CS 435 – Computational Photography
Assignment 4

In this assignment you will demonstrate your ability to perform feature extraction for the purpose of image classification.

Grading Scheme:

1. Theory Questions (20pts)
2. K-NN on Grayscale histogram (40pts)
3. K-NN on Gists (40pts)

Theory Question(s)

1. Given the following image

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

   a. (10pts) Draw a fully connect graph representation of this image. For pixel a, let a_i be its value/intensity, and (a_x, a_y) be its location. We can then compute the similarity/weight between pixels a and b as:

$$w(a,b) = e^{-\left((a_i-b_i)^2+(a_x-b_x)^2+(a_y-b_y)^2\right)}$$

   In addition, consider a pixel **not** to be connected to itself (weight=0).

   b. (10pts) Find the minimum graph cut using the matrix formulation way shown in class. In Matlab you **may** use the `svd` function to do eigen-decomposition on a matrix. You'll likely need to read its documentation to understand how to the inputs and outputs work. Show the computations needed to set up your matrix for eigen-decomponsition and what the chosen eigenvalue/vector pair is. Finally draw your new (cut) graph.

Programming Introduction

The success of traditional machine learning algorithms is highly dependent on *feature extraction.* In this assignment you will get experience:

1. Forming training and testing data sets
2. Implementing and evaluating a K-Nearest-Neighbors classifier.
3. Extracting global and local features

On BBlearn we have provided a dataset for use in training a classifier to detect images as containing a car vs not containing a car.

The dataset is structured as follows: http://cogcomp.org/Data/Car/

Part 1: Classifying an Image using Grayscale Histograms

For each image in your dataset, compute a *normalized grayscale histogram* with 256 bins and extract a class label from the first three characters of the file name, $\{neg, pos\}$. For simplicity you may want to enumerate these class labels.

Note: It may take a while to traverse this directory and make your data matrices. Therefore, you may consider saving your representations and labels in some file and read them in as needed.

Now we want to make our training and testing sets. First set the random number generator's seed to zero so that you can have reproducible results. Next shuffle the data and select 2/3 of it for training and the remaining for testing.

Now that we have our datasets created, let's classify!

Go through each *testing* histogram and classify them as car or not-car using a k-nearest neighbors approach, where $k = 5$. For our similarity metric, we'll use the histogram intersection:

$$sim(a, b) = \frac{\sum_{b=1}^{256} \min(a_j, b_j)}{\sum_{b=1}^{256} b_j}$$

For your report, compute the accuracy as the percentage of the testing images classified correctly. In addition show at least two success and failure cases.

Part 2: Classifying an Image using Gists

Next let's attempt to classification using local gradient information.

We'll pass a 20×20 window around the **grayscale** image (without overlap), computing an 8-bin HOG at each location. Since the images are 40×100 pixels, this will result in 10 HOGs, and thus your total representation of each image will be 10x8=80 values.

Next repeat your evaluation from Part 1, but using this representation.

Submission

1. Assignments must be submitted via Bd Learn
2. Submit a single compressed file (zip, tar, etc..) containing:
   a. A PDF file containing:
      i. Your answer to the theory question(s).
      ii. Two samples of success and failure for Part 1
      iii. Accuracy for Part 1
      iv. Two samples of success and failure for Part 2
      v. Accuracy for Part 2
   b. A README text file (**not** Word or PDF) that explains
      i. Features of your program
      ii. Name of your entry-point script
      iii. Any relevant information on running your script.
   c. Your source files