Script 1: GradientDescentOptimizer

set standard deviation of initial weights to .05

bias = .1

first layer use ReLU with 4096 nodes

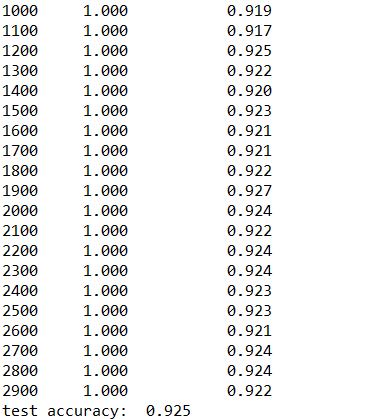
second layer use ReLU with 80 nodes, apply dropouts

third layer use 10 nodes

apply regularization

learning rate = 0.5

accuracy = 0.925



Script 2 AadamOptimizer

set standard deviation of initial weights to .05

bias = .1

first layer use ReLU with 4096 nodes

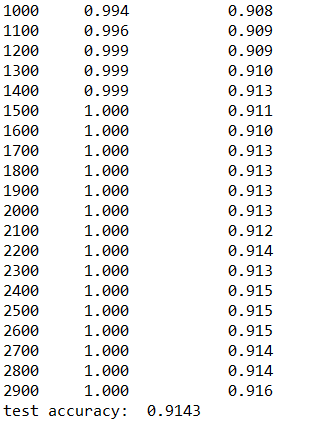
second layer use ReLU with 80 nodes, apply dropouts

third layer use 10 nodes

apply regularization

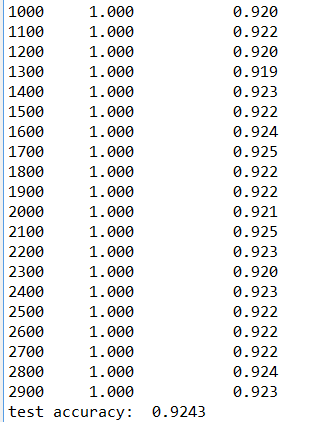
learning rate = 0.5

accuracy = 0.9143



Script 3

Add rate decay, other conditions are same as script 1



Script 4

Convolutional layer works well in accuracy, but pretty time consuming

Set standard deviation as 0.1

Test accuracy = 0.956

