4.

REPORT

a) Process:

🡪Load the training and testing data using setupmnist()

🡪Get the number of hidden nodes from user🡪100/500/1000

🡪Do the following for each number of inputs from the 60,000 training(Case 1:100,Case2:1000,Case3:10000,Case4:60000)

🡪Convert the training labels into 10X1 vectors each

for i=1:length(train.labels)

out(i,(train.labels(i)+1))=1;

end

Now **out** contains the output targets

🡪Now create the net

net\_1000\_100=mlp(784,nhidden,10,'logistic');

🡪Now create the options matrix

🡪Train the net

[net\_1000\_100,options] = netopt(**net\_1000\_100**,options,train.data',out,'scg');

**(net\_1000\_100=The network having 1000hidden nodes and 100 training samples)**

🡪Testing

🡪Feed forward network for testing samples

for i=1:length(testing.data)

y(i,:)=mlpfwd(net\_1000\_100,testing.data(:,i)');

end

[C,I] = max(y,[],2);

fans\_1000\_100=I-1;

🡪Feed forward for cross validation (Test using the remaining training samples)

for i=101:60000

y(i-100,:)=mlpfwd(net\_1000\_100,training.data(:,i)');

end

y=round(abs(y));

test2\_1000\_100=binaryVectorToDecimal(y);

🡪Training error(Run the training matrix itself as input)

y=mlpfwd(net\_100\_60000,training.data');

[C,I] = max(y,[],2);

ftans\_100\_60000=I-1;

🡪Now find the accuracy of each by comparing with base data(example)

ft=[ftans\_500\_60000, ftans\_500\_10000, ftans\_500\_1000, ftans\_500\_100, ftans\_100\_60000, ftans\_100\_10000, ftans\_100\_1000, ftans\_100\_100, ftans\_1000\_60000, ftans\_1000\_10000, ftans\_1000\_1000, ftans\_1000\_100];

for i=1:size(ft,2)

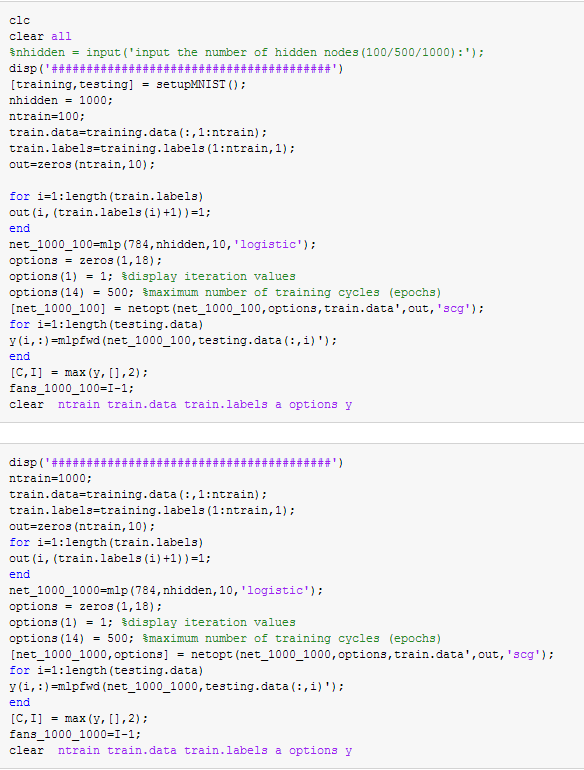
Y(:,i)=(ft(:,i)==training.labels(:,1));

end

yfinal=sum(Y)/60000;

accuracy.training=yfinal2';

##EXAMPLE CODE##





b) net.m has been saved

It was generated using

500 HIDDEN NODES

60000 NUMBER OF TRAINING DATA

0% TRAINING ERROR

1.66% TESTING ERROR

c)

During **training** we adjust the weight of the NN. The training error is actually the error of running through all the trainind data

During **Validation** when verifying if there is an increase in accuracy over data the network hasn't trained on (i.e. validation data set). If the accuracy over the training data set increases, but the accuracy over then validation data set reduces, then we over fit the NN.I used hold-out method to validate the data(ie train the network with some data and test using the remaining data)

During **Testing** we see how the NN performs in actual case.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Training | | Validation | | Testing | |
|  | Accuracy | Error | Accuracy | Error | Accuracy | Error |
| 'net\_100\_60000' | 1 | 0 | 1 | - | 0.968 | 0.032 |
| 'net\_100\_10000' | 0.953383 | 0.046617 | 0.94406 | 0.05594 | 0.9462 | 0.0538 |
| 'net\_100\_1000' | 0.881617 | 0.118383 | 0.87961 | 0.12039 | 0.8857 | 0.1143 |
| 'net\_100\_100' | 0.6831 | 0.3169 | 0.682571 | 0.317429 | 0.6766 | 0.3234 |
| 'net\_500\_60000' | 1 | 0 | 1 | - | 0.9834 | 0.0166 |
| 'net\_500\_10000' | 0.96875 | 0.03125 | 0.9625 | 0.0375 | 0.9621 | 0.0379 |
| 'net\_500\_1000' | 0.880967 | 0.119033 | 0.878949 | 0.121051 | 0.8841 | 0.1159 |
| 'net\_500\_100' | 0.685767 | 0.314233 | 0.685242 | 0.314758 | 0.6807 | 0.3193 |
| 'net\_1000\_60000' | 1 | 0 | 1 | - | 0.9843 | 0.0157 |
| 'net\_1000\_10000' | 0.96935 | 0.03065 | 0.96322 | 0.03678 | 0.9628 | 0.0372 |
| 'net\_1000\_1000' | 0.8733 | 0.1267 | 0.871153 | 0.128847 | 0.8763 | 0.1237 |
| 'net\_1000\_100' | 0.685533 | 0.314467 | 0.685008 | 0.314992 | 0.6779 | 0.3221 |
|  |  |  |  |  |  |  |

The validation accuracy for the NN with 1000 hidden nodes decreases more than the one with the 500 hidden

Nodes(ie 0.96322<0.9625)-🡪 hence we may be over fitting

Since I am getting the almost same accuracy for test data I am going to choose the one with 500 hidden nodes.

Therefore Net\_500\_60000 ie NN with 500 hidden nodes and 60000 training data shall be the “best network”.

d) IMAGE

Please run the following code where fans\_1000\_60000 is the column vector containing the output for the testing the testing data for network with 1000 hidden nodes and 60000 training data.

%PICK OUT ROWS OF WRONG ONES

for i=1:size(fans\_1000\_60000,2)

sl(:,i)=(fans\_1000\_60000(:,i)==testing.labels(:,1));

end

vl=find(sl==0);

for i=1:5

wrongones(:,i)=testing.data(:,vl(i));

end

%DISPLAY THE IMAGES IN MATLAB

for i=1:5

mat = vec2mat(wrongones(:,i),28);

mat=mat';

K = mat2gray(mat);

subplot(1,5,i)

imshow(K)

end

clear K mat vl sl

%CREATE A NEW FIGURE

Figure

%PICK OUT ROWS OF CORRECT ONES

for i=1:size(fans\_1000\_60000,2)

sl(:,i)=(fans\_1000\_60000(:,i)==testing.labels(:,1));

end

vl=find(sl==1);

for i=1:5

correctones(:,i)=testing.data(:,vl(i));

end

%DISPLAY THE IMAGES IN MATLAB

for i=1:5

mat = vec2mat(correctones(:,i),28);

mat=mat';

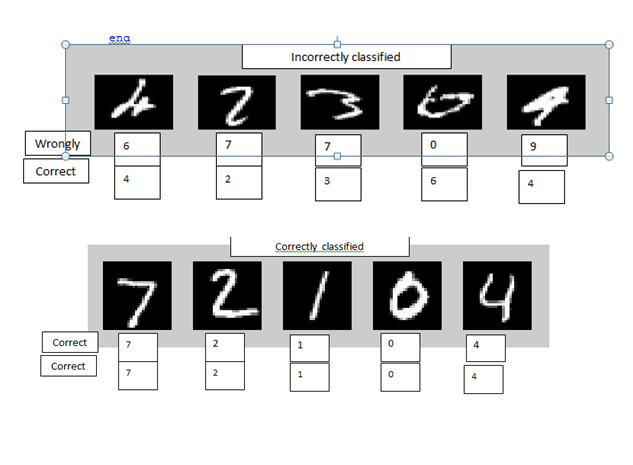
K = mat2gray(mat);

subplot(1,5,i)

imshow(K)

end

((IMAGE IN NEXT PAGE))



4) Extra credit:

🡪**Make sure u have set your default program to open bmp files are MS-PAINT**

**🡪 Save the “net” in Matlab**

**🡪 Run Following code:**

matrix=zeros(28, 28)

K = mat2gray(matrix);

imshow(K);

% A black board has been created->you can close it

imwrite(K,'ath.bmp');

system('ath.bmp');

🡪 **Now you can zoom in and draw on the board with white paint brush. Please save it after doing so(make sure it is still monochrome/16bit bmp).After saving run the following code.**

🡪 KK=imread('ath.bmp' );

%now we have to make KK into 784X1 column matrix

C = KK(:);

C =double(C);

y=mlpfwd(net,A');

[C,I] = max(y,[],2);

finalans=I-1;

C:\Users\Athma\Dropbox\Classes   +Studies\AI\Assignment 3\q4\ath2.bmp

<<Using the Test image on the right that I drew and using the net I was able to correctly predict the number 2(finalans)