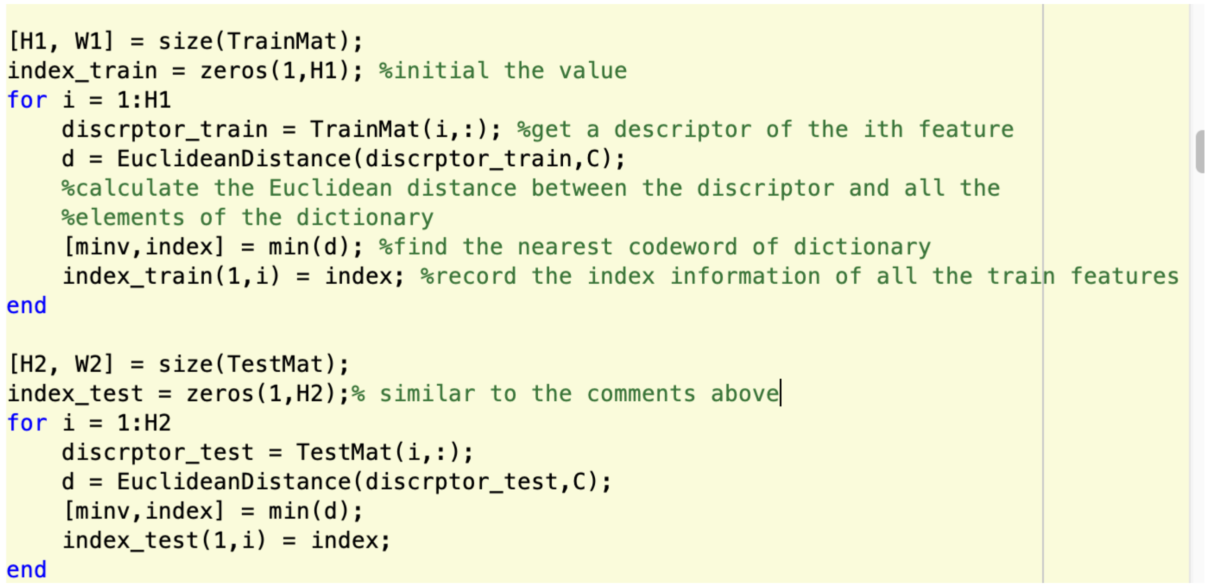
2 Dictionary creation – feature quantization

After running the programs, we can find there are 270000 features in 300 train images and 9000 features in 100 test images, in other words, each image provides 900 features. Besides, every feature is described by 128 double values (calculated using SIFT method).
For word dictionary (the size is specified by the variable 'DictionarySize'), 500 representative features (i.e. codewords) are selected into the dictionary.

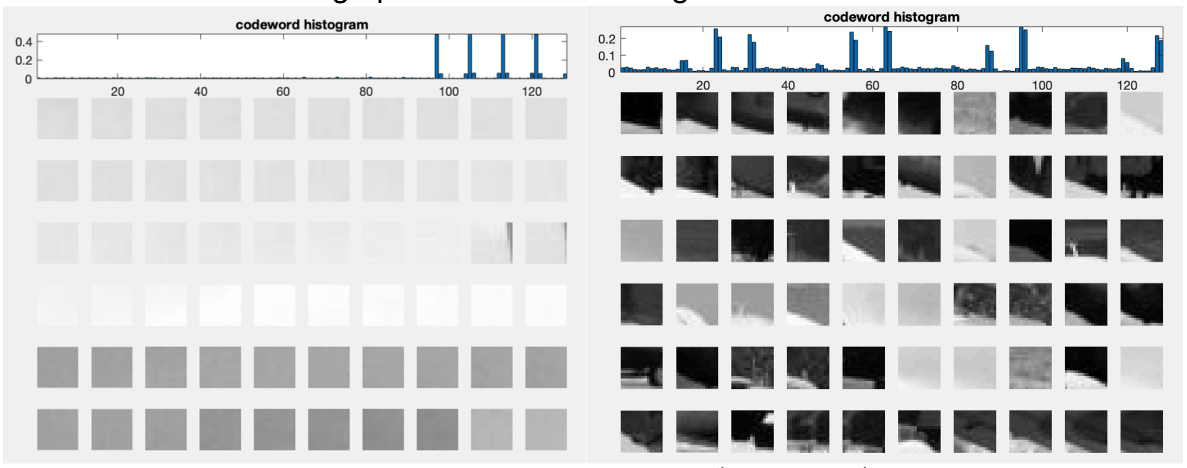
2.4 Write your own code that assigns each descriptor in the training and test images to the nearest codeword cluster.



```
[H1, W1] = size(TrainMat);
index_train = zeros(1,H1); %initial the value
for i = 1:H1
    discrptor_train = TrainMat(i,:); %get a descriptor of the ith feature
    d = EuclideanDistance(discrptor_train,C);
    %calculate the Euclidean distance between the discriptor and all the
    %elements of the dictionary
    [minv,index] = min(d); %find the nearest codeword of dictionary
    index_train(1,i) = index; %record the index information of all the train features
end

[H2, W2] = size(TestMat);
index_test = zeros(1,H2);% similar to the comments above
for i = 1:H2
    discrptor_test = TestMat(i,:);
    d = EuclideanDistance(discrptor_test,C);
    [minv,index] = min(d);
    index_test(1,i) = index;
end
```

Assign each feature to the nearest codeword in the dictionary using Euclidean distance, record all the corresponding index information in the variables 'index_train' or 'index_test'.

2.5 Visualise some image patches that are assigned to the same codeword.



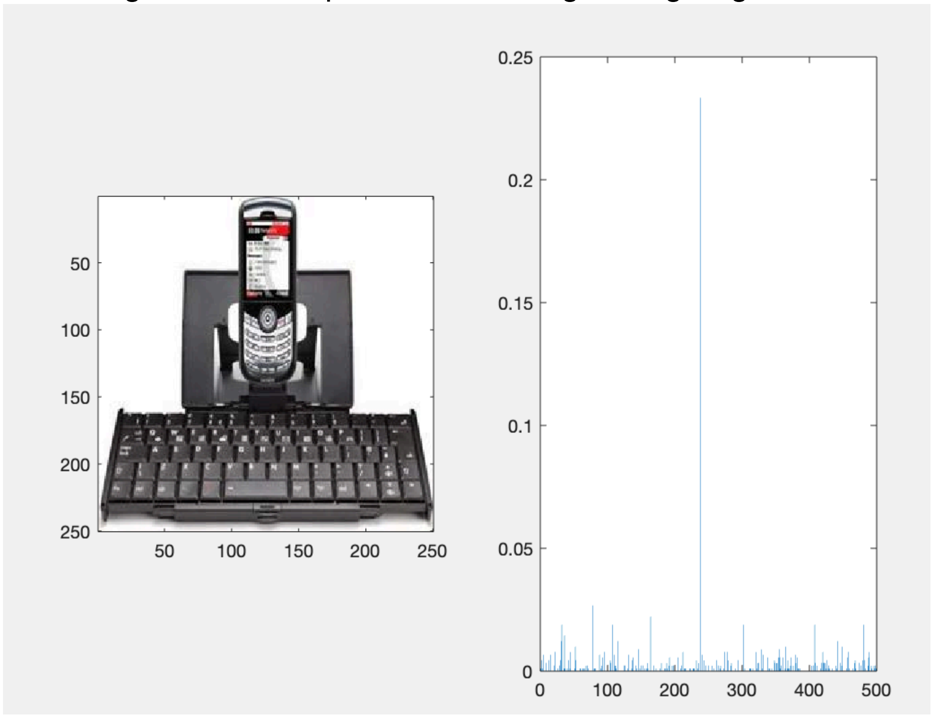
a) From the two images above (I chose the 78th and 397th codeword), we can find that all the train and test image patches have some similar attributes, which is why they can be represented by the same codeword histogram.
b) However, it is clear that some patches corresponding to the same codeword are distinct, such as some texture regions, dark and light background etc., therefore Euclidean distance may have its own weakness i.e. it cannot calculate the similarity of the image patches perfectly.

3. Image representation using Bag of Words representation

3.1 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).

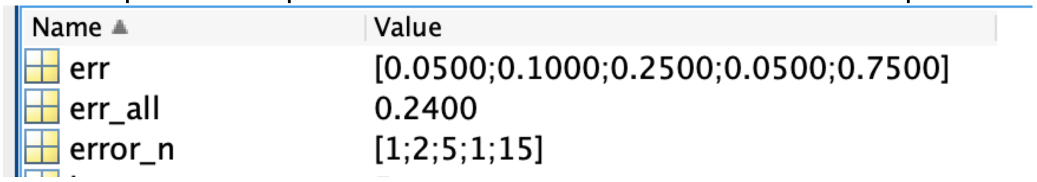
```
%------------------------ write your own code here----------------
%------------------------ write your own code here----------------
d = EuclideanDistance(features.data, C);%calculate the distance
histogram = zeros(1,vocbsize); |
[H, W] = size(d);
for j = 1:H
    [minv, index] = min(d(j,:));
    histogram(1,index) = histogram(1,index)+1;
end
[histogram, c]= do_normalize(histogram);

BoW(ii,:) = histogram;
```

The size of variable d is 900×500, as each image has 900 features and the dictionary has 500 representative codewords. Each row of the matrix d represents the Euclidean distances between the corresponding feature and all the codewords in the dictionary. Then, find the minimal value of each row of d, record the corresponding index and count the indexes into the histogram. After that, normalize the histogram which represents the image using Bag of Words.



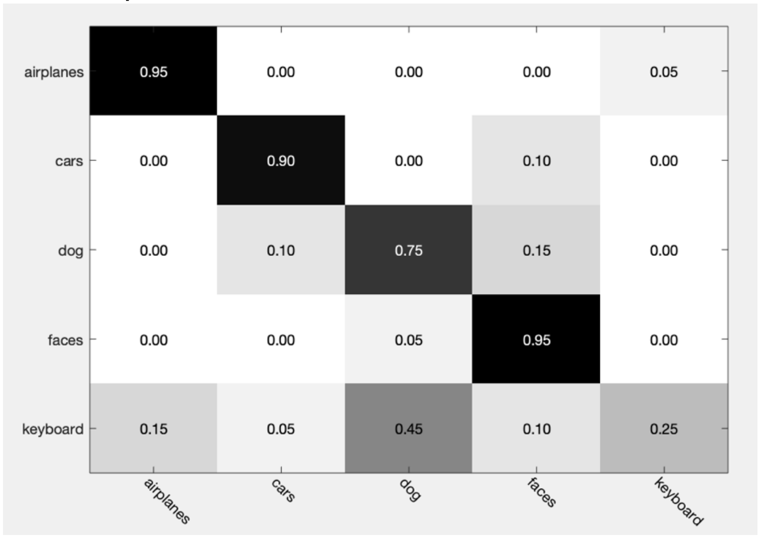
The right histogram is the representation of the left keyboard image.

4. Image classification using a Nearest Neighbour (NN) classifier.
Using the Euclidean distance in the KNN search, i.e. method 1.
4.2 Compute and report the overall and the classification errors per class.



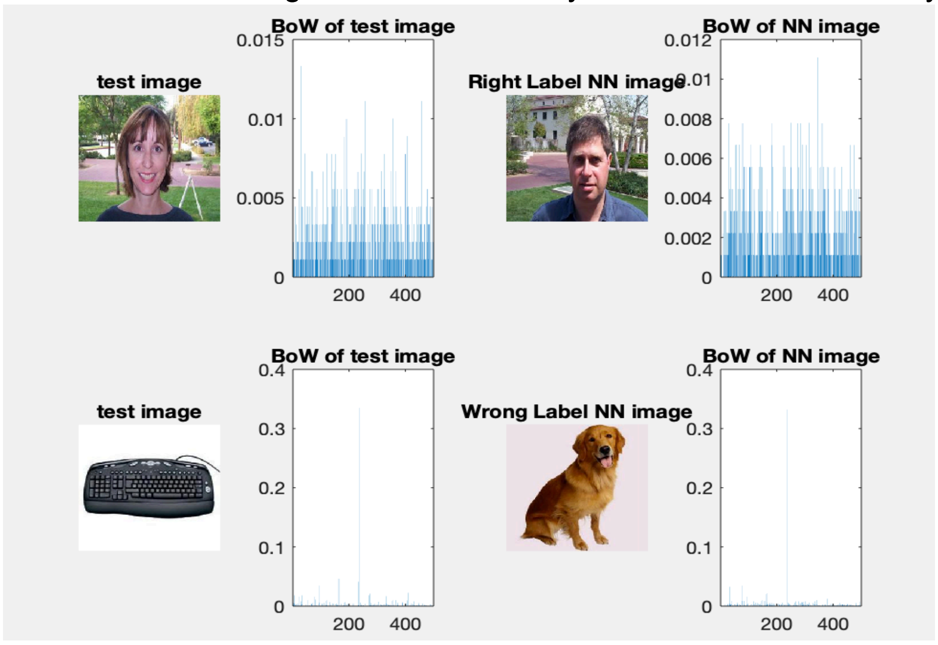
Name	Value
err	[0.0500;0.1000;0.2500;0.0500;0.7500]
err_all	0.2400
error_n	[1;2;5;1;15]

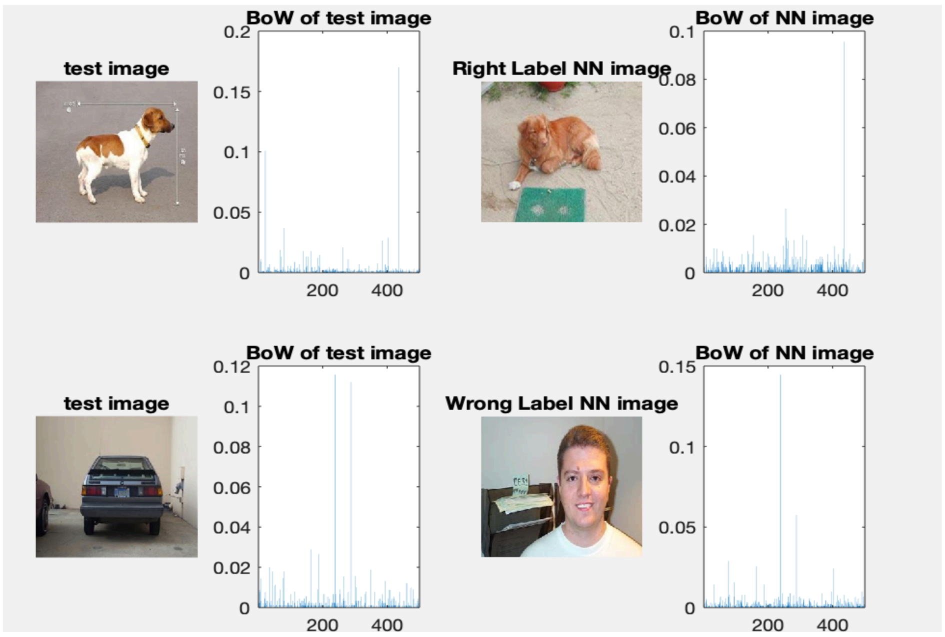
The overall classification error equals 24%, and the numbers and percentages of classification errors per class are shown above by the variable 'error_n' and 'err' respectively.
4.3 Compute and show the confusion matrix.



	airplanes	cars	dog	faces	keyboard
airplanes	0.95	0.00	0.00	0.00	0.05
cars	0.00	0.90	0.00	0.10	0.00
dog	0.00	0.10	0.75	0.15	0.00
faces	0.00	0.00	0.05	0.95	0.00
keyboard	0.15	0.05	0.45	0.10	0.25

From confusion matrix and errors, it is easier to identify these classes of airplanes, cars, and faces, and all their classification accuracies are above 90%. But for dogs and keyboard, the correct classification percentages are relatively low. Especially the classification proportion of keyboard is only 25%, and most keyboard images are misclassified as dog, the percentage of which up to 45%.
4.4 Show some images that are correctly classified and incorrectly classified.





For the images above, these images classified into the same class have very similar histogram distribution, whether they are classified correctly or incorrectly. For these incorrectly classified images, they may be not the typical images of the corresponding class, i.e. the incorrectly classified images above may be affected by illuminance, the background et el. In addition, the wrong label NN image (such as the guy with a white shirt) is not representative image of the class – 'faces', obviously the histogram distribution of image is different from those face histograms in the first row.

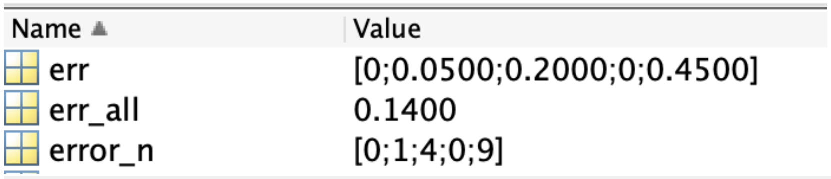
4.5 Write code for computing the histogram intersection between two histograms.

```
    % -------------- write your own code here ---------------
    % -------------- write your own code here ---------------
    sum=0;
    % It is used for aggregating the value of each element of the histogram
    for j = 1:p
        if a(1,j) < b(1,j) % just sum the lower value
            sum = sum + a(1,j);
        else
            sum = sum + b(1,j);
        end
    end
    d = 1 - sum; % the distance of the two histogram

  end
```

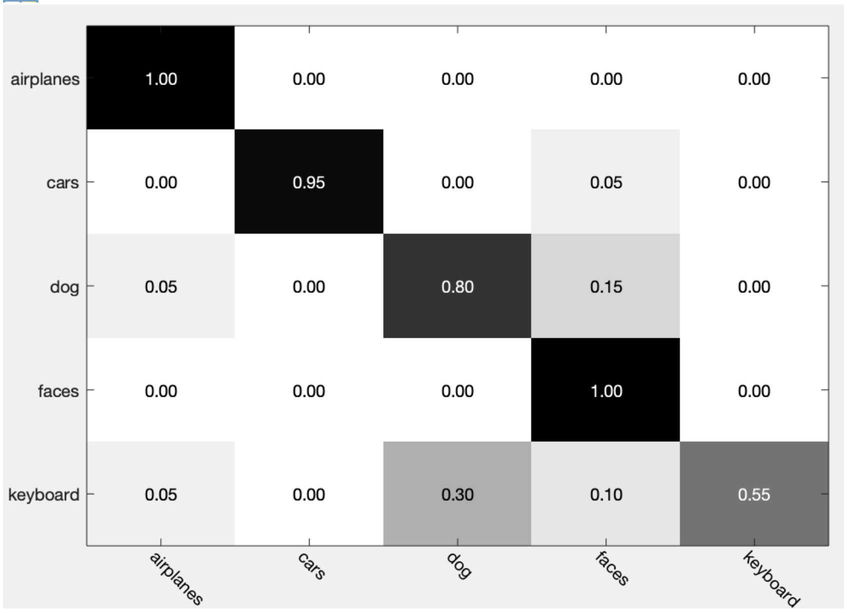
$$histint(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

where h_i, h_j denote two normalized histograms, K is the number of all the elements of the histogram. According to the formula, we can know the range of histogram intersection is between 0 and 1, and when the distance is close to 1 the two histograms are nearly not overlapping, when the distance is close to 0 the two histograms are nearly overlapping.

Now, change the method as method 2, i.e. histogram intersection.

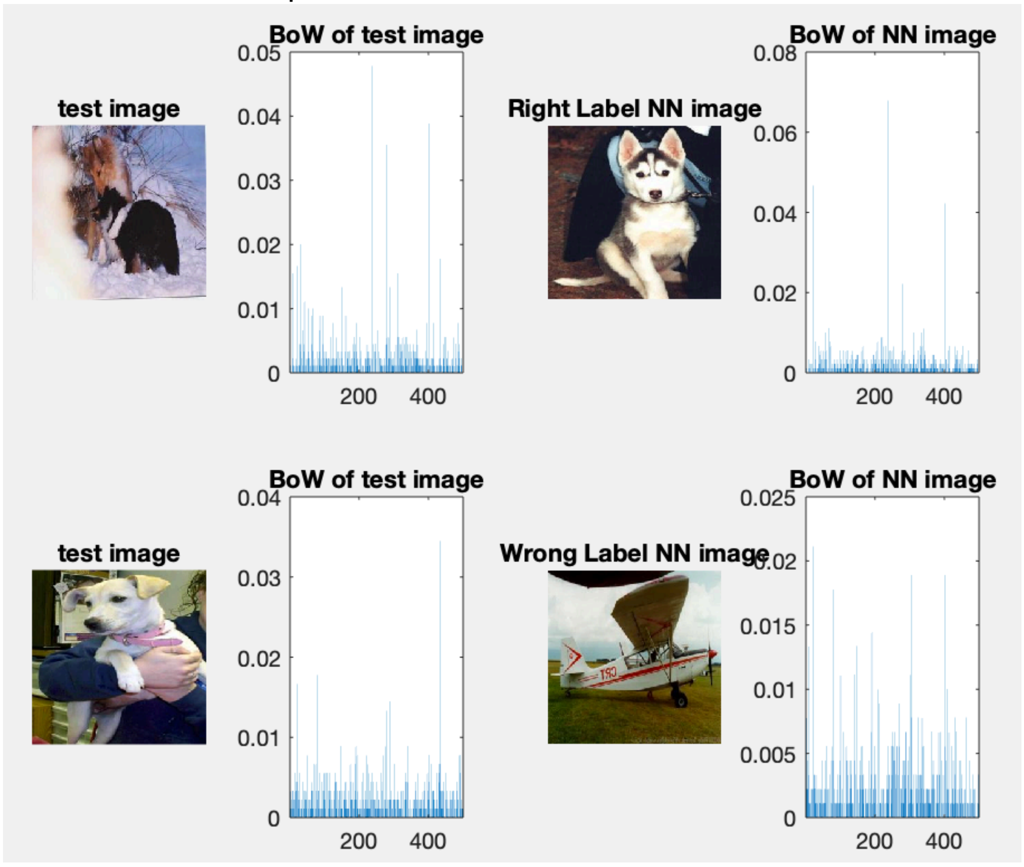


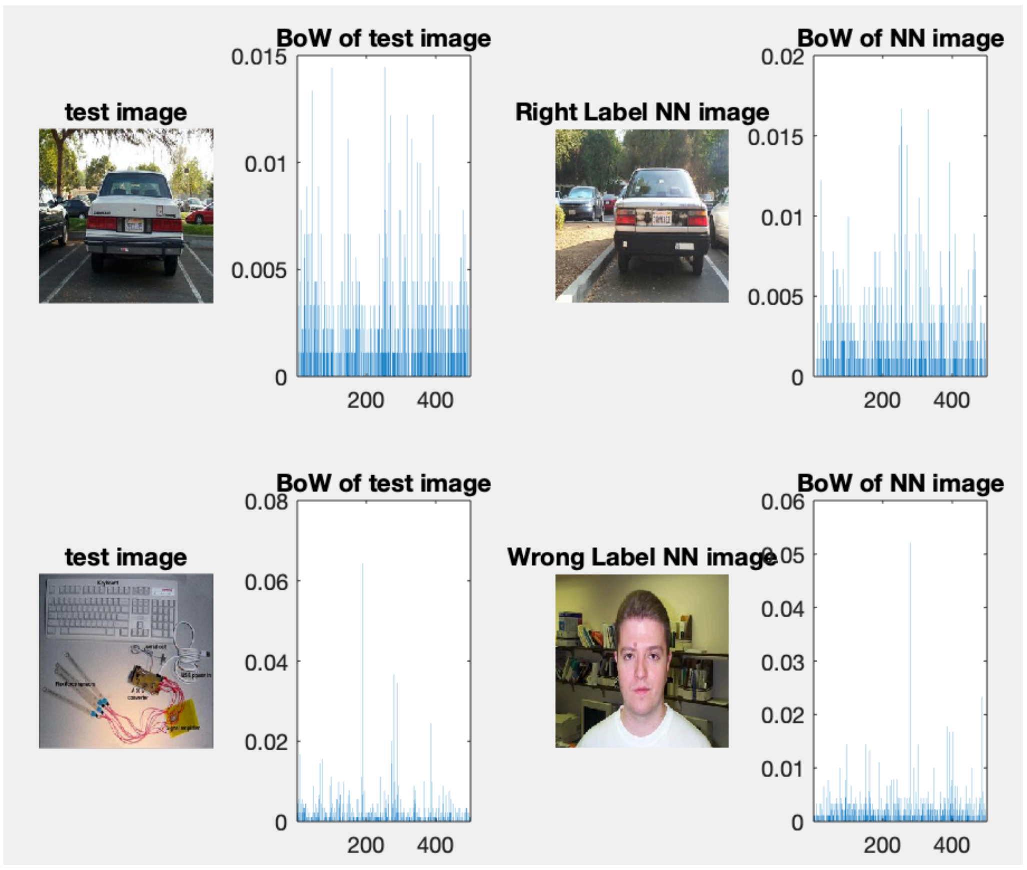
Name ▲	Value
err	[0;0.0500;0.2000;0;0.4500]
err_all	0.1400
error_n	[0;1;4;0;9]



	airplanes	cars	dog	faces	keyboard
airplanes	1.00	0.00	0.00	0.00	0.00
cars	0.00	0.95	0.00	0.05	0.00
dog	0.05	0.00	0.80	0.15	0.00
faces	0.00	0.00	0.00	1.00	0.00
keyboard	0.05	0.00	0.30	0.10	0.55

Using the histogram intersection to compute the similarity of the histograms (bags of words), we can find the overall error rate is 14%, which dramatically lower the first method (24%). Besides, the classification accuracy per class increases, especially for the classes – 'airplanes' and 'faces', both of whose accuracies reach 100%.

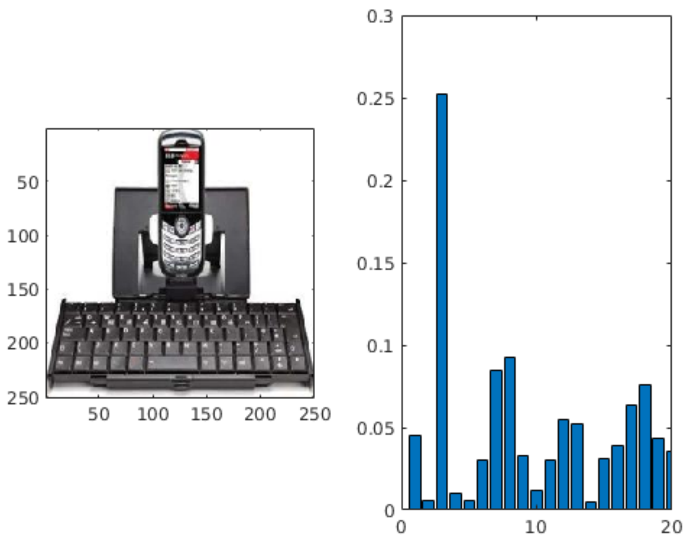




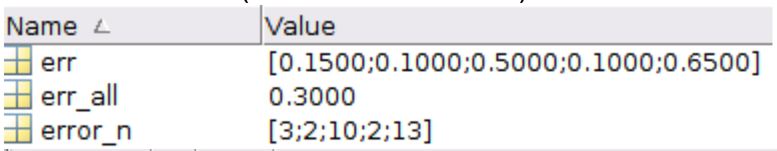
From the images classified incorrectly, their histograms still have a big overlapping part for the test image and the corresponding label image.

5. Dictionary size

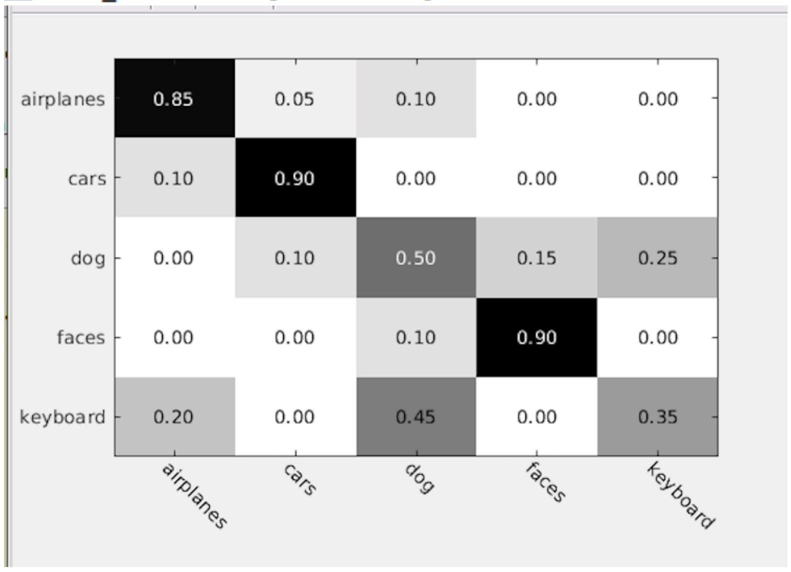
5.1 Perform the classification experiment using a very small dictionary and report the classification error and confusion matrices. Set the dictionary size as 20.
An example for the histogram (bag of words) of the image.



When method 1 (Euclidean distance) is used.



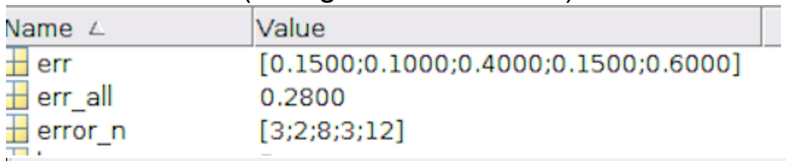
Name	Value
err	[0.1500;0.1000;0.5000;0.1000;0.6500]
err_all	0.3000
error_n	[3;2;10;2;13]



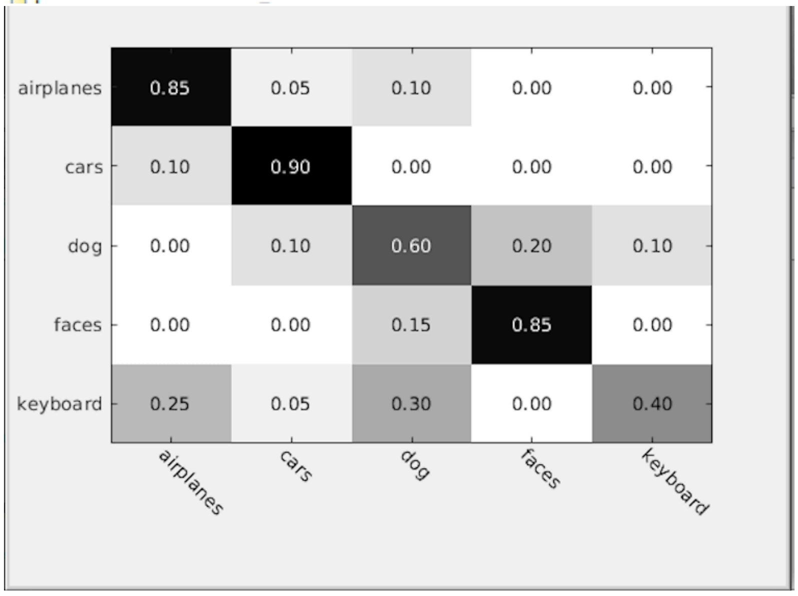
	airplanes	cars	dog	faces	keyboard
airplanes	0.85	0.05	0.10	0.00	0.00
cars	0.10	0.90	0.00	0.00	0.00
dog	0.00	0.10	0.50	0.15	0.25
faces	0.00	0.00	0.10	0.90	0.00
keyboard	0.20	0.00	0.45	0.00	0.35

The overall classification error is 30%, which is 6% higher than that of dictionary size of 500.

When method 2 (histogram intersection) is used.



Name	Value
err	[0.1500;0.1000;0.4000;0.1500;0.6000]
err_all	0.2800
error_n	[3;2;8;3;12]

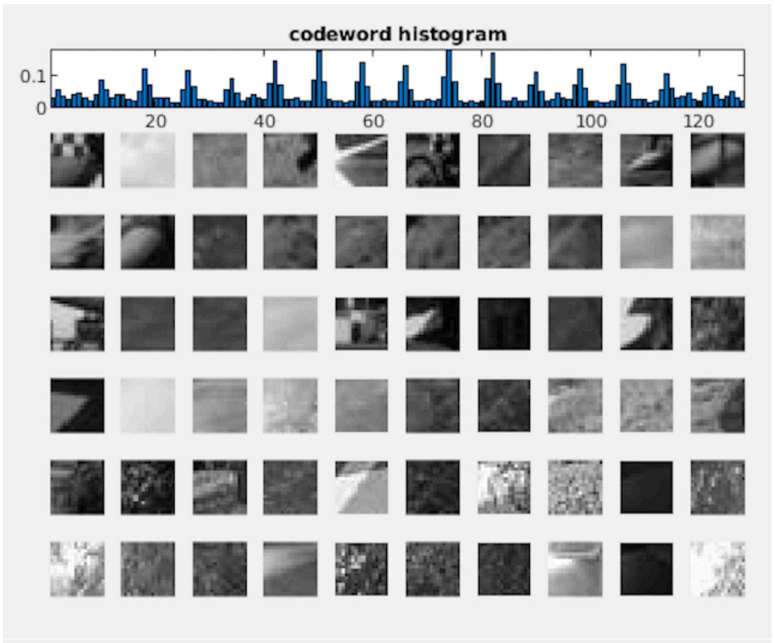


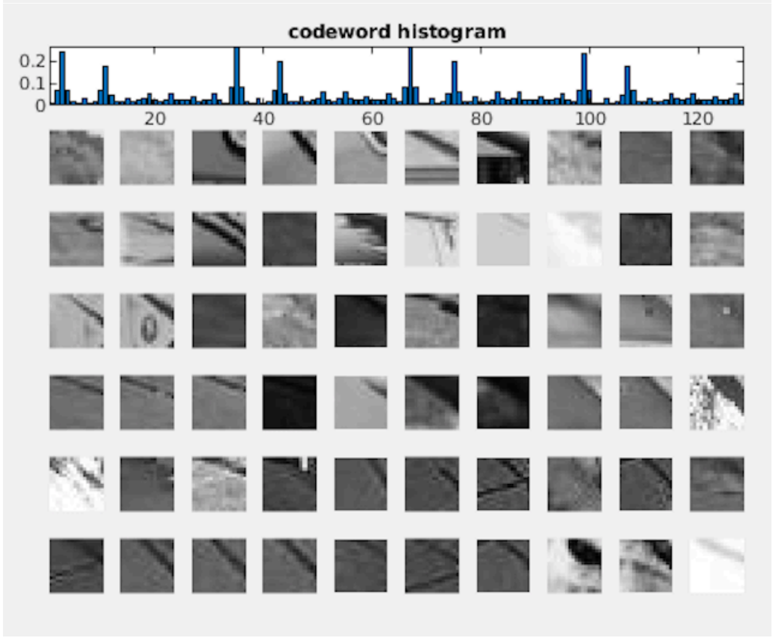
	airplanes	cars	dog	faces	keyboard
airplanes	0.85	0.05	0.10	0.00	0.00
cars	0.10	0.90	0.00	0.00	0.00
dog	0.00	0.10	0.60	0.20	0.10
faces	0.00	0.00	0.15	0.85	0.00
keyboard	0.25	0.05	0.30	0.00	0.40

The overall classification error is 28%, which is dramatically higher than that of dictionary size of 500 (14%).

To sum up, when we use a very small dictionary (20 clusters), the classification performance drops, compared with that when dictionary size is set as 500.

5.2 Explain the drop in the performance.





Perform step 2.5 when selecting ‘wordid’ = 10 or 20.
From these images above, we can find that many feature descriptors are not very similar. This is because the number of codewords is too small (just 20), so many feature descriptors are relatively similar to the corresponding codeword. In other words, the distance between the feature and the codeword is smaller than those between the feature and other 19 codewords, though the distance is still large. That is why the performance drops.

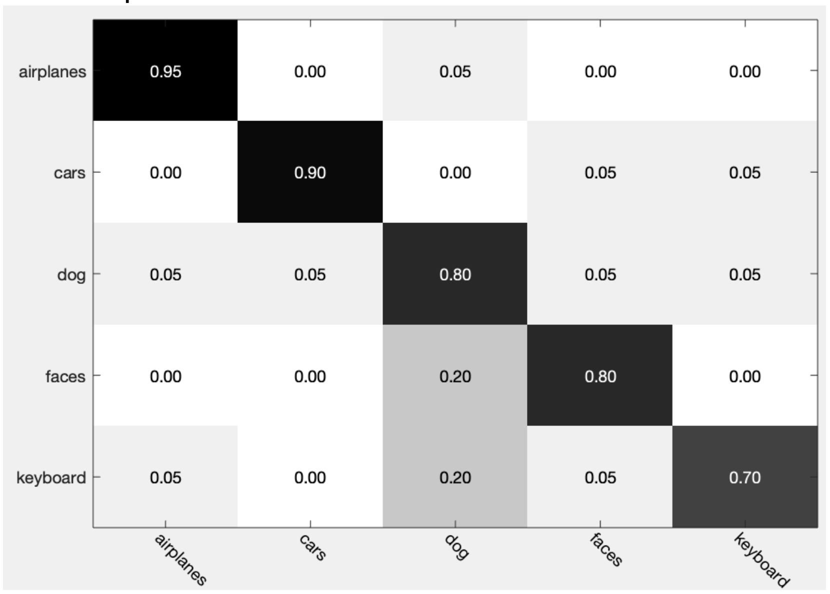
6. Image classification using a Support Vector Machines classifier

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

```
err          [0.0500;0.1000;0.2000;0.2000;0.3000]
err_all      0.1700
error_n      [1;2;4;4;6]
```

(Dictionary size is set as 500)
The overall classification error is 17%, which is lower than that of KNN method using Euclidean distance (24%), and higher than that of KNN method using histogram intersection (14%). And the classification errors per class are shown above, i.e. the error per class is 5%, 10%, 20%, 20%, 30% respectively.

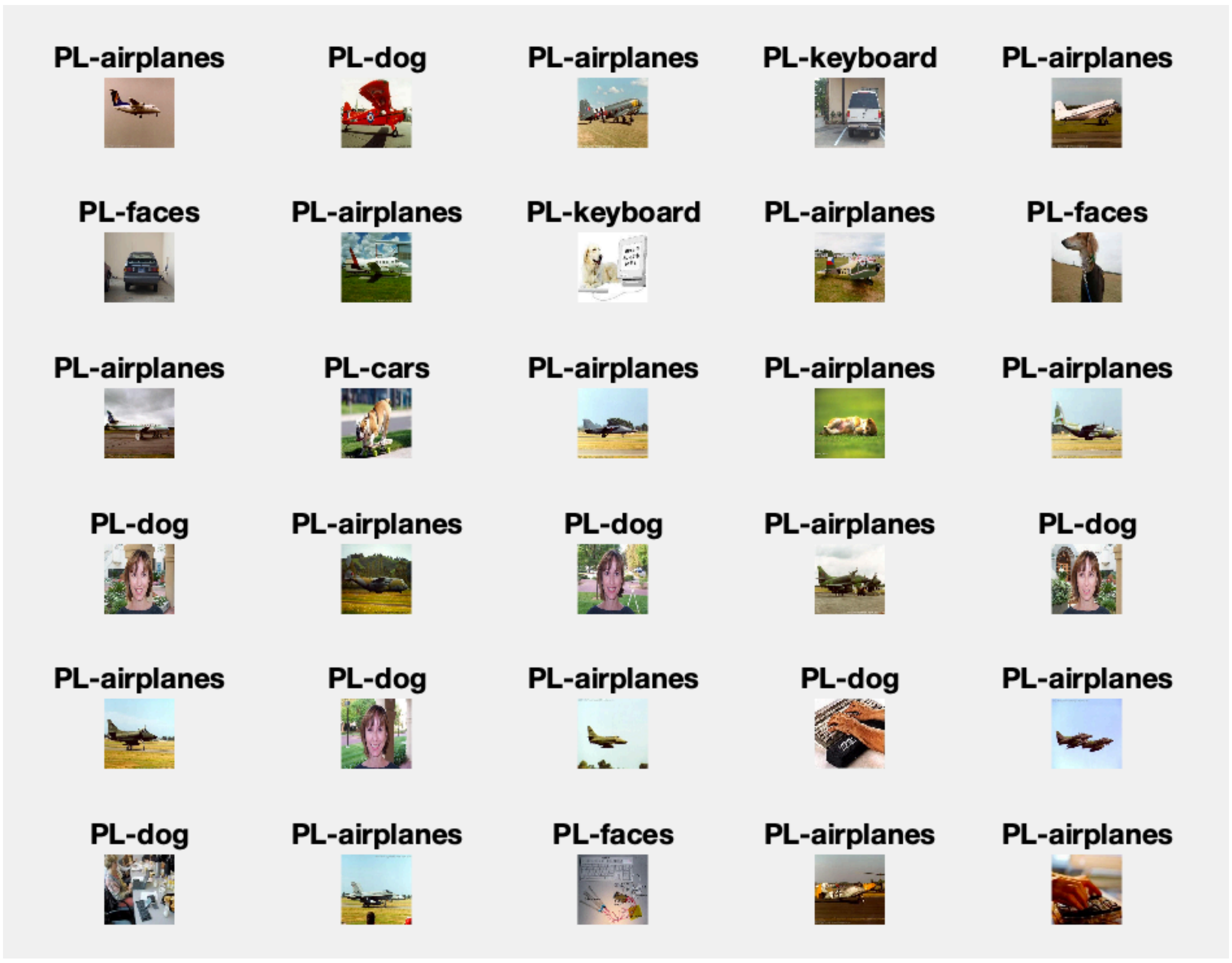
6.4 Compute and show the confusion matrix.



	airplanes	cars	dog	faces	keyboard
airplanes	0.95	0.00	0.05	0.00	0.00
cars	0.00	0.90	0.00	0.05	0.05
dog	0.05	0.05	0.80	0.05	0.05
faces	0.00	0.00	0.20	0.80	0.00
keyboard	0.05	0.00	0.20	0.05	0.70

Overall, compared with KNN, SVM has some improvements in some area, for instance the classification accuracy of the class – ‘keyboard’ is up to 70%, whereas that of using KNN method is just near 50%.

6.5 For each class show some images that are correctly classified and some images that are incorrectly classified.



Some histogram distributions of the images classified into the same class are very similar. For these incorrectly classified images, they may be not the typical images of the corresponding class, i.e. the incorrectly classified images above may be affected by illuminance, the background et el. For example, we can find the image with a face of a lady is incorrectly classified into ‘dog’, which is because their histograms are too similar so the SVM model classify it to ‘dog’.