Machine Learning HW5 Report

Henry Chang 10/30/2020

I set the parameters as follows to train the network. Setting activation types to tanh() had more consistent and smaller training error than logistic(). Both functions types give roughly the same relative error. Setting the learning rate to eta = .1 gives the fastest convergence from experiment.

```
if TEST_TYPE == 'xor':
    net = mlp(inputs =xordata[:,0:2],
                targets=xordata[:,2:3],
                nhid1=2, nhid2=2,
                outtype='tanh',
                hidtype='tanh')
    net.fit(xordata[:,0:2] ,xordata[:,2:3],0.1,10000)
    # fit(input, output, eta, niter )
```

With h = 1-e5, the relative errors of gradient check for all weights are << e-7, so our implementation should be correct.

```
RelErr w3
 [[4.51300334e-12 4.99913276e-11 1.26295502e-10]]
RelErr w2
 [[2.07181755e-11 2.71888915e-12 1.62510524e-12]
 [8.48010247e-12 1.41489074e-12 1.30940543e-10]]
RelErr w1
 [[0.00000000e+00 0.00000000e+00 2.13615673e-11]
 [0.00000000e+00 0.00000000e+00 7.68613762e-11]]
```

The training error are also small consistently after 5000 iterations for multiple runs.

```
Iter   5306   Pttn   0   Error 0.000000028
Iter   5306   Pttn   1   Error 0.000969039
Iter   5306   Pttn   2   Error 0.001003454
Iter   5306   Pttn   3   Error 0.000005628

Iter   9999   Pttn   0   Error 0.000000005
Iter   9999   Pttn   1   Error 0.000053108
Iter   9999   Pttn   2   Error 0.000055707
Iter   9999   Pttn   3   Error 0.000000002
```

Observing from the above two results of relative and training errors, we see that the network has successfully learned to classify the XOR problem with high accuracy.