I400/B457 Intro to Computer Vision - Programming assignment #7 (simple)

This assignment heavily relies on lecture9_1_objects.pptx. We will use the opencv library.

Problem 1: K-nearest neighbor classification [100%]

Similar to PA5, we will be using the subset of the Caltech101 dataset. This time, we use all images of 10 categories instead of 10 per category. We will also use OpenCV SIFT, kmeans clustering, visual words, and bag-of-visual-words identical to the problem 2 of PA5.

The below steps 0-3 are almost identical to the problem 2 of PA5.

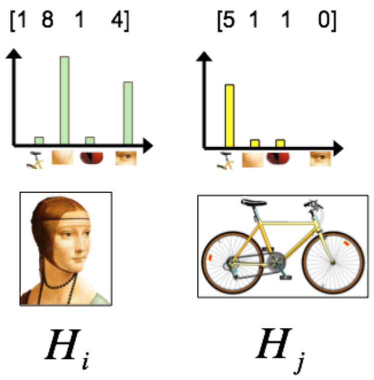
0. Download the Caltech101 dataset from either the original website:
http://www.vision.caltech.edu/Image_Datasets/Caltech101/ or a faster IU server:
http://homes.soic.indiana.edu/classes/spring2016/csci/b490-mryoo/101_ObjectCategories.tar.gz
http://homes.soic.indiana.edu/classes/spring2016/csci/b490-mryoo/101_ObjectCategories.zip

Similar to PA5, we only use the categories {airplanes, camera, chair, crab, crocodile, elephant, headphone, pizza, soccer_ball, starfish}. For each of these categories, we will only use **all** image files: image_0001.jpg~. The first 32 images of each class are considered the **training set**, and the remainder is the testing set.

1. For each image, extract SIFT features.

2. Use k-means to find cluster centers (k=200) of 128-D SIFT descriptors. When doing this, only use SIFT features from “image_0001.jpg” of each category (i.e., we only use 10 images).

3. Based on the learned cluster centers, for each of image, construct a bag-of-visual-words histogram. This can be done by just assigning each SIFT feature to the nearest cluster. Since our k is 200, it will be a 200-D histogram.



It will be important to save the obtained BoW histograms in a hard-disk, since it will take a good amount of computational time to re-compute this every time you debug your code.

4. Classify each test image using k-nearest neighbor classifier. For each test image, find the k=(3 and 5) training images having the smallest Euclidean histogram distance. Decide the

object class of the test image based on ground truth labels of those 3 (or 5) training images. If the object class of the image matches with its ground truth label, it’s called a ‘true positive’.

Report the classification accuracy of each class: #_true_positives_per_class / #_total_test_images_per_class ***[100%]***