MCEN 4125/5125

Hand-Written Character Recognition

Due: TBA

For this project you will be solving the hand-written character recognition (classification) problem we discussed in class.

- You will be turning in 3 files (possibly more but will be discussed further below)

  1. There should be a Matlab script called "Training_*Lastname*.m" (where *Lastname* is replaced by your last name) which will have the optimization code that created all 45 hyperplanes in the form of a's and b's.
  2. There should be a Matlab script called "Testing_*Lastname*.m" (where *Lastname* is replaced by your last name) which will have the Directed-Acyclic-Graph (DAG) code and also your analysis on how well your classification performed.
  3. There should be a Matlab **.mat** file called "DAG_*Lastname*.mat" (where *Lastname* is replaced by your last name) which will include all a's and b's representing the 45 hyperplanes. This .mat file will be used to perform verification analysis to compare with your own analysis.

- You must save your hyperplanes (a's and b's) in the following manner for me to be able to analyze your classification solution:

  1. Store all your a's in an upper-triangular "cell" called A (in Matlab you use curly brackets { }) ... so A{1,10} will store the a vector that represent the hyperplane between 0 and 9 and A{9,10} represent 8 vs 9.
  2. Store all your b's in an upper-triangular "cell" called B... so B{1,10} will store the b scalar that represents 0 vs 9 etc.
  3. Here is example code to make the mat-file:

     ```
     >>save DAG_Ruben A B
     ```

  So the cell-array should look something like this:

  ```
  >> A

  A =

   [] '0 vs 1' '0 vs 2'  ...  '0 vs 9'
   []    []    '1 vs 2'  ...  '1 vs 9'
   []    []       []     ...  '2 vs 9'
   []    []       []     ...  '3 vs 9'
   []    []       []     ...  '4 vs 9'
   []    []       []     ...  '5 vs 9'
   []    []       []     ...  '6 vs 9'
   []    []       []     ...  '7 vs 9'
   []    []       []     ...  '8 vs 9'
  ```

- It is crucial for my ability to analyze your classification that Set1, in the above code, is the smaller of the two numbers you are comparing. So when finding the hyperplane between 0 and 9 (or A{1,10} and B{1,10}), Set1 should be zeros and Set2 should be nines.
- Here is my sample code for checking between zeros and nines in the DAG:

  ```
  Not9 = find(double(images_test)* A{1,10} - B{1,10}>0); % so maybe zero
  ```

- Here is the sample code for displaying an image from its 784 long feature vector:

  ```
  %%%% Load Data
  load mnist;

  %%%% Plotting a HandWritten Digit

  %for ii = 1:10
  nHold = 7, % I chose the 7th image to show
  IMAGE = reshape(images(nHold,:),28,28)';
  labels(nHold)
  figure,imagesc(IMAGE)
  colormap(flipud(gray(256)))
  axis equal
  set(gca, 'YTick', []);
  set(gca, 'XTick', []);
  axis off
  ```

- Now that you have finished solving the problem as described above, by solving for the 45 hyperplanes using the un-altered 784 features, you are now to play with the problem. When I say “play”, I mean try to alter the problem in the following possible two ways (at least choose one):

  1. Reduce (or change) the features in a logical way so as to maintain the essence of the image. One example would be to cut the border pixels and reduce the problem by $2 * 28 + 2 * 26 = 108$. Another option would be to do some image processing on the images (although this may not change the size of the feature vector). Come up with other options! When reducing the feature size, the gain in speed will most likely result in a loss of accuracy but by how much?
  2. Increase the features in a logical way so as to give more more ways for the hyperplane to compare digits. One example would be to append to the original 784 feature another 784 features that come the original image low passed and another 784 features that come from the original image high passed (image processing toolbox will have some of these tools) giving a total of $3 * 784 = 2352$ features. This increase in feature size will definitely increase the time needed to solve the problem but will most likely result in a high accuracy.

- After changing the features in both the training and testing image, and produced new corresponding a's and b's, I need you to save the following items to the **.mat** file called "DAG_*Lastname*.mat" you already made (and already saved the A and B cell array):

  1. Save your new a's and b's in cell arrays name Ap and Bp (where p is for "play")
  2. Save your new, altered, "images_test" as "images_test_play"

- Don't forget about the report :). Although it only get one bullet in the write-up, it is a crucial part of this project. Remember that clearly presenting your results to your peers is an important part of any project. When writing a report or paper, try to ask yourself if another engineering could recreate your results by reading your paper.