Computer Vision Homework 2

Tianwen Chu
chutianwen123@gmail.com

November 26, 2013

1. Representing the World with Visual Words

1.1 Creating Visual Words

Question 1.0

What properties does each of the filter functions pick up? You should group the filters into broad categories (i.e., all the Gaussians). Answer in your write-up.

Gaussian filter	Low pass filter, high frequency section is mitigated (changing fast), the picture becomes smooth and blurred.
Laplacian of Gaussian(Log)	Edge's information is emphasized, it is applied in edge detection.
First derivative of Gaussian(X)	Offset information of y direction, and keep along x direction
First derivative of Gaussian(Y)	Offset information of x direction, and keep along y direction

Question 1.1

You should write the following function to generate a dictionary given a list of images.

[filterBank, dictionary] = getFilterBankAndDictionary(imPaths)

As an input, getFilterBankAndDictionary takes a cell array of strings containing the full path to an image. You can load each file by iterating from 1:length(imPaths), and doing imread(imPathsfig). Generate the αT filter responses over the training files and call kmeans. A sensible initial value to try for K is between 100 and 300, and for α is between 50 and 150, but they depend on your system configuration and you might want to play with these values.

In my implementation, I choose $k = 200, \alpha = 100$. In my program, I utilize two approaches to construct the whole *filterResponses*. Experimentally, the way in comment is a little slower than the present way.

```
%%  Q1.1 Creating Visual Words
function [filterBank, dictionary] =
getFilterBankAndDictionary(toProcess)
tic;
[filterBank] = createFilterBank();
k=200;         % number of features
apha=100;     % number of selected filter_responses
filter_melt=[];
for i=1:length(toProcess)
    I{i}= imread(toProcess{i});
    [filterResponses] = extractFilterResponses(I{i}, filterBank);
    L=size(filterResponses,1);      %  number of pixels
    filter_melt=[filter_melt;filterResponses(randperm(L,apha),:)];
%
filter_melt1((i-1)*apha+1:i*apha,:)=filterResponses(randperm(L,apha),
:);
end
    [unused, dictionary] = kmeans(filter_melt, k, 'EmptyAction','drop');
toc;
end
```

Question 1.2

We want to map each pixel in the image to its closest word in the dictionary. Create the following function to do this

[wordMap] = getVisualWords(I, filterBank, dictionary)

wordMap is a matrix with the same width and height as I, where each pixel in wordMap is assigned the closest visual word of the filter response at the respective pixel in I. We will use the standard Euclidean distance to do this; to do this efficiently, use the MATLAB function pdist2.

I use *min* function to get the corresponding feature of each pixels. The program below is my toy experiment one to get an example. (see in fig 1-1)

```
function wordmap = getVisualWords(I, filterBank, dictionary)
wordmap=I(:,:,1);
filterResponses = extractFilterResponses(I, filterBank);
% each pixels
matrix_distance=pdist2(filterResponses,dictionary,'seuclidean');
[unused index_feature]=min(matrix_distance');     % right
wordmap(:)=index_feature;
imshow(wordmap);
colormap('default');
end
```

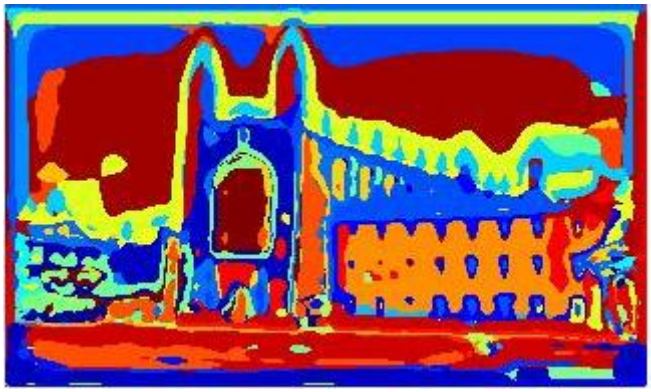


fig 1-1

2. Building a Recognition System

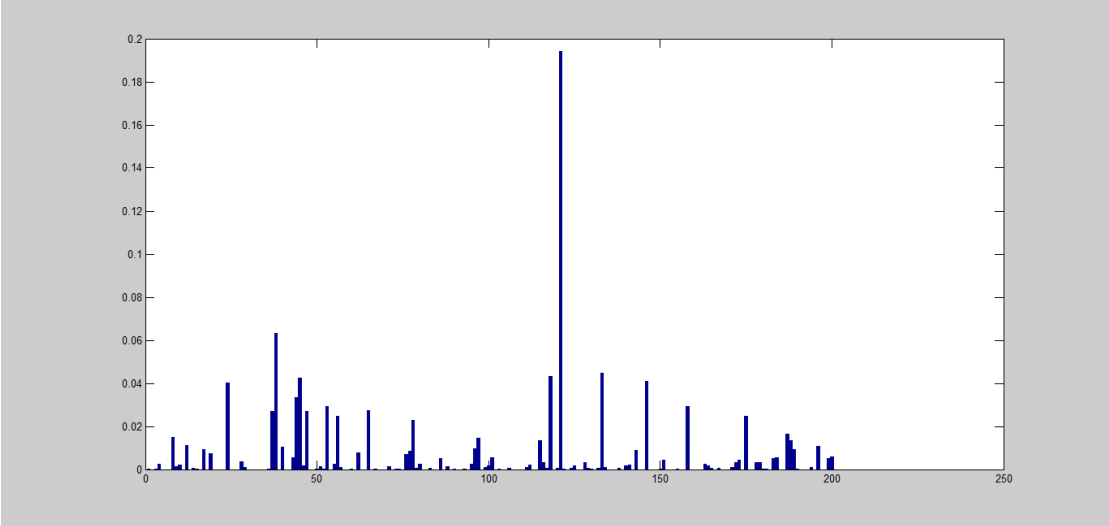
2.1 Extracting Features

Question 2.1

Create a function getImageFeatures that extracts the histogram of visual words within the given image (i.e., the bag of visual words).

[h] = getImageFeatures(wordMap, dictionarySize)

```
function [h] = getImageFeatures(wordMap, dictionarySize)
num_feature=1:dictionarySize;           % construct xcenters parameters
[h]=hist(wordMap(:),num_feature);
[h]=h./sum(h);                          % normalize h
end
```



2.2 Extracting Features

Question 2.2

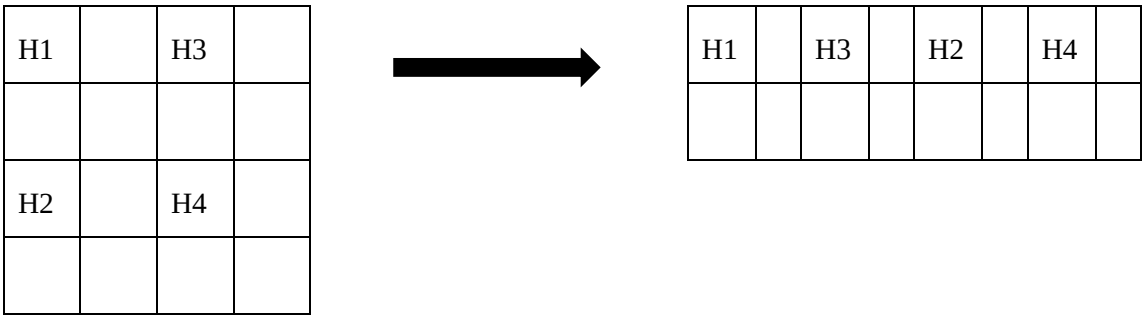
Create a function getImageFeaturesSPM that form a multi-resolution representation of the given image

[h] = getImageFeaturesSPM(layerNum, wordMap, dictionarySize)

In my implementation, there are two approaches; one is merely for three-layer pyramid another one is for arbitrary number layer pyramid. The first one works well, however the second one still needs some improvement to be realized. In both programs, I utilize a special way to preprocess the cell matrix. In all, my intention is to decrease the dimension. I follow the rule that just use *getImageFeatures* once and use relationship between each layers to construct the whole histogram. The first section is to preprocess the highest layer which has finest element cell. In my implementation, I transform the highest layer of finest cell element in to a 2 rows cell arrays. The core principal in my program is that the kernel in 2×2 is the most basic element.

H1	H3
H2	H4

This is the kernel cell matrix, if the number of layer is just two, to make the baser one just add all of them.

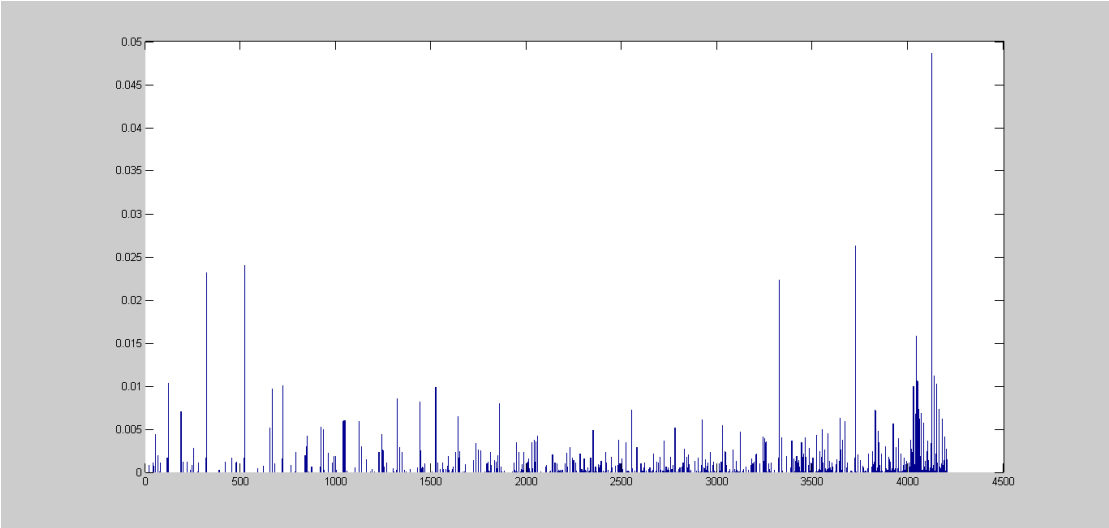


In this case 1, if we have three layers, there are four kernels. There needs to be noticed that when makes second layer, the H3 and H2 are inversed to be presented.

If we want to realize four-layer histogram, I transform 8×8 cell matrix to 2×32. In next step I plus the two rows together so that I get a 1×32 cell matrix. Then I plus each two elements (1+2,3+4,5+6…), then I get 1×16, which indicates the next layer. Afterwards, repeat this principal until reach condition in case 1. The core program is as below. In addition, I use a subfunction to fulfill transition. In every step, the final histogram grows. When the number of layer decreased to only one, I present the final histogram.

```
%% reconstruct a row=2 cell matrix
number_all=4^(layerNum-1);
jmax=(2^(layerNum-1))/4;
imax=2^(layerNum-2);
%% new cell matrix cell_new
for j=1:jmax
    for i=1:imax
     cell_new(1:2,4*i-3+(j-1)*(4*layerNum):4*i+(j-1)*(4*layerNum))=cel
     l_finest(2*i-1:2*i,4*j-3:4*j);
    end
end
```

```
function [H_base,layerNum,H_all]=separate(H_base,layerNum,H_all)
for i=1: 4^(layerNum-2)
H_base{i}=cell_new{4*i-3}+cell_new{4*i-2}+cell_new{4*i-1}+cell_new{4*
i};
end
H_base_processed=cell2mat(H_base);
H_all=[H_all,H_base_processed];
layerNum=layerNum-1;
[H_base,H_all,layerNum]=separate(H_base,layerNum,H_all);
end
end
```



2.3 Comparing images

Question 2.3

Create the function distanceToSet.

```
%% Q2.3 Create the function distanceToSet
function [ histInter ] = distanceToSet( wordHist, histograms )
distance_min = bsxfun(@min,wordHist,histograms);
histInter = sum(distance_min);
end
```

2.4 Building A Model of the Visual World

Question 2.4

Write a script named buildRecognitionSystem.m that produces vision.mat, and submit it as well as any helper functions you write.

```
%   load the files and texton dictionary
```

```
load traintest.mat;
load dictionary.mat filterBank dictionary;
%% create the classTrs
impath_train = strcat('wordmaps/',strrep(imTrs(:),'.jpg','.mat'));    %
paths of imTrs
classTrs=csTrs';
L = length(impath_train);
featureTrs=[];
for i=1:L
    load(impath_train{i});
    histogram = getImageFeaturesSPM( 3, wordMap, size(dictionary,1) );
    featureTrs=[featureTrs,histogram];
end
save vision.mat filterBank dictionary featureTrs classTrs;
```

2.5 Quantitative Evaluation

Question 2.5

Write a script named evaluateRecognitionSystem.m that tests the system and outputs the confusion matrix, and submit it as well as any helper functions you write. Report the confusion matrix for your results in your write-up

The core section in my program is this loop, firstly I match the guess results with the facts, and use *find* to get the index of guess right ones.

```
for i = 1:length(mapping)
    facts(find(strcmp(guess, mapping{i}))) = i;
end
```

3. Part II: Thoughts and More Improvement

Question 3.1

Classi_ers usually work better as the data set size increases, so one option is to just get more data. Yet with just the data we give you, can you think of a way to expand the data set without using external images? A popular way is to left-right ip the image. Will this work under the bag of words representation? How about bag of words with spatial pyramid matching? Why or why not? Include the answers in your write-up (no implementation required).

Because the image is reversed, the spatial pyramid records position information. Within the higher layer, especially for one with finest cells, the order is fixed and corresponding to each position in the image. If the image is left-right flipped, the histogram of highest layer is different arranged. As such, the whole histogram is enlarged, when the testing image comes, we make two comparisons(also flip the test

image), then we may connect two results (A&B) to make the judgment more precise. However, for BOW, it changes very subtlety. If we use normal (don't flip) filterbanks to get the cluster centers, and use same filterbanks process the reversed image, use these filterresponses to compute the distance between cluster centers. Because of property of odd filters, the distance may be different. In some case, the pixel belonged to feature 1 may be changed to feature 2(where feature 1 and feature 2 are quite close).

Question 3.2

Better filter bank

I add two kinds of filters Laplacian and Disk filters to 71.25%, I directly add some codes to creatfilterbanks. The corresponding codes and .mat files are labeled 'add_filters' in the last of file name, you need to delete it to work the code.

Better image similarity function

I use Hellinger and Chisquare to replace sEuclidean approach, the result gets better only for Chisquare. However, each approach have change. In one case, the wrong guess in case1 like fact football outputs airport, in case2 it may become kitchen. Although both wrong results, the extent of incorrect gets lower. Below are the confusion matrixes. The code is referred from Zesheng, I mainly join in discussion of principal.

Case1 Accuracy_result_sEuclidean 71.25%							
13	0	0	3	1	0	1	2
0	15	0	1	2	0	2	0
0	1	18	0	0	0	1	0
1	1	0	16	0	1	1	0
0	1	0	3	13	0	1	2
0	0	1	8	0	10	0	1
0	0	2	2	1	0	15	0
0	2	0	0	3	0	1	14
Case2 Accuracy_result_Hel 71.25%							
12	0	0	4	1	0	1	2
0	13	0	1	1	0	5	0
0	1	19	0	0	0	0	0
1	3	0	13	0	2	1	0
0	0	0	3	15	0	1	1
0	1	0	9	0	10	0	0
0	0	1	2	0	0	17	0
0	3	0	0	1	0	1	15
Case3 Accuracy_result_Chisquare 72.5%							
12	0	0	4	1	0	1	2
0	15	0	1	1	0	3	0
0	1	19	0	0	0	0	0

1	2	1	14	0	1	1	0
0	1	0	2	14	0	1	2
0	1	1	6	0	12	0	0
0	0	2	2	1	0	15	0
0	3	0	0	1	0	1	15

Different Features

In order to change choosing rand filteresponses, I select evenly 100 points in filteresponses. As such, especially in small data, the results will become more accurate.

```
interval=floor(size(filterResponses,1)/num);
start=randperm(interval,1);
filter(start:interval:start+interval*(num-1),:);
```