Deep learning Ex1

Ariel Radomislensky 211991146

Question 1

1. for :

Question 3

1. verified using python:

For x = [0 0] hidden layer output is [0. 0.]

For x = [0 0] output is 0

For x = [0 1] hidden layer output is [0. 0.5]

For x = [0 1] output is 1

For x = [1 0] hidden layer output is [0.5 0. ]

For x = [1 0] output is 1

For x = [1 1] hidden layer output is [0. 0.]

For x = [1 1] output is 0

import numpy as np

U = np.array([[1,-1],[-1,1]])

b1 = -0.5

w = np.array([1,1]).T

b2 = -0.1

X = [np.array([0,0]).T, np.array([0,1]).T, np.array([1,0]).T, np.array([1,1]).T]

for x in X:

h = np.maximum(U.T@x + b1, np.zeros((2,)))

print("For x = ", x, "hidden layer output is ", h)

f = w.T@h + b2

if(f>=0):

print("For x = ", x, "output is 1")

else:

print("For x = ", x, "output is 0")

1. No, it would not be possible, replacing the max function with the identity will effectively eliminate the hidden layer:  
   turning into a linear classifier, and XOR cannot be linearly classified.
2. ReLU(x) can in fact be defined as max(x,0). Maybe the question meant weather we can set ? If so yes, it is still possible:

For x = [0 0] hidden layer output is [0. 0.]

For x = [0 0] output is 0

For x = [0 1] hidden layer output is [1. 0.]

For x = [0 1] output is 1

For x = [1 0] hidden layer output is [0. 1.]

For x = [1 0] output is 1

For x = [1 1] hidden layer output is [0. 0.]

For x = [1 1] output is 0

import numpy as np

U = np.array([[1,-1],[-1,1]])

b1 = 0

w = np.array([1,1]).T

b2 = -0.1

X = [np.array([0,0]).T, np.array([0,1]).T, np.array([1,0]).T, np.array([1,1]).T]

for x in X:

h = np.maximum(U.T@x + b1, np.zeros((2,)))

print("For x = ", x, "hidden layer output is ", h)

f = w.T@h + b2

if(f>=0):

print("For x = ", x, "output is 1")

else:

print("For x = ", x, "output is 0")

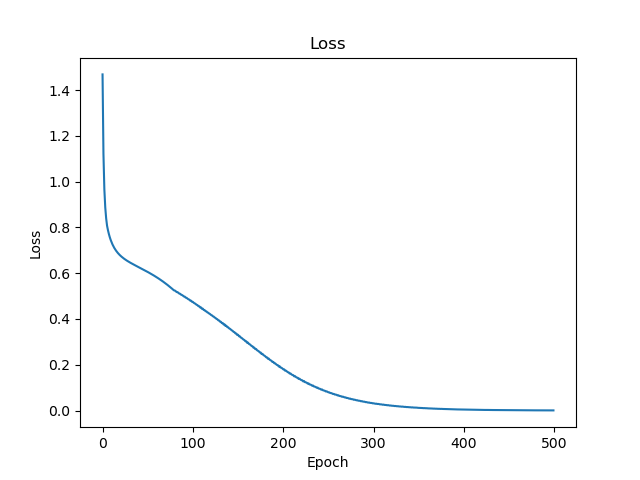
Question 3

To use backward propagation, we need to calculate the gradients:

And implement in the code.

The code uses pandas for visualization, it can be easily installed via `pip install pandas`

To run the code, open a terminal window, cd into the directory where you stored the script, and run it with `python3 Q3.py`, the code will generate 5 png files.

Here are the plots from my run with 500 epochs:

A graph with a line

Description automatically generated

A graph with a line drawn on it

Description automatically generatedA graph of weight loss

Description automatically generatedA graph of weight loss

Description automatically generated

Challenges:

Initially I tried creating the weights and biases history by simply appending the numpy array to a list, I was surprised to see that the result was the same number. My mistake was appending the actual array which is static in memory instead of a copy of it.

Going from calculus to python code was challenging, I ended taking a matlab approach of representing everything as a matrix.