

2) Grad. for log loss (wrt  $w$ ) =  $x(\hat{y} - y)$

If all training examples are true,  $y = 1$

$$\therefore \nabla_w f_{\log} = \underline{x(\sigma(x^T w + b) - 1)}; \nabla_b f_{\log} = \hat{y} - y = (\sigma(x^T w + b) - 1)$$

$$f_{\log} = -y \log \hat{y} - (1-y) \log(1-\hat{y})$$

$$\text{If } y=1 \quad f_{\log} = -\log(\sigma(x^T w + b))$$

The goal of the neural network is  $\therefore$  to minimise  $-\log(\sigma(x^T w + b))$

i.e. maximise  $\log(\sigma(x^T w + b))$

$\log$  is a monotonically increasing function,  $\therefore$  increasing  $\log(\sigma(x^T w + b))$  is the same as increasing  $\sigma(x^T w + b)$

Sigmoid is also a monotonically increasing function,  $\therefore$  increasing  $\sigma(x^T w + b)$  is the same as increasing  $x^T w + b$

Consider the way in which  $b$  changes

$$b_{\text{new}} = b_{\text{old}} - \epsilon \nabla_b f_{\log} = b_{\text{old}} - \epsilon(\hat{y} - y) = b_{\text{old}} - \epsilon(\sigma(z) - 1)$$

$$\epsilon(\sigma(z) - 1) < 0 \quad \because \sigma(z) < 1$$

$\therefore b$  will always keep decreasing and its gradient will never be 0

However the value of  $\hat{y} - 1$  will also get smaller as  $z$  increases

Therefore, the gradient will in fact get smaller but never reach 0.

a) Let us check how a change in  $b$  affects the gradient because this will tell us whether the ~~max~~ value of grad. getting smaller affects the rate at which  $b$  changes

For simplicity, let  $w = 0$

$$\therefore \hat{y} - 1 = \sigma(b) - 1$$

$$\frac{d}{db}(\hat{y} - 1) = \frac{d}{db}(\sigma(b) - 1) = \sigma(b)(1 - \sigma(b))$$

Suppose  $b$  is already very large (as we might expect it to be to correctly output a val close to 1 from the network)  
then  $\sigma(b) \approx 1$

$\therefore$  the rate of change of the grad wrt  $b$  tends to 0

Hence  $b$  will have to change by a very large amount even if grad is small

$\therefore$   $b$  will diverge

b) A similar argument holds for  $w$  except that grad wrt  $w$  depends on  $x$   
hence,  $w$  can converge if  $x$  is 0  
otherwise  $w$  will also diverge

(c) Since the value of weights and bias converge, the training loss will also converge.

(d) Considering that the ~~training~~ training examples were all positive and testing examples can be positive and negative, it will affect the loss function.

Since the ground truth for testing examples will oscillate or change as 0 or 1, the log loss will change ~~form~~.

It will make the log loss oscillate between

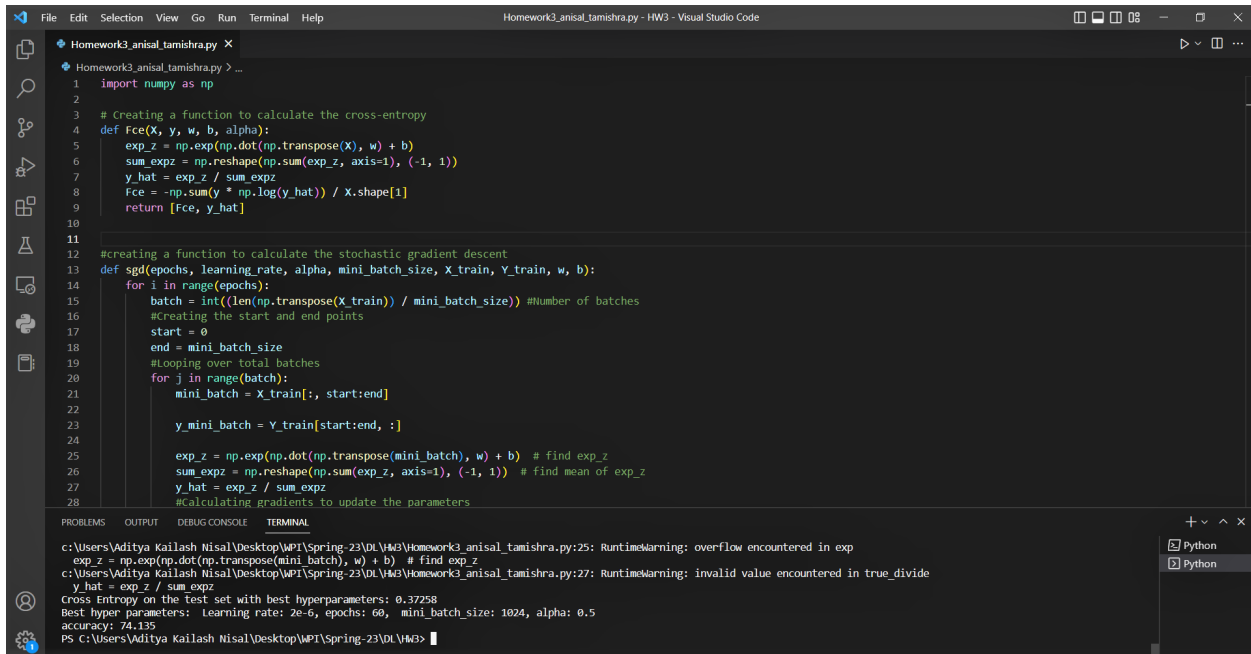
$-\log(\hat{y})$  and  $-(1-\hat{y})\log(1-\hat{y})$ .

Thus, the testing loss doesn't converge or won't converge

Team members:  
Aditya Nisal  
Tanish Mishra

## CS-541 Deep Learning Assignment - 03

Q.4)



```
File Edit Selection View Go Run Terminal Help
Homework3_anisal_tamishra.py - HW3 - Visual Studio Code

Homework3_anisal_tamishra.py X
Homework3_anisal_tamishra.py > ...
1 import numpy as np
2
3 # Creating a function to calculate the cross-entropy
4 def Fce(X, y, w, b, alpha):
5     exp_z = np.exp(np.dot(np.transpose(X), w) + b)
6     sum_expz = np.reshape(np.sum(exp_z, axis=1), (-1, 1))
7     y_hat = exp_z / sum_expz
8     Fce = -np.sum(y * np.log(y_hat)) / X.shape[1]
9     return [Fce, y_hat]
10
11
12 #creating a function to calculate the stochastic gradient descent
13 def sgd(epochs, learning_rate, alpha, mini_batch_size, X_train, Y_train, w, b):
14     for i in range(epochs):
15         batch = int((len(np.transpose(X_train)) / mini_batch_size)) #Number of batches
16         #creating the start and end points
17         start = 0
18         end = mini_batch_size
19         #Looping over total batches
20         for j in range(batch):
21             mini_batch = X_train[:, start:end]
22
23             y_mini_batch = Y_train[start:end, :]
24
25             exp_z = np.exp(np.dot(np.transpose(mini_batch), w) + b) # find exp_z
26             sum_expz = np.reshape(np.sum(exp_z, axis=1), (-1, 1)) # find mean of exp_z
27             y_hat = exp_z / sum_expz
28             #Calculating gradients to update the parameters
29
30 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
c:\Users\Aditya Kailash Nisal\Desktop\WPI\Spring-23\DL\HW3\Homework3_anisal_tamishra.py:25: RuntimeWarning: overflow encountered in exp
exp_z = np.exp(np.dot(np.transpose(mini_batch), w) + b) # find exp_z
c:\Users\Aditya Kailash Nisal\Desktop\WPI\Spring-23\DL\HW3\Homework3_anisal_tamishra.py:27: RuntimeWarning: invalid value encountered in true_divide
y_hat = exp_z / sum_expz
Cross Entropy on the Test set with best hyperparameters: 0.37258
Best hyper parameters: Learning rate: 2e-6, epochs: 60, mini_batch_size: 1024, alpha: 0.5
accuracy: 74.135
PS C:\Users\Aditya Kailash Nisal\Desktop\WPI\Spring-23\DL\HW3> |
```

Optimal Hyperparameters after testing on the validation set : [1024, 2e-6, 60, 0.5]

minibatch\_size = 1024

learning\_rate = 2e-6

Epochs = 60

Alpha = 0.5

1) Linear Regression on the XOR function

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad y^T = [0 \ 0 \ 1 \ 0] \quad \text{Loss function: } f_{\text{mse}}$$

$$f_{\text{mse}} = \frac{1}{2n} \sum_{i=1}^n (y - y^{(i)})^2 = \frac{1}{2n} \sum_{i=1}^n (\tilde{X}^T \tilde{W} - y)^2 \quad \text{Where } \tilde{X} = \begin{bmatrix} X \\ 1 \end{bmatrix} \quad \tilde{W} = \begin{bmatrix} w \\ b \end{bmatrix}$$

$$\nabla_{\tilde{W}} = \frac{1}{n} \tilde{X}^T (\tilde{X} \tilde{W} - y)$$

Eq 6.1 from the DL Textbook:

$$J(\theta) = \frac{1}{4} \sum_{x \in X} (f^*(x) - f(x; \theta))^2 \quad \text{Where } f^*(x) = \text{XOR}$$

For a linear model  $f(x; \theta) = f(x; w, b) = x^T w + b$

$$J(\theta) = \frac{1}{4} \sum (f^*(x) - (x^T w + b))^2$$

$$\nabla_{w,b} J(\theta) = 0 = \nabla_{w,b} \sum (f^*(x) - (x^T w + b))^2$$

$$= \nabla_{w,b} \left( \left[ 0 - \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - b \right]^2 + \left[ 0 - \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - b \right]^2 + \left[ 1 - \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - b \right]^2 + \left[ 0 - \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - b \right]^2 \right)$$

$$\nabla_{w,b} (b^2 + (1 - w_2 - b)^2 + (1 - w_1 - b)^2 + (-w_1 - w_2 - b)^2)$$

$$= \nabla_{w,b} (g(w_1, w_2, b))$$

$$\frac{\partial g}{\partial w_1} = (2(w_1 + b - 1) + 2(w_1 + w_2 + b)) = 2(2w_1 + 2b - 1 + w_2) = 4(w_1 + b - \frac{1}{2}) + \frac{w_2}{2}$$

$$\frac{\partial g}{\partial w_2} = (2(w_2 + b - 1) + 2(w_1 + w_2 + b)) = 4(w_2 + b - \frac{1}{2}) + \frac{w_1}{2}$$

$$\frac{\partial g}{\partial b} = (2b + 2(w_1 + b - 1) + 2(w_2 + b - 1) + 2(w_1 + w_2 + b)) = 4(2b + w_1 + w_2 - 1)$$

$$\nabla_{w,b}(q) = 0 \Rightarrow \begin{bmatrix} w_1 + b - 1/2 + w_2/2 \\ w_2 + b - 1/2 + w_1/2 \\ w_1 + w_2 + 2b - 1 \end{bmatrix} = 0$$

Equating each element to 0

$$w_1 + b - 1/2 + \frac{w_2}{2} = 0 \Rightarrow 2w_1 + 2b + w_2 - 1 = 0 \quad \dots \text{I}$$

similarly

$$2w_2 + 2b + w_1 - 1 = 0 \quad \dots \text{(II)}$$

$$w_1 + w_2 + 2b - 1 = 0 \quad \dots \text{III}$$

$$\text{(I)} - \text{(II)} \Rightarrow w_1 - w_2 = 0 \Rightarrow \boxed{w_1 = w_2 = w}$$

Replacing in (I)

$$3w + 2b - 1 = 0 \quad \dots \text{IV}$$

and (III)

$$2w + 2b - 1 = 0 \quad \dots \text{V}$$

$$\text{(IV)} - \text{(V)} \Rightarrow \boxed{w = 0}$$

Replacing in III

$$2b - 1 = 0$$

$$\Rightarrow \boxed{b = \frac{1}{2}}$$

Hence Proved



$$(Q3) \quad W = [w^{(1)} \dots w^{(c)}]$$

$$\hat{y}_k = \frac{\exp z_k}{\sum_{k'=1}^c \exp z_{k'}}$$

$$z_k = x^T w^{(k)} + b_k$$

$$\nabla_{w^{(l)}} f(\mathcal{E}(W, b)) = -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^c y_k^{(i)} \nabla_{w^{(l)}} \log \hat{y}_k^{(i)}$$

$$= -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^c y_k^{(i)} \left( \frac{\nabla \hat{y}_k^{(i)}}{\hat{y}_k^{(i)}} \right)$$

for  $l = k$ :

$$\nabla_{w^{(k)}} \hat{y}_k^{(i)} =$$

$$\hat{y}_k^{(i)} = \frac{\exp z_k}{\sum_{k'=1}^c \exp z_{k'}}$$

since  $z_k = x^{(i)T} w^{(k)} + b_k$

$$\frac{\exp(x^{(i)T} w^{(k)} + b_k)}{\sum_{k'=1}^c \exp(x^{(i)T} w^{(k')} + b_{k'})}$$

since  $k = l$ .

$$\hat{y}_l^{(i)} = \frac{\exp(x^{(i)T} w^{(l)} + b_l)}{\sum_{k'=1}^c \exp(x^{(i)T} w^{(k')} + b_{k'})}$$

$$\Rightarrow \frac{\exp(x^{(i)T} w^{(l)} + b_l)}{\text{den.}}$$

$$\nabla_{w^{(l)}} \hat{y}_l^{(i)} = \frac{\text{den} \cdot \frac{d}{dw^l} (\exp(x^{(i)T} w^{(l)} + b_l)) - \exp(x^{(i)T} w^{(l)} + b_l) \cdot \frac{d}{dw^l} \text{den}}{(\text{den})^2}$$

(2)

$$= x^{(i)} \frac{(\text{den} \cdot \exp(x^{(i)T} w^l + b_l) - \exp(x^{(i)T} w^l + b_l) \cdot (\exp(x^{(i)T} w^l + b_l)))}{\text{den}^2}$$

$$= \text{let } \exp(x^{(i)T} w^l + b_l) = \text{num}$$

$$= x^{(i)} \frac{(\text{den} \cdot \text{num} - \text{num} \cdot \text{num})}{\text{den}^2}$$

$$= x^{(i)} \left( \frac{\text{num}}{\text{den}} - \frac{\text{num}^2}{\text{den}^2} \right)$$

$$= x^{(i)} \left( \frac{\exp(x^{(i)T} w^l + b_l)}{\sum_{k'=1}^c \exp(x^{(i)T} w^{k'} + b_{k'})} - \left( \frac{\exp(x^{(i)T} w^l + b_l)}{\sum_{k'=1}^c \exp(x^{(i)T} w^{k'} + b_{k'})} \right)^2 \right)$$

$$= x^{(i)} \left( \hat{y}_l^{(i)} - \hat{y}_l^{(i)2} \right)$$

$$= x^{(i)} \hat{y}_l^{(i)} \left( 1 - \hat{y}_l^{(i)} \right) \quad \text{--- ①}$$

$$\text{for } l \neq k: y_k^{(i)} = \frac{\exp z_k}{\text{den}} = \frac{\exp(x^{(i)T} w^{(k)} + b_k)}{\text{den}}$$

$$\nabla_{w^{(k)}} y_k^{(i)} = \frac{\text{den} \times \frac{d}{dw^{(k)}} (x^{(i)T} w^{(k)} + b_k) - (\exp x^{(i)T} w^{(k)} + b_k) \frac{d}{dw^{(k)}} \text{den}}{\text{den}^2}$$

$$= \frac{\text{den} \cdot 0 - x^{(i)} (\exp x^{(i)T} w^{(k)} + b_k) (\exp x^{(i)T} w^{(k)} + b_k)}{\text{den}^2}$$



$$\rightarrow \boxed{\nabla_{w^{(e)}} \hat{y}_{lc}^{(i)} = -x^{(i)} \hat{y}_k^{(i)} \cdot \hat{y}_e^{(i)}} \quad (2)$$

(3)

Now, to find:

$$\nabla_{w^{(e)}} f_{CE}(w, b) = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^c \hat{y}_k^{(i)} \nabla_{w^{(e)}} \log \hat{y}_k^{(i)}$$

$$= \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^c \hat{y}_k^{(i)} \left( \frac{\nabla_{w^{(e)}} \hat{y}_k^{(i)}}{\hat{y}_k^{(i)}} \right)$$

$$= \frac{1}{n} \sum_{i=1}^n \left( \hat{y}_1^{(i)} \left( \frac{\nabla_{w^{(e)}} \hat{y}_1^{(i)}}{\hat{y}_1^{(i)}} \right) + \hat{y}_2^{(i)} \left( \frac{\nabla_{w^{(e)}} \hat{y}_2^{(i)}}{\hat{y}_2^{(i)}} \right) \dots + \hat{y}_c^{(i)} \left( \frac{\nabla_{w^{(e)}} \hat{y}_c^{(i)}}{\hat{y}_c^{(i)}} \right) \right)$$

now, using (1) and (2) we get:

$$= \frac{1}{n} \sum_{i=1}^n \left[ \left( \hat{y}_1^{(i)} \left( \frac{-x \hat{y}_1^{(i)} \cdot \hat{y}_e^{(i)}}{\hat{y}_1^{(i)}} \right) + \left( \hat{y}_2^{(i)} \left( \frac{-x \hat{y}_2^{(i)} \cdot \hat{y}_e^{(i)}}{\hat{y}_2^{(i)}} \right) \dots + \left( \hat{y}_e^{(i)} \left( \frac{x \hat{y}_e^{(i)} - x \hat{y}_e^{(i)}}{\hat{y}_e^{(i)}} \right) + \dots \right. \right. \right. \\ \left. \left. \left. \dots \hat{y}_c^{(i)} \left( \frac{-x \hat{y}_c^{(i)} \cdot \hat{y}_e^{(i)}}{\hat{y}_c^{(i)}} \right) \right) \right]$$

$$= \frac{1}{n} \sum_{i=1}^n \left( -x \hat{y}_1^{(i)} \hat{y}_e^{(i)} - x \hat{y}_2^{(i)} \hat{y}_e^{(i)} \dots + x \hat{y}_e^{(i)} - x \hat{y}_e^{(i)} \hat{y}_e^{(i)} \dots - x \hat{y}_c^{(i)} \hat{y}_e^{(i)} \right)$$

$$= \frac{1}{n} \sum_{i=1}^n \left[ \left( x \hat{y}_e^{(i)} (-\hat{y}_1 - \hat{y}_2 \dots - \hat{y}_c + \hat{y}_e) \right) + x \hat{y}_e^{(i)} \right] \quad \because \sum_{k=1}^c \hat{y}_k^{(i)} = 1$$

$$= \frac{1}{n} \sum_{i=1}^n \left( x \hat{y}_e^{(i)} (-1) + x \hat{y}_e^{(i)} \right)$$

$$= \frac{1}{n} \sum_{i=1}^n x^{(i)} (\hat{y}_e^{(i)} - \hat{y}_e^{(i)})$$

$$\boxed{\nabla_{w^{(e)}} f_{CE}(w, b) = \frac{1}{n} \sum_{i=1}^n x^{(i)} (\hat{y}_e^{(i)} - \hat{y}_e^{(i)})}$$

(4)

Similarly, to compute the gradient of  $f_{CE}$  w.r.t  $b_k$ , we follow the same procedure.

Since,  $x^{(i)}$  was a coefficient of  $W_k^{(i)}$  so we had it ~~output~~ multiplied with the derivative.

Now,  $b_k$  here doesn't have any coeff but 1.

Hence,

$$\text{for } k=l: \nabla_{b_k} f_{CE} = \hat{y}_l^{(i)} (1 - \hat{y}_l^{(i)}).$$

$$k \neq l: \nabla_{b_k} f_{CE} = -\hat{y}_k^{(i)} \cdot \hat{y}_l^{(i)}.$$

thus we get:

$$\nabla_{b_k} f_{CE} = -\frac{1}{n} \sum_{i=1}^n [y_l^{(i)} - \hat{y}_l^{(i)}]$$

Combining all these scalars into one vector, we get:

$$\boxed{\nabla_b f_{CE} = -\frac{1}{n} \sum_{i=1}^n [\text{ } y^{(i)} - \hat{y}^{(i)}]}$$