CSE 573: Computer Vision and Image Processing

HW 1: Spatial Pyramid Matching for Scene Classification

Instructor: Kevin Keane
TAs: Radhakrishna Dasari, Yuhao Du, Niyazi Sorkunlu

Due Date: September 25, 2017

Submitted by,
Muthuvel Palanisamy,
muthuvel@buffalo.edu
Person # - 50246815

1. Representing the World with Visual Words

Q1.0 (5 points) What properties do each of the filter functions (see Figure 3) pick up? You should group the filters into broad categories (i.e., all the Gaussians). Answer in your write-up.

Filters:

- **Gaussian Filters**– Gaussian filter picked up the pixels with high light intensity which is mostly the while pixels.
- **Log Filters –** Log filter extracted features such ah edges and boundaries. It detects the parallel line in the image which may help in determination of vanishing point of parallel lines
- **dx Filter: Gaussian derivative x –** dx filter captured the features that have transition with respect to horizontal axis
- **dy Filter: Gaussian derivative x –** dx filter captured the features that have transition with respect to vertical axis

Refer montage image in Q1.2. All images in a row correspond to a specific filter respectively

Q1.1 (10 points) Extracting Filter Responses

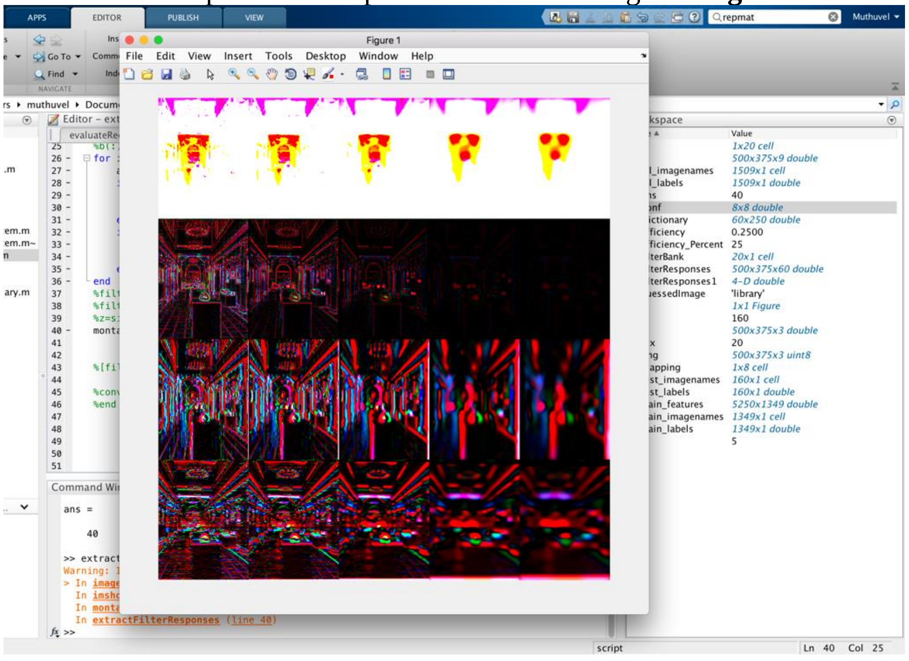
Design Steps Implemented:

- Conversion of image to double
- Checking for grey scale images
- Conversion of image to L*a*b
- Extraction of filter responses using **imfilter** command and the filterBank provided

Primary Command Used:

- Imfilter – extracting filter responses
- Monatage – visualization of filter responses

Sample Filter Responses visualized using **montage**



	Value
	1x20 cell
	500x375x9 double
l_imagenames	1509x1 cell
l_labels	1509x1 double
ns	40
onf	8x8 double
ictionary	60x250 double
ficiency	0.2500
ficiency_Percent	25
lterBank	20x1 cell
lterResponses	500x375x60 double
lterResponses1	4-D double
uessedImage	'library'
	1x1 Figure
	160
	500x375x3 double
x	20
ng	500x375x3 uint8
apping	1x8 cell
st_imagenames	160x1 cell
st_labels	160x1 double
ain_features	5250x1349 double
ain_imagenames	1349x1 cell
ain_labels	1349x1 double
	5

```
%b(:
for i
end
%filt
%filt
%z=si
monta

%[fil

%conv
%end
```

```
ans =

    40

>> extract
Warning: I
> In image
  In imsho
  In monta
  In extractFilterResponses (line 40)
fx >>
```

Q1.2 (10 points) Creating Visual Words

A dictionary of **Cluster Size(K) = 250** with **Number of Pixels (Alpha) = 200** is created and saved as Dictionary.mat. Tried implementing the function with dictionary of K=150 and Alpha=50 which resulted in lower accuracy.

Design Steps Implemented:

- Reads Image
- Extract Filter Response using **extractFilterResponses** Function
- Chose random pixel from the filter responses for T → 1349 training images
- Called kmeans and computes dictionary

Primary Commands used :

- randperm- Used to choose the x and y co-ordinates of Alpha random pixels
- waitbar – to keep tab on the progress of training the dictionary
- reshape – conversion from matrix to vector
- kmeans- computing dictionary

Q1.3 (10 points) Computing Visual Words

.

Design Steps Implemented:

- Loads Dictionary and Filter Responses of an image
- Calculates Euclidean distance using pdist2 and find the minimum distance
- Generates wordmap containing Euclidean distance

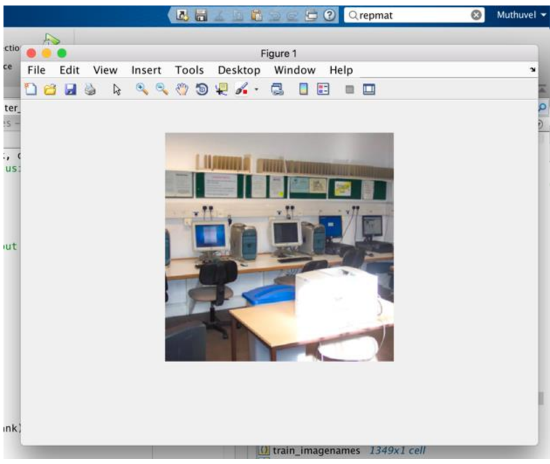
Primary Commands used :

- Imagesc – visualization of wordamps
- Pdist2 – calculation of Euclidean distance

--------------------- Sample Images and their Wordmaps from **Computer_Room** -------------------------

Image 1

WordMap_Image1


repmat
Muthuvel
Figure 1
File Edit View Insert Tools Desktop Window Help
train_imagenames 1349x1 cell

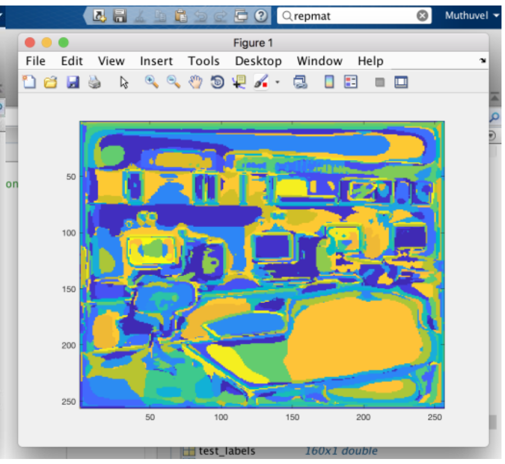
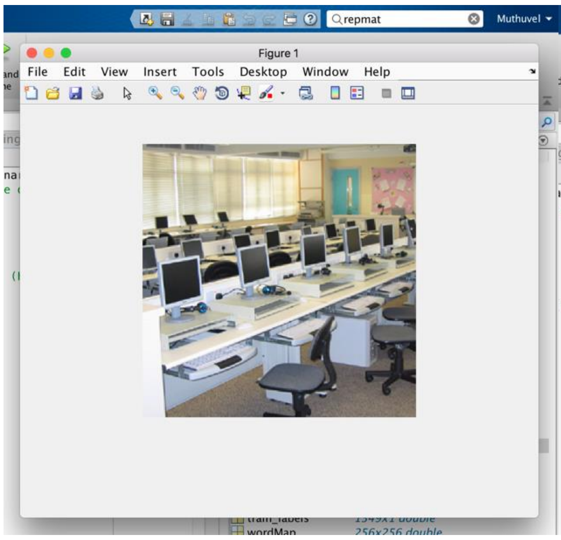


Image2



WordMap_Image2

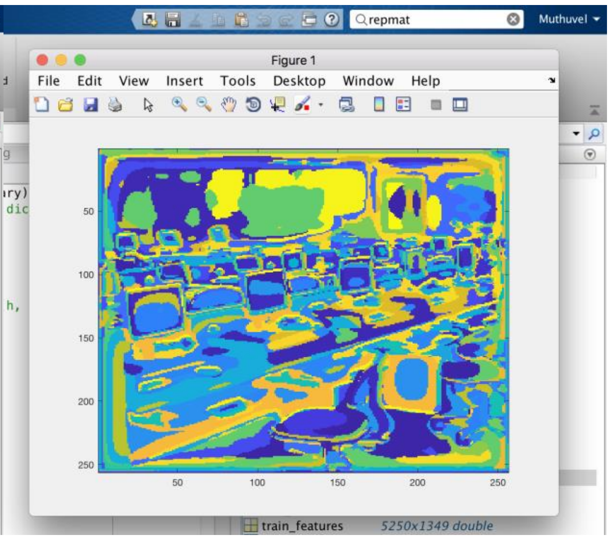


Figure 1
File Edit View Insert Tools Desktop Window Help
repmat
Muthuvel
50
100
150
200
250
50
100
150
200
250
train_features 5250x1349 double

Image 3

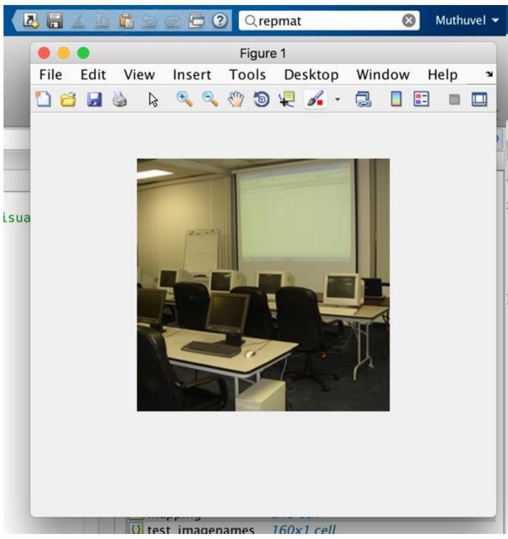


Figure 1
File Edit View Insert Tools Desktop Window Help
repmat
Muthuvel
test_imagenames 160x1 cell

WordMap_Image3

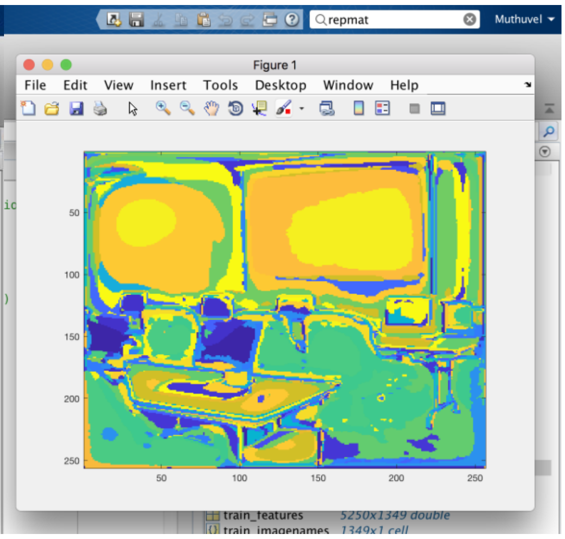


Figure 1
File Edit View Insert Tools Desktop Window Help
repmat
Muthuvel
50
100
150
200
250
50
100
150
200
250
train_features 5250x1349 double
train_imagenames 1349x1 cell

Comments:
 Visualization gives an idea about how the values of filter response of each pixel in an image are matched with one another. While testing the values of filter responses of the filter responses, it was found that a unique colour region in (like yellow, blue, etc.….) will denote the pixels with similar values . wordmap also picks up the lights reflected on walls with varying intensity in neighbouring pixels that can be seen on Image 3 above

2. Build Recognition System

Q2.1 (10 points) Extracting Features

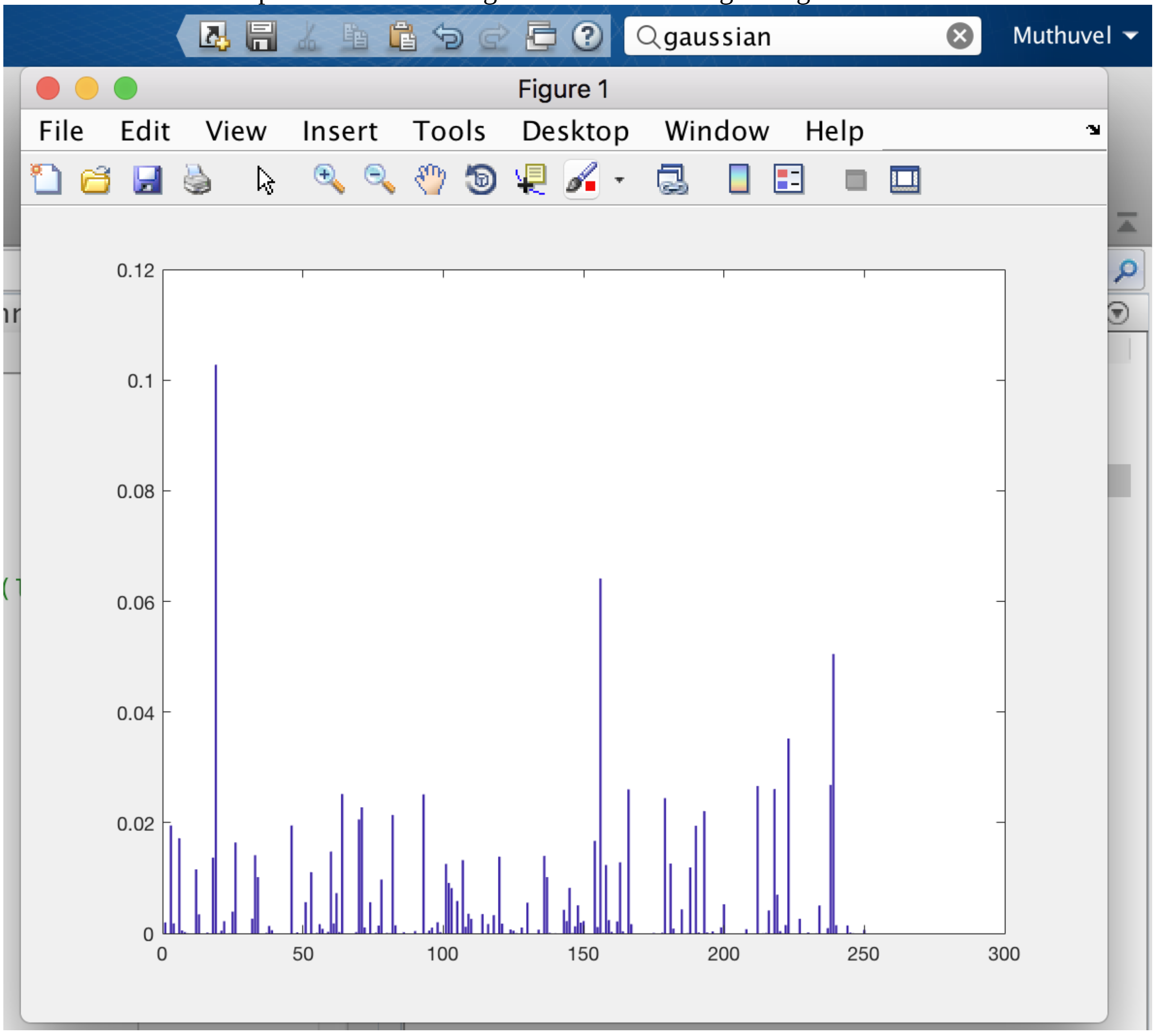
Design Steps Implemented:

- Defined a vector that denotes how many times each cluster value in a wordmap is present
- Calulate histogram of the vector
- L1 Normalization of histograms

Primary Commands Used:

- histcounts – for calculation of histograms
- h=h/sum(h) – for normalization of histogram

Sample bar char of histogram extracted from getImageFeatures.m



Q2.2 (15 points) Multi – Resolution: Spatial Pyramid Matching

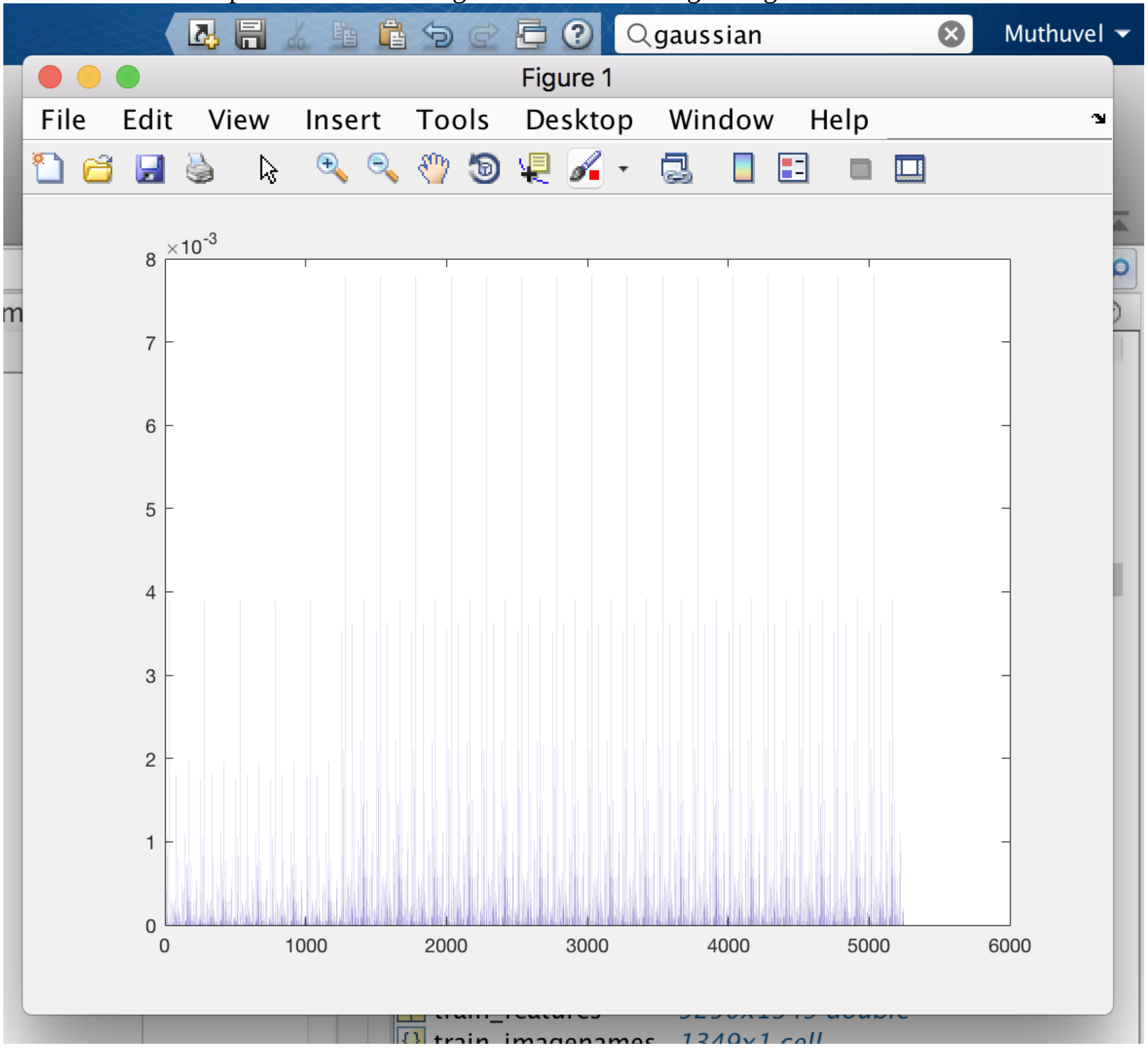
Design Steps Implemented:

- Load wordMap from the path
- Calculate the dimensios of wordMap
- L2 – layer - Split the image in 2l*2l = 16 cells in accordance with the dimension of the image (I used if statements to check if size of rows & columns are not divisible by 4 and defined commands accordingly for each possibility to split up the image)
- Convert individual cells into histograms and concatenate them
- L1 Layer – Repeat the same procedure and calculate a concatenated histograms of 2*2=4 cells
- L0 Layer – Calculate histogram for wordmap as a whole
- Concatenate all histograms and makes it L1 normalized

Primary commands used:

- If statements – to check if dimensions are multiple of 4 or not
- Mat2cell- split images to n number of cells
- histcounts – to determine histogram of wordmaps
- h=h/Sum(h) – L1 Normalization of histograms, where h is histogram
- Cat – concatenation of histograms

Sample bar char of histogram extracted from getImageFeaturesSPM.m



Q2.3 (10 points) Comparing Images

Design Steps Implemented:

- Creates a matrix – histInter with same dimensions as histograms matrix
- Calculates minimum of histograms and wordHist and assign to histinter
- Calculates the sum

Primary commands used:

- Min – to find the minimum of two values
- Repmat – to copy values of wordHist in n colums for finding distance

Q2.4 (15 points) Build a Model of the Visual Word

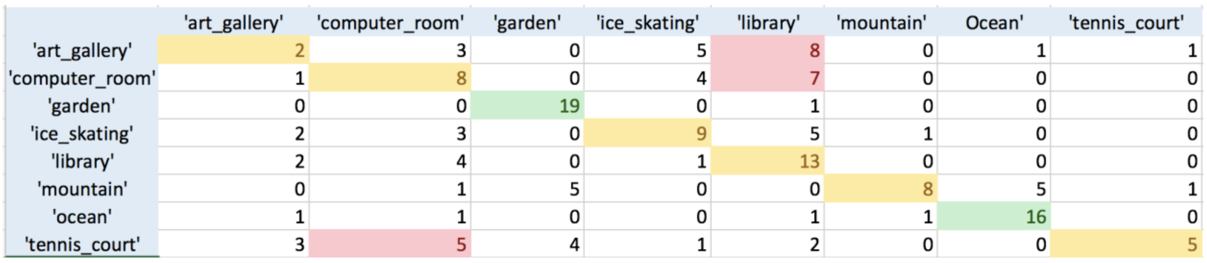
Design Steps Implemented:

- Load training wordmaps for extracting histograms by calling **getImageFeaturesSPM.m**
- Concatenates all histograms and creates **train_features** containing histograms of the training set
- Creates **vision.mat** file containing Dictionary, filterBank, train_Features,train_labels

Q2.5 (10 points) Qualitative Evaluation

- Loads Vision.mat and traintest.mat
- Calls guessImage.m function to calculate recognize images of all test_imagenames
- Check the guessedImage with test_labels and mapping variables
- Write the result in the confusion matrix – C(i,j) that contains number of instances of class i that were predicted as class j .
- Calcualates efficiency using trace(C) / sum(C(:))

CONFUSION MATRIX OUTPUT



	'art_gallery'	'computer_room'	'garden'	'ice_skating'	'library'	'mountain'	Ocean'	'tennis_court'
'art_gallery'	2	3	0	5	8	0	1	1
'computer_room'	1	8	0	4	7	0	0	0
'garden'	0	0	19	0	1	0	0	0
'ice_skating'	2	3	0	9	5	1	0	0
'library'	2	4	0	1	13	0	0	0
'mountain'	0	1	5	0	0	8	5	1
'ocean'	1	1	0	0	1	1	16	0
'tennis_court'	3	5	4	1	2	0	0	5

Program Output:

- **Efficiency Percentage** of Recognizing test_Imagenames: **50.00%**
- Number of correct test cases**: 80/160**
- Classes with the high number of correct test cases: **Garden →19/20 = 95% Efficiency** and **Ocean→16/20=80%**
- Class with the most number of failed test cases: **Art_gallery → 2/2 = 10% Efficiency and Tennis_Court → 5/20 = 25% Efficiency**

Q2.6 (5 points) List some of these classes/samples and discuss why they are more difficult in your write-up.

Ocean Class:

- Absence of lot of foreign objects like chair, table or people makes it easier to for detection
- Most of the images contain blue pixels with only very few training and test images containing red-dawn/dust and boats, which results in trained histograms that shares a lot of common features with testing images

Garden Class

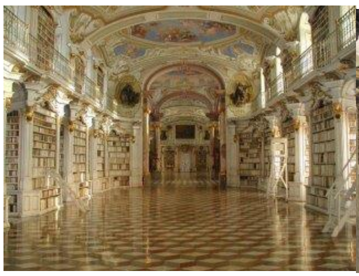
- Similar to Ocean images, garden class is primarily built of images which does not have a lot of other objects. Moreover, the overwhelming present green pixels in almost every garden images make it easier to identify common features
- This is also the reason why the images from other classes are rarely detected as Garden class or even Ocean class. This is evident from the confusion matrix
-

Art_Gallery Class:

- Art_Gallery Classes contains image taken which contains objects such as wooden walls, desk, floor and desk which is also present in library and computer_room classes.
- This causes the histogram extracted from a art gallery class to share significant amount of common features with libary and ice-skating classes.
- Art_Gallery class in test_imagenames that are classified as libary or ice-skating account for 13 out of 20 samples = 65%, which supports the claim.
- The maximum instance of efficiency for art_gallery class reached was only 25% which is still low
- A similar defect is also present in computer_rooms class.

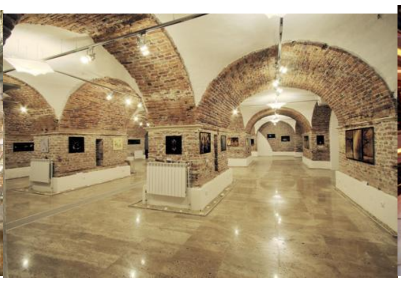
Example:

Library image



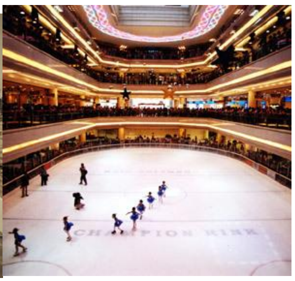
sun_akwfxelxtnwbxotp,jpg

Art gallery Image



sun_bejswhsjkngsdsvd.jpg

Ice-skating image



sun_bfuludctnmspioyb.jpg

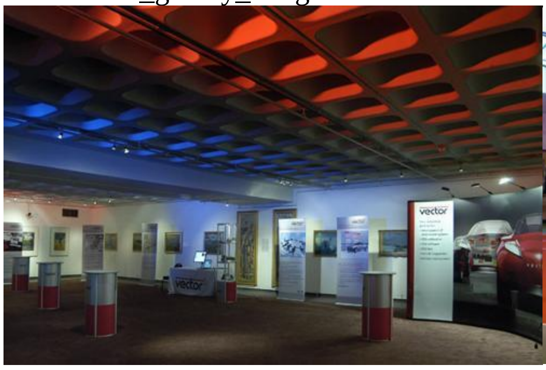
- We can see that these images share many features like floor, walls with round arches and edges that paves way for deviations

TENNIS_COURT CLASS:

- Tennis court class has a efficiency percentage of only 25%
- Long open areas of a tennis court with green floor resembled samples from garden in some cases. This causes a few occurrences to be classified as gardens
- 5 of the test images are classified as computer_room due to similar reasons.

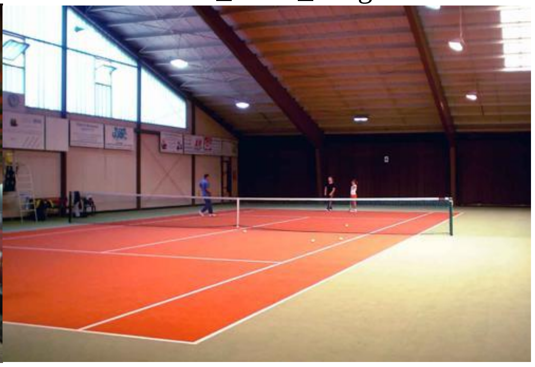
Example:

Art_gallery_Image



sun_amalnipvvupcaegy.jpg

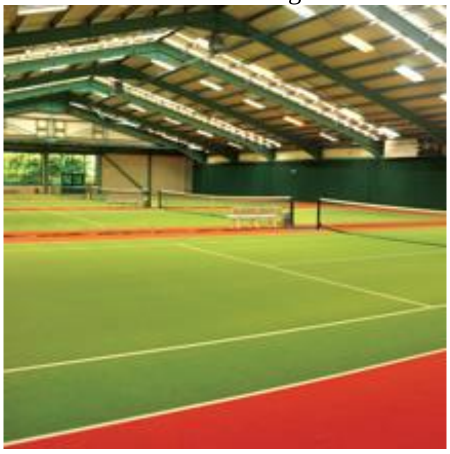
Tennis_Court_Image



sun_bgzapkejibzyvvqy.jpg

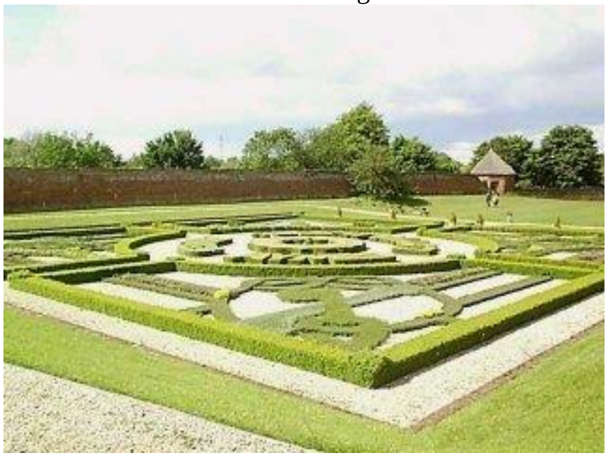
- We could not the similarity between in the ceiling, wide open areas and almost similar lightning in both the images.

Tennis Court Image



sun_bzbaifnovkzxbkxd.jpg

Garden Class image



sun_apraridkrvcdkoda.jpg

- These pictures show how a grass court tennis class image could easily be mistaken for a garden due to the presence of a lot of green pixels in the image.,

Q. 2.7 (Extra Credit, up to 20 points) Improving Performance

Methods I implement to try improve the performance:

- I tried changing the number of random pixels and clusters which caused the efficiency to increase slightly though not being significant
- While extracting the features through histograms, I tried using a different code with hist() function which eventually turned out to be a step down and resulted in a confusion matrix with efficiency percentage on only around 25%
- Though spatial pyramid matching is speculated to give a higher recognition efficiency over bag of words approach, upon implementation, I could not find any change in the efficiency percentage

Methods which I believe will give an increase in efficiency

- Though the values of K and Alpha has a role in increasing efficiency, a different filtering type like Gabor filters or median filter could increase the efficiency
- Different distances such as cosine or correlation distances could be used in Computing Visual Words to increase the efficiency of recognition instead of Euclidean distance.