JI Xuanjin

16251180

COMP3046\_PhasE3

Design and Implementation of an Artificial Neural Network

1. ANN Class Design
   1. Globe Variable:
      1. **float r**: learning rate
      2. **int epochs:** epochs
      3. **int mbz**: number of batchs
      4. **int layers**: number of layers not including input layer
      5.  : Vector to store the number of neurons of each layer
      6.  : 2D-Vector to store the neurons not including input layer
      7.  : 2D-Vector to store the all errors
      8.  : 2D-Vector to store the all bias
      9.  : 3D-Vector to store the all weights
   2. Function:
      1.  : Three functions to set learning rate, number of epochs and number of mini-batch
      2.  : Bias and Weights initialization for random float number in (-0.5, 0.5)
      3.  : Use activation function to calculate the value of each neuron; Calculate the derivative of each neuron for future backpropagation
      4.  : 2D-vector to store the all neurons
      5.  : Calculate the value of each neuron from first hidden to last layer
      6.  : Calculate last layer’s error according to loss function and neurons’ value sigmoid derivative of last layer as well as the ground truth 
      7.  : Calculate each neuron’s error from second last year to first hidden layer according to current layer’s neurons’ error and (current+1) layer’s weights and error
      8. : Core part of ANN

First, random shuffle the Train set.

For each epoch, build a new Grad\_W and Grad\_B to temporarily store the sum of gradients.

Divide the training set into different batches. For each sample of training data set, run the **feedforward(),** **Output\_error()** and **Backpropate\_error()**

From last layer to first hidden layer, calculate sum of gradients and store them into Grad\_B and Grad\_W.

After each batch, update the weight and bias

* + 1.  : Calculate the loss value
    2.  : Test the trained weight and bias. For each test sample, use **feedforward()** and return prediction value

1. Main function

User could choose **Train ANN** or **Test ANN** service

1. Experimental Result
   1. After many times experiments, total **3 layers** (1 hidden layer) and hidden layer has **30 neurons**, **learning rate is 0.2** and there are **50 epochs**, **10 mini-batch** with **4000** sample which can get the better result.
   2. The accuracy is (0.7, 0.89)
   3. Loss value can reach (0.2,0.3) at the end of training
   4. CPU time per epoch is around (33000, 34000) milliseconds (Using RRS638 computer)
2. Contribution

As I handle it as the individual project for phase 3 and phase 4, thus the contribution is 100%