Homework 2: Intro to Deep Learning (Spring 2020)

Name: Pratik Mistry
NetID: pdm79
RUID: 194008675

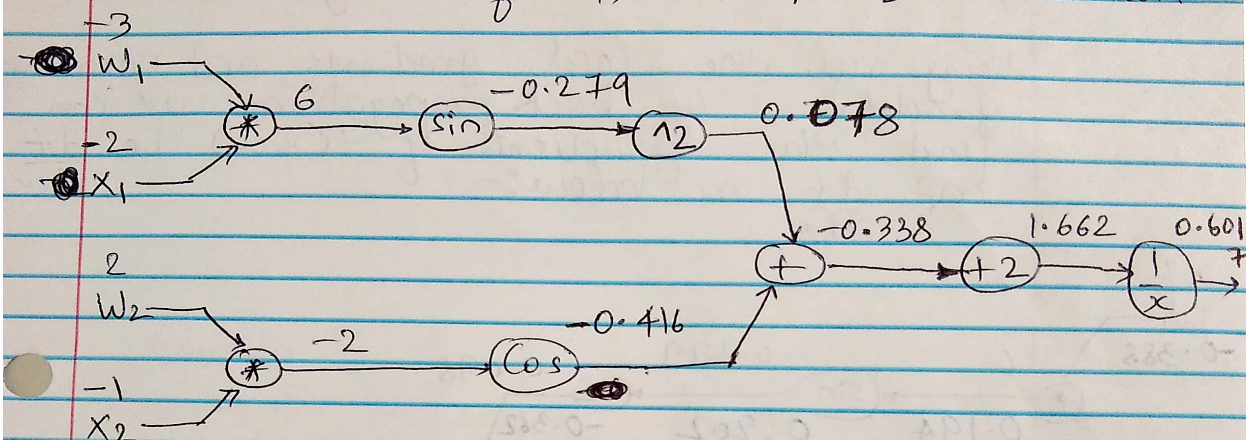
Solution 1:

Part A: Please find the below attached images of the hand-written computational graph calculations to calculate output of given function using forward propagation and also the gradients of input i.e. W, X by back propagation.

Homework 2 : Question 1

Solution: Given: $f(x,w) = \dfrac{1}{2 + \sin^2(w_1 x_1) + \cos(w_2 x_2)}$

→ Below is the graphical representation of forward propagation for the given function and values of w_1, w_2 & x_1 & x_2 as shown.



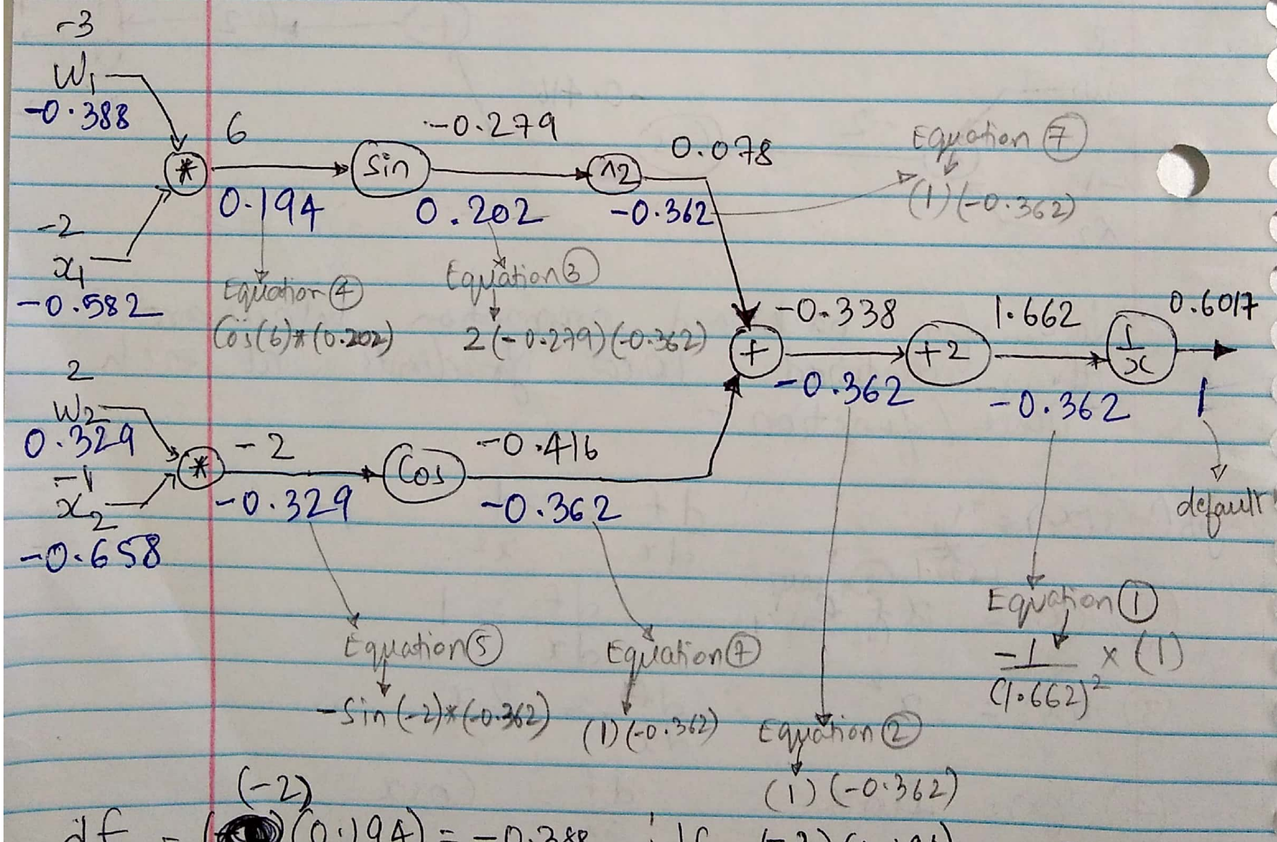
Now, for backward propagation below are the identified local gradients at each node / function:-

1. $f(x) = \frac{1}{x}$ $\quad \therefore \frac{df}{dx} = \frac{-1}{x^2}$
2. $f(x) = x + c$ $\quad \therefore \frac{df}{dx} = 1$
3. $f(x) = x^2$ $\quad \therefore \frac{df}{dx} = 2x$
4. $f(x) = \sin x$ $\quad \therefore \frac{df}{dx} = \cos x$
5. $f(x) = \cos x$ $\quad \therefore \frac{df}{dx} = -\sin x$

(6) $f(x,y) = xy \quad \therefore \frac{df}{dx} = y$ & $\frac{df}{dy} = x$ ... switcher node

(7) $f(x,y) = x+y \quad \therefore \frac{df}{dx} = 1$ & $\frac{df}{dy} = 1$ .... distributor node.

$\therefore$ Using all the local gradients and upstream gradients, by back propogation we can find the gradients of $f(x,w)$ ie $\frac{df}{dw}$ and $\frac{df}{dx}$ as below:-



$\frac{df}{dw_1} = (-2)(0.194) = -0.388$ $\quad$ $\frac{df}{dx_1} = (-3)(0.194) = -0.582$

& $\frac{df}{dw_2} = (-1)(-0.329) = 0.329$ $\quad$ & $\frac{df}{dx_2} = (2)(-0.329) = -0.658$

Part B: Please find the uploaded python code which will calculate the output of the given function using forward propagation and also the gradients of input values of W, X using back propagation. Below is the screenshot of the output of the python code with input same as the ones taken in handwritten computation.

Q1-HW2.py × Q2-HW2.py

Q1-HW2.py > ...

```
        return dW1,dX1,dW2,dX2


river Function
__name__ == "__main__":
 computationalGraph = ComputationalGraphFunction(-3,-2,2,-1)
 forward_feed_output = computationalGraph.forward()
 print("\nThe output of given computational function by forward propogation is: ", forward_feed_output)
 print("-------------------------------------------------------------------------------------------------

 dW1, dx1, dW2, dx2 = computationalGraph.backward()
 print("-------------------------------------------------------------------------------------------------
 print("\nThe local gradient of W i.e. dW1 and dW2 by back propogation is: ", dW1, "and" , dW2)
 print("\nThe local gradient of X i.e. dx1 and dx2 by back propogation is: ", dx1, "and", dx2)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
(base) suketuvsmacbook:HW2 learning$ python3 Q1-HW2.py

The output of W1*X1 function is:  6
The output of Sin(W1*X1) function is:  -0.27941549819892586
The output of Sin^2(W1*X1) function is:  0.07807302063375395

The output of W2*X2 function is:  -2
The output of Cos(W2*X2) function is:  -0.4161468365471424

The output of Sin^2(W1.X1) + Cos(W2.X2) function is:  -0.3380738159133885
The output of (Sin^2(W1.X1) + Cos(W2.X2) + 2) function is:  1.6619261840866115
The output of inverse of (Sin^2(W1.X1) + Cos(W2.X2) + 2) function is:  0.6017114415641729

The output of given computational function by forward propogation is:  0.6017114415641729
----------------------------------------------------------------------------------------------------

The local gradient at inverse function i.e. 1/(Sin^2(W1.X1) + Cos(W2.X2) + 2) is:  -0.36205665890923505
The local gradient at linear function i.e. (Sin^2(W1.X1) + Cos(W2.X2) + 2) is:  1
The local gradient at distributor function i.e. Sin^2(W1.X1) + Cos(W2.X2) is:  1
The local gradient at Square Function i.e. Sin^2(W1.X1) is:  -0.5588309963978517
The local gradient at Sine Function i.e. Sin(W1.X1) is:  0.9601702866503661

The local gradient at Cosine Function i.e. Cos(W2.X2) is:  0.9092974268256817
----------------------------------------------------------------------------------------------------

The local gradient of W i.e. dW1 and dW2 by back propogation is:  -0.3885395959048329 and 0.32921718831127095

The local gradient of X i.e. dx1 and dx2 by back propogation is:  -0.5828093938572494 and -0.6584343766225419
```

Conclusion: Thus, by looking at both Part A and B of the solution, we can see that the output of the function and also gradients for the given set of inputs using forward and backward propagation is the same.

NOTE: Input values taken are: W1 = -3, X1 = -2, W2 = 2, X2 = -1

Solution 2:

Part A: Please find the below attached images of the hand-written computational graph calculations to calculate output of given function using forward propagation and also the gradients of input i.e. W, X by back propagation.

Homework 2: Question 2

Solution:-

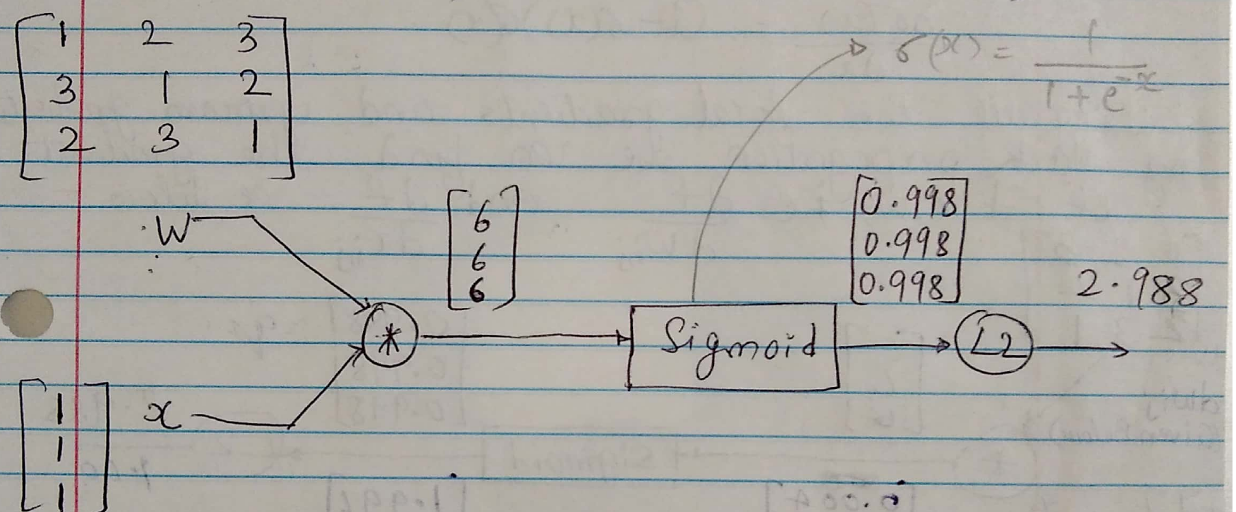
→ Given $f(x, W) = \|\sigma(Wx)\|^2$ .... W is 3×3 matrix

... x is 3×1 matrix

... $\sigma(\cdot)$ is Sigmoid function

... $\|\cdot\|^2$ is L2 loss.

→ Below is the graphical representation of forward propogation for given function and values of W (3×3 matrix) and x (3×1 matrix)



Now, for backward propogation, to find gradiants of $W_{3\times3}$ and $x_{3\times1}$ matrixes, ~~below~~ below are the identified local gradients at each node/function:

① $f(q) = \|q\|^2 = q_1^2 + q_2^2 + \dots + q_n^2$ - L2 loss function

$\therefore \frac{df}{dq_i} = 2q_i$ $\quad \therefore \nabla_q f = 2q$.

② $q = W \cdot x$ $\quad \therefore \frac{df}{dW_{i,j}} = \sum_k \frac{df}{\partial q_k} \cdot \frac{\partial q_k}{\partial W_{i,j}} = 2q_i x_j$

$\therefore \nabla_W f = 2q \cdot x^T$.

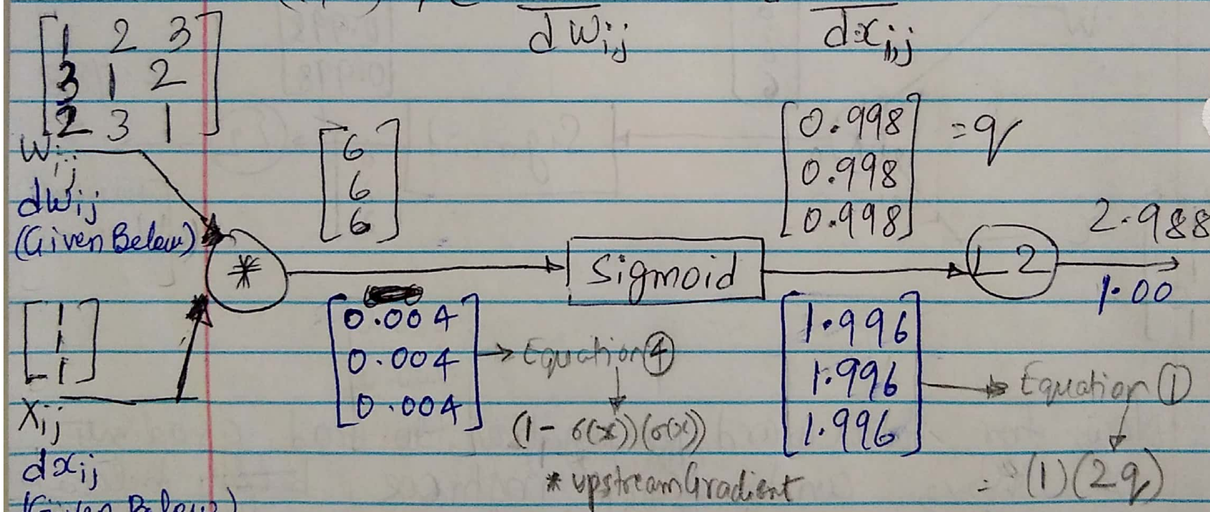
Similarly, (3) $q = W.x$ $\quad \therefore \dfrac{df}{dx_i} = \sum_k \dfrac{\partial f}{\partial q_k} \dfrac{\partial q_k}{\partial x_i}$

$$\therefore \nabla_x f = 2W^T q.$$

(4). Sigmoid function: $\sigma(x) = \dfrac{1}{1+e^{-x}}$

$$\therefore \frac{\partial \sigma(x)}{\partial x} = (1-\sigma(x))\sigma(x).$$

$\therefore$ Using all the local gradients and upstream gradients by back propogation, we can find the gradient of $f(x,w)$ i.e $\dfrac{df}{dW_{ij}}$ and $\dfrac{df}{dx_{ij}}$ as below:-



$$\frac{df}{dW_{ij}} = \underbrace{2q.x^T}_{\text{Equation (2)}} \approx \begin{bmatrix} 0.004 & 0.004 & 0.004 \\ 0.004 & 0.004 & 0.004 \\ 0.004 & 0.004 & 0.004 \end{bmatrix}$$

$$\frac{df}{dx_{ij}} = \underbrace{2W^T.q}_{\text{Equation (3)}} \approx \begin{bmatrix} 0.024 \\ 0.024 \\ 0.024 \end{bmatrix}$$

Note:- The effect of rounding/approximation in decimal values gives a bit of different dx_{ij}

Part B: Please find the uploaded python code which will calculate the output of the given function using forward propagation and also the gradients of input values of W, X using back propagation. Below is the screenshot of the output of the python code with input same as the ones taken in handwritten computation.

Q1-HW2.py   Q2-HW2.py ×

Q2-HW2.py > ...

```python
# Driver Function
if __name__ == "__main__":
    computationalGraph = ComputationalGraphFunction([[1,2,3],[3,1,2],[2,3,1]],[[1],[1],[1]])            # Creating object of
    forward_feed_output = computationalGraph.forward()            # Calculating output forward propogration
    print("\nThe output of given computational function by forward propogation is: ", forward_feed_output)
    print("--------------------------------------------------------------------------------------------------------"

    dW, dx = computationalGraph.backward()                        # Calculating gradients of W and X using backward propogat
    print("--------------------------------------------------------------------------------------------------------"
    print("\nThe local gradient of W i.e. dW by back propogation is: \n", dW)
    print("\nThe local gradient of X i.e. dx by back propogation is: \n", dx)
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

```
The output of multiplication function of W and X is:
 [[6]
 [6]
 [6]]
The output of sigmoid function is:
 [[0.99752738]
 [0.99752738]
 [0.99752738]]
The output of L2 loss function is:  2.985182602656016

The output of given computational function by forward propogation is:  2.985182602656016
------------------------------------------------------------------------------------------------------------

The local gradient at L2 Loss function is:
 [[1.99505475]
 [1.99505475]
 [1.99505475]]
The local gradient at sigmoid function is:
 [[0.00246651]
 [0.00246651]
 [0.00246651]]
------------------------------------------------------------------------------------------------------------

The local gradient of W i.e. dW by back propogation is:
 [[0.00492082 0.00492082 0.00492082]
 [0.00492082 0.00492082 0.00492082]
 [0.00492082 0.00492082 0.00492082]]

The local gradient of X i.e. dx by back propogation is:
 [[0.02952493]
 [0.02952493]
 [0.02952493]]
```

Conclusion: Thus, by looking at both Part A and B of the solution, we can see that the output of the function and also gradients for the given set of inputs using forward and backward propagation is the same.

NOTE:

- Input values taken are: W (i,j) = [[1,2,3],[3,1,2],[2,3,1]] and X (i,j) = [[1],[1],[1]]
- We can see a bit different output for dX (i,j) because of precision in approximation/rounding off. But, the output is nearly same upto 2 decimal points. E.g. in handwritten gradient value is 0.024 while gradient is 0.029 in output of python code.