CmpE 537
Fall 2019
Assignment 3: Image Classification with Bag-of-Features

Gurkan Demir
2015400177

December 4, 2019

1 Introduction

Bag of visual words (BOVW) is commonly used in image classification. Its concept is adapted from information retrieval and NLP's bag of words (BOW).

The general idea of bag of visual words (BOVW) is to represent an image as a set of features. Features consists of keypoints and descriptors. Keypoints are the "stand out" points in an image, so no matter the image is rotated, shrink, or expand, its keypoints will always be the same. And descriptor is the description of the keypoint. We use the keypoints and descriptors to construct vocabularies and represent each image as a frequency histogram of features that are in the image. From the frequency histogram, later, we can find another similar images or predict the category of the image.

In this assignment, we are expected to develop an image classifier based on Bag-of-Features model using Python. We are given a dataset which contains variable number of instances per class (There are 7 classes: City, Face, Greenery, Building, House Indoor, Office, Sea). The dataset is also divided into two as training and test. We are expected to train our classifier using the training image set and test it using the test image set.

2 Implementation

1. Find training images path.
2. Read train images, and compute SIFT descriptors.
3. Find the dictionary.
4. Quantize features.
5. Classify test images.

3 How to Run?

In order to run image classifier, execute the following from the command line:

python3 BoW.py –train_path [TRAIN_PATH] –test_path [TEST_PATH] –no_clusters [NO_CLUSTERS] –kernel [KERNEL]

where;

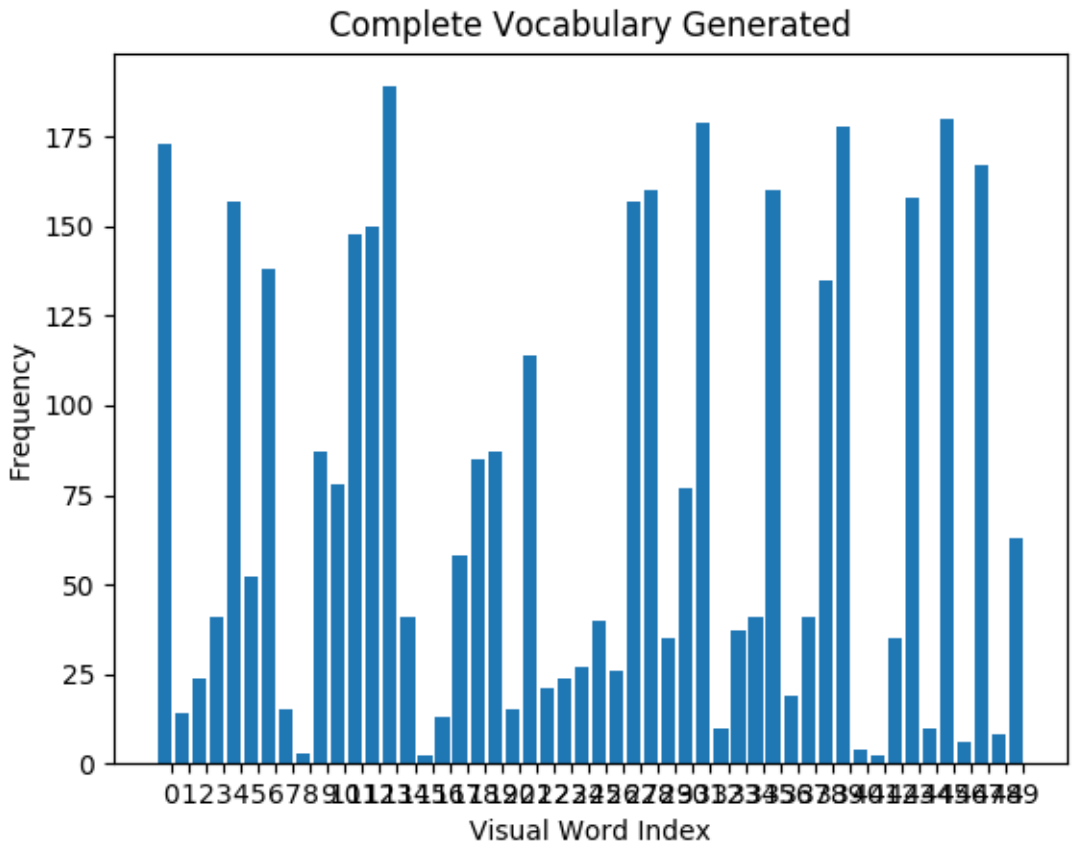
1. **TRAIN_PATH :** Path of train dataset.
2. **TEST_PATH :** Path of test dataset.
3. **NO_CLUSTERS :** Number of clusters, default 50.
4. **KERNEL :** Type of kernel, linear or precomputed, default linear.

For example, if you want to classify images with precomputed kernel, where train path is **dataset/train** and test path is **dataset/test** by choosing **100** cluster centers. You have to execute following command:

python3 BoW.py –train_path dataset/train –test_path dataset/test –no_clusters 100 –kernel precomputed

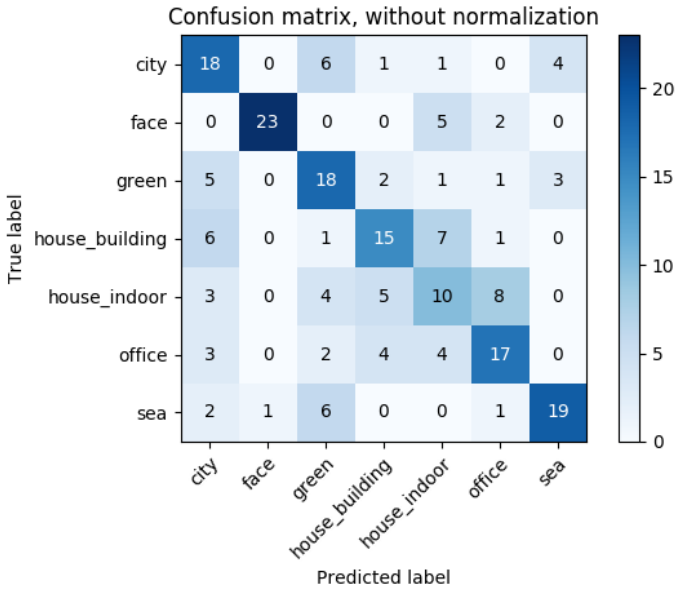
4 Outputs

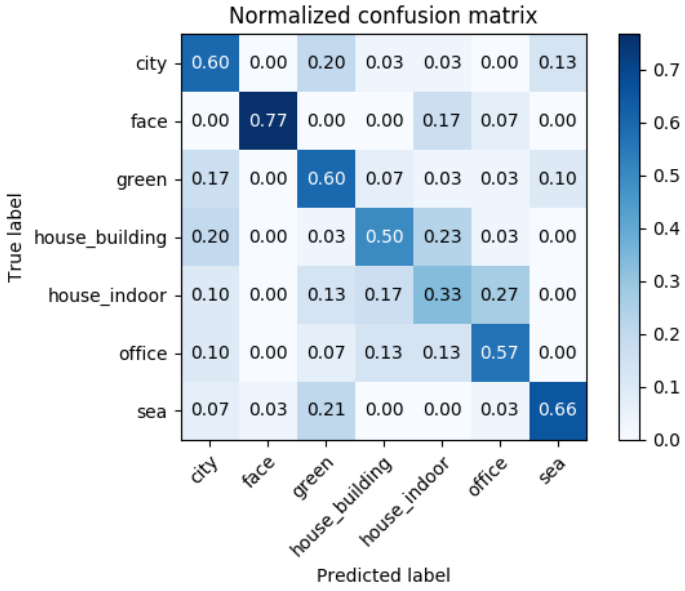
4.1 K = 50



1. Linear Kernel
   - Training
     - Average Training Accuracy: 0.736
     - Selected Parameters: C = 0.1

- Tests
  - Confusion Matrix

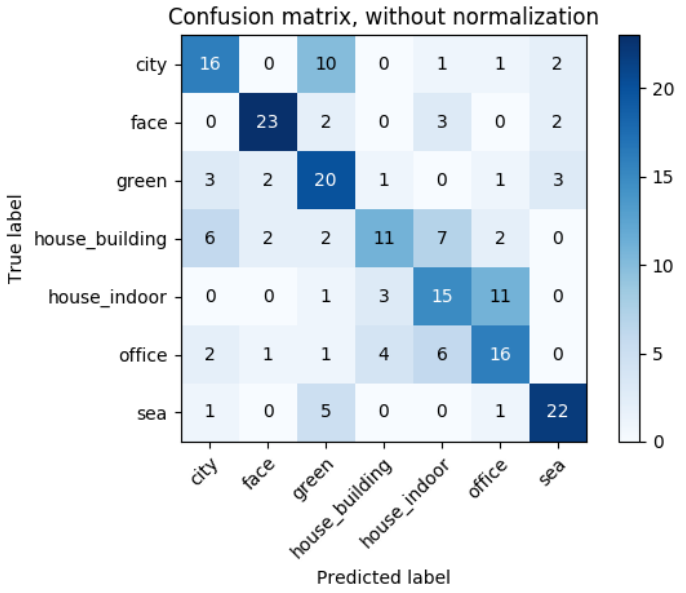


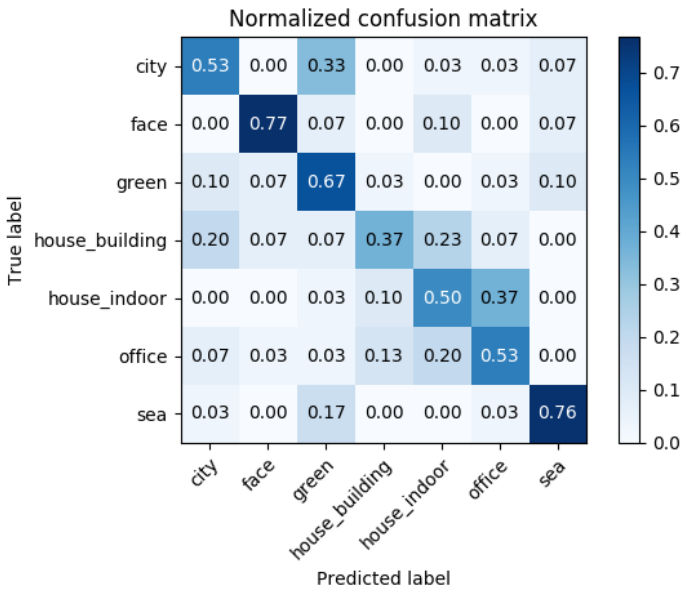


  - Average Testing Accuracy: 0.574

2. Precomputed Kernel

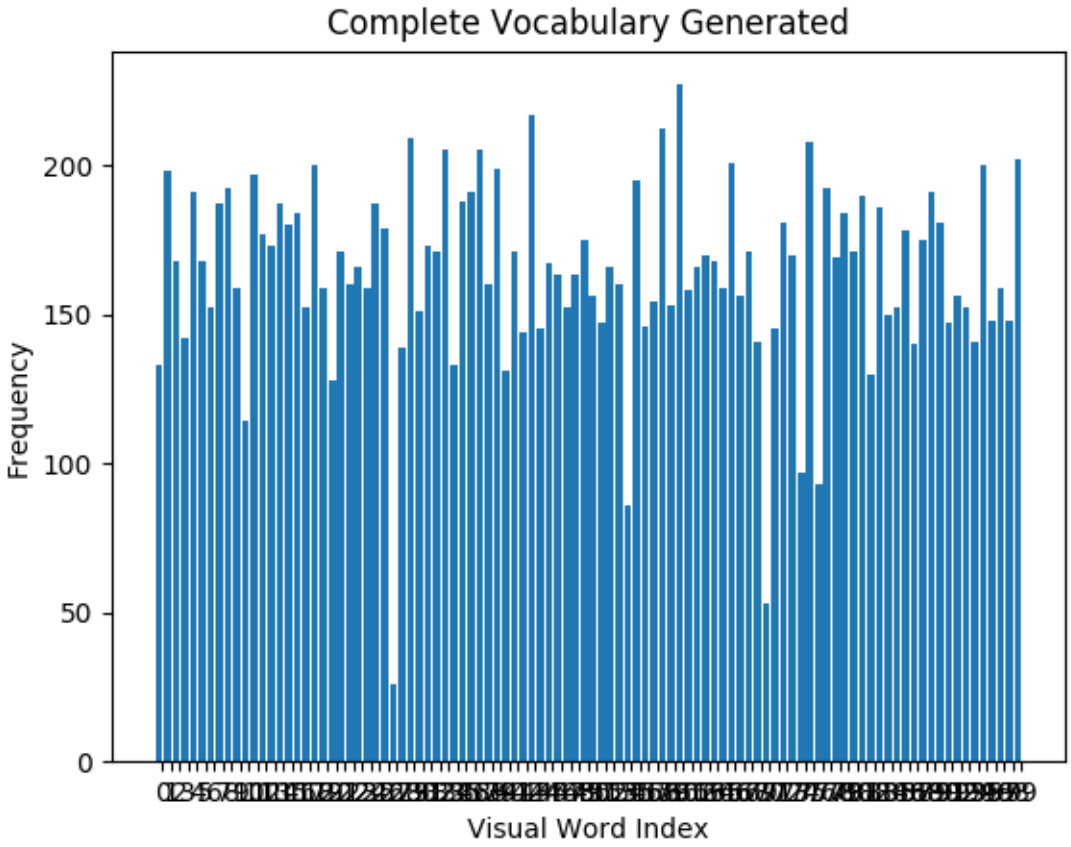
- Training
  - Average Training Accuracy: 0.76
  - Selected Parameters: C = 0.1
- Tests
  - Confusion Matrix



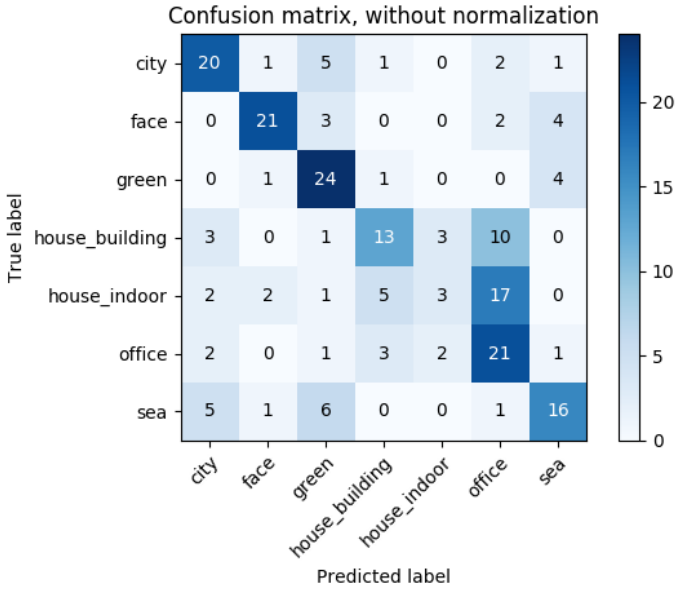


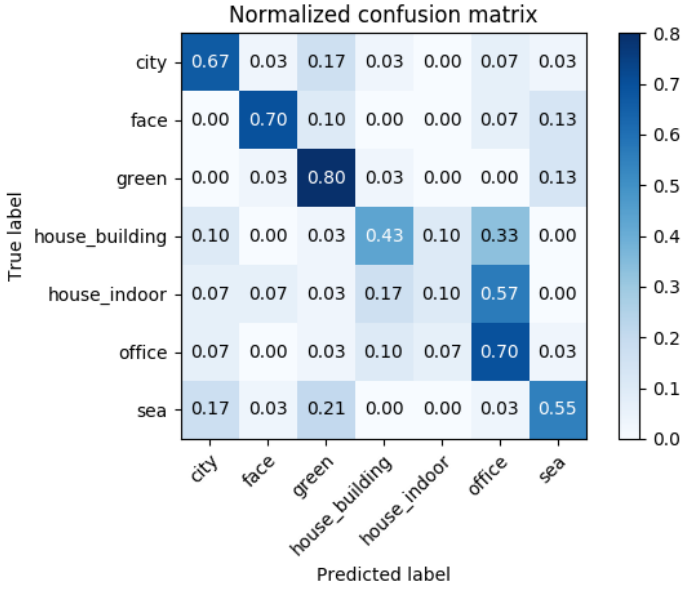
- Average Testing Accuracy: 0.589

4.2 K = 100



1. Linear Kernel
   - Training
     - Average Training Accuracy: 0.922
     - Selected Parameters: C = 0.2
   - Tests
     - Confusion Matrix



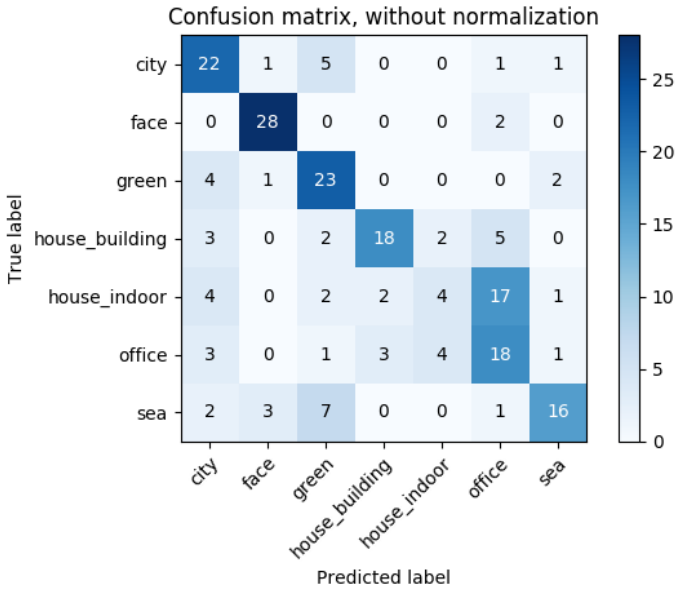


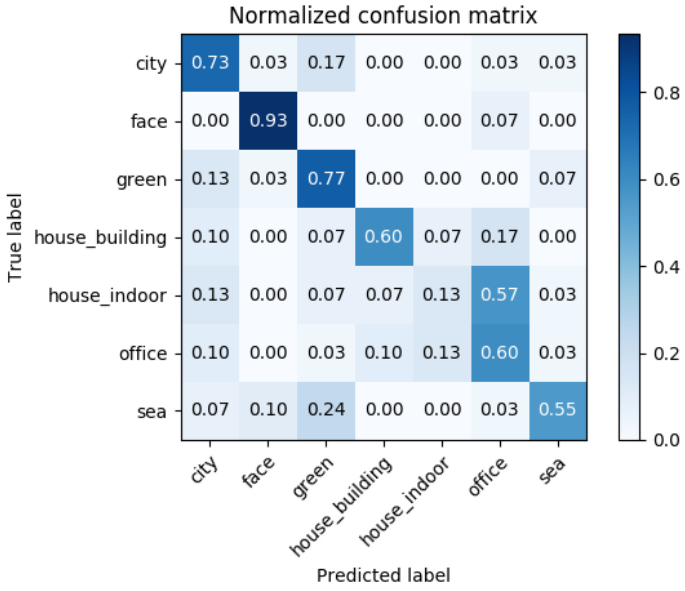
  - Average Testing Accuracy: 0.565

2. Precomputed Kernel

   - Training
     - Average Training Accuracy: 0.931
     - Selected Parameters: C = 0.1

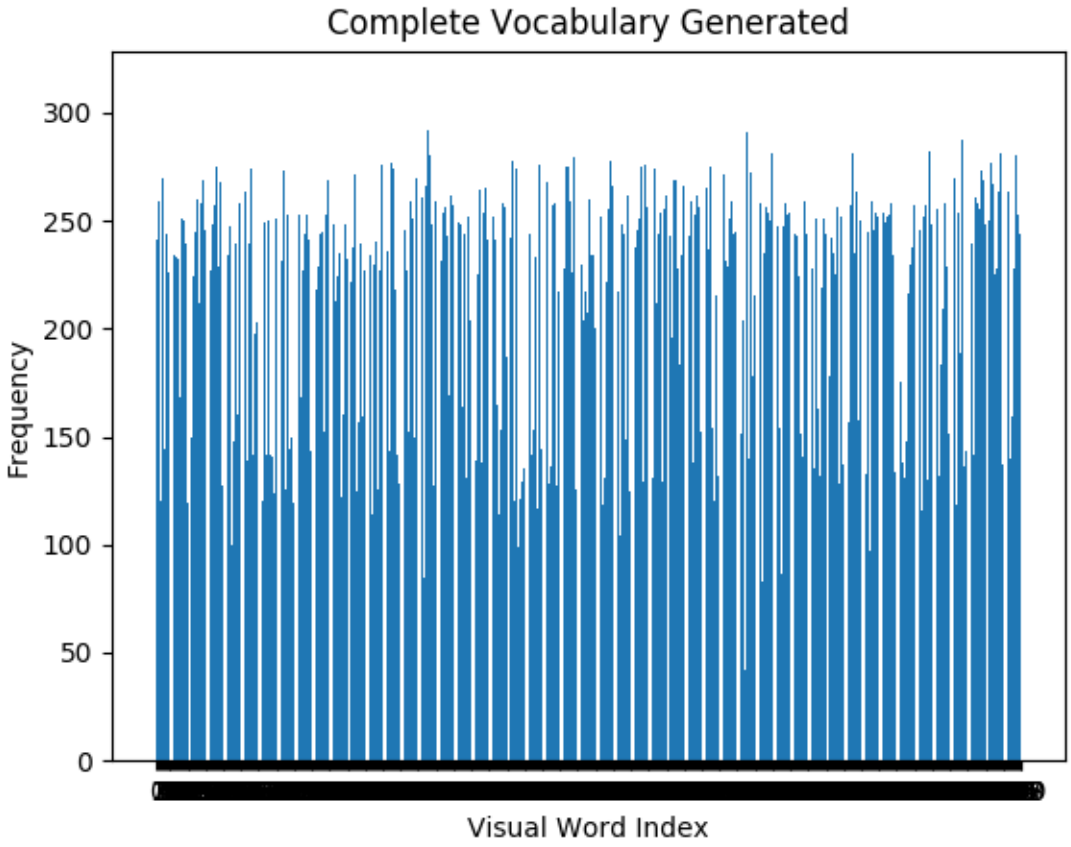
- Tests
  - Confusion Matrix



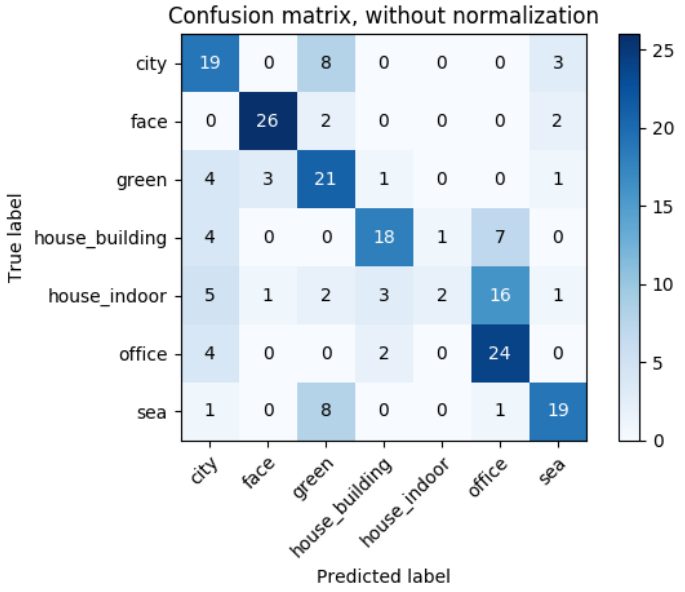


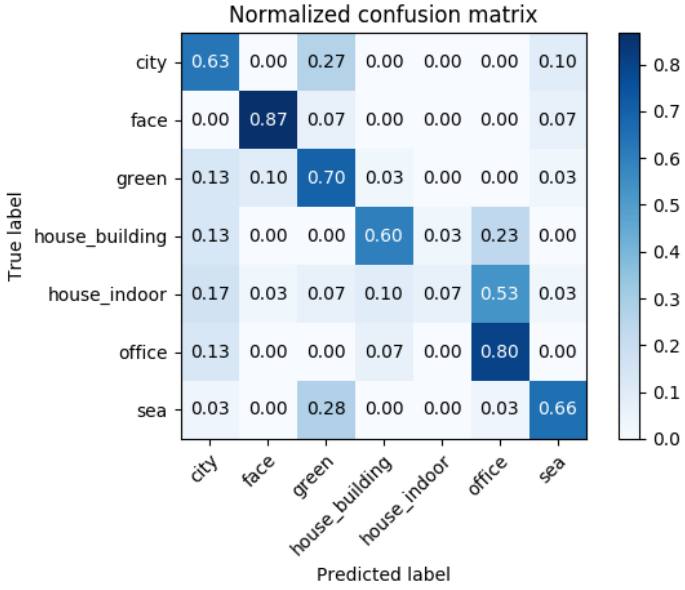
  - Average Testing Accuracy: 0.617

4.2.1 K = 500



1. Linear Kernel
   - Training
     - Average Training Accuracy: 0.999
     - Selected Parameters: C = 0.5
   - Tests
     - Confusion Matrix



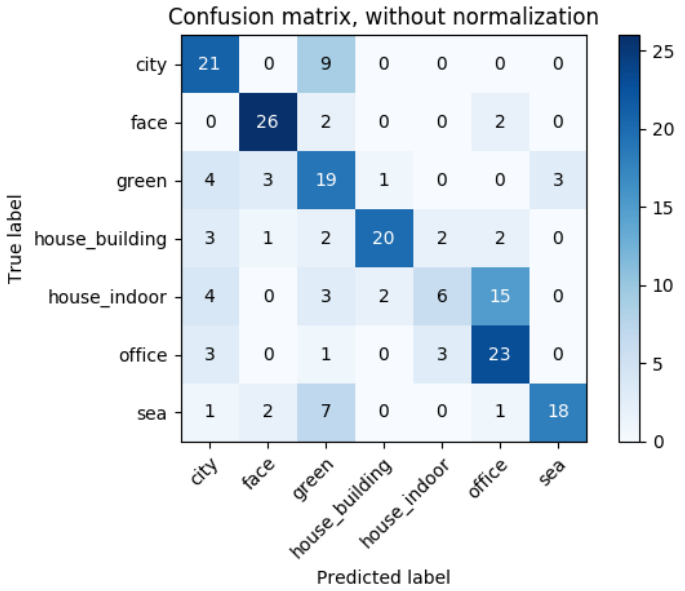


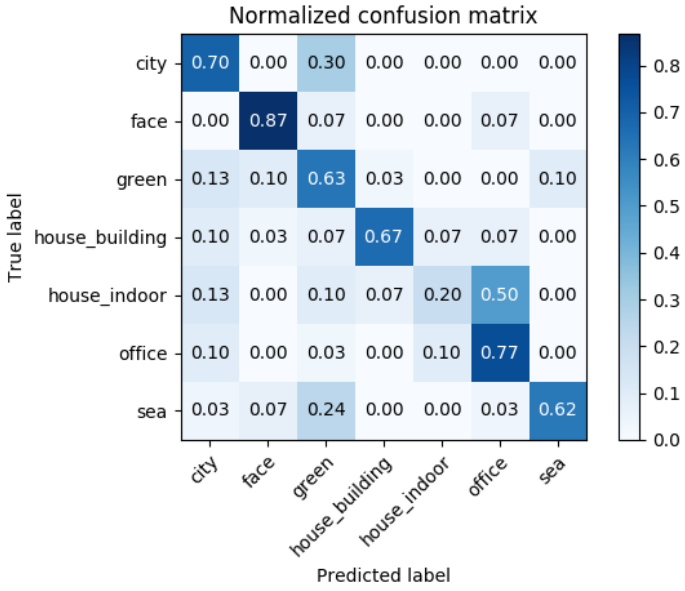
  - Average Testing Accuracy: 0.61

2. Precomputed Kernel

   - Training
     - Average Training Accuracy: 0.998
     - Selected Parameters: C = 0.5

- Tests
  - Confusion Matrix





  - Average Testing Accuracy: 0.636

5 Results

In this homework, we are expected to classify test images according to training images. While doing this, we are expected to use Support Vector Machine as a classifier with linear and precomputed kernels.

Training images are classified into 7 classes which are city, face, green, house building, house indoor, office and sea. In city, face and office classes there are 140 train images, in green class there exists 133 images, also there are 70 images in house building, 42 house indoor, and 142 sea images. Since there exists different number of train images for each class, it is an unbalanced situation. This makes our model biased, which means classes which has small number of train images might be predicted less. In order to prevent it, I changed each class weight. I assigned each class weight to:

(# of images) / (# of class * # of images for that class)

By doing this, prediction rate of house indoor and house building increases.

Beside of those, in this homework we are expected to cluster image descriptors into 50, 100 and 500 using KMeans. When we use original image size, clustering them takes too much time. In order to save from time, I resized all input images into (150 x 150). By doing this, clustering image descriptors takes much less time. However it is a trade-off, when we resize images, number of descriptors for each image and accuracy of testing decreases, but we save from time.

Additionally, in this homework, we use 2 different kernels for SVM classifier, which are linear and precomputed. As can be seen from the outputs above, experiments with precomputed kernel, has much more accuracy than linear kernel. Since precomputed Chi-Squared kernels are useful for handling discrete features such as bag-of-features, it results better than linear kernel.

Furthermore, in this homework, we cluster image descriptors into 50, 100 and 500 clusters. For different cluster sizes, our training accuracy differs. Higher cluster size means higher training accuracy. From this point of view, we are expected to say that if we increase cluster size, our testing accuracy increases too. But according to outputs listed above, testing accuracy sometimes decreases, sometimes increases. Of course, it can sourced from KMeans, descriptors etc.

Finally, thanks to this homework, I had hand-on experience on image classification using Support Vector Machines. Also, I had a chance to think on how to increase accuracy, or how to decrease computation time while doing this homework.