Home Work #3

API (Class: NeuralNetwork)

The NeuralNetwork class consists of five methods for executing the standard feedforward and backpropagation passes using python. A valid initializing of class could be **nn= NeuralNetwork(4,4,2)**. On initializing the class, two dictionaries **network** and **dE_Theta** will be created. The network will be populated with the matrices of weights (parameters) randomly initialized using numpy random number generator. The **dE_Theta** will hold all the gradients as calculated using the **SGD** during back propagation. The forward propagation passes were implemented using sigmoid nonlinearities, while back propagation used SGD to calculate the change in the weights with respect to the mean square error calculated using the predefined target values. Finally, the parameters were updated using the **updateParams** function with an **eta** (learning rate) as input. A valid call of getLayer, forward, forward2D, backward, and updateParams methods could be **getLayer(0)**, **forward(1D Column Tensor)** and **forward2D(2D Tensor)**, **backwards(target)**, **updateParams(eta)** respectively. The input tensor of forward and forward2D function will be automatically appended with '1s' to accommodate the bias values.

Secondary API (logic_gates):

This API contains four classes for AND, OR, NOT, and EXOR logic gates. Each class constructor called the NeuralNetwork API and initialized the net with random weights. A function t**rain()** was created to call the forward, backward, updateParams function from the NeuralNetwork API in all four classes. The train function generated the input tensor using the different combination of Boolean (True, False) values along with the "and, or, not or combination of these" for target calculation. The input tensor and target were converted to integers. The valid initialization of any of these classes could be **And=AND()**. The train function will be called like **And.train()**. Below are the tables comparing the manually adjusted weights and weights learnt. The EXOR gate was the most difficult to learn with a lot of adjustments in learning rate.

AND gate		
	Learned Weights	Manual Weights
Bias	-9.12929487/-7.09996893	-15
1st Weight	6.04586438/4.67501323	10
2nd Weight	6.00817235/4367124873	10

OR gate		
	Learned Weights	Manual Weights
Bias	-2.5024063 /-2.38398222	-5
1st Weight	5.48629118/5.00331366	10
2nd Weight	5.54339968/9.41786322	10

Not gate		
	Learned Weights	Manual Weights
Bias	8.35154257/2.7174827	5
1st Weight	-11.9624607/-5.66034312	-10

EXOR gate		
	Learned Weights	Manual Weights

1st Layer		
1st Neuron		
Bias-1	-0.60678445	-10
1st Weight	1.1726274	20
2nd Weight	1.13926089	20
2nd Neuron		
Bias-2	3.63177333	30
3rd Weight	-2.0595226	-20
4th Weight	-2.105067	-20
2nd Layer		
Bias-1	-3.2905667	-30
1st Weight	2.589783	20
2nd Weight	2.4171351	20