School of Electronic Engineering and Computer Science

ECS797 Machine Learning for Visual Data Analysis

Lab 1: Image Classification using a Bag-of-Words model

1. DICTIONARY CREATION: Feature Quantization

1. In feature quantization, we executed the commands in lib1.m file and obtained the two variables, TrainMat(270000*128) and TestMat(90000*128) are loaded. The sift features has 128 dimensions.

2. After creating the dictionary of 500 words which is controlled by DictionarySize variable parameter in the code, we obtained the values in C.mat file in data/global/dictionary.

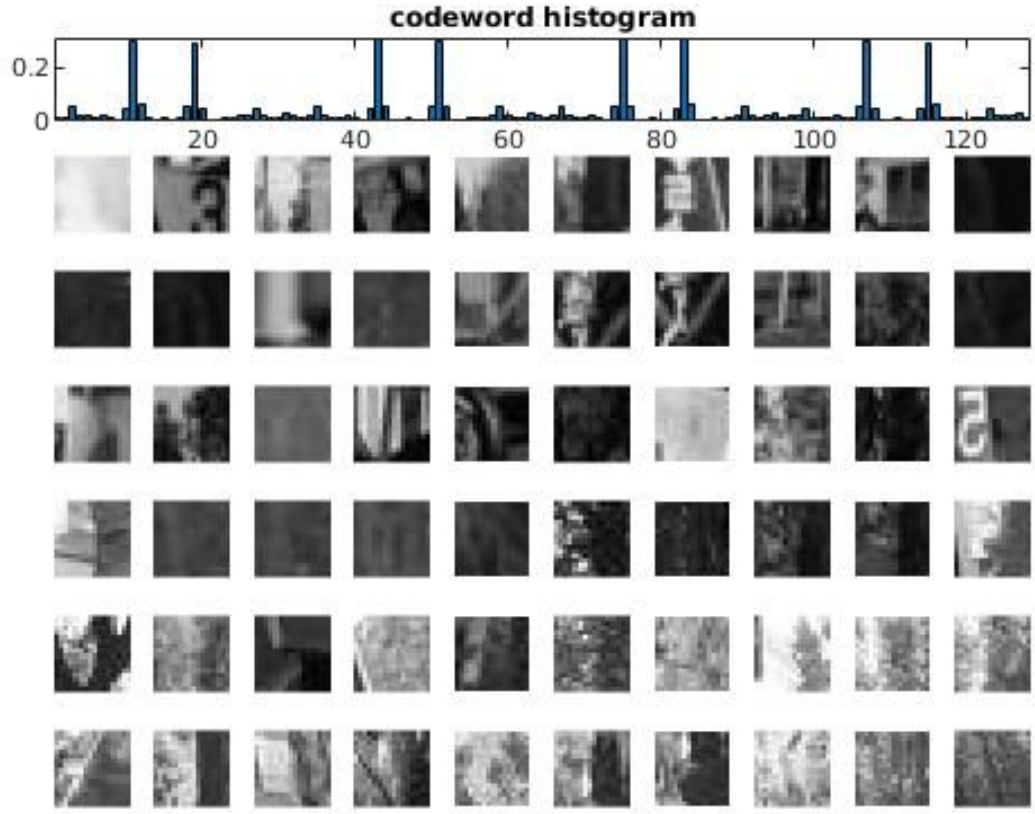
3. We obtained the Euclidean distance by implementing the Euclidean distance section in lab1.m file.

4. Write your own code that assigns each descriptor in the training and test images to the nearest codeword cluster. (Hint: use the function min())
% Assume the assignment vector of training images is index_train and that of test images is index_test, the dimensions of which should be 1x270000 and 1x90000 respectively. The entry in the vector is the codeword ID. Save them like this:

```
[n,~] = size(TestMatrix);
for i=1:n
 test = TestMatrix(i,:);
 dist = EuclideanDistance(test,C);
 [minv,index] = min(dist);
 index_test(1:i) = index;
end
```

```
[n,~] = size(TrainMatrix);
for i=1:n
 test = TrainMatrix(i,:);
 dist = EuclideanDistance(test,C);
 [minv,index] = min(dist);
 index_train(1:i) = index;
End
```

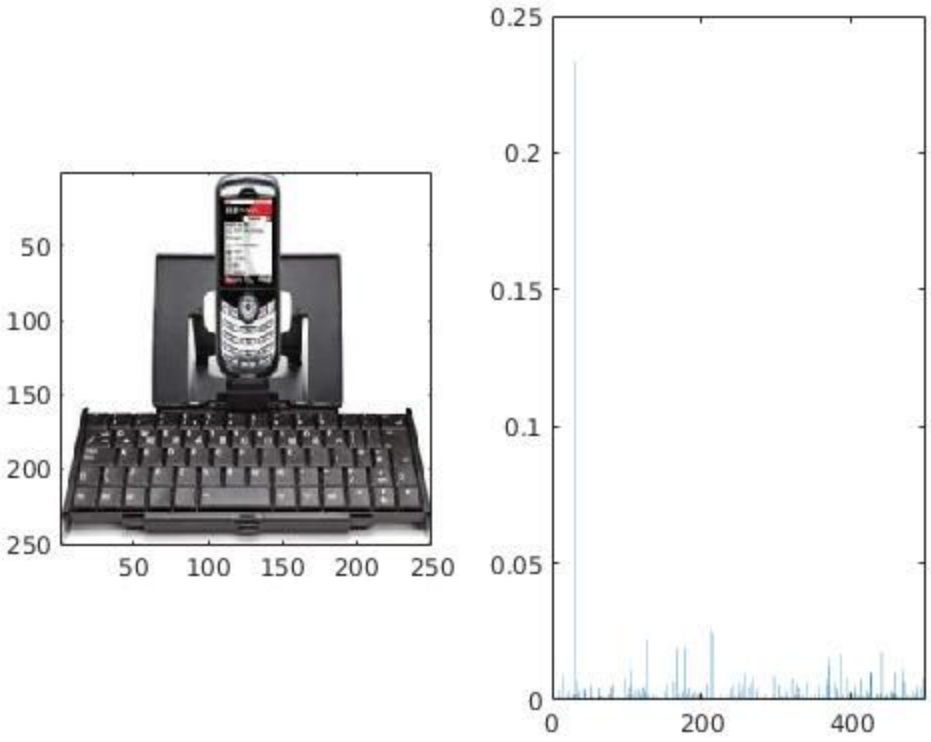
5. Visualize some image patches that are assigned to the same codeword



2. Image representation using Bag of Words representation

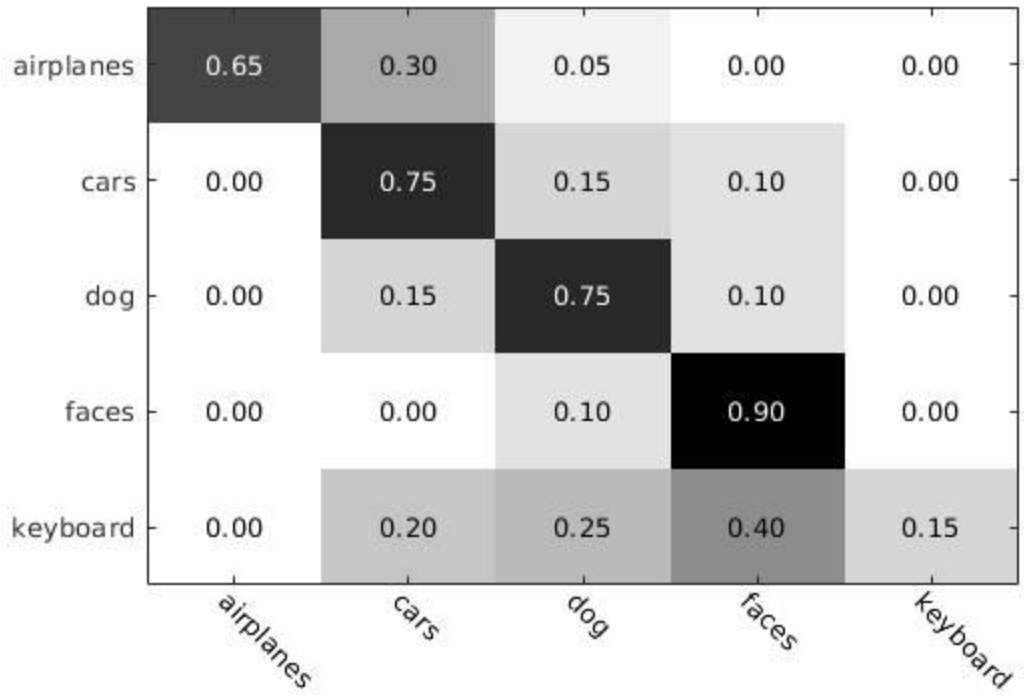
1. Represent each image in the training and the test dataset as a histogram of visual words (i.e. represent each image using the Bag of Words representation). Normalise the histograms by their L1 norm. For normalisation you may want to call the function do_normalize() that is in the file do_normalize.m in the software directory, with the appropriate arguments.
   %Here we should read the feature data for each image which is stored in data/local/*ID*

```
dist = EuclideanDistance(features.data, C);

        [test,~]= size(dist);

        points = zeros(1,vocbsize);

        for k=1:test

        [minVal,  indxmin] = min(dist(k,:));

        points(1, indxmin)= points(1, indxmin)+1;

        End

BoW(ii, :) = do_normalize(points);
```

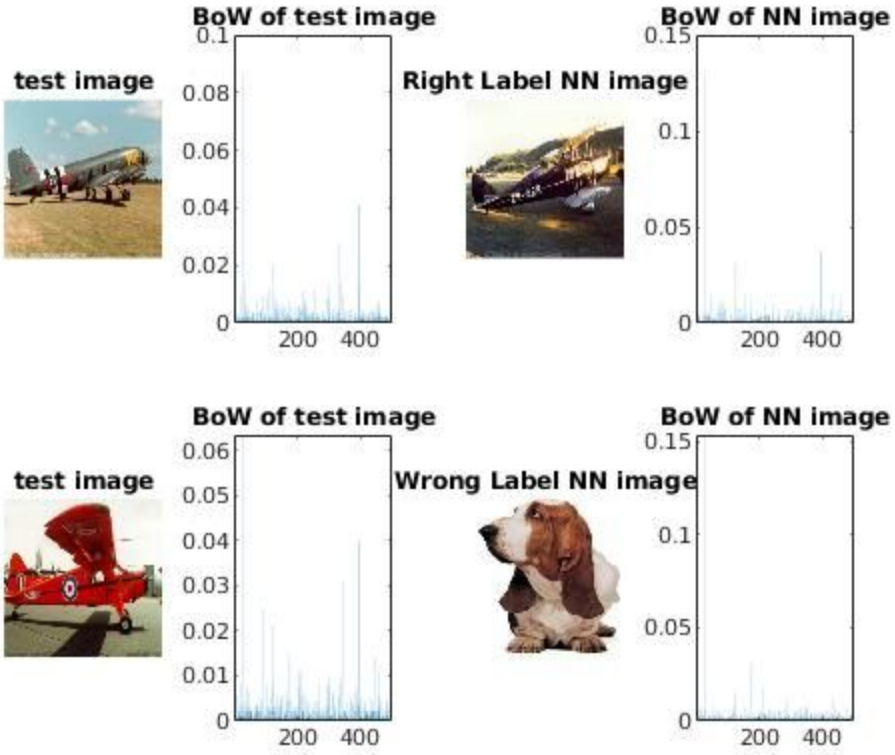


3.Image classification using a Nearest Neighbour (NN) classifier

1. After applying NN classifier, we obtained NN result Mat file with 100*1 double
2. Compute and report the overall and the classification errors per class.
   0.400000000000000 - err-all
   Err
   [0.250000000000000
   0.200000000000000
   0.450000000000000
   0.250000000000000
   0.850000000000000]
3. Compute and show the confusion matrix.



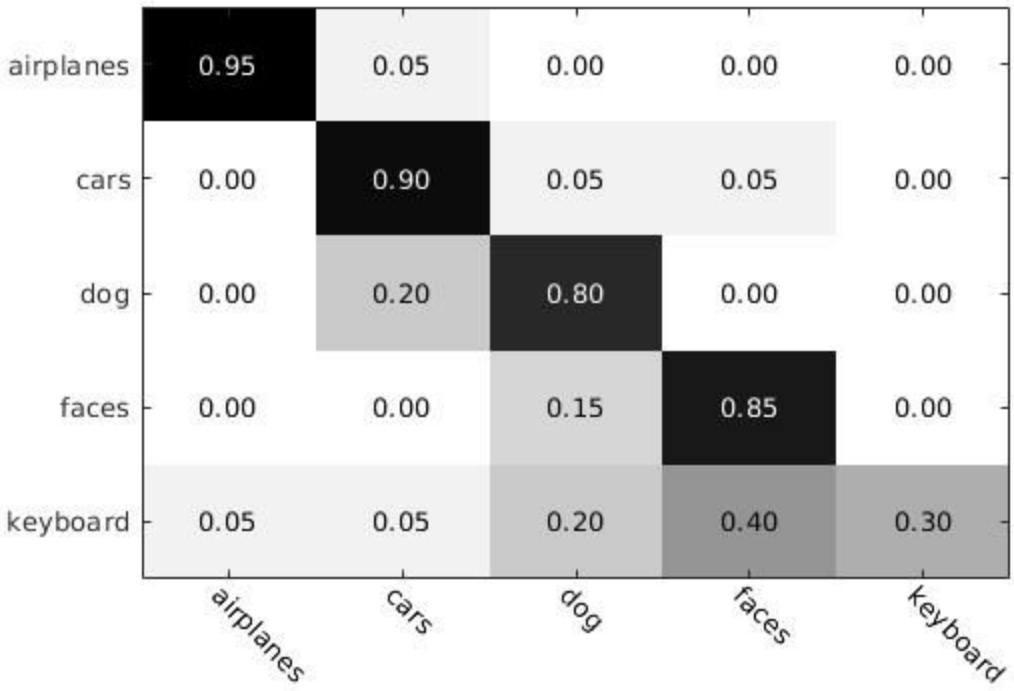
4. For each class show some images that are correctly classified and some images that are incorrectly classified. Can you explain some of the failures?
Ans: As we can observe below in the figure, we obtained test image and right label NN image correctly classified in above row but incorrectly classified test image in the below section. The apparent reason for this can be the random allocation of K value in the beginning, as we alter the value with each iteration, we obtain different accuracies in upcoming confusion matrices.



5. Write code for computing the histogram intersection between two histograms.
% Change the 'method' option in 4.1 and write the code for computing the histogram intersection in the file knnsearch.m

Ans: We need to make following changes in the knnsearch.m file to obtain histogram intersection. Following is the Confusion matrix obtained.

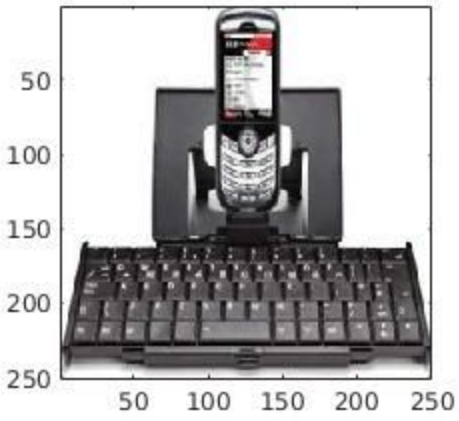
```
dist = 1;
        for i = 1:p
        dist= dist-min(a(1,i),b(1,i));
end
```

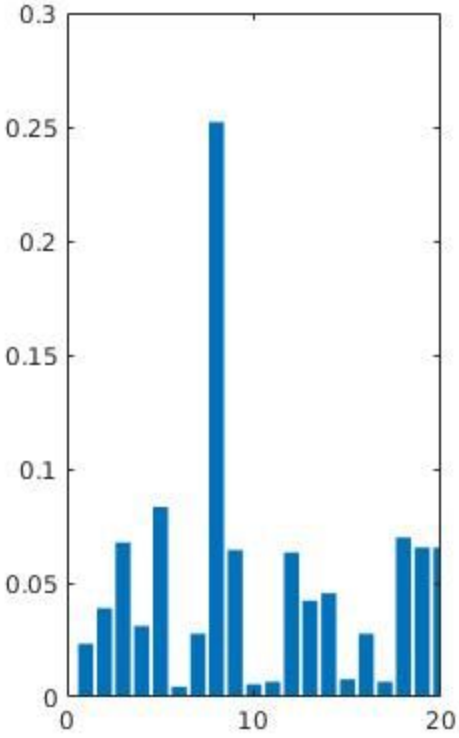


4. Dictionary size

1. Perform the classification experiment using a very small dictionary and report the classification error and confusion matrices. Hint: Cluster the descriptors into a small number of clusters (e.g. 20).

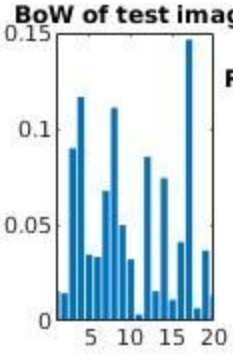
   Ans: We obtained following changes in the histogram, correctly and incorrectly classified, confusion matrix before and after histogram intersection.


50
100
150
200
250
50
100
150
200
250


0.3
0.25
0.2
0.15
0.1
0.05
0
0
10
20

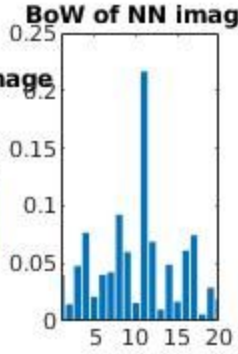
test image

BoW of test image



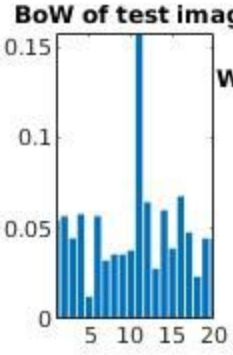
Right Label NN image

BoW of NN image



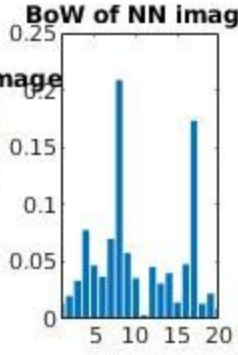
test image

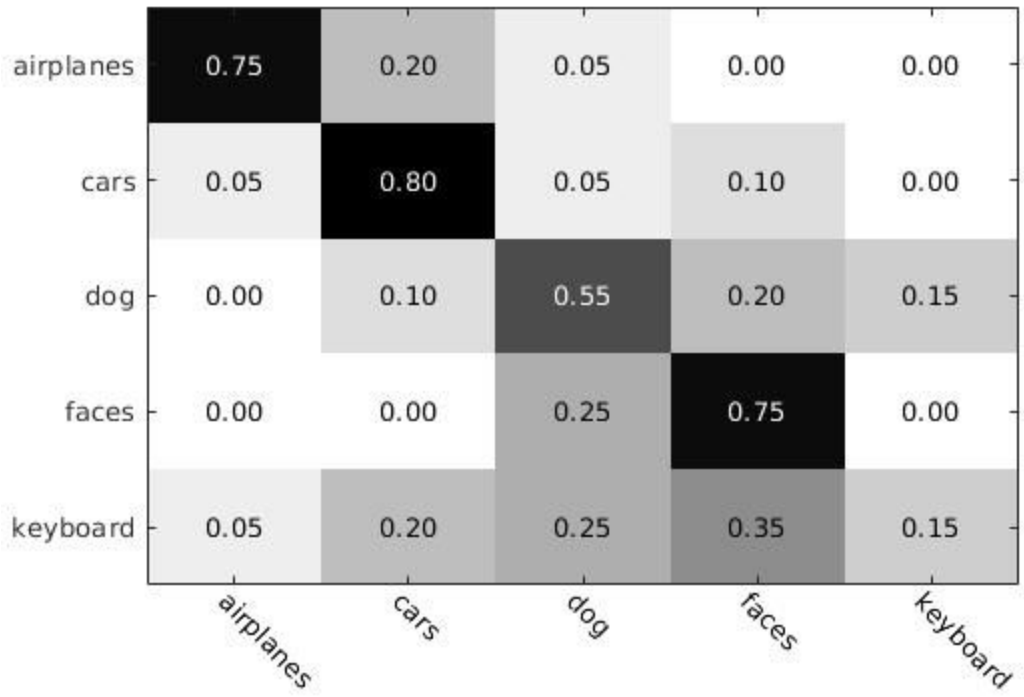
BoW of test image

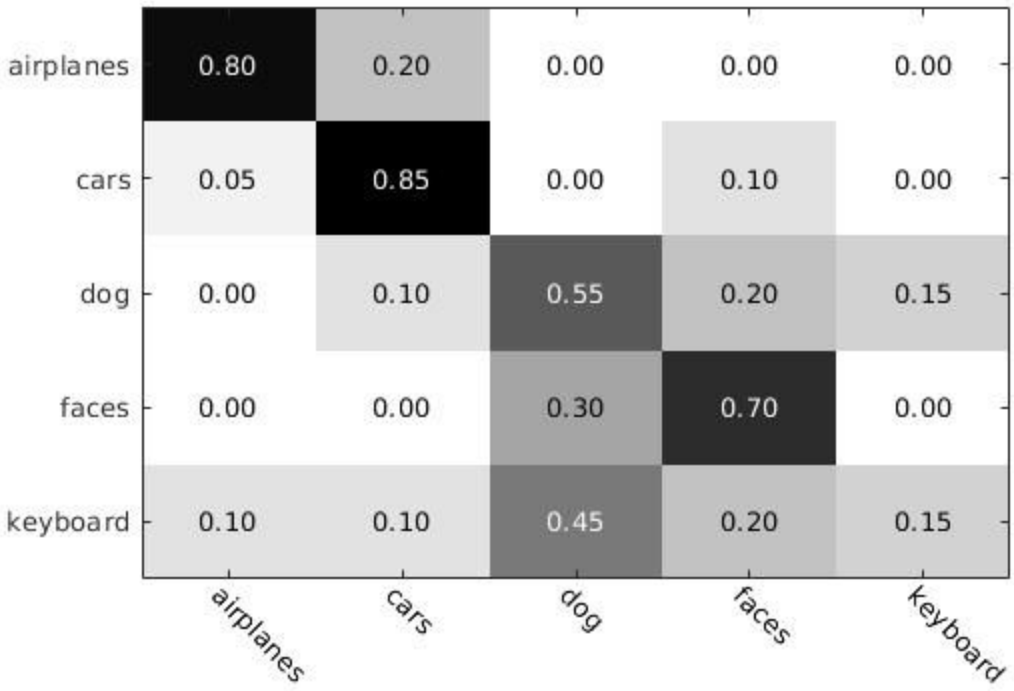


Wrong Label NN image

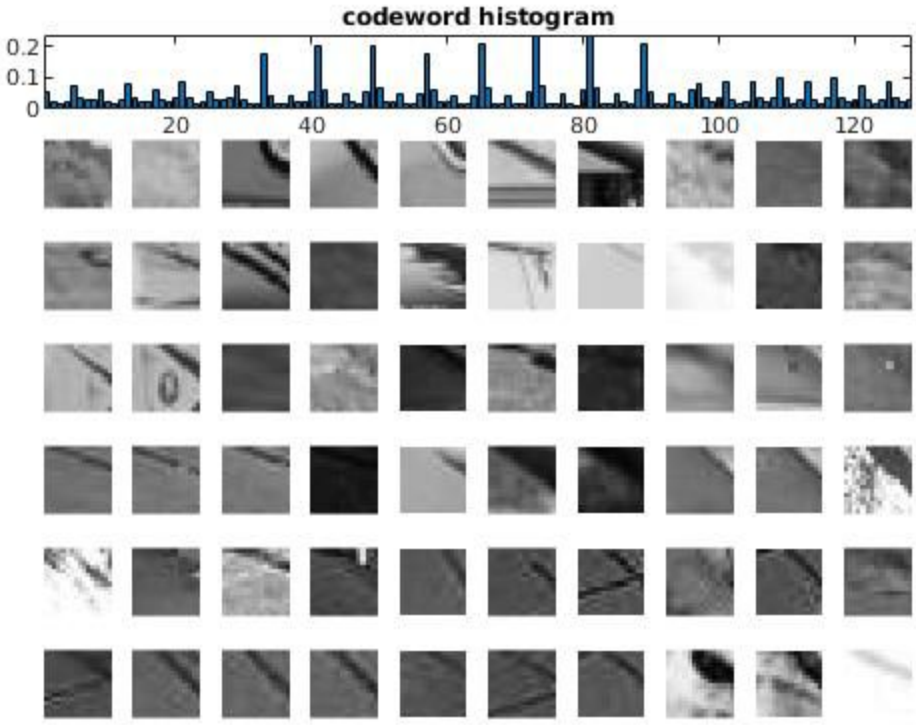
BoW of NN image




airplanes
cars
dog
faces
keyboard
0.75 0.20 0.05 0.00 0.00
0.05 0.80 0.05 0.10 0.00
0.00 0.10 0.55 0.20 0.15
0.00 0.00 0.25 0.75 0.00
0.05 0.20 0.25 0.35 0.15
airplanes
cars
dog
faces
keyboard



2. Explain the drop in the performance. To support your argument you may want to perform step 2.5 in order to visualize descriptors that belong to some of the visual words.
   Ans: We observe the drop in performance due to decreased number of dictionary size which resulted in decreased number of clusters or centers.Due to less information data clustering, we obtained drop in the performance. We can strengthen this analysis by observing the histogram codeword below.



4. Image classification using a Support Vector Machines classifier

1. For each of the image classes train a linear multiclass SVM classifier (using the libsvm library). Pay attention to the procedure that calculates the optimal values for the parameter C using cross validation.

   Classification using BOW rbf_svm

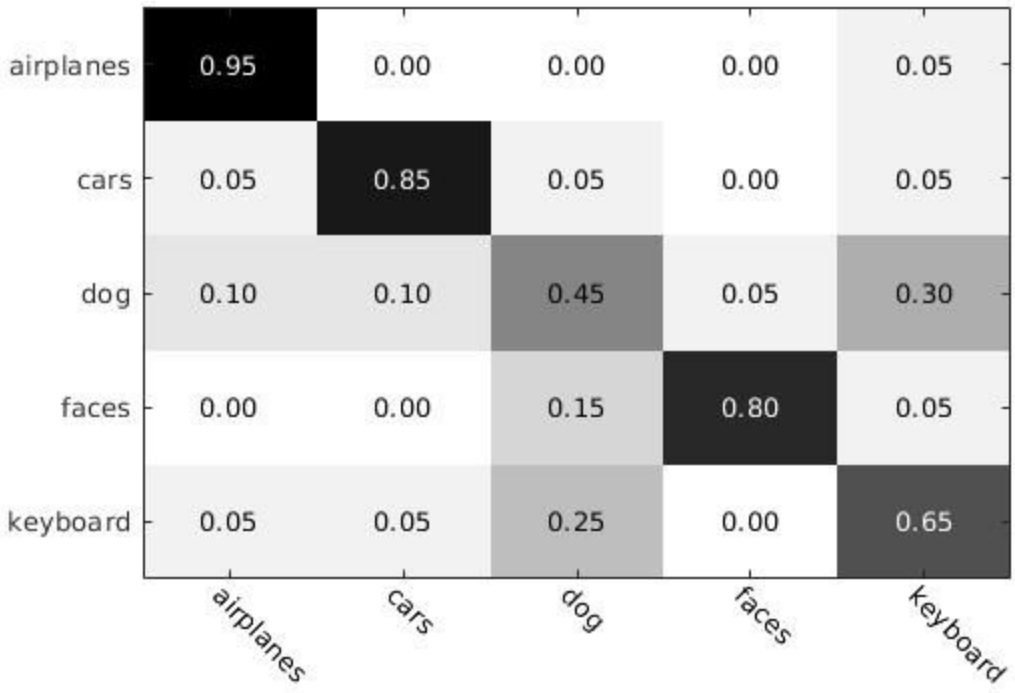
2. Apply the linear SVM classifiers to the test images and classify them.

   Accuracy = 74% (74/100) (classification)

3. Compute and report the overall and the classification errors per class.

   0.260000000000000

4. Compute and show the confusion matrix.



5. For each class show some images that are correctly classified and some images that are incorrectly classified. Can you explain some of the failures?

Ans: We can observe the correctly and incorrectly classified images below. The chances of certain failures are due to the varying values of the k involved in the classification algorithm.

PL-airplanes 	PL-keyboard 	PL-airplanes 	PL-dog 	PL-airplanes 
PL-keyboard 	PL-airplanes 	PL-airplanes 	PL-airplanes 	PL-keyboard 
PL-airplanes 	PL-airplanes 	PL-airplanes 	PL-keyboard 	PL-airplanes 
PL-keyboard 	PL-airplanes 	PL-faces 	PL-airplanes 	PL-keyboard 
PL-airplanes 	PL-cars 	PL-airplanes 	PL-airplanes 	PL-airplanes 
PL-cars 	PL-airplanes 	PL-keyboard 	PL-airplanes 	PL-keyboard 
PL-airplanes 	PL-keyboard 	PL-airplanes 	PL-dog 	PL-airplanes 
PL-dog 	PL-airplanes 	PL-dog 	PL-cars 	PL-dog