Multi-Layer Perceptron

Zhuoran Liu

January 30, 2017

1 Introduction

In this report, we will consider the neural network multi-layer perceptron. Multi-layer perceptron is a neural network with multiple hidden layers and in each hidden layer there are multiple neurons. Between different neurons in adjacent layers, there exist weights to connect different neurons. In every hidden layer, there exist biases to tune some outcomes of neurons. The aim of training the network is to find the proper weights and biases which can be used to predict the new data.

1.1 Underlying Theory

Firstly, we will talk about the working mechanism of the MLP. Given the neural network structure(weights matrix W and biases vector b) and input data vector a, we can calculate the feed forward process by formula

$$\mathbf{z} = g(\mathbf{w}a + \mathbf{b})$$

Here g is the activation function and output z is a vector. Use z as the input of the next layer, we can do similar calculation again until we get the output. Given the last output z, we will calculate the $argmax(z)$. The category of the $argmax(z)$ is the predication of the input a. Known the right category, we will tune the weights and biases to predict better and better. This is the learning process of MLP. Given the input

1.2 Learning Algorithm

Backpropagation algorithm is the learning algorithm we will use. It consists of 4 steps.

$$\delta^L = \nabla_a C \circ \sigma'(z^L)$$

$$\delta^L = ((w^{l+1})^T \delta^{l+1} \circ \sigma'(z^l)$$

$$\frac{\partial C}{\partial b^l_j} = \sigma^l_j$$

$$\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \sigma^l_j$$

This four step will backpropagate error and output the gradient change of cost function. The whole process is input, feedforward, backpropagate, update weights and biases, input again... Finally after many epoches we will get the final neural network with particular weights and biases, then we finish training.

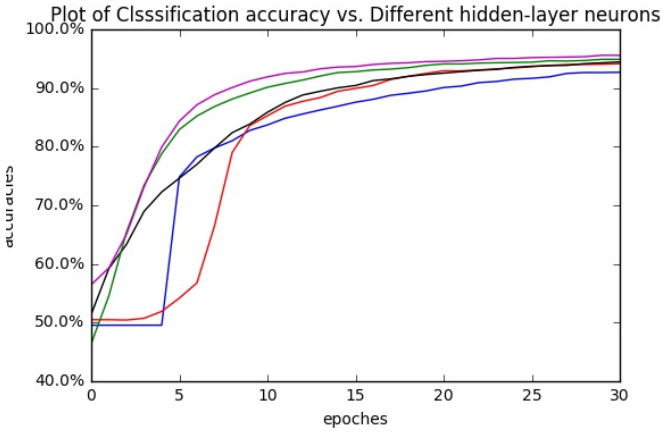
2 Problem statement

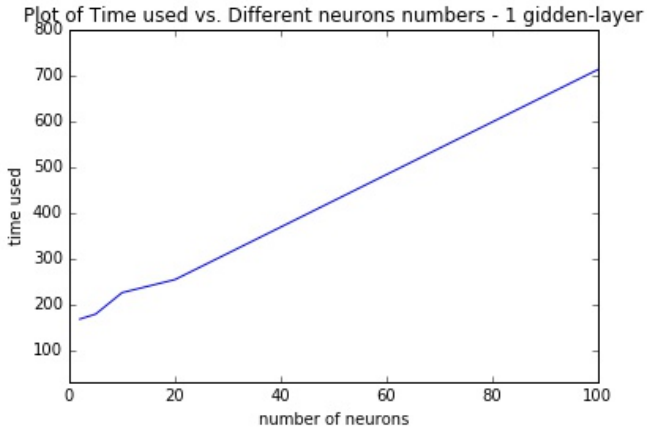
1. The influence of different layers and different number of neurons.
2. The different initializations of the weights and biases before learning.
3. Different chooses of activation functions.
4. different learning method Stochastic Gradient Descent vs. Momentum.

3 Results

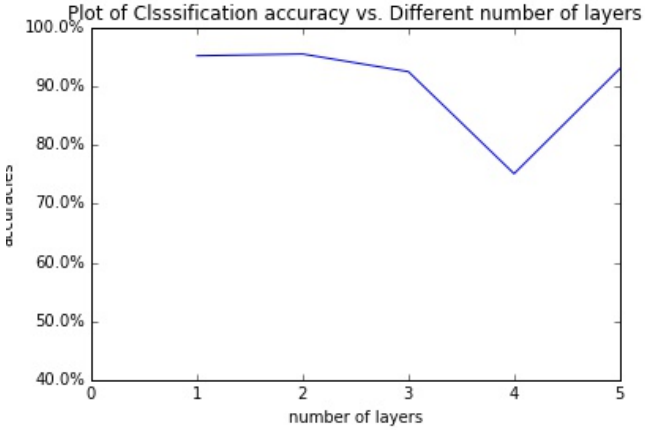
3.1 The structure of Multi-Layer Perceptron

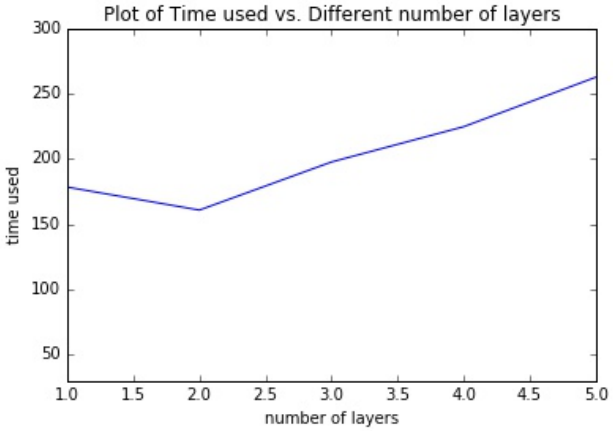
3.1.1 Different neurons





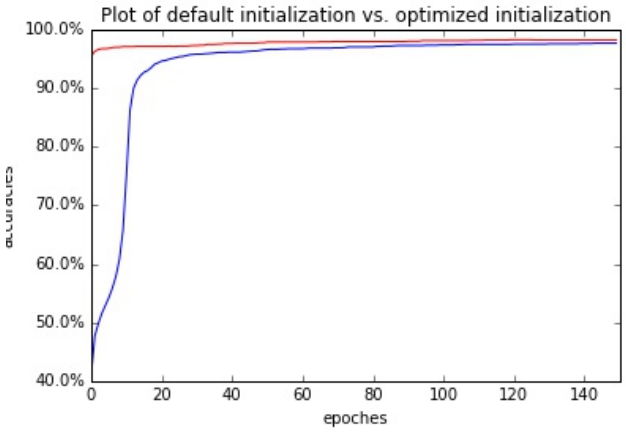
3.1.2 Different layers



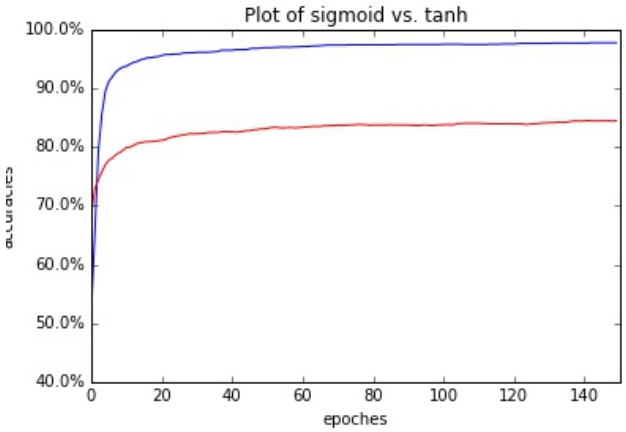


3.2 The initialization of the weights and biases

3.2.1 different initialization of weights and biases



3.2.2 different activation functions



3.3 Different learning methods

3.3.1 Stochastic gradient descent

3.3.2 Momentum learning

4 Discussion

5 Conclusion

Multi-layer perceptron is perceptron which can be trained by various patterns. It is not easy to be trained well. This training process depended on many aspects. For example number of neurons, number of layers, initialization of weights and biases, exact learning method, learning rate, size of mini-batches and so on. From what we have explored, more hidden layers and more neurons involved will have better results, but it will also take more time. The time has a similar linear relationship with the number of neurons and the number of hidden layers. A good initialization can obviously improve the initial accuracy of the learning process, and it also help tune the weights and biases better. Different activation functions have different influences on the accuracies of test, and they have also different running time. The choose of learning rate, epoches and mini batch sizes are also import parts. But they can be investigated by some experiments and choose the comparable better one.

6 Appendix