Report - **Assignment2**
CS519F16
Xi Yu

1. **A function that evaluates the trained network, as well as computes all the subgradients using backpropagation.**
   Fun: Cross_entropy(prob_class, labels)

2. **A function that performs stochastic mini-batch gradient descent training**
   Fun: update(self,
   delta_nextlayer,
   g_inputs_nextlayer = None,
   learning_rate = 0.02,
   momentum = 0.8)

3. **The best accuracy**
   The best accuracy of training data = 0.85
   The best accuracy of testing data = 0.80

4. **Code Description**
   See source code file, FNN_classification_python.py

5. **Training Monitoring**
   Please run the source code, FNN_classification_python.py

6. **Tuning Parameters**

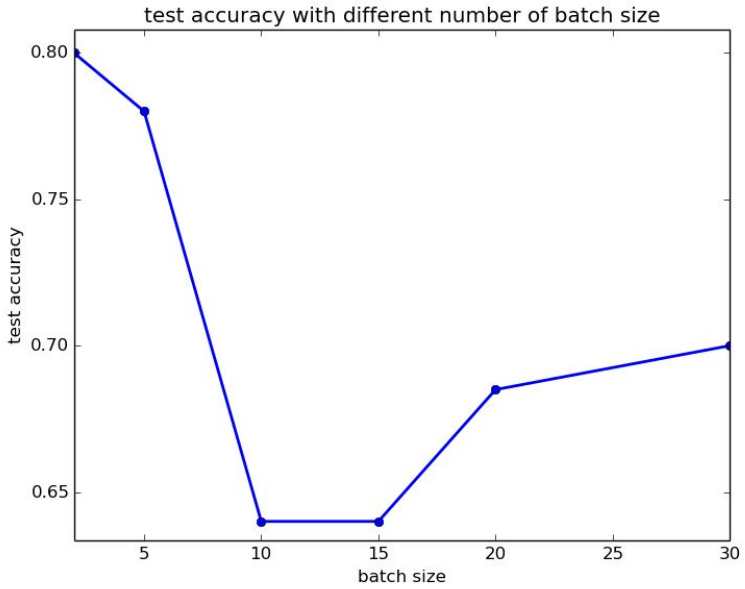

Figure 1. different number of batch size

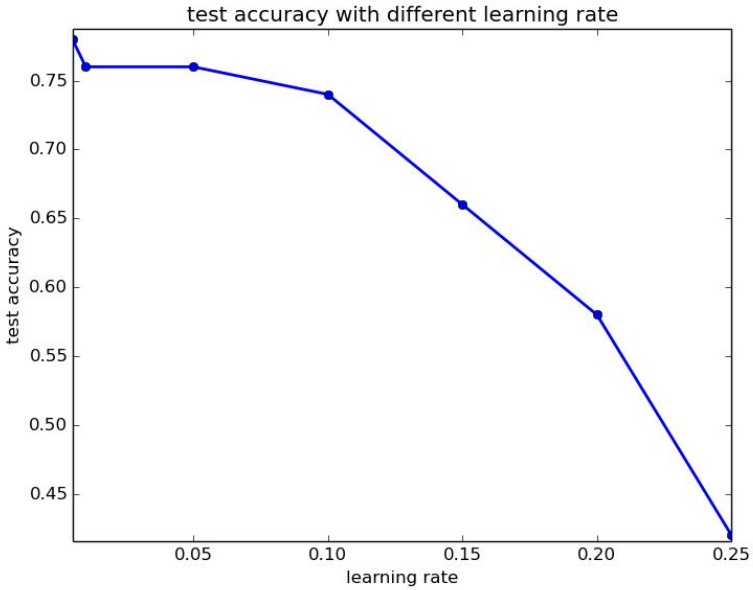

Figure 2. Different learning rate

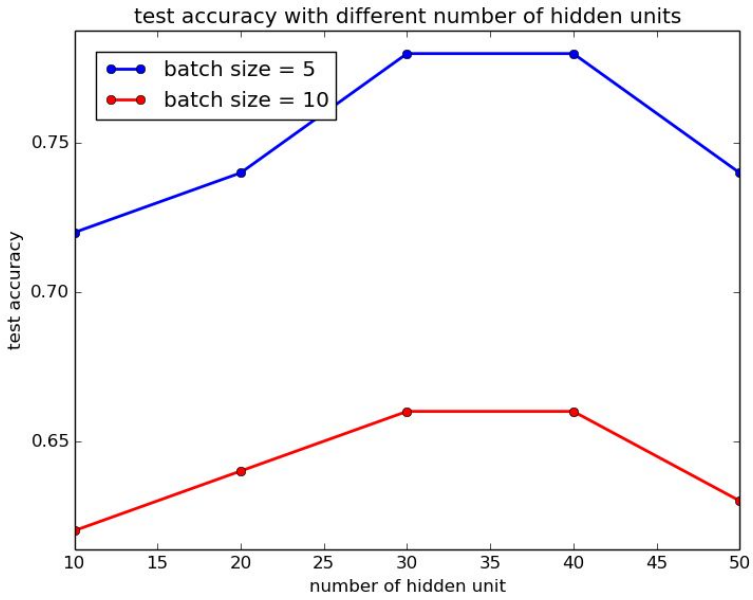

Figure 3. Different number of hidden units

7. Discussion about the performance of your neural network

This neural network perform the highest test accuracy when batchSize≈2, learningRate≈0.01, #ofHiddenUnits≈35.

Batch size affects the test accuracy. Small and large batch size lead to high test accuracy. Small batch size is like an online gradient descent algorithm. And large batch size is working like a full batch algorithm. Some batch size might reduce the classification accuracy.

Learning rate also affects the test accuracy. The accuracy reduces as the learning rate increases in figure 2. Because large learning rate causes the convergence oscillation. So the test accuracy may never achieve the optimal point.

The number of hidden unit slightly affects the test accuracy. In figure 3, there is a peak representing the optimal number of hidden unit. But the slope of the peak is not very large.

BONUS:

This neural network is constructed on layer modular. It is easy to add layers by instant layer classes. I

Take input: setInput(input)

Returns output: Output()

Returns gradients w.r.t input: Layer_module.g_inputs

Returns gradients w.r.t parameter: Layer_module.delta