Math 350 Final Exam

Instructions: This is a take home final exam. You may use your notes and anything on our class website to assist you in writing solutions to the following, but you may not use anything else on the internet. You may also use the help files available in Matlab. **All of your work must be your own.** If you have questions, you may ask me (I may not be able to answer fully, but it is better to ask me than not to know).

You shouldn't have to write much "new code"- we've done sample sessions for each problem, so you should feel free to copy/paste from our samples and homework.

To turn in your solutions, create a new folder on CLEo called "Final", and upload **two** files per problem: One file is the script m-file that you typed, and the second file is the PDF version of the published m file (that way, I see what you saw). If you're not sure how to do this, come see me.

Due: The solutions are due by 8PM on Thursday, Dec 13th.

No late solutions will be accepted, so start early!

Problem 1: Feature Extraction

The goal: You'll have a collection of 26 photographs. The data is on our class website as `Photos01.mat` (See the description on the very last page). Each photo is 162×149, so the matrix is 24138×26.

1. Find the best four dimensional basis for the 26 points, and visualize the mean and four basis vectors. Also plot the 26 photos in $\mathbb{R}^2$
2. Use an autoencoder to go from $\mathbb{R}^{24138}$ to $\mathbb{R}^4$ (use mean-subtracted data), and visualize the weights of the encoder- This is the matrix that is 4×24138. If `net` is the autoencoder, we can extract the matrix as:

   ```
   W=net.EncoderWeights;
   ```

   Then visualize each row of W as a grayscale image (in a 2×2 subplot). NOTE: You can use the same training parameters that we used in class in the handwritten digit example, but make the MaxEpochs about 1000 or so. It may take a few minutes to train.

Problem 2: Mushroom Data Classification

The data consists of 4062 points, where each point has 22 characteristics of a mushroom. We want to classify the mushroom as poisonous or edible. When you load the data set `mushrooms.mat` from the class website, you'll have a matrix X that is 22×4062 and a target matrix T that is 2×4062.

1. Divide the data into training and testing sets, using about 70% for training and 30% for testing.
2. Train a $22 - 10 - 2$ feedforward neural network using the default transfer function and training values. Be sure to use the only the training set!
3. Give the confusion matrix using the testing set.

Problem 3: Mushrooms, Part II

Repeat Problem 2, but use Radial Basis Functions. Do this on your own by using the `edm.m` and `rbf1.m` functions (downloadable from the class website).

1. For the transfer function ϕ, use the cubic function.
2. For the centers, use Matlab's `k-means` function to find 30 of them.
3. Use the SVD to compute the pseudoinverse and find the weights and bias terms.
4. Output your confusion matrix.

Problem 4: Deep Net Training

Use the stacked autoencoders and handwritten digit example to build a deep net classifier for our cats and dogs data! If you download the data `CatsDogs.mat`, you'll have a matrix X that is 4096×198, so there are 198 photos, each 64×64. The first 99 are cats, the next 99 are dogs. Since we don't have much data, go ahead and use all the data for training, but do randomize the order of the data.

Your autoencoders should go from 4098 dimensions to 50, then 10 (and then 2). The other parameters are fine from the sample.

When you're finished training, output the confusion matrix. You'll also have a set of encoder weights 50×4096. Visualize the first 25 of them in a 5×5 subplot (like you did in the first problem).

Description of the Data: Photos01.mat

- When you load the data, you'll load a matrix X that is 24138×26.
- The first number comes from the size of each "photo": $162 \times 149 = 24138$.

If you want to visualize the photos, you would "reshape" the corresponding vector. For example, to visualize the first four faces:

```
for jj=1:4
   subplot(2,2,jj)
   imagesc(reshape(X(:,jj),162,149));
   axis equal; colormap(gray)
end
```