

Image Retrieval System

HE Chencheng

January 2019

1 Introduction

This first project targets to build a system to retrieve near duplicate images. The retrieval system can be divided into two main part as following:

- **Image representation:** There are three main representation: local representation, global representation and deep representation.
- **Comparison:** After we represent images properly, we can compare our target image with our image databases to retrieve the most similar image.

2 Implementation Steps

Step1: Local image representation

Extracting keypoints and local invariant descriptors from **all the images** in my training dataset by using SIFT function in OpenCV module.

Step2: Global image representation: Bag of Words

All the descriptors extracted from all the images in training dataset is the Bag of Words. Clustering these descriptors to form a codebook/vocabulary using K-means algorithm. Based on the result of K-means algorithm, we will get K centers, and thus we can Quantify the local invariant descriptors from each image to form a single histogram that represents the visual contents of an image. Through this process we should get a vector with K dimensions to represent a image.

Step3: Comparison

We transform all our training images into vectors representation. Now we want to retrieve a target image to get a most similar image from our database. We should do the same operations to the target image to form a vector representation. And then we can compare the distance between target image vector and

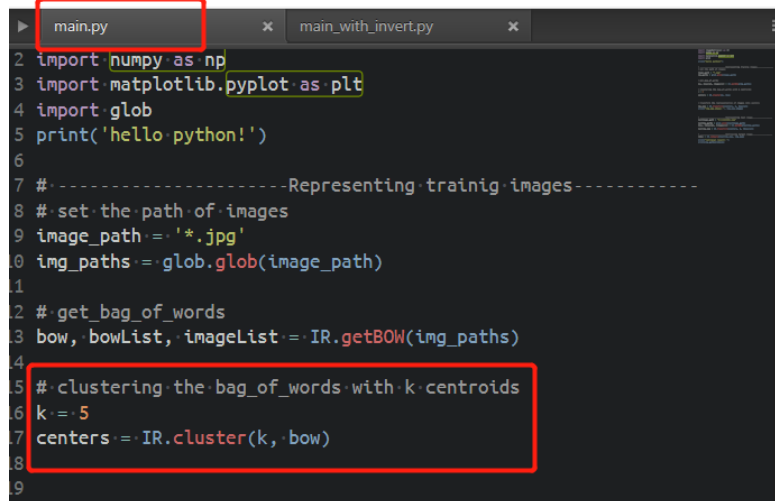
every image in our database. We will return the index of image, which has the shortest **Norm distance** with our target image.

3 Testing My Code

If you want to test my code, just have a few things to remind you:

- if you just stay in my folder, you can just run the **main.py** file, directly. The test image here is '115602.jpg' which is included in the training image datasets, so you should get the result: 115602.jpg
- you can also change the value of K here to test different examples.

Figure 1: you can modify the value of K



```
1 main.py x main_with_invert.py x
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import glob
5 print('hello python!')
6
7 #-----Representing trainig images-----
8 # set the path of images
9 image_path = '*.jpg'
10 img_paths = glob.glob(image_path)
11
12 # get bag of words
13 bow, bowList, imageList = IR.getBOW(img_paths)
14
15 # clustering the bag of words with k centroids
16 k = 5
17 centers = IR.cluster(k, bow)
18
19
```

- you should have the both **main.py** and **imageRetrieval.py** under your training image datasets. Make sure they are in the same directory.

Figure 2: Make sure python files and images are in the same folder

115100	2006/5/13 8:56	JPG 文件	1,030 KB
115101	2006/5/13 8:59	JPG 文件	1,040 KB
115200	2006/5/13 8:51	JPG 文件	1,054 KB
115201	2006/5/13 8:52	JPG 文件	1,044 KB
115300	2006/5/12 17:47	JPG 文件	2,132 KB
115301	2006/5/12 18:00	JPG 文件	1,029 KB
115400	2006/5/12 16:25	JPG 文件	1,037 KB
115401	2006/5/12 16:27	JPG 文件	1,040 KB
115500	2006/5/11 9:44	JPG 文件	1,045 KB
115501	2006/5/11 9:45	JPG 文件	1,050 KB
115600	2006/5/10 18:51	JPG 文件	1,053 KB
115601	2006/5/10 18:51	JPG 文件	1,096 KB
115700	2006/5/10 16:40	JPG 文件	1,052 KB
115701	2006/5/10 16:41	JPG 文件	1,054 KB
115702	2006/5/10 16:43	JPG 文件	1,060 KB
imageRetrieval	2019/1/26 15:56	PY 文件	3 KB
main	2019/1/26 23:13	PY 文件	1 KB
Readme	2019/1/26 2:18	文本文档	1 KB

main
imageRetrieval
.jpg files

- you can also move the test image '115602.jpg' to other place in your computer, but just make sure that you modify the test image path in my code

Figure 3: you can set the path of your test image

```

23
24
25 # ----- Representing test image -----
26 testImage_path = 'D://115602.jpg'
27 testImg_paths = glob.glob(testImage_path)
28 tbow, tbowList, timageList = IR.getBOW(testImg_paths)
29 testing_rep = IR.transform(centers, k, tbowList)
30
31 # ----- Retrieving target image -----

```