Deep Leaning

Exercise 1

Assignment Code:

3.

Output:

X size: torch.Size([2000, 50])

Y size: torch.Size([2000, 1])

X test size: torch.Size([100, 50])

Y test size: torch.Size([100, 1])

Number of Epochs: 40.

Model accuracy: 100%

Number of Weights:4106001

4.

A screenshot of a cell phone

Description automatically generated

Tanh is the best.

I chose Tanh to be the activation function.

6.

Output:

X size: torch.Size([2000, 50])

Y size: torch.Size([2000, 1])

X test size: torch.Size([100, 50])

Y test size: torch.Size([100, 1])

Number of Epochs: 40.

Model accuracy: 99%

Number of Weights:842001

Report

1.

I used 4 layers(1 input,2 hidden,1 output)

X size: 50

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | Input | Hidden1 | Hidden2 | output |
| #Neurons | 800 | 800 | 200 | 1 |

2.

95%-100%

3.

Not necessarily. the Model could be overfitting the sample and not generalizing well.

A screenshot of a cell phone

Description automatically generated

4. Yes. See image at #3.

5. The **diff** parameter differentiate between the classes.

The 2 classes. The more **diff** is bigger the more the model can easily differentiate between the classes. Meaning less epochs are needed.

And a simple model could be used.

For example I used diff = 5

Y size: torch.Size([2000, 1])

X test size: torch.Size([100, 50])

Y test size: torch.Size([100, 1])

Number of Epochs: **10**.

Model accuracy: 100%

Number of Weights:842001