**CSC 578 Project 2**

**Yiyang Yang**

Early stopping

Cost functions ×

Quadratic ×

Cross-entropy ×

log-likelihood ×

allow choice of cost function with a parameter ×

Momentum ×

L2 Regularization ×

Better initial weights ×

Transfer functions ×

tanh

softmax

ReLU

Minibatch shuffling ×

Returning learned network ×

Returning accuracy and costs for plotting ×

Did NOT include the MNIST data with my submission ×

Incomplete, Details \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Not sure, Details \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Instructions**:

New variables:

split: Percentage of train, test and validation datasets.

momentum: momentum coefficient.

actFunction: activation function.

costFunction: cost function.

l2Lambda: lambda of L2 regularization.

LearnedNet: If it is in the learned neural network.

**Description**:

This is similar neural network as the first project, but add some new algorithm and functions, such as momentum and L2 regularization.

**Code**:

%Date: 10/21/2017

%Author: Yiyang Yang

function [ weight, bias, accuracy, cost ] = Yiyang\_NNet2(inputs, targets, nodeLayers, numEpochs, batchSize, eta, split, momentum, actFunction, costFunction, l2Lambda, learnedNet)

%Get train, test and valid data sets.

[train, test, valid] = dividerand(length(inputs), split(1), split(2), split(3));

inputTrain = inputs(:, train);

inputTest = inputs(:, test);

inputValid = inputs(:, valid);

targetTrain = targets(:, train);

targetTest = targets(:, test);

targetValid = targets(:, valid);

%To check Train sets in learned network

if learnedNet == 'None'

weight = {};

weightD = {};

bias = {};

biasD = {};

for i = 2: length(nodeLayers)

weight{i} = normrnd(0, 1/sqrt(length(inputTrain)), nodeLayers(i), nodeLayers(i - 1));

bias{i} = normrnd(0,1/sqrt(length(inputTrain)), nodeLayers(i), 1);

end

else

nodeLayers = learnedNet{0};

weight = learnedNet{1};

bias = learnedNet{2};

end

accuracy = {};

cost = {};

%Batching

b = {};

targetB = {};

n = 1;

for i =1 : batchSize : length(inputTrain)

if length(inputTrain) - i < batchSize

miniB = inputTrain(:, i : end);

b{n} = miniB;

target = targetTrain(:, i: end);

targetB{n} = target;

else

miniB = inputTrain(:, i: i + batchSize - 1);

b{n} = miniB;

target = targetTrain(:, i: i + batchSize - 1);

targetB{n} = target;

n = n + 1;

end

end

costTrain = [];

costTest = [];

costValid = [];

if strcmp(actFunction, 'softmax') == 1

actHide = input('Choose activation function for hidden layers:\n');

elseif strcmp(actFunction, 'relu') == 1

actLast = input('Choose activation function for ouput layers:\n');

end

fprintf(' | TRAIN || TEST || VALIDATION \n');

fprintf('--------------------------------------------------------------------------\n');

fprintf('Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc\n');

fprintf('--------------------------------------------------------------------------\n');

%Epochs

for p = 1 : numEpochs

bRandom = randperm(length(b));

bN = 1;

for q = 1 : length(b)

z = {};

activation = {};

activation{1} = b{bRandom(bN)};

delta = {};

for l = 2 : length(nodeLayers)

z{l} = bsxfun(@plus, (weight{l} \* activation{l - 1}), bias{l});

if strcmp(actFunction, 'sigmoid') == 1

activation{l} = logsig(z{l});

elseif strcmp(actFunction, 'tanh') == 1

activation{l} = tanh(z{l});

elseif strcmp(actFunction, 'softmax') == 1

if l ~= length(nodeLayers)

activation{l} = ft(z{l}, actHide);

else

activation{l} = ft(z{l}, actFunction);

end

elseif strcmp(actFunction, 'relu') == 1

if l ~= length(nodeLayers)

activation{l} = max(0, z{l});

else

activation{l} = ft(z{l}, actLast);

end

end

end

%Apply Backpropagation

e = (activation{length(nodeLayers)} - targetB{bRandom(bN)});

if strcmp(actFunction, 'tanh') == 1

tanhP = 1 - tanh(z{length(nodeLayers)}) .^2;

delta{length(nodeLayers)} = e .\* tanhP;

elseif strcmp(actFunction, 'softmax') == 1

delta{length(nodeLayers)} = e .\* ones(size(z{length(nodeLayers)}));

elseif strcmp(actFunction, 'relu') == 1

if strcmp(actLast, 'softmax') == 1

delta{length(nodeLayers)} = e .\* ones(size(z{length(nodeLayers)}));

elseif strcmp(actLast, 'softmax') == 0

reluP = dFt(z{length(nodeLayers)}, actLast);

delta{length(nodeLayers)} = e .\* reluP;

end

elseif strcmp(actFunction, 'sigmoid') == 1

lg = logsig(z{length(nodeLayers)});

sigP = lg .\* (1 - lg);

delta{length(nodeLayers)} = e .\* sigP;

end

%Apply gradient descent and get new wegiht and bias

for l = (length(nodeLayers) - 1) : -1 : 2

if strcmp(actFunction, 'softmax') == 1

delta{l} = (weight{l + 1}.' \* delta{l + 1}) .\* dFt(z{l}, actHide);

elseif strcmp(actFunction, 'softmax') == 0

delta{l} = (weight{l + 1}.' \* delta{l + 1}) .\* dFt(z{l}, actFunction);

end

end

for l = length(nodeLayers) : -1 : 2

if p == 1 && q == 1

weight{l} = weight{l} - eta / length(b{bRandom(bN)}) \* delta{l} \* activation{l - 1}.';

weightD{l} = eta / length(b{bRandom(bN)}) \* delta{l} \* activation{l - 1}.';

bias{l} = bias{l} - eta / length(b{bRandom(bN)}) \* sum(delta{l}, 2);

biasD{l} = eta / length(b{bRandom(bN)}) \* sum(delta{l}, 2);

else

weight{l} = weight{l} + weightD{l};

weightD{l} = momentum .\* weightD{l} - eta / length(b{bRandom(bN)}) \* delta{l} \* activation{l - 1}.';

bias{l} = bias{l} + biasD{l};

biasD{l} = momentum .\* biasD{l} - eta / length(b{bRandom(bN)}) \* sum(delta{l}, 2);

end

end

bN = bN + 1;

end

%Get outputs

outputTrain = {};

outputTest = {};

outputValid = {};

outputTrain{1} = inputTrain;

outputTest{1} = inputTest;

outputValid{1} = inputValid;

for l = 2 : length(nodeLayers)

zTrain = bsxfun(@plus, (weight{l} \* outputTrain{l - 1}), bias{l});

zTest = bsxfun(@plus, (weight{l} \* outputTest{l - 1}), bias{l});

zValid = bsxfun(@plus, (weight{l} \* outputValid{l - 1}), bias{l});

if strcmp(actFunction, 'softmax') == 0

outputTrain{l} = ft(zTrain, actFunction);

outputTest{l} = ft(zTest, actFunction);

outputValid{l} = ft(zValid, actFunction);

elseif strcmp(actFunction, 'relu') == 1

if l == length(nodeLayers)

outputTrain{l} = ft(zTrain, actLast);

outputTest{l} = ft(zTest, actLast);

outputValid{l} = ft(zValid, actLast);

else

outputTrain{l} = max(0, zTrain);

outputTest{l} = max(0, zTest);

outputValid{l} = max(0, zValid);

end

else

if l == length(nodeLayers)

outputTrain{l} = ft(zTrain, actFunction);

outputTest{l} = ft(zTest, actFunction);

outputValid{l} = ft(zValid, actFunction);

else

outputTrain{l} = ft(zTrain, actHide);

outputTest{l} = ft(zTest, actHide);

outputValid{l} = ft(zValid, actHide);

end

end

end

%Correct cases and accuracy rate of train, test and validation

if size(targets, 1) <= 1

aTrain = targetTrain - round(outputTrain{length(nodeLayers)});

cTrain = sum(aTrain(:) == 0);

accTrain = cTrain / size(inputTrain, 2);

aTest = targetTest - round(outputTest{length(nodeLayers)});

cTest = sum(aTest(:) == 0);

accTest = cTest / size(inputTest, 2);

aValid = targetValid - round(outputValid{length(nodeLayers)});

cValid = sum(aValid(:) == 0);

accValid = cValid / size(inputValid, 2);

else

[iTrain, vTrain] = max(outputTrain{length(nodeLayers)});

[itTrain, vtTrain] = max(targetTrain);

aTrain = vtTrain - vTrain;

cTrain = sum(aTrain(:) == 0);

accTrain = cTrain / length(inputTrain);

[iTest, vTest] = max(outputTest{length(nodeLayers)});

[itTest, vtTest] = max(targetTest);

aTest = vtTest - vTest;

cTest = sum(aTest(:) == 0);

accTest = cTest / length(inputTest);

[iValid, vValid] = max(outputValid{length(nodeLayers)});

[itValid, vtValid] = max(targetValid);

aValid = vtValid - vValid;

cValid = sum(aValid(:) == 0);

accValid = cValid / length(inputValid);

end

weightSum = 0;

for l = 2 : length(nodeLayers)

weightSum = weightSum + sum(sum(weight{l} .^2));

end

l2Train = l2Lambda/(2 \* size(inputTrain, 2)) \* weightSum;

l2Test = l2Lambda/(2 \* size(inputTest, 2)) \* weightSum;

l2Valid = l2Lambda/(2 \* size(inputValid, 2)) \* weightSum;

if strcmp(costFunction, 'cross') == 1

ctTrain = l2Train - 1 / length(inputTrain) .\* sum(sum((1 - targetTrain) .\* log(1 - outputTrain{length(nodeLayers)}) + targetTrain .\* log(outputTrain{length(nodeLayers)} + eps)) + eps);

ctTest = l2Test - 1 / length(inputTest) .\* sum(sum((1 - targetTest) .\* log(1 - outputTest{length(nodeLayers)}) + targetTest .\* log(outputTest{length(nodeLayers)} + eps)) + eps);

ctValid = l2Valid - 1 / length(inputValid) .\* sum(sum((1 - targetValid) .\* log(1 - outputValid{length(nodeLayers)}) + targetValid .\* log(outputValid{length(nodeLayers)} + eps)) + eps);

elseif strcmp(costFunction, 'quad') == 1

ctTrain = l2Train + 1 / sum(sum((1/2 \* (targetTrain - outputTrain{length(nodeLayers)}) .^2))) \* (2 \* size(inputTrain, 2));

ctTest = l2Test + 1 / sum(sum((1/2 \* (targetTest - outputTest{length(nodeLayers)}) .^2))) \* (2 \* size(inputTest, 2));

ctValid = l2Valid + 1 / sum(sum((1/2 \* (targetValid - outputValid{length(nodeLayers)}) .^2))) \* (2 \* size(inputValid, 2));

elseif strcmp(costFunction, 'log') == 1

ctTrain = l2Train + sum(eps - log(max(outputTrain{length(nodeLayers)})) / size(inputTrain, 2));

ctTest = l2Test + sum(eps - log(max(outputTest{length(nodeLayers)})) / size(inputTest, 2));

ctValid = l2Valid + sum(eps - log(max(outputValid{length(nodeLayers)})) / size(inputValid, 2));

end

costTrain(p) = ctTrain;

costTest(p) = ctTest;

costValid(p) = ctValid;

fprintf('%d | %0.3f | %d/%d | %0.3f || %0.3f | %d/%d | %0.3f || %0.3f | %d/%d | %0.3f\n', p, costTrain(p), cTrain, size(inputTrain, 2), accTrain, costTest(p), cTest, size(inputTest, 2), accTest, costValid(p), cValid, size(inputValid, 2), accValid);

%Accuracy of train, test and validation

accuracy{1}(p) = accTrain;

accuracy{2}(p) = accTest;

accuracy{3}(p) = accValid;

%Cost of train, test and validation

cost{1}(p) = costTrain(p);

cost{2}(p) = costTest(p);

cost{3}(p) = costValid(p);

%Early stopping

if cTrain == length(inputTrain) && cTest == length(inputTest) && cValid == length(inputValid)

finputf('Good model.\n');

subplot(1, 2, 1);

plot(accuracy{1});

hold on;

plot(accuracy{2}); plot(accuracy{3});

title('Accuracy'); xlabel('Number of Epoches'); ylabel('Accuracy');

legend('Train', 'Test', 'Valid');

hold off;

subplot(1, 2, 2);

plot(cost{1});

hold on;

plot(cost{2}); plot(cost{3});

title('Cost'); xlabel('Number of Epoches'); ylabel('Cost');

legend('Train', 'Test', 'Valid');

hold off;

break

elseif p >= round(numEpochs \* 0.8)

if costValid(p) > costValid(p - 1)

fprintf('Early Stopping at 80 percent epochs.\n');

subplot(1, 2, 1);

plot(accuracy{1});

hold on;

plot(accuracy{2}); plot(accuracy{3});

title('Accuracy'); xlabel('Number of Epoches'); ylabel('Accuracy');

legend('Train', 'Test', 'Valid');

hold off;

subplot(1, 2, 2);

plot(cost{1});

hold on;

plot(cost{2}); plot(cost{3});

title('Cost'); xlabel('Number of Epoches'); ylabel('Cost');

legend('Train', 'Test', 'Valid');

hold off;

break

end

end

subplot(1, 2, 1);

plot(accuracy{1});

hold on;

plot(accuracy{2}); plot(accuracy{3});

title('Accuracy'); xlabel('Number of Epoches'); ylabel('Accuracy');

legend('Train', 'Test', 'Valid');

hold off;

subplot(1, 2, 2);

plot(cost{1});

hold on;

plot(cost{2}); plot(cost{3});

title('Cost'); xlabel('Number of Epoches'); ylabel('Cost');

legend('Train', 'Test', 'Valid');

hold off;

end

end

%Activation function in different case

function act = ft(x, y)

switch y

case 'sigmoid'

act = logsig(x);

case 'tanh'

act = tanh(x);

case 'softmax'

act = softmax(x);

case 'relu'

act = max(0, x);

end

end

%Derivative activation function in different case

function dAct = dFt(x, y)

switch y

case 'sigmoid'

dAct = ft(x, y) .\* (1 - ft(x, y));

case 'tanh'

dAct = 1 - ft(x, y) .^2;

case 'softmax'

dAct = ft(x, y) .\* (1 - ft(x, y));

case 'relu'

dAct = double(x > 0);

end

end

**Analysis**:

This neural network I create could work on the three datasets provided. But to iris dataset the accuracy was not that good than the accuracy of minist and xor dataset. Transform function sigmoid and momentum 0.3 get better results than relu with momentum 0.3.

**Ideas for enhancement:**

This neural network still a simple network, I think there are still need some new features. Such as search, it will be more useful to find out what parameters will be good fit to the specific dataset. I think I still need to learn more about neural network to find out something new and add it to the network.

**Outputs**:

*Iris Data:*

| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iris.csv | 40 | 20 | 10 | 0.1 | sigmoid | cross | .3 | 5 |

>> Yiyang\_NNet2(input, target, [4, 20, 3], 40, 10, 0.1, [80, 10, 10], 0.3, 'sigmoid', 'cross', 5, 'None');

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 1.975 | 41/120 | 0.342 || 2.079 | 6/15 | 0.400 || 2.236 | 3/15 | 0.200

2 | 1.920 | 82/120 | 0.683 || 2.025 | 13/15 | 0.867 || 2.227 | 5/15 | 0.333

3 | 1.901 | 42/120 | 0.350 || 2.075 | 7/15 | 0.467 || 2.107 | 2/15 | 0.133

4 | 1.889 | 82/120 | 0.683 || 2.030 | 13/15 | 0.867 || 2.199 | 5/15 | 0.333

5 | 1.880 | 48/120 | 0.400 || 2.072 | 7/15 | 0.467 || 2.128 | 3/15 | 0.200

6 | 1.871 | 82/120 | 0.683 || 2.028 | 13/15 | 0.867 || 2.230 | 5/15 | 0.333

7 | 1.856 | 82/120 | 0.683 || 2.067 | 13/15 | 0.867 || 2.151 | 5/15 | 0.333

8 | 1.844 | 82/120 | 0.683 || 2.072 | 13/15 | 0.867 || 2.164 | 5/15 | 0.333

9 | 1.836 | 52/120 | 0.433 || 2.095 | 7/15 | 0.467 || 2.148 | 4/15 | 0.267

10 | 1.814 | 82/120 | 0.683 || 2.060 | 13/15 | 0.867 || 2.228 | 5/15 | 0.333

11 | 1.798 | 82/120 | 0.683 || 2.074 | 13/15 | 0.867 || 2.231 | 5/15 | 0.333

12 | 1.782 | 82/120 | 0.683 || 2.102 | 13/15 | 0.867 || 2.225 | 5/15 | 0.333

13 | 1.764 | 82/120 | 0.683 || 2.103 | 13/15 | 0.867 || 2.279 | 5/15 | 0.333

14 | 1.742 | 82/120 | 0.683 || 2.124 | 13/15 | 0.867 || 2.281 | 5/15 | 0.333

15 | 1.724 | 91/120 | 0.758 || 2.150 | 14/15 | 0.933 || 2.294 | 8/15 | 0.533

16 | 1.706 | 82/120 | 0.683 || 2.147 | 13/15 | 0.867 || 2.384 | 5/15 | 0.333

17 | 1.679 | 82/120 | 0.683 || 2.172 | 13/15 | 0.867 || 2.408 | 5/15 | 0.333

18 | 1.654 | 82/120 | 0.683 || 2.198 | 13/15 | 0.867 || 2.452 | 5/15 | 0.333

19 | 1.635 | 82/120 | 0.683 || 2.215 | 13/15 | 0.867 || 2.538 | 5/15 | 0.333

20 | 1.612 | 82/120 | 0.683 || 2.264 | 13/15 | 0.867 || 2.548 | 5/15 | 0.333

21 | 1.590 | 82/120 | 0.683 || 2.291 | 13/15 | 0.867 || 2.617 | 5/15 | 0.333

22 | 1.582 | 82/120 | 0.683 || 2.286 | 13/15 | 0.867 || 2.823 | 5/15 | 0.333

23 | 1.547 | 82/120 | 0.683 || 2.373 | 13/15 | 0.867 || 2.736 | 5/15 | 0.333

24 | 1.538 | 82/120 | 0.683 || 2.379 | 13/15 | 0.867 || 2.931 | 5/15 | 0.333

25 | 1.511 | 82/120 | 0.683 || 2.465 | 13/15 | 0.867 || 2.875 | 5/15 | 0.333

26 | 1.498 | 82/120 | 0.683 || 2.504 | 13/15 | 0.867 || 2.955 | 5/15 | 0.333

27 | 1.484 | 82/120 | 0.683 || 2.568 | 13/15 | 0.867 || 2.982 | 5/15 | 0.333

28 | 1.479 | 82/120 | 0.683 || 2.608 | 13/15 | 0.867 || 3.098 | 5/15 | 0.333

29 | 1.465 | 82/120 | 0.683 || 2.639 | 13/15 | 0.867 || 3.216 | 5/15 | 0.333

30 | 1.448 | 82/120 | 0.683 || 2.728 | 13/15 | 0.867 || 3.150 | 5/15 | 0.333

31 | 1.440 | 86/120 | 0.717 || 2.801 | 13/15 | 0.867 || 3.164 | 5/15 | 0.333

32 | 1.432 | 87/120 | 0.725 || 2.852 | 13/15 | 0.867 || 3.215 | 5/15 | 0.333

Early Stopping at 80 percent epochs.

Plot:



| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iris.csv | 40 | 20 | 10 | 0.1 | relu | cross | .3 | 5 |

>> Yiyang\_NNet2(input, target, [4, 20, 3], 40, 10, 0.1, [80, 10, 10], 0.3, 'relu', 'cross', 5, 'None');

Choose activation function for ouput layers:

'softmax'

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 18.379 | 42/120 | 0.350 || 18.474 | 5/15 | 0.333 || 19.136 | 3/15 | 0.200

2 | 13.874 | 37/120 | 0.308 || 10.020 | 7/15 | 0.467 || 16.929 | 6/15 | 0.400

3 | 19.125 | 42/120 | 0.350 || 21.969 | 5/15 | 0.333 || 18.342 | 3/15 | 0.200

4 | 13.940 | 37/120 | 0.308 || 10.224 | 7/15 | 0.467 || 17.088 | 6/15 | 0.400

5 | 3.017 | 41/120 | 0.342 || 4.247 | 3/15 | 0.200 || 4.037 | 6/15 | 0.400

6 | 2.902 | 41/120 | 0.342 || 4.071 | 3/15 | 0.200 || 4.036 | 6/15 | 0.400

7 | 3.397 | 42/120 | 0.350 || 5.004 | 5/15 | 0.333 || 4.836 | 3/15 | 0.200

8 | 3.490 | 42/120 | 0.350 || 4.610 | 5/15 | 0.333 || 5.005 | 3/15 | 0.200

9 | 12.991 | 42/120 | 0.350 || 19.453 | 5/15 | 0.333 || 17.354 | 3/15 | 0.200

10 | 12.752 | 42/120 | 0.350 || 19.336 | 5/15 | 0.333 || 17.101 | 3/15 | 0.200

11 | 14.023 | 42/120 | 0.350 || 10.399 | 5/15 | 0.333 || 17.241 | 3/15 | 0.200

12 | 3.210 | 41/120 | 0.342 || 4.732 | 3/15 | 0.200 || 4.503 | 6/15 | 0.400

13 | 13.263 | 42/120 | 0.350 || 19.575 | 5/15 | 0.333 || 17.746 | 3/15 | 0.200

14 | 3.047 | 41/120 | 0.342 || 4.347 | 3/15 | 0.200 || 4.096 | 6/15 | 0.400

15 | 12.772 | 42/120 | 0.350 || 19.358 | 5/15 | 0.333 || 17.096 | 3/15 | 0.200

16 | 12.823 | 42/120 | 0.350 || 19.398 | 5/15 | 0.333 || 17.116 | 3/15 | 0.200

17 | 12.804 | 41/120 | 0.342 || 19.392 | 3/15 | 0.200 || 17.094 | 6/15 | 0.400

18 | 14.093 | 42/120 | 0.350 || 10.521 | 5/15 | 0.333 || 17.345 | 3/15 | 0.200

19 | 12.829 | 42/120 | 0.350 || 19.400 | 5/15 | 0.333 || 17.126 | 3/15 | 0.200

20 | 12.825 | 42/120 | 0.350 || 19.389 | 5/15 | 0.333 || 17.136 | 3/15 | 0.200

21 | 3.151 | 41/120 | 0.342 || 4.576 | 3/15 | 0.200 || 4.271 | 6/15 | 0.400

22 | 3.054 | 41/120 | 0.342 || 4.347 | 3/15 | 0.200 || 4.091 | 6/15 | 0.400

23 | 2.916 | 37/120 | 0.308 || 4.010 | 7/15 | 0.467 || 4.081 | 6/15 | 0.400

24 | 12.829 | 42/120 | 0.350 || 19.369 | 5/15 | 0.333 || 17.184 | 3/15 | 0.200

25 | 14.058 | 42/120 | 0.350 || 10.446 | 5/15 | 0.333 || 17.271 | 3/15 | 0.200

26 | 3.108 | 42/120 | 0.350 || 4.225 | 5/15 | 0.333 || 4.473 | 3/15 | 0.200

27 | 3.418 | 41/120 | 0.342 || 5.052 | 3/15 | 0.200 || 4.703 | 6/15 | 0.400

28 | 12.833 | 42/120 | 0.350 || 19.380 | 5/15 | 0.333 || 17.169 | 3/15 | 0.200

29 | 3.361 | 41/120 | 0.342 || 4.713 | 3/15 | 0.200 || 4.270 | 6/15 | 0.400

30 | 3.080 | 42/120 | 0.350 || 4.314 | 5/15 | 0.333 || 4.451 | 3/15 | 0.200

31 | 2.996 | 41/120 | 0.342 || 4.288 | 3/15 | 0.200 || 4.085 | 6/15 | 0.400

32 | 3.011 | 41/120 | 0.342 || 4.381 | 3/15 | 0.200 || 4.252 | 6/15 | 0.400

Early Stopping at 80 percent epochs.

Plot:



| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iris.csv | 40 | 20 | 10 | 0.1 | relu | cross | 0 | 5 |

>> Yiyang\_NNet2(input, target, [4, 20, 3], 40, 10, 0.1, [80, 10, 10], 0, 'relu', 'cross', 5, 'None');

Choose activation function for ouput layers:

'softmax'

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 11.374 | 87/120 | 0.725 || 22.695 | 6/15 | 0.400 || 20.578 | 7/15 | 0.467

2 | 36.075 | 44/120 | 0.367 || 36.292 | 1/15 | 0.067 || 36.292 | 5/15 | 0.333

3 | 24.101 | 43/120 | 0.358 || 26.334 | 5/15 | 0.333 || 33.073 | 2/15 | 0.133

4 | 23.965 | 44/120 | 0.367 || 34.202 | 1/15 | 0.067 || 25.330 | 5/15 | 0.333

5 | 32.168 | 44/120 | 0.367 || 36.418 | 1/15 | 0.067 || 26.025 | 5/15 | 0.333

6 | 23.683 | 44/120 | 0.367 || 34.214 | 1/15 | 0.067 || 25.077 | 5/15 | 0.333

7 | 22.930 | 44/120 | 0.367 || 34.217 | 1/15 | 0.067 || 24.543 | 5/15 | 0.333

8 | 11.584 | 85/120 | 0.708 || 23.271 | 6/15 | 0.400 || 20.197 | 7/15 | 0.467

9 | 11.566 | 84/120 | 0.700 || 23.247 | 6/15 | 0.400 || 20.189 | 7/15 | 0.467

10 | 13.245 | 78/120 | 0.650 || 25.582 | 5/15 | 0.333 || 24.618 | 5/15 | 0.333

11 | 11.893 | 82/120 | 0.683 || 25.555 | 5/15 | 0.333 || 22.576 | 6/15 | 0.400

12 | 1.401 | 87/120 | 0.725 || 4.051 | 6/15 | 0.400 || 3.155 | 7/15 | 0.467

13 | 0.983 | 87/120 | 0.725 || 3.248 | 6/15 | 0.400 || 2.433 | 7/15 | 0.467

14 | 0.850 | 87/120 | 0.725 || 3.063 | 6/15 | 0.400 || 2.286 | 7/15 | 0.467

15 | 0.739 | 87/120 | 0.725 || 2.868 | 6/15 | 0.400 || 2.111 | 7/15 | 0.467

16 | 0.702 | 87/120 | 0.725 || 2.936 | 6/15 | 0.400 || 2.196 | 7/15 | 0.467

17 | 0.649 | 87/120 | 0.725 || 2.839 | 6/15 | 0.400 || 2.105 | 7/15 | 0.467

18 | 0.656 | 87/120 | 0.725 || 2.988 | 6/15 | 0.400 || 2.262 | 7/15 | 0.467

19 | 0.586 | 87/120 | 0.725 || 2.687 | 6/15 | 0.400 || 1.944 | 7/15 | 0.467

20 | 0.600 | 87/120 | 0.725 || 2.760 | 6/15 | 0.400 || 2.045 | 7/15 | 0.467

21 | 0.574 | 87/120 | 0.725 || 2.798 | 6/15 | 0.400 || 2.069 | 7/15 | 0.467

22 | 0.597 | 87/120 | 0.725 || 2.849 | 6/15 | 0.400 || 2.135 | 7/15 | 0.467

23 | 0.557 | 87/120 | 0.725 || 2.779 | 6/15 | 0.400 || 2.034 | 7/15 | 0.467

24 | 0.565 | 87/120 | 0.725 || 2.901 | 6/15 | 0.400 || 2.156 | 7/15 | 0.467

25 | 0.575 | 87/120 | 0.725 || 2.894 | 6/15 | 0.400 || 2.149 | 7/15 | 0.467

26 | 0.568 | 87/120 | 0.725 || 2.855 | 6/15 | 0.400 || 2.096 | 7/15 | 0.467

27 | 0.613 | 87/120 | 0.725 || 3.098 | 6/15 | 0.400 || 2.345 | 7/15 | 0.467

28 | 0.575 | 87/120 | 0.725 || 2.903 | 6/15 | 0.400 || 2.126 | 7/15 | 0.467

29 | 0.578 | 87/120 | 0.725 || 2.913 | 6/15 | 0.400 || 2.123 | 7/15 | 0.467

30 | 0.613 | 87/120 | 0.725 || 3.036 | 6/15 | 0.400 || 2.253 | 7/15 | 0.467

31 | 0.678 | 87/120 | 0.725 || 3.280 | 6/15 | 0.400 || 2.496 | 7/15 | 0.467

32 | 0.610 | 87/120 | 0.725 || 3.000 | 6/15 | 0.400 || 2.182 | 7/15 | 0.467

33 | 0.660 | 87/120 | 0.725 || 3.197 | 6/15 | 0.400 || 2.384 | 7/15 | 0.467

Early Stopping at 80 percent epochs.

Plot:



MNIST Data:

| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MNIST | 30 | 30 | 10 | 3.0 | sigmoid | quad | .3 | 5 |

>> load('D:\CSC 578\Project 2\mnistTrn.mat');

>> Yiyang\_NNet2(trn, trnAns, [784, 30, 10], 30, 10, 3, [80, 10, 10], 0.3, 'sigmoid', 'quad', 5, 'None');

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 4.462 | 4524/40000 | 0.113 || 4.463 | 556/5000 | 0.111 || 4.467 | 602/5000 | 0.120

2 | 5.613 | 18221/40000 | 0.456 || 5.643 | 2279/5000 | 0.456 || 5.679 | 2307/5000 | 0.461

3 | 8.652 | 28815/40000 | 0.720 || 8.750 | 3602/5000 | 0.720 || 8.774 | 3591/5000 | 0.718

4 | 12.079 | 33825/40000 | 0.846 || 12.370 | 4258/5000 | 0.852 || 12.159 | 4217/5000 | 0.843

5 | 15.725 | 35216/40000 | 0.880 || 16.358 | 4420/5000 | 0.884 || 15.737 | 4398/5000 | 0.880

6 | 18.706 | 35683/40000 | 0.892 || 19.582 | 4480/5000 | 0.896 || 18.657 | 4466/5000 | 0.893

7 | 20.999 | 36021/40000 | 0.901 || 22.029 | 4516/5000 | 0.903 || 20.947 | 4511/5000 | 0.902

8 | 22.729 | 36149/40000 | 0.904 || 23.874 | 4543/5000 | 0.909 || 22.688 | 4524/5000 | 0.905

9 | 24.270 | 36354/40000 | 0.909 || 25.394 | 4556/5000 | 0.911 || 24.134 | 4554/5000 | 0.911

10 | 25.572 | 36498/40000 | 0.912 || 26.682 | 4572/5000 | 0.914 || 25.391 | 4563/5000 | 0.913

11 | 26.659 | 36606/40000 | 0.915 || 27.716 | 4582/5000 | 0.916 || 26.415 | 4580/5000 | 0.916

12 | 27.730 | 36704/40000 | 0.918 || 28.718 | 4601/5000 | 0.920 || 27.442 | 4594/5000 | 0.919

13 | 28.652 | 36808/40000 | 0.920 || 29.549 | 4605/5000 | 0.921 || 28.222 | 4596/5000 | 0.919

14 | 29.553 | 36908/40000 | 0.923 || 30.326 | 4621/5000 | 0.924 || 29.013 | 4612/5000 | 0.922

15 | 30.331 | 36969/40000 | 0.924 || 31.084 | 4624/5000 | 0.925 || 29.798 | 4619/5000 | 0.924

16 | 31.148 | 37041/40000 | 0.926 || 31.793 | 4631/5000 | 0.926 || 30.474 | 4627/5000 | 0.925

17 | 31.845 | 37118/40000 | 0.928 || 32.358 | 4641/5000 | 0.928 || 31.027 | 4631/5000 | 0.926

18 | 32.587 | 37195/40000 | 0.930 || 33.012 | 4644/5000 | 0.929 || 31.697 | 4637/5000 | 0.927

19 | 33.306 | 37260/40000 | 0.931 || 33.646 | 4652/5000 | 0.930 || 32.305 | 4647/5000 | 0.929

20 | 33.950 | 37306/40000 | 0.933 || 34.194 | 4663/5000 | 0.933 || 32.816 | 4650/5000 | 0.930

21 | 34.494 | 37349/40000 | 0.934 || 34.649 | 4668/5000 | 0.934 || 33.267 | 4650/5000 | 0.930

22 | 35.180 | 37398/40000 | 0.935 || 35.263 | 4668/5000 | 0.934 || 33.868 | 4667/5000 | 0.933

23 | 35.803 | 37430/40000 | 0.936 || 35.725 | 4670/5000 | 0.934 || 34.287 | 4668/5000 | 0.934

24 | 36.367 | 37488/40000 | 0.937 || 36.266 | 4680/5000 | 0.936 || 34.774 | 4668/5000 | 0.934

Early Stopping at 80 percent epochs.

Plot:



| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MNIST | 30 | 30 | 10 | 3.0 | softmax | log | .3 | 0 |

>> Yiyang\_NNet2(trn, trnAns, [784, 30, 10], 30, 10, 3, [80, 10, 10], 0.3, 'softmax', 'log', 0, 'None');

Choose activation function for hidden layers:

'sigmoid'

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 0.228 | 35985/40000 | 0.900 || 0.239 | 4442/5000 | 0.888 || 0.232 | 4482/5000 | 0.896

2 | 0.143 | 36884/40000 | 0.922 || 0.150 | 4550/5000 | 0.910 || 0.147 | 4588/5000 | 0.918

3 | 0.123 | 37326/40000 | 0.933 || 0.132 | 4597/5000 | 0.919 || 0.126 | 4637/5000 | 0.927

4 | 0.107 | 37648/40000 | 0.941 || 0.116 | 4658/5000 | 0.932 || 0.109 | 4679/5000 | 0.936

5 | 0.096 | 37993/40000 | 0.950 || 0.105 | 4697/5000 | 0.939 || 0.096 | 4713/5000 | 0.943

6 | 0.086 | 38102/40000 | 0.953 || 0.095 | 4704/5000 | 0.941 || 0.089 | 4720/5000 | 0.944

7 | 0.083 | 38240/40000 | 0.956 || 0.091 | 4729/5000 | 0.946 || 0.085 | 4736/5000 | 0.947

8 | 0.075 | 38367/40000 | 0.959 || 0.084 | 4737/5000 | 0.947 || 0.076 | 4746/5000 | 0.949

9 | 0.074 | 38475/40000 | 0.962 || 0.082 | 4745/5000 | 0.949 || 0.074 | 4761/5000 | 0.952

10 | 0.071 | 38621/40000 | 0.966 || 0.079 | 4758/5000 | 0.952 || 0.071 | 4777/5000 | 0.955

11 | 0.067 | 38667/40000 | 0.967 || 0.075 | 4764/5000 | 0.953 || 0.068 | 4768/5000 | 0.954

12 | 0.064 | 38745/40000 | 0.969 || 0.072 | 4768/5000 | 0.954 || 0.065 | 4777/5000 | 0.955

13 | 0.061 | 38806/40000 | 0.970 || 0.069 | 4772/5000 | 0.954 || 0.062 | 4777/5000 | 0.955

14 | 0.058 | 38903/40000 | 0.973 || 0.067 | 4769/5000 | 0.954 || 0.059 | 4782/5000 | 0.956

15 | 0.057 | 38917/40000 | 0.973 || 0.064 | 4781/5000 | 0.956 || 0.058 | 4784/5000 | 0.957

16 | 0.055 | 38968/40000 | 0.974 || 0.063 | 4790/5000 | 0.958 || 0.056 | 4785/5000 | 0.957

17 | 0.053 | 38977/40000 | 0.974 || 0.061 | 4788/5000 | 0.958 || 0.055 | 4789/5000 | 0.958

18 | 0.052 | 39042/40000 | 0.976 || 0.060 | 4778/5000 | 0.956 || 0.053 | 4789/5000 | 0.958

19 | 0.051 | 39093/40000 | 0.977 || 0.060 | 4787/5000 | 0.957 || 0.053 | 4800/5000 | 0.960

20 | 0.050 | 39091/40000 | 0.977 || 0.058 | 4786/5000 | 0.957 || 0.053 | 4791/5000 | 0.958

21 | 0.048 | 39130/40000 | 0.978 || 0.056 | 4786/5000 | 0.957 || 0.050 | 4801/5000 | 0.960

22 | 0.046 | 39186/40000 | 0.980 || 0.055 | 4793/5000 | 0.959 || 0.048 | 4801/5000 | 0.960

23 | 0.044 | 39233/40000 | 0.981 || 0.052 | 4788/5000 | 0.958 || 0.048 | 4798/5000 | 0.960

24 | 0.044 | 39215/40000 | 0.980 || 0.053 | 4798/5000 | 0.960 || 0.048 | 4803/5000 | 0.961

Early Stopping at 80 percent epochs.

Plot:



| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MNIST | 30 | 30 | 10 | 1.0 | softmax | log | .3 | 5 |

>> Yiyang\_NNet2(trn, trnAns, [784, 30, 10], 30, 10, 3, [80, 10, 10], 0.3, 'softmax', 'log', 5, 'None');

Choose activation function for hidden layers:

'sigmoid'

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 0.238 | 35919/40000 | 0.898 || 0.327 | 4511/5000 | 0.902 || 0.331 | 4498/5000 | 0.900

2 | 0.166 | 36857/40000 | 0.921 || 0.307 | 4594/5000 | 0.919 || 0.307 | 4597/5000 | 0.919

3 | 0.148 | 37299/40000 | 0.932 || 0.320 | 4656/5000 | 0.931 || 0.321 | 4650/5000 | 0.930

4 | 0.137 | 37650/40000 | 0.941 || 0.342 | 4693/5000 | 0.939 || 0.342 | 4672/5000 | 0.934

5 | 0.131 | 37833/40000 | 0.946 || 0.364 | 4716/5000 | 0.943 || 0.365 | 4706/5000 | 0.941

6 | 0.127 | 38049/40000 | 0.951 || 0.386 | 4736/5000 | 0.947 || 0.388 | 4717/5000 | 0.943

7 | 0.124 | 38173/40000 | 0.954 || 0.407 | 4754/5000 | 0.951 || 0.409 | 4744/5000 | 0.949

8 | 0.125 | 38316/40000 | 0.958 || 0.433 | 4751/5000 | 0.950 || 0.435 | 4753/5000 | 0.951

9 | 0.123 | 38448/40000 | 0.961 || 0.456 | 4759/5000 | 0.952 || 0.457 | 4763/5000 | 0.953

10 | 0.121 | 38559/40000 | 0.964 || 0.475 | 4782/5000 | 0.956 || 0.478 | 4777/5000 | 0.955

11 | 0.123 | 38543/40000 | 0.964 || 0.500 | 4773/5000 | 0.955 || 0.501 | 4767/5000 | 0.953

12 | 0.122 | 38676/40000 | 0.967 || 0.521 | 4785/5000 | 0.957 || 0.521 | 4789/5000 | 0.958

13 | 0.123 | 38701/40000 | 0.968 || 0.540 | 4793/5000 | 0.959 || 0.543 | 4794/5000 | 0.959

14 | 0.123 | 38850/40000 | 0.971 || 0.561 | 4793/5000 | 0.959 || 0.562 | 4796/5000 | 0.959

15 | 0.123 | 38874/40000 | 0.972 || 0.583 | 4796/5000 | 0.959 || 0.583 | 4803/5000 | 0.961

16 | 0.124 | 38924/40000 | 0.973 || 0.602 | 4800/5000 | 0.960 || 0.603 | 4792/5000 | 0.958

17 | 0.125 | 38991/40000 | 0.975 || 0.623 | 4798/5000 | 0.960 || 0.624 | 4815/5000 | 0.963

18 | 0.127 | 39010/40000 | 0.975 || 0.643 | 4809/5000 | 0.962 || 0.643 | 4808/5000 | 0.962

19 | 0.127 | 39056/40000 | 0.976 || 0.662 | 4802/5000 | 0.960 || 0.663 | 4818/5000 | 0.964

20 | 0.128 | 39096/40000 | 0.977 || 0.680 | 4806/5000 | 0.961 || 0.681 | 4805/5000 | 0.961

21 | 0.130 | 39158/40000 | 0.979 || 0.700 | 4802/5000 | 0.960 || 0.701 | 4816/5000 | 0.963

22 | 0.130 | 39175/40000 | 0.979 || 0.718 | 4813/5000 | 0.963 || 0.719 | 4814/5000 | 0.963

23 | 0.132 | 39207/40000 | 0.980 || 0.737 | 4809/5000 | 0.962 || 0.738 | 4820/5000 | 0.964

24 | 0.133 | 39260/40000 | 0.982 || 0.756 | 4805/5000 | 0.961 || 0.755 | 4820/5000 | 0.964

Early Stopping at 80 percent epochs.

Plot:



*XOR Data:*

| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| xor.csv | 20 | [3 2] | 1 | 0.1 | sigmoid | cross | .3 | 5 |

>> inputx = [0 0 1 1; 0 1 0 1];

>> targetx = [0 1 1 0];

>> Yiyang\_NNet2(inputx, targetx, [2, 3, 2, 1], 20, 1, 0.1, [50, 25, 25], 0.3, 'sigmoid', 'cross', 5, 'None');

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 7.164 | 2/2 | 1.000 || 14.578 | 0/1 | 0.000 || 14.593 | 0/1 | 0.000

2 | 7.164 | 2/2 | 1.000 || 14.584 | 0/1 | 0.000 || 14.598 | 0/1 | 0.000

3 | 7.165 | 2/2 | 1.000 || 14.591 | 0/1 | 0.000 || 14.606 | 0/1 | 0.000

4 | 7.166 | 2/2 | 1.000 || 14.599 | 0/1 | 0.000 || 14.614 | 0/1 | 0.000

5 | 7.167 | 2/2 | 1.000 || 14.607 | 0/1 | 0.000 || 14.622 | 0/1 | 0.000

6 | 7.168 | 2/2 | 1.000 || 14.615 | 0/1 | 0.000 || 14.630 | 0/1 | 0.000

7 | 7.169 | 2/2 | 1.000 || 14.623 | 0/1 | 0.000 || 14.638 | 0/1 | 0.000

8 | 7.169 | 2/2 | 1.000 || 14.630 | 0/1 | 0.000 || 14.646 | 0/1 | 0.000

9 | 7.170 | 2/2 | 1.000 || 14.638 | 0/1 | 0.000 || 14.654 | 0/1 | 0.000

10 | 7.171 | 2/2 | 1.000 || 14.646 | 0/1 | 0.000 || 14.662 | 0/1 | 0.000

11 | 7.172 | 2/2 | 1.000 || 14.654 | 0/1 | 0.000 || 14.670 | 0/1 | 0.000

12 | 7.173 | 2/2 | 1.000 || 14.661 | 0/1 | 0.000 || 14.677 | 0/1 | 0.000

13 | 7.174 | 2/2 | 1.000 || 14.669 | 0/1 | 0.000 || 14.685 | 0/1 | 0.000

14 | 7.175 | 2/2 | 1.000 || 14.676 | 0/1 | 0.000 || 14.693 | 0/1 | 0.000

15 | 7.176 | 2/2 | 1.000 || 14.684 | 0/1 | 0.000 || 14.701 | 0/1 | 0.000

16 | 7.177 | 2/2 | 1.000 || 14.692 | 0/1 | 0.000 || 14.708 | 0/1 | 0.000

Early Stopping at 80 percent epochs.

Plot:



| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| xor.csv | 20 | [3 2] | 1 | 0.1 | tanh | cross | .3 | 5 |

>> Yiyang\_NNet2(inputx, targetx, [2, 3, 2, 1], 20, 1, 0.1, [50, 25, 25], 0.3, 'tanh', 'cross', 5, 'None');

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 11.818 | 0/2 | 0.000 || 21.764 | 1/1 | 1.000 || 21.515 | 1/1 | 1.000

2 | 11.680 | 0/2 | 0.000 || 21.886 | 0/1 | 0.000 || 21.432 | 1/1 | 1.000

3 | 11.596 | 1/2 | 0.500 || 21.850 | 0/1 | 0.000 || 21.361 | 1/1 | 1.000

4 | 11.480 | 0/2 | 0.000 || 21.691 | 0/1 | 0.000 || 21.280 | 1/1 | 1.000

5 | 11.409 | 0/2 | 0.000 || 21.645 | 0/1 | 0.000 || 21.240 | 1/1 | 1.000

6 | 11.351 | 1/2 | 0.500 || 21.626 | 0/1 | 0.000 || 21.222 | 1/1 | 1.000

7 | 11.304 | 1/2 | 0.500 || 21.623 | 0/1 | 0.000 || 21.223 | 1/1 | 1.000

8 | 11.267 | 1/2 | 0.500 || 21.636 | 0/1 | 0.000 || 21.241 | 1/1 | 1.000

9 | 11.271 | 1/2 | 0.500 || 21.753 | 0/1 | 0.000 || 21.274 | 1/1 | 1.000

10 | 11.221 | 2/2 | 1.000 || 21.715 | 0/1 | 0.000 || 21.322 | 1/1 | 1.000

11 | 11.233 | 2/2 | 1.000 || 21.833 | 0/1 | 0.000 || 21.371 | 1/1 | 1.000

12 | 11.231 | 2/2 | 1.000 || 21.901 | 0/1 | 0.000 || 21.444 | 1/1 | 1.000

13 | 11.213 | 2/2 | 1.000 || 21.908 | 0/1 | 0.000 || 21.553 | 1/1 | 1.000

14 | 11.238 | 2/2 | 1.000 || 22.056 | 0/1 | 0.000 || 21.629 | 1/1 | 1.000

15 | 11.258 | 2/2 | 1.000 || 22.163 | 0/1 | 0.000 || 21.735 | 1/1 | 1.000

16 | 11.271 | 2/2 | 1.000 || 22.219 | 0/1 | 0.000 || 21.886 | 1/1 | 1.000

Early Stopping at 80 percent epochs.

Plot:



| **data set** | **epochs** | **hids** | **batch** | **eta** | **trans.** | **cost** | **mom.** | **reg.** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| xor.csv | 20 | [3 2] | 1 | 0.1 | relu | cross | .3 | 5 |

>> Yiyang\_NNet2(inputx, targetx, [2, 3, 2, 1], 20, 1, 0.1, [50, 25, 25], 0.3, 'relu', 'cross', 5, 'None');

Choose activation function for ouput layers:

'sigmoid'

| TRAIN || TEST || VALIDATION

--------------------------------------------------------------------------

Ep | Cost | Corr | Acc || Cost | Corr | Acc || Cost | Corr | Acc

--------------------------------------------------------------------------

1 | 7.191 | 0/2 | 0.000 || 11.422 | 0/1 | 0.000 || 10.874 | 1/1 | 1.000

2 | 6.792 | 0/2 | 0.000 || 11.418 | 0/1 | 0.000 || 10.825 | 1/1 | 1.000

3 | 6.392 | 0/2 | 0.000 || 11.458 | 0/1 | 0.000 || 10.771 | 1/1 | 1.000

4 | 6.065 | 0/2 | 0.000 || 11.571 | 0/1 | 0.000 || 10.729 | 1/1 | 1.000

5 | 5.824 | 2/2 | 1.000 || 11.898 | 0/1 | 0.000 || 10.719 | 1/1 | 1.000

6 | 5.628 | 2/2 | 1.000 || 28.500 | 0/1 | 0.000 || 10.736 | 1/1 | 1.000

7 | 5.473 | 2/2 | 1.000 || 28.448 | 0/1 | 0.000 || 10.780 | 0/1 | 0.000

8 | 5.330 | 2/2 | 1.000 || 28.398 | 0/1 | 0.000 || 10.863 | 0/1 | 0.000

9 | 5.219 | 2/2 | 1.000 || 28.367 | 0/1 | 0.000 || 10.983 | 0/1 | 0.000

10 | 5.164 | 2/2 | 1.000 || 28.345 | 0/1 | 0.000 || 11.181 | 0/1 | 0.000

11 | 5.155 | 2/2 | 1.000 || 28.331 | 0/1 | 0.000 || 11.579 | 0/1 | 0.000

12 | 5.153 | 2/2 | 1.000 || 28.327 | 0/1 | 0.000 || 28.327 | 0/1 | 0.000

13 | 5.154 | 2/2 | 1.000 || 28.331 | 0/1 | 0.000 || 28.331 | 0/1 | 0.000

14 | 5.159 | 2/2 | 1.000 || 28.341 | 0/1 | 0.000 || 28.341 | 0/1 | 0.000

15 | 5.167 | 2/2 | 1.000 || 28.356 | 0/1 | 0.000 || 28.356 | 0/1 | 0.000

16 | 5.177 | 2/2 | 1.000 || 28.375 | 0/1 | 0.000 || 28.375 | 0/1 | 0.000

Early Stopping at 80 percent epochs.

Plot:

