Logo

Description automatically generated

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

-------🙡🙣-------

**BÁO CÁO ĐỒ ÁN CÁ NHÂN**

**TOÁN ỨNG DỤNG VÀ THỐNG KÊ CHO CÔNG NGHỆ THÔNG TIN**

**HỌ VÀ TÊN:**

21127635 - Nguyễn Khánh Anh Kiệt

**GIÁO VIÊN HƯỚNG DẪN:**

Ngô Đình Hy

Nguyễn Văn Quang Huy

Trần Hà Sơn

Nguyễn Đình Thúc

TP. Hồ Chí Minh – 2023

**Mục lục**

[**A.** **Ý tưởng thực hiện và mô tả các hàm** 3](#_Toc140754108)

[**1.** **Ý tưởng thực hiện** 3](#_Toc140754109)

[**2.** **Mô tả các hàm** 3](#_Toc140754110)

[**a.** **Hàm kmeans** 3](#_Toc140754111)

[**b.** **Hàm save\_image** 4](#_Toc140754112)

[**c.** **Hàm main** 4](#_Toc140754113)

[**B.** **Hình ảnh kết quả** 6](#_Toc140754114)

[**C.** **Nhận xét kết qủa** 7](#_Toc140754115)

[**D.** **Tài liệu tham khảo** 9](#_Toc140754116)

1. **Ý tưởng thực hiện và mô tả các hàm**
2. **Ý tưởng thực hiện**

Trong thuật toán k-Means mỗi cụm dữ liệu được đặc trưng bởi một tâm (centroid). tâm là điểm đại diện nhất cho một cụm và có giá trị bằng trung bình của toàn bộ các quan sát nằm trong cụm. Chúng ta sẽ dựa vào khoảng cách từ mỗi quan sát tới các tâm để xác định nhãn cho chúng trùng thuộc về tâm gần nhất. Ban đầu thuật toán sẽ khởi tạo ngẫu nhiên một số lượng xác định trước tâm cụm. Sau đó tiến hành xác định nhãn cho từng điểm dữ liệu và tiếp tục cập nhật lại tâm cụm. Thuật toán sẽ dừng cho tới khi toàn bộ các điểm dữ liệu được phân về đúng cụm hoặc số lượt cập nhật tâm chạm ngưỡng.

Các bước thực hiện:

1. Khởi tạo ngẫu nhiên k centroids
2. Gán nhãn cho từng pixels dựa vào khoảng cách tới cụm
3. Tinh toán lại tâm cho từng cụm theo trung bình
4. **Mô tả các hàm**
5. **Hàm kmeans**

Khởi tạo trung tâm: Hàm bắt đầu bằng cách khởi tạo centroids. Người dùng có thể chọn 1 trong 2 phương pháp để khởi tạo:

* 'random': Tạo các centroids với các giá trị RGB ngẫu nhiên trong khoảng từ 0 đến 255
* 'in\_pixels': Tạo các centroids với các giá trị ngẫu nhiên từ các điểm pixels trong img\_1d

Phần chính của hàm là một vòng lặp chạy đối ta max\_iter lần lặp (do người dùng chỉ định) hoặc cho đến khi đạt được sự hội tụ:

* Tính khoảng cách và gán labels: Tính khoảng cách Euclidean giữa từng điểm ảnh và tất cả centroids. Khoảng cách biểu thị mức độ giống hoặc khác nhau của một điểm dữ liệu đối với mỗi trọng tâm. Label của 1 điểm ảnh được đặt theo centroid gần nhất.
* Cập nhật centroids: sau khi gán labels đến tất cả điểm ảnh, trọng tâm mới sẽ được tính toán. Mỗi trọng tâm được update bằng cách lấy giá trị trung bình của tất cả các điểm ảnh dữ liệu thuộc về cụm tương ứng.
* Kiểm tra: Vào cuối mỗi lần lặp hàm sẽ kiểm tra xem centroid mới có giống với centroid cũ hay không. Nếu giống nhau điều đó có nghĩa là thuật toán đã hội tụ và vòng lặp bị ngắt dừng các lần lặp tiếp theo.

def kmeans(img\_1d, k\_clusters, max\_iter, init\_centroids='random'):

    # Initialize the centroids

    if init\_centroids == 'random':

        centroids = np.random.randint(0, 256, size=(k\_clusters, 3))

    elif init\_centroids == 'in\_pixels':

        centroids = img\_1d[np.random.choice(img\_1d.shape[0], k\_clusters, False)]

    else:

        raise ValueError("Invalid 'init\_centroids' option.")

    # Run the k-means algorithm

    for \_ in range(max\_iter):

        # Calculate the distances and labels

        temp = img\_1d[:, np.newaxis] - centroids

        distances = np.sqrt(np.sum(np.square(temp), axis=2))

        #distances = np.linalg.norm(temp, axis=-1)

        labels = np.argmin(distances, axis=1)

        # Update the centroids

        new\_centroids = np.array([np.mean(img\_1d[labels == i], axis=0) for i in range(k\_clusters)])

        # Check if the centroids have changed

        if np.all(centroids == new\_centroids):

            break

        # Update the centroids

        centroids = new\_centroids

    return centroids, labels

1. **Hàm save\_image**

Hàm dùng để chuyển đổi mảng numpy thành một hình ảnh và lưu nó ở định dạng nhất định với thư viện PIL (Python Imaging Library)

def save\_image(img\_array, output\_format='png'):

    # Convert the array into a PIL image and save it

    image = Image.fromarray(img\_array)

    file\_name = input("Enter the output file name (without extension): ")

    image.save(f"{file\_name}.{output\_format}")

1. **Hàm main**

Hàm để thực thi chương trình

* Đọc ảnh: Chương trình yêu cầu người dùng nhập tên ảnh để mở ảnh
* Chuyển ảnh đã đọc được thành mảng 1D array bằng cách sử dụng các hàm như ravel() và reshape()
* Chạy kmeans function: Yêu cầu người dùng nhập 2 thông số: số clusters và số lần tối đa vòng lặp chạy
* Chuyển đổi từ array thành ảnh: Chuyển từ array thành PIL image
* Hiển thị ảnh gốc và ảnh đã qua thuật toán K-means cho người dùng và hỏi người dùng sử dụng chức năng lưu ảnh.

def main():

    # Read the image

    filename = input("Enter the file name: ")

    img = Image.open(filename)

    # Convert the image into a numpy array

    img\_array = np.array(img)

    # Convert the image into a 1D array

    shape = img\_array.shape

    flat\_arr = img\_array.ravel()

    flat\_arr = flat\_arr.reshape(-1, 3)

    # Get the number of clusters and maximum number of iterations

    k = int(input("Enter the number of clusters: "))

    max\_iter = int(input("Enter the maximum number of iterations: "))

    # Run the k-means algorithm

    centroids, labels = kmeans(flat\_arr, k, max\_iter)

    print(centroids)

    # Reduce the image

    reduce\_image = centroids[labels]

    # Convert the reduced image into a 3D array

    reduce\_arr = reduce\_image.reshape(shape)

    # Convert the array into a PIL image

    reduce\_arr = reduce\_arr.astype(np.uint8)

    # Display the reduced image

    reduce\_img = Image.fromarray(reduce\_arr)

    # Display the original and reduced images

    fig, axes = plt.subplots(1, 2, figsize=(12, 6))

    # Display the first image on the first subplot

    axes[0].imshow(img)

    axes[0].axis('off')  # Remove axes for the first image

    axes[0].set\_title('Image 1')  # Set title for the first image

    # Display the second image on the second subplot

    axes[1].imshow(reduce\_img)

    axes[1].axis('off')  # Remove axes for the second image

    axes[1].set\_title('Image 2')  # Set title for the second image

    # Adjust the spacing between subplots to add space for descriptions

    plt.subplots\_adjust(bottom=0.2)

    # Add descriptions below the images

    axes[0].text(0.5, -0.15, 'Original Image', transform=axes[0].transAxes,

                ha='center', fontsize=12)

    axes[1].text(0.5, -0.15, 'After Colours compressed', transform=axes[1].transAxes,

                ha='center', fontsize=12)

    # Show the plot with both images and descriptions

    plt.show()

    if(input("Do you want to save the image? (y/n): ") == 'y'):

        output\_format = input("Enter the output format: ")

        save\_image(reduce\_arr, output\_format)

    else:

        print("Thank you for using this program")

1. **Hình ảnh kết quả**

Với K=3:

A comparison of a picture of a canal and a boat

Description automatically generated

Với K=5:

A comparison of a picture of a canal and a boat

Description automatically generated

Với K=7:

A comparison of a canal and a city

Description automatically generated

1. **Nhận xét kết qủa**

Các kết quả đều trả về đúng với số clusters cũng như giảm được chất lượng hình ảnh. Tuy nhiên theo thuật toán và sự lựa chọn centroids đầu tiên, chúng ta không thể luôn luôn có được số lượng k màu mong muốn cũng như nhiều clusters có khả năng trống và trả về kết quả sai. Điều này xãy ra vì nhiều centroids có vị trí quá xa so với các điểm ảnh. Vì vậy các centroids này trống.

A bowl of rice with meat and vegetables

Description automatically generatedA brown square with a white border

Description automatically generated

A computer screen shot of a computer

Description automatically generated

1. **Tài liệu tham khảo**

*Bài 4: K-means Clustering*. (n.d.). Retrieved from machinelearningcoban: https://machinelearningcoban.com/2017/01/01/kmeans/

*Calculate the Euclidean distance using NumPy*. (n.d.). Retrieved from Geeksforgeeks: https://www.geeksforgeeks.org/calculate-the-euclidean-distance-using-numpy/

*Convert Image To Matrix And Then To 1D Array*. (n.d.). Retrieved from stackoverflow: https://stackoverflow.com/questions/15612373/convert-image-png-to-matrix-and-then-to-1d-array

*How to Convert images to NumPy array?* (n.d.). Retrieved from Geeksforgeeks: https://www.geeksforgeeks.org/how-to-convert-images-to-numpy-array/

*K means Clustering - Introduction*. (n.d.). Retrieved from Geeksforgeeks: https://www.geeksforgeeks.org/k-means-clustering-introduction/