Home Work #2

API (Class: NeuralNetwork)

The NeuralNetwork class consists of three methods for executing the standard neural net feedforward pass using python. The class can be initialized using a list of input arguments with the first and last elements being the size (integer value) of input layer and output layer respectively. The class can handle the variable number of hidden layers and their size should be mentioned in between input and output layer. A valid initializing could be **nn= NeuralNetwork(4,4,2)**. On initializing the class, a network (dictionary) will be populated with the matrices of weights (parameters) randomly initialized using numpy random number generator. The first row each matrix contains the weights for the bias term. Three methods of this class included **getLayer** for accessing the weights of an individual layer (integer value), **forward** for performing forward propagation on a single "column" vector, and **forward2D** for performing a forward pass on 2D input tensor. All forward propagation passes were implemented using sigmoid nonlinearities. A valid call of getLayer, forward, and forward2D methods could be **getLayer(0)**, **forward(1D Column Tensor)** and **forward2D(2D Tensor)**, respectively (See the test.py for more information). 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 class and then the weights of each layer of network were manually set using the getLayer function of NeuralNetwork class. The input tensor for all these four classes were Boolean (True, False) and were converted to integers. The forward propagation on the adjusted weights and input tensor was performed by calling forward method of NeuralNetwork

class. The final results were parsed back to Boolean values. The valid initialization of any of these classes could be **And=AND()** and forward propagation could be performed using the **And(True,True)**. Same rules imply for other classes in this API.