Nén ảnh với thuật toán phân cụm K-means

Họ tên: Đinh Duyên Bảo Duy
MSSV: 18520658
Lớp: CS23.K22.KHCL
Công việc: tìm hiểu về thuật toán K-means, viết code.

Họ tên: Nguyễn Hoàng Phúc
MSSV: 18521256
Lớp: CS23.K22.KHCL
Công việc: tìm hiểu về thuật toán K-means, viết báo cáo, làm powerpoint.

Họ tên: Huỳnh Ngọc Trân
MSSV: 18520385
Lớp: CS23.K22.KHCL
Công việc: tìm hiểu và tìm phương pháp cải tiến thuật toán K-means, viết code, viết báo cáo, thuyết trình.

***Tóm tắt* — Nén ảnh là một bài toán khá cơ bản trong xử lý ảnh nói riêng cũng như khoa học máy tính nói chung. Trong đồ án này, chúng tôi tập trung nghiên cứu một phương pháp nén ảnh dựa trên kỹ thuật phân cụm dữ liệu, đó là thuật toán phân cụm K-means (K- means clustering). Trong báo cáo này, chúng tôi trình bày và phân tích các nội dung liên quan đến cơ sở lý thuyết của thuật toán phân cụm K-means cũng như tiến hành thực nghiệm trên ảnh cụ thể. Bên cạnh đó, từ thuật toán truyền thống, chúng tôi cũng tìm hiểu phương pháp hỗ trợ để có thể áp dụng thuật toán hiệu quả hơn.**

I. GIỚI THIỆU

Nén hình ảnh là một loại nén dữ liệu được áp dụng cho hình ảnh kỹ thuật số, nhằm giảm chi phí cho việc lưu trữ hoặc truyền tải. Các thuật toán nén ảnh có thể tận dụng các đặc điểm của thị giác và các thuộc tính thống kê của dữ liệu hình ảnh để cho ra kết quả.

Có rất nhiều thuật toán mà ta có thể áp dụng để nén ảnh. Riêng trong báo cáo này, chúng tôi tập trung phân tích một thuật toán khá cơ bản, đó là phân cụm K-means (K-means clustering). Khác với các thuật toán nén không mất mát như JPEG, Huffman, Arimethic Coding,..., phân cụm K-means là một thuật toán nén có mất mát, tức là sẽ có một số chi tiết bị mất sau khi nén ảnh, và ta không thể xây dựng lại ảnh gốc một cách hoàn toàn sau khi đã nén. Phân cụm K-means hoạt động trên một tập dữ liệu với ý tưởng chính là phân dữ liệu thành K cụm khác nhau sao có các điểm dữ liệu trong cùng một cụm thì có tính chất giống nhau. Dựa trên ý tưởng đó, ta có thể áp dụng K-means để nén ảnh màu bằng cách phân cụm theo giá trị các pixels của ảnh.

II. THUẬT TOÁN PHÂN CỤM K-MEANS

1. Giới thiệu thuật toán:

K-means clustering (phân cụm K-means) là một kỹ thuật học không giám sát (unsupervised learning). Khi áp dụng K-means, chúng ta không biết được nhãn (label) của từng điểm dữ liệu. Mục đích là làm thế nào để phân dữ liệu thành các cụm (cluster) khác nhau sao cho dữ liệu trong cùng một cụm có tính chất giống nhau.

Về ý tưởng, đầu tiên, thuật toán chọn ngẫu nhiên K điểm dữ liệu, mỗi điểm này coi như biểu diễn đại diện cho một cụm, lúc này trong mỗi cụm thì mỗi điểm đó cũng là điểm trung tâm (center) của cụm đó. Các điểm dữ liệu còn lại được gán vào một cụm nào đó trong K cụm đã có sao cho tổng khoảng cách từ các điểm đó đến tâm của cụm là nhỏ nhất. Sau đó tính lại tâm của các cụm bằng cách tìm giá trị trung bình của các điểm dữ liệu ở mỗi cụm và lặp lại quá trình đó cho đến khi hàm tiêu chuẩn hội tụ. Hàm tiêu chuẩn hay được dùng nhất là hàm tiêu chuẩn sai số vuông.

2. Mô tả thuật toán

Giả sử có N điểm dữ liệu là $[\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_N}] \in \mathbb{R}^{d\times N}$ và K là số cụm $S = \{S_1, S_2, \dots, S_K\}$ mà chúng ta muốn phân chia $(K < N)$.Chúng ta cần tìm các điểm trung $\boldsymbol{m_1}, \boldsymbol{m_2}, \dots, \boldsymbol{m_K} \in \mathbb{R}^{d\times 1}$ và nhãn của mỗi điểm dữ liệu. Trong đó, $\boldsymbol{x_i}$, $\boldsymbol{m_i}$ là các vector.

Input: Một tập gồm N điểm dữ liệu, số nguyên K là số cụm mà ta muốn phân chia dữ liệu.

Output: Một tập dữ liệu gồm K cụm.

Các bước của thuật toán:

Bước 1: Chọn ngẫu nhiên K điểm bất Kỳ làm điểm trung tâm.

$$\mathbb{C}^{(0)} = \{\boldsymbol{m_1}^{(0)}, \boldsymbol{m_2}^{(0)}, \dots, \boldsymbol{m_k}^{(0)}\}$$

Bước 2: Phân các điểm dữ liệu vào cụm có điểm trung tâm gần nó nhất.

$$S_i^{(t)} = \{x_p : \| x_p - m_i^{(t)} \|^2 \leq \| x_p - m_p^{(t)} \|^2 \ \forall j, 1 \leq j \leq k$$

Nếu các cụm không thay đổi so với trước khi phân lại cụm thì ta dừng thuật toán.

Bước 3: Sau khi phân cụm, ta cập nhật lại điểm trung tâm của cụm bằng cách lấy trung bình cộng các điểm dữ liệu.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

Sau khi các điểm trung tâm được cập nhật, ta quay lại bước 2.

Độ phức tạp của thuật toán này là $O(tKN)$. Trong đó N là số điểm dữ liệu đầu vào, K là số cụm, t là số lần lặp. Thông thường $t, K << N$, nên thuật toán này có hiệu quả tương đối với các cơ sở dữ liệu lớn.

III. NÉN ẢNH VỚI K-MEANS

1. *Dữ liệu ảnh:*

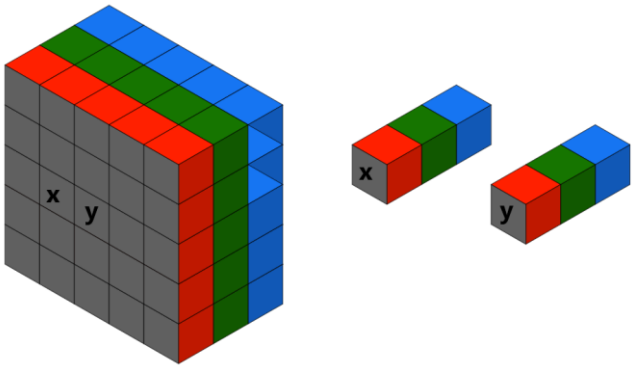
Khi được lưu trữ trong máy tính, một bức ảnh có thể được xem là một ma trận các điểm ảnh mang giá trị, hay còn được gọi là các pixels. Đối với ảnh màu, mỗi pixel gồm 3 bytes lưu trữ giá trị cường độ của 3 màu Red – Green – Blue (RGB). Có thể thấy, kích thước của một bức ảnh màu kỹ thuật số có thể rất lớn, do mỗi pixel cần tới 3 bytes (hay 3*8 = 24 bits) để lưu trữ. Do vậy, các bức ảnh này thường được lưu trữ dưới dạng ảnh đã nén với Kích thước nhỏ hơn (số bits ít hơn) và tốn ít dung lượng bộ nhớ hơn. Điều này có thể được thực hiện bằng kỹ thuật phân cụm K-means.

Để áp dụng thuật toán K-means, dữ liệu ảnh đầu vào cần phải qua một bước xử lý là vector hóa. Qua đó, mỗi pixel của ảnh sẽ được lưu dưới dạng vector, mỗi vector gồm 3 phần tử lưu lần lượt 3 giá trị cường độ của 3 màu RGB. Các vector này được xem là các điểm dữ liệu trong thuật toán phân cụm K-means.

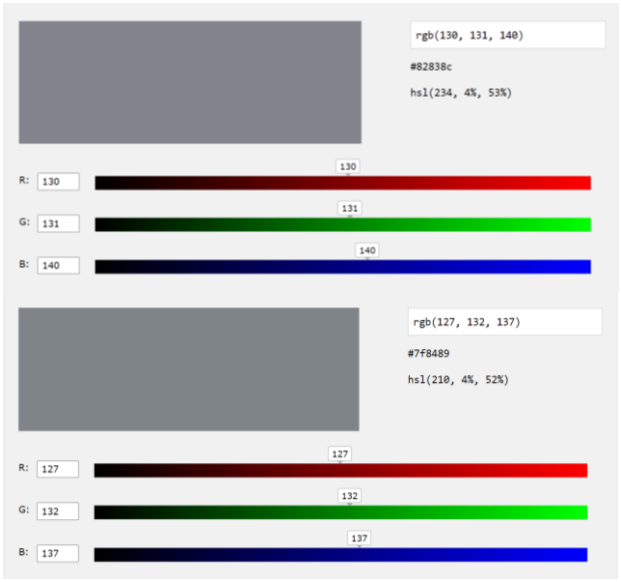
2. *Thuật toán phân cụm K-means trong bài toán nén ảnh:*

Như đã đề cập, mỗi pixel trong ảnh màu gồm 3 bytes lưu trữ giá trị cường độ của 3 màu RGB, giá trị này có độ lớn từ 0 đến 255 (đây cũng là giá trị của 3 phần tử trong vector màu). Theo đó, tổng số màu có thể được biểu diễn lên đến 256*256*256 = 16.777.216 màu. Tuy nhiên, trong thực tế, số màu mà chúng ta có thể nhận diện được bằng mắt thường khi nhìn vào một bức ảnh là ít hơn con số đó rất nhiều. Nói cách khác, có rất nhiều màu tuy có giá trị RGB Khác nhau nhưng đối với mắt thường thì chúng gần như là một. Lợi dụng điểm này, thuật toán K-means có thể phân các màu tương tự nhau vào một cụm.

Giả sử ta có cụm các pixels từ một ảnh màu đầu vào bất kỳ như hình trên. Cho hai pixels x và y có giá trị RGB lần lượt là (130, 131, 140) và (127, 132, 137), thì 2 pixels đó có màu như sau:



Hình 1.
Bên trái: Minh họa một cụm pixels trong một ảnh màu bất Kỳ.
Bên phải: Hai pixels ***x*** và ***y*** cạnh nhau từ cụm pixels.



Hình 2. Lần lượt 2 màu của 2 pixels ***x*** và ***y***.

Có thể thấy, sự khác biệt giữa hai màu trên là không đáng kể đối với mắt thường. Như vậy, thuật toán phân cụm K-means có thể gom hai màu này vào cùng một cụm và sử dụng một điểm trung tâm có màu tương tự để biểu diễn đại diện.

Vậy, thuật toán phân cụm K-means đối với nén ảnh được mô tả cụ thể hơn như sau: ta chọn K màu để biểu diễn những màu tương tự chúng trong ảnh, K màu này sẽ là các điểm trung tâm và ta sẽ thay thế mỗi giá trị của pixel bằng giá trị của điểm trung

tâm. Số lượng K màu mà ta dùng là ít hơn tổng số lượng màu gốc rất nhiều, nhờ đó ta có thể giảm kích thước dữ liệu.

3. Thí nghiệm và kết quả:

Việc hiện thực hóa thuật toán K-means để nén ạn là tương đối dễ dàng với sự hỗ trợ của thư viện SciKit-Learn. Khi sử dụng hàm KMeans sẵn có trong thư viện SciKit-Learn, có một số tham số đáng chú ý như sau:

- ***n_clusters***: số cụm mà ta muốn phân chia dữ liệu. Trong trường hợp này, chúng tôi thực hiện thí nghiệm với nhiều giá trị khác nhau.
- ***init***: phương pháp chọn các điểm trung tâm để bắt đầu. Theo định nghĩa của thuật toán K-means thì tham số có giá trị *'random'*. Tuy nhiên, chúng tôi chọn giá trị *'kmeans++'* do đây là cách chọn các điểm trung tâm "thông minh hơn".
- ***max_iter***: số lần lặp lại thuật toán nhiều nhất trong một lần chạy. Ở đây, chúng tôi cho giá trị của tham số theo mặc định là 300.
- ***n_init***: số lần chạy thuật toán K-means với các lần chọn các điểm trung tâm ban đầu khác nhau. Ở đây, chúng tôi cho giá trị của tham số theo mặc định là 10.

Chúng tôi thực hiện thí nghiệm trên ảnh sau:

Một số kết quả với các giá trị K khác nhau:

Hình 4. Ảnh kết quả với K = 2

Hình 5. Ảnh kết quả với K = 15

Hình 6. Ảnh kết quả với K = 32

Hình 7. Ảnh kết quả với K = 64

Đối với mỗi lần thí nghiệm, chúng tôi cũng ghi nhận lại hệ số nén (compression ratio) được tính toán theo công thức:

$$r = \frac{compressed\ size}{uncompressed\ size}$$

(kích thước file ảnh sau khi nén / kích thước file ảnh gốc)

Cụ thể, với ảnh gốc có kích thước 343KB, ta có các giá trị hệ số nén ở từng trường hợp như sau:

Số cụm K	Kích thước sau khi nén	Hệ số nén r
2	7KB	0.02
5	24KB	0.07
8	37KB	0.11
12	54KB	0.16
15	64KB	0.19
18	72KB	0.22
20	76KB	0.23
25	93KB	0.28
32	112KB	0.34
64	168KB	0.50

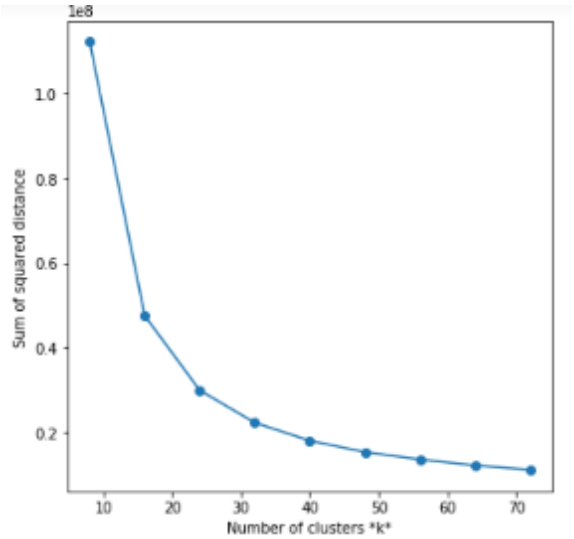
Ngoài ảnh trên, chúng tôi cũng đã thử nghiệm trên nhiều ảnh khác. Trong tất cả trường hợp, thuật toán phân cụm K-means đều thể hiện được ưu điểm là có khả năng giảm kích thước ảnh đi rất nhiều lần.

IV. TỐI ƯU HÓA VÀ HƯỚNG PHÁT TRIỂN

Bên cạnh điểm mạnh là khả năng giảm kích thước ảnh một cách hiệu quả, thuật toán phân cụm K-means cũng có một vấn đề quan trọng cần lưu ý: chọn số cụm K sao cho phù hợp. Nếu ta chọn K quá nhỏ, ảnh sau khi nén sẽ có chất lượng kém và xấu hơn ảnh gốc rất nhiều. Tuy nhiên, nếu chọn K quá lớn thì hệ số nén sẽ tăng, đồng thời, thời gian thực hiện nén cũng lâu hơn.

Để giải quyết vấn đề chọn ra số K tối ưu nhất, chúng tôi đã tiến hành nghiên cứu và tìm được *Elbow method* (tạm dịch: phương pháp gấp khúc). Phương pháp Elbow là một heuristic trong kỹ thuật phân cụm dữ liệu, dùng để chọn số cụm dữ liệu phù hợp. Trong kỹ thuật phân cụm K-means, khi áp dụng phương pháp Elbow, ta vẽ đồ thị biểu diễn *tổng bình phương độ lỗi* (error sum of squares – sse) theo số cụm K. Tổng bình phương độ lỗi là tổng bình phương khoảng cách giữa các điểm dữ liệu tới các điểm trung tâm của các cụm mà chúng được phân vào. Sau khi vẽ được đồ thị, ta chọn giá trị K tại điểm gấp khúc của đường biểu diễn.

Trong trường hợp nén ảnh đã nêu ở phần thí nghiệm trên, với các giá trị $K \in \{8, 16, 24, 32, 40, 48, 56, 64, 72\}$ ta có thể vẽ được đồ thị theo phương pháp Elbow như sau:



Hình 8. Đồ thị biểu diễn tổng độ lỗi bình phương theo số cụm K

Theo đồ thị, ta có thể chọn được K tại điểm gấp khúc với giá trị 32 hoặc 40. Đó là giá trị K có thể xem như phù hợp: không quá lớn để đảm bảo tốc độ và hệ số nén, cũng như không quá nhỏ để đảm bảo chất lượng ảnh.

Hình 9. Ảnh khi nén với K = 40 (được xem là K phù hợp)

Tuy phương pháp Elbow được xem là có thể chọn ra số K tối ưu cho thuật toán K-means, phương pháp này vẫn còn một số bất cập. Đầu tiên, bản chất của phương pháp là thí nghiệm thuật toán với nhiều giá trị K để chọn ra giá trị tối ưu, vì thế, thời gian tính toán là tương đối lớn. Ngoài ra, trong một số trường hợp, phương pháp Elbow không đạt được hiệu quả mà ta mong muốn: đường biểu diễn không thể hiện một điểm gấp khúc rõ ràng nào mà chỉ giảm một cách từ từ, không có tác dụng chỉ ra giá trị tối ưu.

Vì thế, trong tương lai, chúng tôi mong có thể tìm được cách giải quyết vấn đề chọn K một cách tối ưu hơn nữa để có thể áp dụng thuật toán phân cụm K-means trong bài toán nén ảnh một cách hiệu quả nhất.

V. KẾT LUẬN

Phân cụm K-means là một thuật toán khá cơ bản trong kỹ thuật phân cụm dữ liệu. Ưu điểm của thuật toán là đơn giản, dễ hiểu, dễ cài đặt, cũng như có thể áp dụng cho nhiều bài toán khác nhau. Riêng đối với bài toán nén ảnh, thuật toán K-means thể hiện độ hiệu quả cao với khả năng giảm kích thước dữ liệu đi rất nhiều lần. Tuy nhiên, thuật toán còn một số nhược điểm như: hiệu quả phụ thuộc vào việc chọn số cụm K (phải xác định trước), chi phí cho việc thực hiện vòng lặp tính toán khoảng cách lớn khi số cụm K và dữ liệu phân cụm lớn, và tùy vào các center ban đầu mà thuật toán có thể có tốc độ hội tụ rất chậm.

Có thể thấy, việc chọn số cụm K phù hợp cho thuật toán là một vấn đề quan trọng. Phương pháp Elbow được xem là một cách tiếp cận để giải quyết vấn đề này, tuy nhiên phương pháp này cũng còn nhiều bất cập.

Nhìn chung, việc áp dụng thuật toán phân cụm K-means vẫn còn nhiều hạn chế. Tuy nhiên, đây vẫn là một thuật toán cơ bản và quan trọng, là nền tảng cho nhiều kỹ thuật phức tạp hơn

TÀI LIỆU THAM KHẢO

[1] Aurlien Gron. 2017. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (1st. ed.). O'Reilly Media, Inc.
[2] Vũ Hữu Tiệp. 2018. Machine Learning cơ bản. Nxb. Khoa học và Kỹ thuật