TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN 1: COLOR COMPRESSION

TOÁN ỨNG DỤNG VÀ THỐNG KÊ

Triệu Nhật Minh — 21127112

Ngày 16 tháng 7 năm 2023

Mục lục

1 Giới thiệu

2 Ý tưởng thực hiện

3 Hướng dẫn sử dụng

4 Mô tả

4.1 Nhóm hàm chính

4.1.1 get_labels

4.1.2 initialize_centroids

Input: Mảng 1 chiều các điểm ảnh (img_1d), số cụm (k_cluster) và kiểu khởi tạo (init_centroids)
Output: Mảng các centroids

Mô tả: Đầu tiên, ta cần đếm số màu phân biệt trong bức ảnh (sử dụng numpy.unique) và so sánh chúng với số màu cần "nén". Sử dụng hàm min để lấy giá trị nhỏ nhất giữa 2 giá trị để đảm bảo số màu cần "nén" không vượt quá số màu phân biệt trong bức ảnh. Sau đó, tuỳ thuộc vào kiểu khởi tạo centroids người dùng nhập vào (random hoặc in_pixels) thì hàm sẽ khởi tạo các centroids theo kiểu tương ứng:

random: Sử dụng numpy.random.randint để tạo ra một mảng các số nguyên ngẫu nhiên trong khoảng từ 0 đến 255 với số sample trả về bằng số cụm, mỗi cụm có số kênh màu giá trị (ở đây số kênh màu là img_1d.shape[1] do số kênh màu trong đa số các trường hợp test là 3 (tương ứng với hệ màu RGB), tuy nhiên có trường hợp số kênh màu trả về là 4. Do đó để đảm bảo tính tổng quát, ta sẽ sử dụng img_1d.shape[1]), đồng thời ép kiểu giá trị trả về là số nguyên không âm 8bit để đảm bảo tính chính xác của giá trị trả về.

in_pixels: Sử dụng numpy.random.choice đồng thời tận dụng mảng unique_img_1d lưu trữ các màu phân biệt trong bức ảnh để tạo ra mảng các màu ngẫu nhiên với số kết quả trả về là số cụm. Sau đó lấy ra phần tử tương ứng trong mảng unique_img_1d và gán chúng làm giá trị cho phần tử trong mảng centroids. Mỗi cụm có số kênh màu bằng với số kênh màu của phần tử trong mảng unique_img_1d. Trong lúc random xuất hiện một tham số *replace=False* nhằm đảm bảo các màu được chọn không có sự trùng lặp.

Nếu giá trị của tham số init_centroids không phải là "random" hoặc "in_pixels" thì hàm sẽ báo lỗi.

Và dù bằng bất kì cách random hợp lệ nào thì mảng centroid trả về đều được chuyển thành ndarray 2 chiều với số hàng (rows) bằng số cụm, số cột (cols) bằng số kênh màu của phần tử trong mảng img_1d (img_1d.shape[1]).

4.1.3 update_centroids

4.1.4 kmeans

4.2 Nhóm hàm bổ trợ

4.2.1 convert_1d_array

4.2.2 convert_2d_array

4.2.3 show_image

Input: Danh sách các ảnh được lưu từ phương thức Image.fromarray của thư viện PIL
Output: Không có

Mô tả: Để có thể hiển thị nhiều ảnh trên một figure, ở đây để đối chiếu ảnh trước và sau xử lý, ta sử dụng phương thức subplot. Đầu tiên, ta sẽ tạo ra một figure với số hàng bằng 1, số cột bằng số ảnh được truyền vào. Sau đó, ta sẽ duyệt qua từng ảnh trong danh sách ảnh được truyền vào và hiển thị chúng lên figure với vị trí tương ứng và hiển thị figure ra màn hình.

Dòng code *plt.figure(figsize=(20,10))* có tác dụng tạo figure hiển thị ra màn hình với kích thước 20x10 inches thay vì thông số mặc định 6.4x4.8 nhằm đảm bảo tính thẩm mỹ, không có tác động đến kết quả xử lý.

4.2.4 write_image

4.2.5 execute

5 Hình ảnh đầu ra

5.1 Hình ảnh gốc

5.2 Hình ảnh lúc sau

5.3 So sánh với scikit-learn

5.4 Nhận xét

6 Tài liệu tham khảo

- How to Code K-Means in Python (No Sklearn)
- numpy.random.randint
- numpy.random.choice
- matplotlib.pyplot.figure