Homework-2: In this assignment you will practice HoG, DenseSIFT and SIFT feature extractions from images,and then testing out VLAD and Fisher Vector aggregation schemes to generate an uniform dimension feature representation for images. Then for a given set of images and their pair wise matching/non-matching info from the CDVS data set, you will test the performance of different combination of features and aggregation schemes to see which one gives us the best performance in TPR-FPR ROC.

[Q1, 20pts] Compute Image Features, create a matlab/python function that compute n x d features by calling vl\_feat HoG, DSIFT and SIFT functions (notice that vl\_feat also has Python version), implementing the following function:

|  |
| --- |
| % im - input images, let us make them all grayscale only, so it is a h x w matrix  % opt.type = {‘hogf’, ‘sift’, ‘dsft’} for hog, sift and densesift  % f - n x d matrix containing n features of d dimension  function [f]=**getImageFeatures**(im, opt) |

[Q2, 20pts] Compute VLAD and Fisher Vector models of image features, for this purpose you need to first compute a Kmeans model for HoG, DenseSIFT and SIFT. Use the CDVS data set given as both training and testing for convenience (not the right way in research though, should use a different data set, say FLICKR MIR, or ImageNet), implementing the following functions:

|  |
| --- |
| % f - n x d matrix containing training features from say 100 images.  % k - VLAD kmeans model number of cluster  % kd - desired dimension of the feature  % vlad\_km - VLAD kmeans model  % A - PCA projection for dimension reduction  function [vlad\_km, A]=**getVladModel**(f, kd, k)  % PCA dimension reduction of the feature  [A,s,lat]=princomp(f);  f0 = f\*A(:,1:kd); % this is the feature with desired d-dimensions  ...... |

|  |
| --- |
| % f - n x d matrix containing n features from say 100 images.  % k - number of GMM components  % kd - desired lower dimension of the feature  % fv\_gmm - FisherVector GMM model:  % fv\_gmm.m - mean, fv\_gmm.cov - variance, fv\_gmm.p - prior  % A - PCA for dimension reduction  function [fv\_gmm, A]=**getFisherVectorModel**(f, kd, k)  [A,s,lat]=princomp(f);  f0 = f\*A(:,1:kd); % this is the feature with desired d-dimensions  ...... |

%hint from last year’s gmm training with a set of dimensions and number of clusters

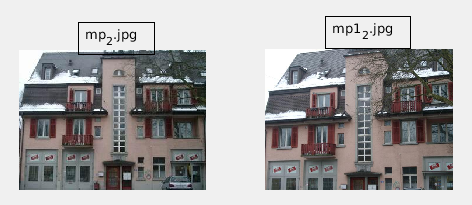
|  |
| --- |
|  |

[Q3, 20pts] Compute VLAD and Fisher Vector Aggregation of Images, from the given VLAD and FV models, implementing the following functions. Notice that the feature nxd f need to be projected to the desired lower dimension via, f0=f\*A(:,1:kd), to match the VLAD model dimension before calling this function.

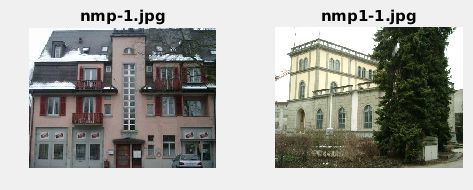
|  |
| --- |
| % f - n x d matrix containing a feature from an image by calling f=getImageFeature(im,..  % vlad\_km - VLAD kmeans model  function [vlad]=**getVladAggregation**(vlad\_km, f) |

|  |
| --- |
| % f - n x d matrix containing a feature from an image by calling f=getImageFeature(im,..  % fv\_gmm - GMM Model from features, has m, cov, and p.  function [fv]=**getFisherVectorAggregation**(fv\_gmm, f) |

[Q4, 20pts] Now benchmarking the TPR-FPR performance of various feature and aggregation scheme performance against the mini CDVS data set: <https://app.box.com/s/oea1ng52b3ghac813qgry6v6xds3rpzy.> For HoG feature, let us have kd=[8, 16] and number of cluster nc=[32, 64] for VLAD and FV models, and for the SIFT and DenseSIFT features, let us have kd=[24, 48], nc=[32, 64, 96]. So for each image, we will have 4 + 6 = 10 different feature + aggregation representations. For the total of N images in the mini CDVS dataset, we have M=N\*(N-1)/2 total image pairs, and the matching pairs ground truth are given, we only care about the first 100 matching pairs and first 100 non-matching pairs in the fid.mat, which has two variables mp and nmp. Each has a row of two image filenames to their associated images, e.g, mp(1,:): mp\_2.jpg and mp1\_2.jpg are two matching pairs:



And nmp(1,:) contains file names for the following non-matching pairs



Let us use the Euclidean distance to on those features to compute the TPR-FPR ROCs and find out which one have the best performance. Last year’s plots are attached below for example, you only need to plot the last row for 10 feature-aggregation combinations.

