmatrix}$ Remeber: For every function except for Softmax, it gives an identity matrix with the calculated answer on the diagonal and zeros on the spots where index i $\neq$ index j. So we will have that disclaimer here instead of writing it each time. All answers will return a 4x4 matrix

   (a) Relu Layer:

   When i = j, Relu returns 1 if input value is ≥ 0 or returns 0 if input value < 0. Since all elements in h are positive this is the same as a regular linear function which just returns an identity matrix. So for input h it will return:

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

   (b) Softmax:

   As stated above Softmax is out exception. So when i = j it returns the result of:

$$g(h) * (1 - g(h))$$

   When i $\neq$ j however we can determine those values in the matrix with the formula

$$-g_i(h) * g_j(h)$$

   We will iterate through i and j in the same format as a nested for loop, so i = 1, j = 1 $\rightarrow$ 2 $\rightarrow$ 3 $\rightarrow$ 4. Increment i and repeat process. As with all gradients these are dependant on the outputs of the activation functions. So let's calculate the outputs:

$$g_1 = \frac{e^1}{e^1 + e^2 + e^3 + e^4} = 0.03103085$$

$$g_2 = \frac{e^2}{e^1 + e^2 + e^3 + e^4} = 0.08714431874$$

$$g_3 = \frac{e^3}{e^1 + e^2 + e^3 + e^4} = 0.23688282$$

$$g_4 = \frac{e^4}{e^1 + e^2 + e^3 + e^4} = 0.64391426$$

With this info, let's manually fill out the gradient matrix for the first row and then show the final product:

$$i = 1, j = 1$$

$$g_1 * (1 - g_1) = 0.03103085 * (1 - 0.03103085) = 0.03103085$$

$$i = 1, j = 2$$

$$-g_1 * g_2 = -0.03103085 * 0.08714431874 = -0.00279373$$

$$i = 1, j = 3$$

$$-g_1 * g_3 = -0.03103085 * 0.23688282 = -0.00759413$$

$$i = 1, j = 4$$

$$-g_1 * g_4 = -0.03103085 * 0.23688282 = -0.02064299$$

Repeating this process for each combination of i and j, we get the following matrix:

$$\begin{bmatrix} 0.03103085 & -0.00279373 & -0.00759413 & -0.02064299 \\ -0.00279373 & 0.07955019 & -0.02064299 & -0.05611347 \\ -0.00759413 & -0.02064299 & 0.18076935 & -0.15253222 \\ -0.02064299 & -0.05611347 & -0.15253222 & 0.22928869 \end{bmatrix}$$

(c) Sigmoid:
Calculate values from activation:

$$g(z) = \frac{1}{1 + e^-z}$$

$$g_1 = \frac{1}{1 + e^-1} = 0.73105858$$

$$g_2 = \frac{1}{1 + e^-2} = 0.88079708$$

$$g_3 = \frac{1}{1+e^-3} = 0.95257413$$

$$g_4 = \frac{1}{1+e^-4} = 0.98201379$$

Where i = j do $g(z) * (1 - g(z)) + \epsilon$ where $\epsilon = .0000001$
Else, return 0

$$i = 1, j = 1$$

$$g_1 {*} (1 - g_1) + \epsilon = 0.73105858 {*} (1 - 0.73105858) + .0000001 = 0.19661203$$

$$i = 2, j = 2$$

$$g_2 {*} (1 - g_2) + \epsilon = 0.88079708 {*} (1 - 0.88079708) + .0000001 = 0.10499369$$

$$i = 3, j = 3$$

$$g_3 {*} (1 - g_3) + \epsilon = 0.95257413 {*} (1 - 0.95257413) + .0000001 = 0.04517676$$

$$i = 4, j = 4$$

$$g_4 {*} (1 - g_4) + \epsilon = 0.98201379 {*} (1 - 0.98201379) + .0000001 = 0.01766281$$

This give a final matrix of:

$$\begin{bmatrix} 0.19661203 & 0 & 0 & 0 \\ 0 & 0.10499369 & 0 & 0 \\ 0 & 0 & 0.04517676 & 0 \\ 0 & 0 & 0 & 0.01766281 \end{bmatrix}$$

(d) TanH
Calculate values from activation:

$$g(z) = \frac{e^z - e^-z}{e^z + e^-z}$$

$$g_1 = \frac{e^1 - e^-1}{e^1 + e^-1} = 0.76159416$$

$$g_2 = \frac{e^2 - e^-2}{e^2 + e^-2} = 0.96402758$$

$$g_3 = \frac{e^3 - e^-3}{e^3 + e^-3} = 0.99505475$$

$$g_4 = \frac{e^4 - e^-4}{e^4 + e^-4} = 0.9993293$$

Where i = j do $(1 - g^2(z)) + \epsilon$ where $\epsilon = .0000001$ Else, return 0

$$(1 - g_1^2) + \epsilon = (1 - 0.76159416^2) + .0000001 = 0.41997444$$

$$(1 - g_2^2) + \epsilon = (1 - 0.96402758^2) + .0000001 = 0.07065092$$

$$(1 - g_3^2) + \epsilon = (1 - 0.99505475^2) + .0000001 = 0.00986614$$

$$(1-g_4^2)+\epsilon=(1-0.9993293^2)+.0000001=0.00134105$$

This gives a final matrix of:

$$\begin{bmatrix} 0.41997444 & 0 & 0 & 0 \\ 0 & 0.07065092 & 0 & 0 \\ 0 & 0 & 0.00986614 & 0 \\ 0 & 0 & 0 & 0.00134105 \end{bmatrix}$$

2. With weight matrix $W=\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{bmatrix}$, the graident is just W tranposed. So the gradient of our fully connected layer would be

$$W=\begin{bmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{bmatrix}$$

3. $y=0$, $\hat{y}=.2$, $\epsilon=.0000001$ Calculate the loss:

   (a) Squared Error:

   $$J=(y-\hat{y})^2=(0-.2)^2=.04$$

   (b) Log Loss:

   $$J=-1*(y*\ln(\hat{y}+\epsilon)+(1-y)*\ln(1-\hat{y}+\epsilon))$$

   $$J=-1*(0*\ln(.2+.0000001)+(1-0)*\ln(1-.2+.0000001))$$

   $$J=-1*(0*-1.6094374+1*-0.2231437)$$

   $$J=-1*(-0.2231437)$$

   $$J=0.2231437$$

4. Given $y=[1,0,0]$, $\hat{y}=[0.2,0.2,0.6]$, $\epsilon=.0000001$ Calculate Cross Entropy

$$J=-1*y*\ln(\hat{y}^T+\epsilon)$$

$$J=-1*\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}*\ln\left(\begin{bmatrix} .2 & .2 & .6 \end{bmatrix}^T+.0000001\right)$$

$$J=-1*\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}*\ln\left(\begin{bmatrix} .2000001 \\ .2000001 \\ .6000001 \end{bmatrix}\right)$$

$$J=-1*\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}*\begin{bmatrix} -1.6094374 \\ -1.6094374 \\ -0.5108255 \end{bmatrix}$$

1x3 matrix * 3x1 matrix gives an answer of 1x1 matrix so out answer is

$$J=-1*[-1.6094374]$$

$$J=1.6094374$$

5. Given $y = 0$, $\hat{y} = .2$, $\epsilon = .0000001$ calculate the gradients:

   (a) Squared Error:

$$\delta = -2 * (y - \hat{y})$$
$$\delta = -2 * (0 - .2)$$
$$\delta = -2 * -.2$$
$$\delta = -2 * -.2$$
$$\delta = .4$$

   (b) Log Loss:

$$\delta = -1 * \frac{y - \hat{y}}{\hat{y} * (1 - \hat{y}) + \epsilon}$$
$$\delta = -1 * \frac{0 - .2}{.2 * (1 - .2) + .0000001}$$
$$\delta = -1 * \frac{-.2}{.2 * .8 + .0000001}$$
$$\delta = -1 * \frac{-.2}{.1600001}$$
$$\delta = -1 * -1.2499992$$
$$\delta = 1.2499992$$

6. Given $y = [1, 0, 0]$, $\hat{y} = [0.2, 0.2, 0.6]$, $\epsilon = .0000001$ Calculate Cross Entropy gradient:

$$\delta = -1 * \frac{y}{\hat{y} + \epsilon}$$
$$\delta_1 = -1 * \frac{1}{.2 + .0000001} = -4.9999975$$
$$\delta_2 = -1 * \frac{0}{.2 + .0000001} = 0$$
$$\delta_3 = -1 * \frac{0}{.6 + .0000001} = 0$$
$$\delta = [-4.9999975, 0, 0]$$

Activation/FC Layer

```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code> python main.py
Fully Connected:
[[0.0009792  0.00099362]]
Fully Connected Gradient:
[[6.46440512e-05 1.14556810e-04 8.08290128e-05 6.34649549e-05]
 [2.70964398e-05 9.37682339e-05 3.12761634e-05 1.67531900e-04]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 1: FC Layer

```
Linear
[[1 2 3 4]]
Activation Gradient:
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 2: Linear

```
RELU
[[1 2 3 4]]
Activation Gradient:
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 3: ReLu

```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code> python main.py
Pick the number coresponding to the activation rate you wish to use
1) Linear
2) ReLu Linear
3) Sigmoid
4) TanH
5) Softmax
3
Sigmoid
[[0.73105858 0.88079708 0.95257413 0.98201379]]
Activation Gradient:
[[0.19661203 0.         0.         0.        ]
 [0.         0.10499369 0.         0.        ]
 [0.         0.         0.04517676 0.        ]
 [0.         0.         0.         0.01766281]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 4: Sigmoid

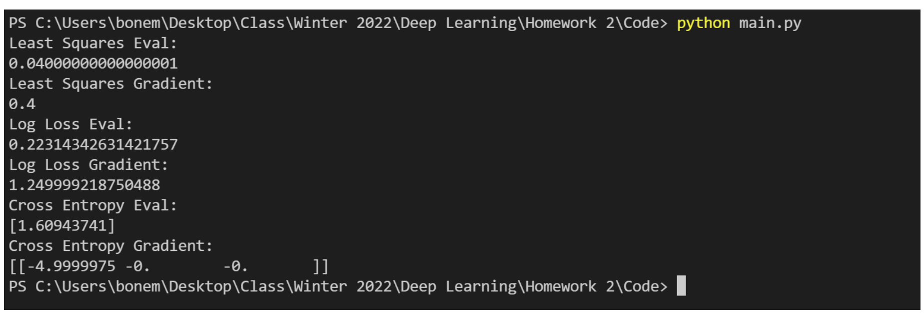
```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code> python main.py
Pick the number coresponding to the activation rate you wish to use
1) Linear
2) ReLu Linear
3) Sigmoid
4) TanH
5) Softmax
4
Tanh
[[0.76159416 0.96402758 0.99505475 0.9993293 ]]
Activation Gradient:
[[0.41997444 0.         0.         0.        ]
 [0.         0.07065092 0.         0.        ]
 [0.         0.         0.00986614 0.        ]
 [0.         0.         0.         0.00134105]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 5: TanH

```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code> python main.py
Pick the number coresponding to the activation rate you wish to use
1) Linear
2) ReLu Linear
3) Sigmoid
4) TanH
5) Softmax
5
Softmax
[[0.0320586  0.08714432 0.23688282 0.64391426]]
Activation Gradient:
[[ 0.03103085 -0.00279373 -0.00759413 -0.02064299]
 [-0.00279373  0.07955019 -0.02064299 -0.05611347]
 [-0.00759413 -0.02064299  0.18076935 -0.15253222]
 [-0.02064299 -0.05611347 -0.15253222  0.22928869]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 6: Softmax

Objective Layer



```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code> python main.py
Least Squares Eval:
0.04000000000000001
Least Squares Gradient:
0.4
Log Loss Eval:
0.22314342631421757
Log Loss Gradient:
1.249999218750488
Cross Entropy Eval:
[1.60943741]
Cross Entropy Gradient:
[[-4.9999975 -0.        -0.       ]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 2\Code>
```

Figure 7: Objective Layers