CS 435 - Computational Photography

Assignment 4 - Classification

Introduction

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

Grading

Theory Questions	20pts
K-NN on Grayscale histograms	40pts
K-NN on Gists	40pts
TOTAL	100pts

Table 1: Grading Rubric

1 (20pts) Theory Questions

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:

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

   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-decomposition and what the chosen eigenvalue/vector pair is. Finally draw your new (cut) graph.

The Dataset

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. Since we need labels (for our training set and to evaluate our testing set) we will only use the TrainImages subdirectory for both training and and testing data.

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

The following code can be used to parse this subdirectory:

```
files = dir('/CarData/TrainImages');
X = [];
Y = [];
for f = files'
   if ~f.isdir
        im = imread([d, '/', f.name]);
 %do whatever you need to in order to generate feature vector for im
 %which I'll call fv
       X(end+1,:) = fv;
       Y(end+1,1) = ~strcmp(f.name(1:3),'neg');
   end
 end
```

And to shuffle and divide your data in to training and testing subsets...

```
rng(0);
inds = randperm(size(X,1));
num = size(X,1)/3;
X = X(inds,:);
Y = Y(inds,:);

Xtrain = X(1:2*num,:);
Ytrain = Y(1:2*num,:);

Xtest = X(2*num+1:end,:);
Ytest = Y(2*num+1:end,:);
```

2 (40 points) Classifying an Image using Grayscale Histograms

For each image in your dataset, compute a 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 (where D is the number of bins):

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

For your report, compute the accuracy as the percentage of the testing images classified correctly. In addition show:

1. One image correctly labeled as a car
2. One image correctly labeled as not a car
3. One image incorrectly labeled as a car
4. One image incorrectly labeled as not a car

3 (40 points) Classifying an Image using Gists

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

Divide your grayscale images into 10 non-overlapping 20×20 sub-images, computing an 8-bin HOG at each sub-image. Concatenate these ten 8-bin HOGs to form an 80-feature representation of the image.

Now repeat your classification (kNN with k=5) and evaluation as in Part 1, but using this representation. Note that now your histogram intersection similar metric will just sum over 80 bins.

Submission

For your submission, upload to Blackboard a single zip file containing:

1. PDF writeup that includes:
   (a) Your answer to the theory question(s).
   (b) Accuracy for Part 2
   (c) Sample images for Part 2
   (d) Accuracy for Part 3
   (e) Sample images for Part 3
2. A README text file (not Word or PDF) that explains
   - Features of your program
   - Name of your entry-point script
   - Any useful instructions to run your script.
3. Your source files