TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN, ĐHQG-TPHCM
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO

Môn: Toán ứng dụng và thống kê cho công nghệ thông tin

Chủ đề: Color Compression.

Họ tên: Kuo Yung Sheng

MSSV: 21127684

Lớp: 21CLC07

Thành phố Hồ Chí Minh, ngày 19 tháng 6 năm 202

1. Ý tưởng thực hiện.

1. Tạo ngẫu nhiên các điểm dữ liệu để làm điểm tâm(k).
2. Gán label cho từng điểm dữ liệu dựa trên điểm tâm (label sẽ được xét dựa trên khoảng cách từ điểm dữ liệu đến điểm tâm gần nhất).
3. Cập nhật lại điểm tâm dựa trên các điểm dữ liệu đã gán label (điểm tâm mới sẽ là tâm của các điểm có chung label).
4. Lặp lại quá trình 2,3 đến khi max_iterator (được gán sẵn) hoặc đến khi điểm tâm mới không thay đổi so với điểm tâm cũ.

2. Mô tả các hàm.

def get_random_centroids(dataSet, k): khởi tạo điểm tâm ngẫu nhiên từ điểm dữ liệu..
Input là số k cần khởi tạo, danh sách các điểm dữ liệu (dataSet).

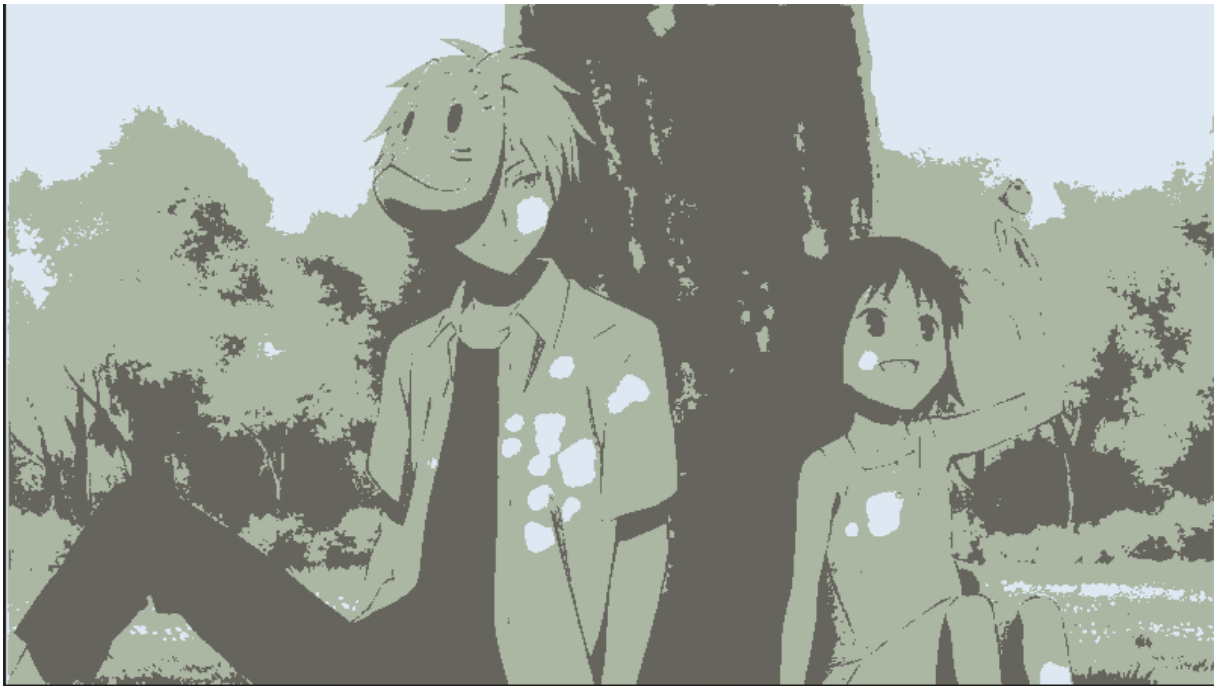
def get_labels(dataSet, centroids): cập nhật label cho điểm dữ liệu.
Input là danh sách các điểm dữ liệu (dataSet), và các điểm tâm đã được khởi tạo (centroids).

def get_centroids(img_1d, labels, k_clusters): cập nhật lại điểm tâm.
Input là danh sách các điểm dữ liệu (img_1d), nhãn của từng điểm dữ liệu (labels), số điểm tâm (k_clusters).

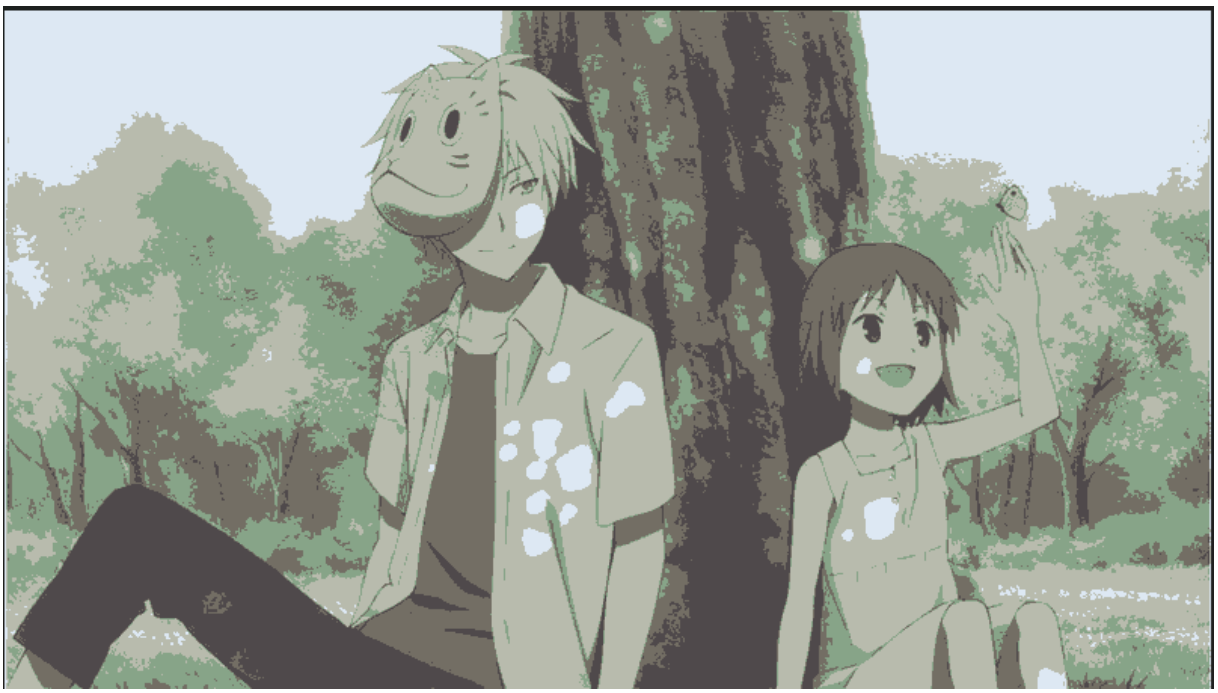
def kmeans(img_1d, k_clusters, max_iter=10, init_centroids='random'): thuật toán Kmeans trả về danh sách các điểm tâm(centroids), và nhãn của các điểm dữ liệu (label).
Input là danh sách các điểm dữ liệu (img_1d), số điểm cluster, số vòng lặp (max_iter).

3. Thử nghiệm

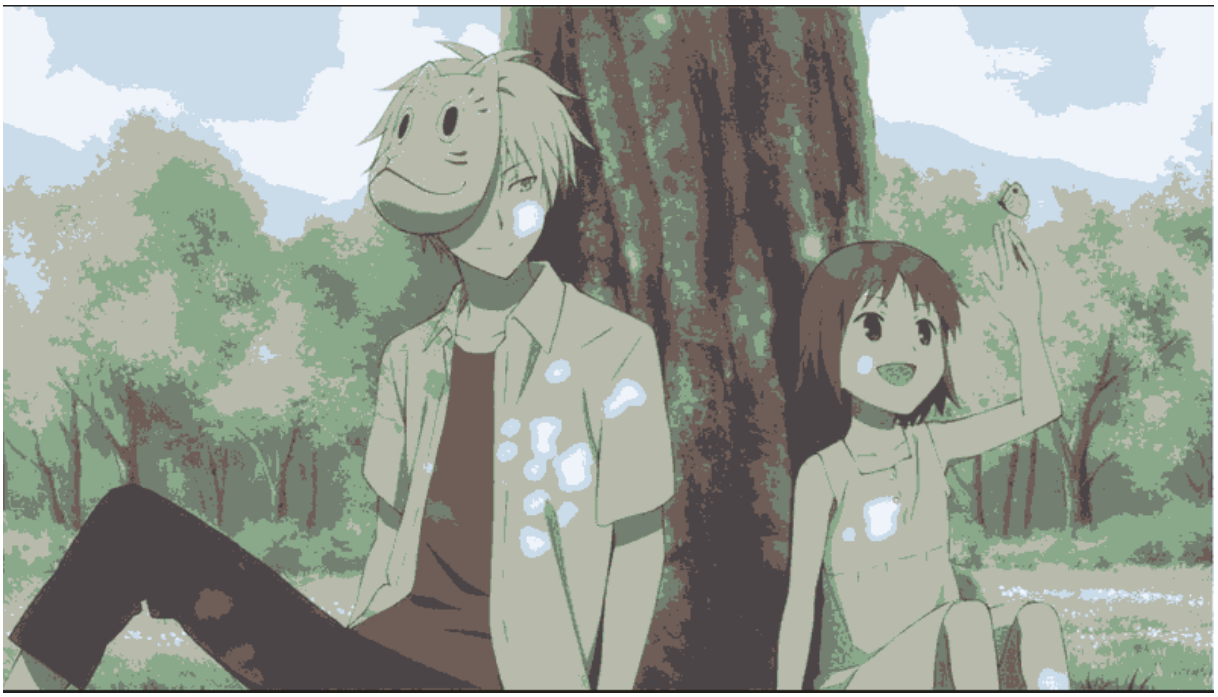
Hình ảnh kết quả với từng số lượng màu k = {3, 5, 7}



k=3



k=5



k=7

Nhận xét:

Với k càng lớn thì ảnh càng rõ nét.

Khi k nhỏ sẽ đỡ tốn chi phí lưu trữ, ảnh không rõ nét nhưng vẫn giữ được nội dung của ảnh.

Càng iterator càng lớn (số lần chạy) thì độ chính xác của thuật toán K-mean càng cao và có xu hướng hội tụ về một điểm.

4. Tài liệu tham khảo

https://phamdinhkhanh.github.io/deepai-book/ch_ml/KMeans.html