Trường Đại Học Công Nghệ Thông Tin
*****

Báo Cáo Tiến Độ Lần 1
Môn Truy Vấn Thông Tin Đa Phương Tiện

Nguyễn Ngọc Thừa – 15520859
Trần Minh Thuận – 15520863
Đinh Nguyễn Tiến Đạt - 15520099

I. Yêu Cầu Bài Toán

- Bài toán tìm kiếm tập các ảnh có thông tin liên quan đến một bức ảnh input đầu vào từ tập dữ liệu(dataset) cho trước.
- Input: là một ảnh đầu vào cần tìm kiếm.
- Output: một danh sách các tên file ảnh có chứa thông tin liên quan đến ảnh input đầu vào.

II. Các Phương Pháp Được Áp Dụng

- Thuật toán nhận diện và mô tả đặc trưng cục bộ Scale-invariant feature transform(SIFT).
- Mô hình truy xuất thông tin Bag of Words
- Phương pháp gôm cụm Kmeans
- Phương pháp Inverted index
- Độ đo sự tương đồng Cosin

III. Bộ Dữ Liệu

- Bộ dữ liệu được sử dụng để test hệ thống là German Traffic Sign Benchmarks bao gồm 900 ảnh.
-http://benchmark.ini.rub.de/?section=gtsdb&subsection=dataset

IV. Các Bước Thực Hiện

-Load toàn bộ tập ảnh từ dataset.
-Tính đặc trưng cục bộ (SIFT) cho toàn bộ tập ảnh.
-Phân n cụm(n word) cho toàn bộ các đặc trưng thu được ở trên với KMeans.
-Xây dựng tập vocabulary từ n word thu được ở trên.
- Với từng file ảnh tính đặc trưng cục bộ cho ra m đặc trưng
- Với từng đặc trưng trong m đặc trưng thu được ở trên phân nó vào một trong n cụm của tập vocabulary. Khi đó với mỗi ảnh ta lập được một vector thể hiện tần số xuất hiện của từng word(bag of word)
- Thành lập inverted index thể hiện sự liên hệ của từng file ảnh với các cụm.
- Với ảnh query ta cũng thực hiện như các bước trên: tính đặc trưng cục bộ → phân từng đặc trưng thu được vào từng cụm → vector thể hiện số lượng đặc trưng tương ứng với từng cụm (query vector)
- Từ inverted index → các file cần so sánh.
- Tính sự tương đồng giữa query vector và vector của từng file → n file có độ tương đồng cao nhất.

V. Code

-Tính toán đặc trưng cục bộ SIFT

```
###Computing SIFT features
if not os.path.isfile("all_feature_array.data"):
        print("Computing sift feature...")
        for filename in os.listdir(dataset_path):
                img = cv2.imread(os.path.join(dataset_path, filename), 0)

                kp = sift.detect(img, None)
                kp, des = sift.compute(img, kp)
                file_feature_list.append((filename, des))
                for kp_feature in des:
                        all_feature_list.append(kp_feature)
                #print(des)
        all_feature_array = np.array(all_feature_list)
        pickle.dump(all_feature_array, open("all_feature_array.data", "wb"))
        pickle.dump(file_feature_list, open("file_feature_list.data", "wb"))
else:
        print("Load all_feature_array(sift feature) file...")
        all_feature_array = pickle.load(open("all_feature_array.data", "rb"))
        file_feature_list = pickle.load(open("file_feature_list.data", "rb"))
print("Sift features array:")
print(all_feature_array)
```

-Phân cụm các đặc trưng

```
###KMeans clustering
kmeans = KMeans(n_clusters=vocabulary_len)
if not os.path.isfile("kmeans.model"):
        print("Computing KMeans...")
        kmeans = kmeans.fit(all_feature_array)
        joblib.dump(kmeans, "kmeans.model")
else:
        print("Load KMeans...")
        kmeans = joblib.load("kmeans.model")
#print(kmeans.predict([all_feature_array[0], all_feature_array[1], all_feature_array[2]]))
print("KMeans Centers:")
print(kmeans.cluster_centers_)
#
```

-Tính toán các vector tần số

```
###Computing bag of words
for file_list in file_feature_list:
        tmp = np.zeros((vocabulary_len), dtype=int)
        for predict in kmeans.predict(file_list[1]):
                tmp[predict] = tmp[predict] + 1
        bag_of_words.append((file_list[0], tmp))
#print(bag_of_words)
#----------------------------
```

-Xử lý ảnh đầu vào và tính toán đưa ra kết quả

```
#query_img = cv2.imread("./Dataset/0817.png", 0)
query_img = cv2.imread("sample1.jpg", 0)
kp, des = sift.detectAndCompute(query_img, None)
query_vector = np.zeros((vocabulary_len), dtype=int)
for predict in  kmeans.predict(des):
        query_vector[predict] = query_vector[predict] + 1

#----------------------------
angle_best = spatial.distance.cosine(query_vector, bag_of_words[0][1])
filename_result = ''
for item in bag_of_words[1:]:
        angle = spatial.distance.cosine(query_vector, item[1])
        print(angle, item[0])
        if angle < angle_best:
                angle_best = angle
                filename_result = item[0]
print(filename_result)
```

VI. Kết Quả Thu Được Sơ Bộ

- Cho ảnh query là ảnh trong tập dataset. Kết quả: cho kết quả chính xác 100%.
- Cho ảnh query là một phần cắt ra từ một file ảnh trong dataset. Kết quả: cho kết quả tương đối chính xác 70% phụ thuộc vào kích thước ảnh được cắt ra.
- Cho ảnh query là một ảnh ngoài dataset. Kết quả: cho kết quả không ổn định.