Code for Classification challenge

```
% Classification challange written by: Axel Qvarnström
clear all
close all
clc
xTest2 = loadmnist2();
xTest2 = cast(xTest2,'double');
[xTrain, tTrain, xValid, tValid, xTest, tTest] = LoadMNIST(3);

% Casting the data so it becomes 4-D doubles
xTrain = cast(xTrain,'double');
xValid = cast(xValid,'double');
xTest = cast(xTest,'double');

% Preprocessing the data
trainMean = mean(xTrain,4);
trainStd = std(xTrain,1,4);

validMean = mean(xValid,4);
validStd = std(xValid,1,4);

testMean = mean(xTest,4);
testStd = std(xTest,1,4);

xTest2Mean = mean(xTest2,4);
xTest2Std = std(xTest2,1,4);

xTrain = (xTrain - trainMean)./max(max(trainStd));
xValid = (xValid - validMean)./max(max(validStd));
xTest = (xTest - testMean)./max(max(testStd));
xTest2 = (xTest2 - xTest2Mean)./max(max(xTest2Std));


% Creating the network
layers= [
    imageInputLayer([28 28 1])
    convolution2dLayer(3,64,'Padding','same')
    batchNormalizationLayer
    reluLayer

    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(3,64,'Padding','same')
    batchNormalizationLayer
    reluLayer

    maxPooling2dLayer(2,'Stride',2)
    fullyConnectedLayer(100)  % Hidden layer 1 with 100 neurons
    batchNormalizationLayer
    reluLayer

    fullyConnectedLayer(10)
    softmaxLayer
    classificationLayer];

% Options
options = trainingOptions(...
    'sgdm',...
    'ExecutionEnvironment',...
    'gpu',...
    'InitialLearnRate',0.01,...
    'Momentum',0.9,...
    'MaxEpochs',30,...
    'MiniBatchSize',100,...
    'ValidationFrequency',30,...
    'ValidationPatience',5,...
    'ValidationData',{xValid,tValid});

% Training the network
trainingNetwork = trainNetwork(xTrain,tTrain,layers,options);
save trainingNetwork       % Saving the network, so I don't need to run
                           % it again

%% Making the classification with the saved network written above
load trainingNetwork.mat   % Getting the above saved network
                           % to run this section


% Just to see the accuracy of classify the xTest data
classifyXTest = classify(trainingNetwork,xTest);
classificationAccuracy1 = sum(classifyXTest==tTest)/numel(tTest);

% Classify xTest2 and write the result to a csv file
classifyXTest2 = classify(trainingNetwork,xTest2);
classifyXTest2 = string(classifyXTest2);
classifyXTest2 = double(classifyXTest2);
csvwrite('classifications.csv',classifyXTest2);
```