Lab0 - Warm Up

Wen-Jie Tseng (0556146)
Deep Learning and Practice - 2018 Spring Semester

1 Introduction

In this report, I will simply introduce my understanding about sigmoid function, neural networks, and backpropagation. Some descriptions of implementing the concepts above was also included. A three input nodes, three hidden nodes, and one output node, one hidden layer neural networks for training XOR gate was implemented in nn-xor.py file. Boldface means the code segments in .py file. For the detailed implementation, see source code (nn-xor.py) in .zip file.

2 Experiment Setups

2.1 Sigmoid Functions

Proof of the derivative of Sigmoid function, first we have:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{1}$$

Next, find the derivative of Sigmoid function with a tricky step, add one minus one, then we can obtain:

$$\frac{d}{dx}\sigma(x) = \frac{e^{-x}}{(1+e^{-x})^2} = (\frac{1+e^{-x}-1}{1+e^{-x}})(\frac{1}{1+e^{-x}}) = (1-\sigma(x))(\sigma(x)) \tag{2}$$

The equation (2) helps us in finding gradients of neural networks.

2.2 Neural Networks

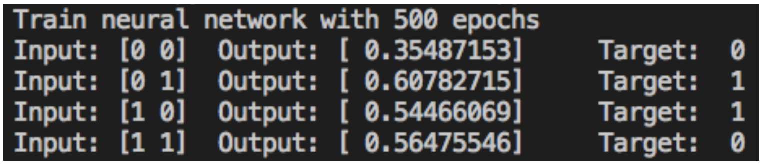
The neural networks was implemented as a **neural_network** class, which contains the neurons declared in **__init__**. There were two weight matrices, from input layer to hidden layer and from hidden layer to output layer. The activation units for input, hidden, and output layers are also declared in **__init__**. The weight matrices must be assigned some randomized values through the **randomized_matrix** function. At last, we created two arrays for recording weights in backpropagation. All the variables above were declared as array type in numpy.

2.3 Backpropagation

The process of backpropagation was divided into two functions, 1) **forward_propagation** and 2) **backward_propagation**. The **forward_propagation** means computing the final output of a network. The input unit has to multiply weight matrix between input layer and hidden layer. Update these values to units in hidden layer. Numpy provides arithmetic within arrays saving much efforts instead of using for loop. In **backward_propagation**, we are ultimately interesting in the gradient of each input neuron. By chain rule, we can derive the gradient of each neuron from the output layer, hidden layer, and back to input layer. Multiply the gradient and activation unit of each layer, we can update this value into weight matrix. One complete process including forward and backward name as epoch. We can run a certain amount of epochs, and check the training result.

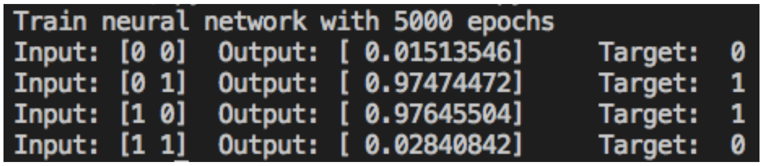
3 Results

I trained the networks with 500, 5000, and 50000 epochs. As shown in Figure 1, the output values still ranged from targets. While increasing epochs from 500 to 5000 (Figure 2) and 50000 (Figure 3), one can observe the outputs converged towards targets.



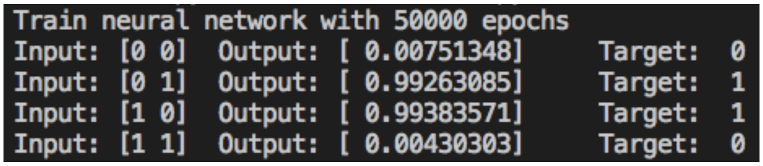
```
Train neural network with 500 epochs
Input: [0 0]  Output: [ 0.35487153]     Target:  0
Input: [0 1]  Output: [ 0.60782715]     Target:  1
Input: [1 0]  Output: [ 0.54466069]     Target:  1
Input: [1 1]  Output: [ 0.56475546]     Target:  0
```

Figure 1: Training neural network with 500 epochs.



```
Train neural network with 5000 epochs
Input: [0 0]  Output: [ 0.01513546]     Target:  0
Input: [0 1]  Output: [ 0.97474472]     Target:  1
Input: [1 0]  Output: [ 0.97645504]     Target:  1
Input: [1 1]  Output: [ 0.02840842]     Target:  0
```

Figure 2: Training neural network with 5000 epochs.



```
Train neural network with 50000 epochs
Input: [0 0]  Output: [ 0.00751348]     Target:  0
Input: [0 1]  Output: [ 0.99263085]     Target:  1
Input: [1 0]  Output: [ 0.99383571]     Target:  1
Input: [1 1]  Output: [ 0.00430303]     Target:  0
```

Figure 3: Training neural network with 50000 epochs.

4 Discussion

As one can see in the previous section, after increasing epochs, the result became much better. When I was training network with 500 epochs, the output was varied each time. When using 50000 epochs, the outputs were stable and very close to targets.

I am a master student and my research interests is Human-Computer Interaction. For the final project, I hope we can build an artistic project such like Deep Style, producing paintings or music. Or I would like to investigate applying deep learning technique to computer vision and build a system can have attention.

5 References

1. CS231: Backpropagation, Intuitions
2. "OR" example provided in Lab0 materials