

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
VIỆN TRÍ TUỆ NHÂN TẠO



BÁO CÁO MÔN HỌC KỸ THUẬT VÀ CÔNG NGHỆ DỮ LIỆU LỚN
ĐỀ TÀI: THUẬT TOÁN KMEAN & LẬP TRÌNH MAPREDUCE HÓA
KMEAN TRONG PHẦN CỤM ẢNH

GVHD: TS. Trần Hồng Việt
CN. Đỗ Thu Uyên

Nhóm sinh viên thực hiện:

1. Nguyễn Huy Hoàng
2. Nguyễn Phan Hiên
3. Trần Văn Dy
4. Bùi Thê Long

Năm 2024

MỞ ĐẦU

Trước đây, chúng ta mới chỉ biết đến dữ liệu có cấu trúc (structure data), ngày nay, với sự kết hợp của dữ liệu và internet, đã xuất hiện một dạng khác của dữ liệu - Big data (dịch là “dữ liệu lớn”). Dữ liệu này có thể từ các nguồn như: hồ sơ hành chính, giao dịch điện tử, dòng trạng thái (status), chia sẻ hình ảnh, bình luận, tin nhắn... của chính chúng ta, nói cách khác chúng là dữ liệu được sản sinh qua quá trình chia sẻ thông tin trực tuyến liên tục của người sử dụng.

Tuy nhiên, việc xử lý Big Data đặt ra những thách thức không nhỏ cho các hệ thống và công nghệ truyền thống. Để giải quyết vấn đề này, các framework tính toán phân tán như Hadoop đã ra đời. Hadoop, với mô hình lập trình MapReduce, cung cấp khả năng xử lý song song và phân tán, cho phép xử lý hiệu quả các tập dữ liệu lớn.

Trong lĩnh vực thị giác máy tính, phân cụm ảnh là một bài toán quan trọng với nhiều ứng dụng thực tế. Phân cụm ảnh giúp nhóm các ảnh tương tự lại với nhau, hỗ trợ cho các tác vụ như nén ảnh, tổ chức và truy xuất ảnh, nhận dạng đối tượng, và phân tích hình ảnh y tế. Thuật toán K-Means là một trong những thuật toán phân cụm phổ biến nhất, được biết đến với sự đơn giản và hiệu quả.

Nhận thấy tiềm năng của việc kết hợp K-Means với MapReduce trong việc phân cụm ảnh, nhóm chúng tôi đã lựa chọn đề tài: **"Thuật toán K-Means & Lập trình MapReduce hóa K-Means trong phân cụm ảnh"** cho báo cáo môn học Kỹ thuật và Công nghệ dữ liệu lớn. Báo cáo này trình bày chi tiết về thuật toán K-Means, mô hình MapReduce, cách thức tích hợp K-Means với MapReduce, và ứng dụng cụ thể trong phân cụm ảnh. Qua đó, chúng tôi hy vọng mang đến một giải pháp hiệu quả để xử lý các tập dữ liệu ảnh lớn, đồng thời làm rõ hơn tiềm năng của việc ứng dụng các công nghệ Big Data trong lĩnh vực thị giác máy tính.

MỤC LỤC	
CHƯƠNG 1: TỔNG QUAN VỀ DỮ LIỆU LỚN	4
1.1 Định nghĩa.....	4
1.2 Đặc trưng cơ bản của dữ liệu lớn.....	5
1.3 Tổng quan về Hadoop.....	6
1.4 Tổng quan về MapReduce.....	7
CHƯƠNG 2: PHÂN CỤM ẢNH BẰNG THUẬT TOÁN KMEANS	9
2.1 Giới thiệu thuật toán K-Means	9
2.2 Triển khai thuật toán phân cụm ảnh.....	9
2.3 Ví dụ minh họa - bài toán nén ảnh.....	10
CHƯƠNG 3: ỨNG DỤNG MAPREDUCE TRONG PHÂN CỤM ẢNH	13
3.1 Ý tưởng tích hợp K-Means với MapReduce.....	13
3.2 Mô hình và thuật toán chi tiết	15
3.3 Áp dụng trên tập dữ liệu ảnh	17
3.4 Cải tiến K-Means	17
3.5 Demo chương trình cài đặt	17
CHƯƠNG 4: THUẬT TOÁN K-MEANS CẢI TIẾN VỚI ARTIFICIAL BEE COLONY.....	23
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	24
5.1 Kết luận.....	26
5.2 Hướng phát triển	27
TÀI LIỆU THAM KHẢO	28

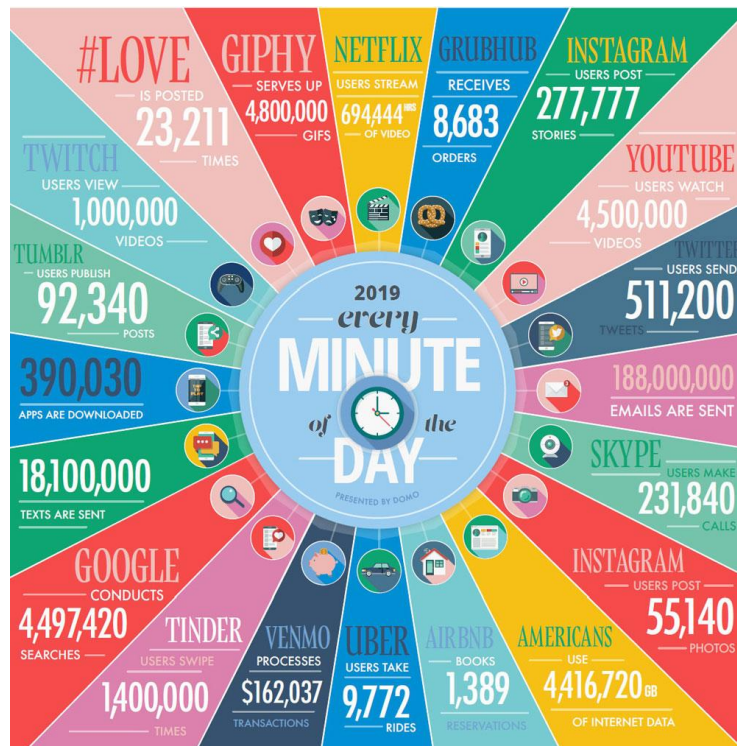
CHƯƠNG 1: TỔNG QUAN VỀ DỮ LIỆU LỚN

1.1 Định nghĩa.

- Theo wikipedia: Dữ liệu lớn là một thuật ngữ chỉ bộ dữ liệu lớn hoặc phức tạp mà các phương pháp truyền thống không đủ các ứng dụng để xử lý dữ liệu này.
- Theo Gartner : Dữ liệu lớn là những nguồn thông tin có đặc điểm chung khối lượng lớn, tốc độ nhanh và dữ liệu định dạng dưới nhiều hình thức khác nhau, do đó muốn khai thác được phải đòi hỏi phải có hình thức mới để đưa ra quyết định khám phá và tối ưu hóa quy trình.

Dữ liệu đến từ rất nhiều nguồn khác nhau, bao gồm:

- Hoạt động của con người trên Internet: Mạng xã hội, email, tìm kiếm web, thương mại điện tử, ...
- Thiết bị di động: Dữ liệu vị trí, thông tin liên lạc, ứng dụng di động, ...
- Internet of Things (IoT): Cảm biến, thiết bị thông minh, máy móc kết nối mạng, ...
- Doanh nghiệp: Dữ liệu giao dịch, dữ liệu khách hàng, dữ liệu vận hành, ...
- Khoa học: Dữ liệu nghiên cứu, thí nghiệm, quan sát, ...
- Y tế: Hồ sơ bệnh án điện tử, thiết bị y tế, ...



Hình 1.1 Minh họa nguồn gốc của dữ liệu.

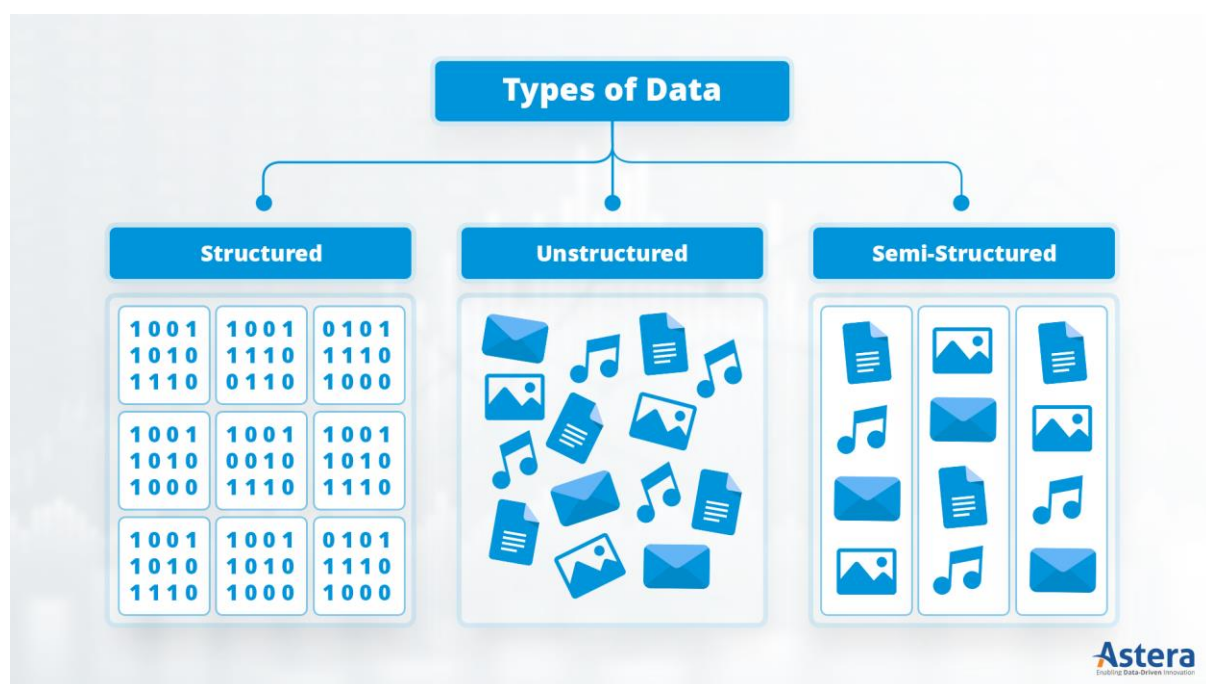
Một số lợi ích có thể mang lại như: Cắt giảm chi phí, tiết kiệm thời gian và giúp tối ưu hóa sản phẩm, hỗ trợ con người đưa ra những quyết định đúng và hợp lý hơn.

1.2 Đặc trưng cơ bản của dữ liệu lớn.

(1) *Volume (Khối lượng)*: Đây là đặc trưng dễ nhận thấy nhất của Big Data. Khối lượng dữ liệu được tạo ra và lưu trữ ngày càng tăng theo cấp số nhân, từ megabyte, gigabyte, terabyte đến petabyte, exabyte, zettabyte và yottabyte. Ví dụ, Facebook xử lý hơn 500 terabyte dữ liệu mỗi ngày (theo một báo cáo năm 2012, con số hiện tại chắc chắn lớn hơn rất nhiều), bao gồm hình ảnh, video, bình luận và các tương tác khác.

(2) *Velocity (Tốc độ)*: Dữ liệu lớn không chỉ lớn về kích thước mà còn được tạo ra với tốc độ cực kỳ nhanh, đòi hỏi khả năng xử lý gần như tức thời (real-time). Ví dụ, các giao dịch chứng khoán, dữ liệu từ cảm biến IoT, lượt click chuột trên website, các bài đăng trên mạng xã hội đều được tạo ra liên tục và cần được phân tích ngay lập tức để đưa ra quyết định kịp thời.

(3) *Variety (Đa dạng)*: Dữ liệu lớn bao gồm nhiều loại dữ liệu khác nhau, từ dữ liệu có cấu trúc (structured data) như cơ sở dữ liệu quan hệ (bảng, cột), dữ liệu bán cấu trúc (semi-structured data) như JSON, XML đến dữ liệu phi cấu trúc (unstructured data) như văn bản, hình ảnh, âm thanh, video.



Hình 1.2 Phân loại dữ liệu: Có cấu trúc, Bán cấu trúc và Phi cấu trúc

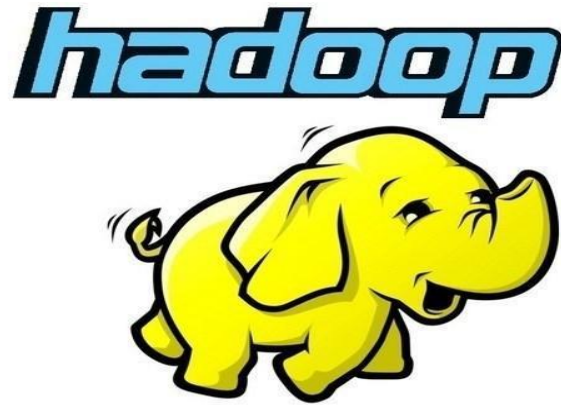
(4) *Veracity (Tính xác thực)*: Tính xác thực đề cập đến độ tin cậy và chính xác của dữ liệu. Với khối lượng dữ liệu khổng lồ và đa dạng, việc đảm bảo chất lượng và tính toàn vẹn của dữ liệu là một thách thức lớn. Dữ liệu nhiễu, không chính xác hoặc không đầy đủ có thể dẫn đến những phân tích sai lệch và quyết định sai lầm.

(5) *Value (Giá trị)*: Đây là đặc trưng quan trọng nhất của Big Data. Mục đích cuối cùng của việc thu thập và xử lý dữ liệu lớn là để trích xuất ra những thông tin hữu ích,

có giá trị, từ đó hỗ trợ việc ra quyết định, tối ưu hóa quy trình, tạo ra sản phẩm, dịch vụ mới hoặc mang lại lợi thế cạnh tranh.

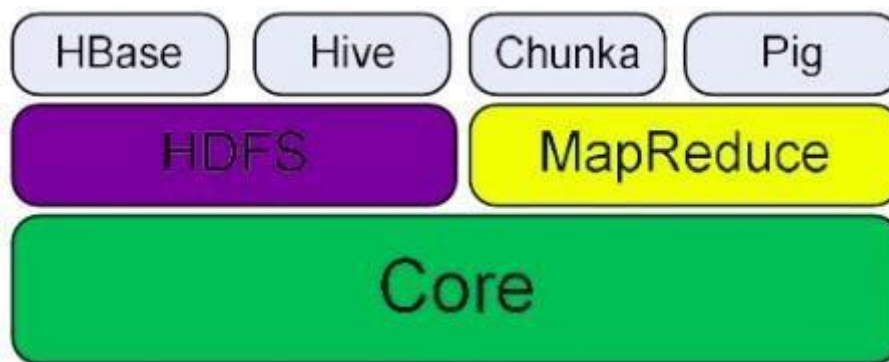
1.3 Tổng quan về Hadoop.

Theo apache hadoop: Apache Hadoop là một framework dùng để chạy những ứng dụng trên 1 cluster lớn được xây dựng trên những phần cứng thông thường.



Hình 1.3 Biểu tượng của Hadoop

Các thành phần của hadoop: Core, MapReduce engine, HDFS, HBase, Hive, Pig, Chukwa,.. Tuy nhiên *tập chung* vào 2 thành phần quan trọng nhất: HDFS và MapReduce.



Hình 1.4 Thành phần của Hadoop

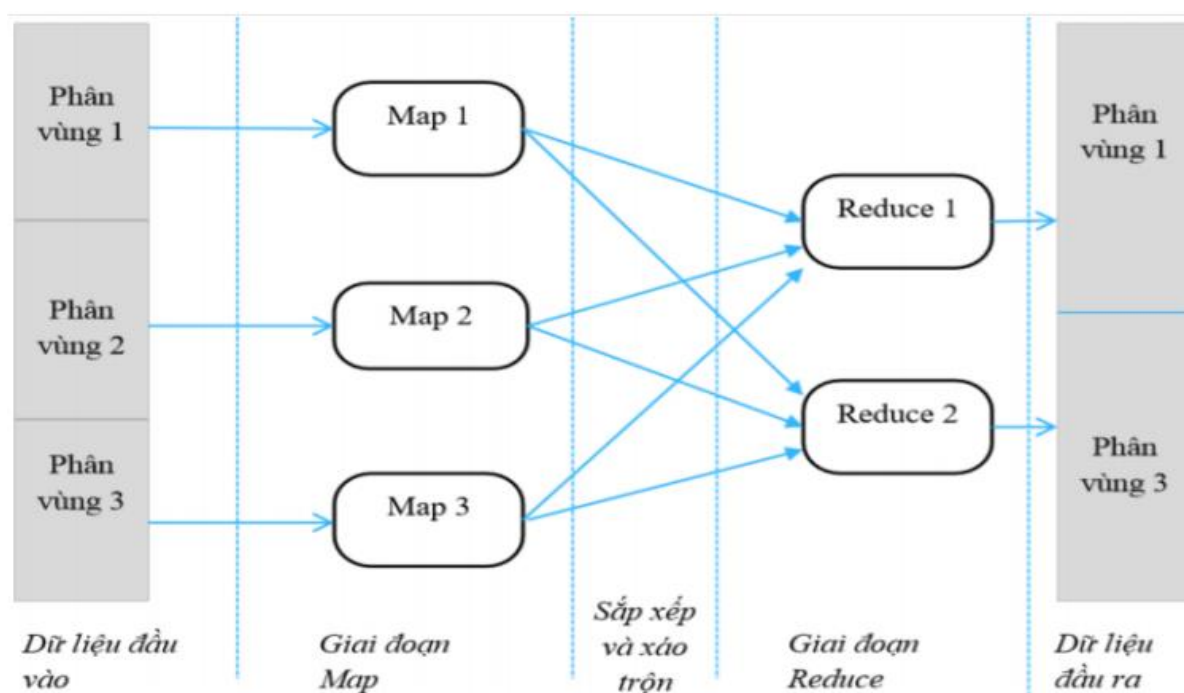
Hadoop hiện thực mô hình Map/Reduce, đây là mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau, và các phần này sẽ được chạy song song trên nhiều node khác nhau.

Thêm vào đó, Hadoop cung cấp 1 hệ thống file phân tán (HDFS) cho phép lưu trữ dữ liệu lên trên nhiều node. Cả Map/Reduce và HDFS đều được thiết kế sao cho framework sẽ tự động quản lý được các lỗi, các hư hỏng về phần cứng của các node.

=> *Kết luận:* Là một framework cho phép phát triển các ứng dụng phân tán. Viết bằng java

1.4 Tổng quan về MapReduce.

- *Định nghĩa:* Theo Google, MapReduce là mô hình dùng cho xử lý tính toán song song và phân tán trên hệ thống phân tán.
- Ứng dụng của MapReducer:
- Dữ liệu cần xử lý kích thước lớn.
- Các ứng dụng thực hiện xử lý, phân tích dữ liệu, thời gian xử lý đáng kể, có thể tính bằng phút, giờ, ...
- Thực thi mô hình MapReduce:



Hình 1.5 Thực thi mô hình Mapreduce

B1: Phân rã từ nghiệp vụ chính (do người dùng muốn thể hiện) thành các công việc con để chia từng công việc con này về các máy tính trong hệ thống thực hiện xử lý một cách song song.

B2: Thu thập lại các kết quả

- **Giai đoạn Map:** Hàm Map nhận đầu vào là các cặp khóa/giá trị (key/value) và xử lý chúng để tạo ra các cặp khóa/giá trị trung gian. Các cặp khóa/giá trị trung gian này được sắp xếp và phân nhóm theo khóa.
- **Giai đoạn Reduce:** Hàm Reduce nhận đầu vào là các khóa trung gian và danh sách các giá trị tương ứng với khóa đó. Hàm Reduce thực hiện tổng hợp, tính toán trên các giá trị này để tạo ra kết quả cuối cùng.

Ví dụ đơn giản về MapReduce: Đếm số lần xuất hiện của các từ trong một tập tài liệu.

- **Input:** Tập tài liệu lớn.
- **Map:**
 - Đầu vào: Một dòng văn bản.
 - Xử lý: Tách dòng văn bản thành các từ riêng lẻ.
 - Đầu ra: Các cặp <từ, 1> (ví dụ: <hello, 1>, <world, 1>, <hello, 1>).
- **Reduce:**
 - Đầu vào: <từ, danh sách các số 1> (ví dụ: <hello, [1, 1, 1]>).
 - Xử lý: Tính tổng các số 1 trong danh sách.
 - Đầu ra: <từ, tổng số lần xuất hiện> (ví dụ: <hello, 3>).

5. Lợi ích của Big Data

Xử lý và phân tích Big Data mang lại nhiều lợi ích to lớn cho các tổ chức và cá nhân, bao gồm:

- **Ra quyết định tốt hơn:** Phân tích dữ liệu lớn giúp hiểu rõ hơn về khách hàng, thị trường, xu hướng, từ đó đưa ra các quyết định kinh doanh chính xác và kịp thời hơn.
- **Tối ưu hóa hoạt động:** Big Data giúp xác định các điểm nghẽn, cải thiện hiệu quả quy trình, giảm chi phí và tăng năng suất.
- **Phát triển sản phẩm và dịch vụ mới:** Hiểu rõ nhu cầu và hành vi của khách hàng từ dữ liệu lớn giúp tạo ra các sản phẩm và dịch vụ mới đáp ứng tốt hơn mong đợi của khách hàng.
- **Cá nhân hóa trải nghiệm khách hàng:** Big Data cho phép các doanh nghiệp cá nhân hóa các sản phẩm, dịch vụ và tương tác với khách hàng, từ đó nâng cao sự hài lòng và lòng trung thành của khách hàng.
- **Dự đoán xu hướng và rủi ro:** Phân tích dữ liệu lớn giúp dự đoán các xu hướng tương lai, phát hiện sớm các rủi ro tiềm ẩn và đưa ra các biện pháp phòng ngừa kịp thời.

CHƯƠNG 2: PHÂN CỤM ẢNH BẰNG THUẬT TOÁN KMEANS

2.1 Giới thiệu thuật toán K-Means

- Khái niệm: Thuật toán K-Means là một phương pháp học không giám sát phổ biến trong phân cụm dữ liệu. K-Means phân chia dữ liệu thành k cụm dựa trên khoảng cách giữa các điểm dữ liệu và tâm cụm (centroids). Ứng dụng của K-Means rất đa dạng, bao gồm: phân cụm ảnh, nén ảnh, phân tích khách hàng,...
- Nguyên lý hoạt động:
- Bước 1: Khởi tạo: Chọn ngẫu nhiên k tâm cụm từ tập dữ liệu.
- Bước 2: Phân cụm: Gán mỗi điểm dữ liệu vào cụm có tâm gần nhất (thường tính bằng khoảng cách Euclidean).
- Bước 3: Cập nhật tâm cụm: Tính toán trung bình của tất cả các điểm trong cùng một cụm và cập nhật tâm cụm.
- Bước 4: Lặp lại cho đến khi thay đổi tâm cụm không đáng kể (hội tụ)
- Ưu điểm khi sử dụng MapReduce: Khả năng xử lý dữ liệu lớn, phân phối tài nguyên hiệu quả và đảm bảo tính mở rộng

2.2 Triển khai thuật toán phân cụm ảnh

- Input dữ liệu:
 - Ảnh đầu vào: Ảnh cần phân cụm được biểu diễn dưới dạng ma trận pixel hoặc vector đặc trưng (feature vector).
 - Số cụm (k): Xác định số cụm mà bạn muốn chia ảnh (ví dụ, k=3 cho phân cụm thành 3 vùng).
- Output:
 - Ảnh phân cụm, trong đó mỗi pixel được gán vào một cụm cụ thể.
- Gán tâm cụm cho mỗi điểm dữ liệu
 - Mỗi pixel của ảnh hoặc vector đặc trưng được coi là một điểm dữ liệu.
 - Tính khoảng cách từ mỗi điểm dữ liệu đến các tâm cụm.
 - Gán điểm dữ liệu vào cụm gần nhất trực tiếp vào bộ nhớ
- Cập nhật tâm cụm
 - Với mỗi cụm, tính trung bình tất cả các tọa độ điểm dữ liệu để cập nhật tâm cụm mới.
 - Gửi tâm cụm mới này đến vòng lặp tiếp theo.
- Lặp lại:
 - Lặp lại quá trình K-Means cho đến khi tâm cụm không thay đổi (hội tụ) hoặc đạt số vòng lặp tối đa.

2.3 Ví dụ minh họa - bài toán nén ảnh

Mục tiêu: Giảm số lượng màu trong ảnh mà vẫn giữ được chất lượng thị giác tốt

Input dữ liệu:

- Ảnh đầu vào: Được biểu diễn dưới dạng ma trận (N x M), với mỗi pixel là một vector [R,G,B]. Ví dụ:
Ảnh gốc

$$\begin{bmatrix} [255, 0, 0] & [254, 1, 1] & [0, 255, 0] & [0, 254, 1] \\ [0, 0, 255] & [1, 1, 254] & [255, 1, 1] & [1, 255, 1] \\ [0, 1, 254] & [128, 128, 128] & [130, 130, 130] & [127, 127, 127] \\ [0, 0, 0] & [255, 255, 255] & [250, 250, 250] & [5, 5, 5] \end{bmatrix}$$

- Số cụm (k): Ví dụ là 3

Ví dụ: Với k = 3, khởi tạo tâm cụm:

- C1=[255,0,0] (đỏ).
- C2=[0,255,0] (xanh lá).
- C3=[0,0,255] (xanh dương).

Giai đoạn gán tâm cụm:

- Hàm tính khoảng cách Euclid

```
def compute_distance(a, b):  
    return np.sqrt(np.sum((a - b) ** 2))
```

- Hàm gán tâm cụm

```
def update_centroids(X, clusters, k):  
    new_centroids = []  
    for i in range(k):  
        cluster_points = X[clusters == i]  
        if cluster_points.size == 0: # Xử lý cụm rỗng  
            new_centroids.append(np.random.rand(X.shape[1]))  
        else:  
            new_centroids.append(np.mean(cluster_points, axis=0))  
    return np.array(new_centroids)
```

Cập nhật tâm cụm

```
def update_centroids(X, clusters, k):
    new_centroids = []
    for i in range(k):
        cluster_points = X[clusters == i]
        if cluster_points.size == 0: # Xử lý cụm rỗng
            new_centroids.append(np.random.rand(X.shape[1]))
        else:
            new_centroids.append(np.mean(cluster_points, axis=0))
    return np.array(new_centroids)
```

Thuật toán K-Means tổng thể

```
def k_means(X, k, max_iters=100):
    centroids = initialize_centroids(X, k)
    for _ in range(max_iters):
        clusters = assign_clusters(X, centroids)
        new_centroids = update_centroids(X, clusters, k)
        if np.all(centroids == new_centroids): # Kiểm tra hội tụ
            break
        centroids = new_centroids
    return clusters, centroids
```

Trong thực tế, không cần thiết new_centroids phải trùng với centroids mà khoảng cách tâm cụm mới và cũ có thể bé hơn một hyper parameter alpha là đã có thể coi là hội tụ.

Output:

- Tâm cụm cuối cùng:
Sau khi hội tụ, ảnh sẽ có 3 màu chính (mỗi màu đại diện cho một cụm).

[254.67, 0.67, 0.67] [89.6, 165.9, 89.7] [0.33, 0.67, 254.33]

- Ảnh sau phân cụm: Mỗi pixel trong ảnh gốc được thay thế bằng màu đại diện của cụm tương ứng.

Ảnh nén:

$$\begin{bmatrix} [254, 0, 0] & [254, 0, 0] & [89, 165, 89] & [89, 165, 89] \\ [0, 0, 254] & [0, 0, 254] & [254, 0, 0] & [89, 165, 89] \\ [0, 0, 254] & [89, 165, 89] & [89, 165, 89] & [89, 165, 89] \\ [89, 165, 89] & [89, 165, 89] & [89, 165, 89] & [89, 165, 89] \end{bmatrix}$$

(Các số đã được làm tròn xuống)

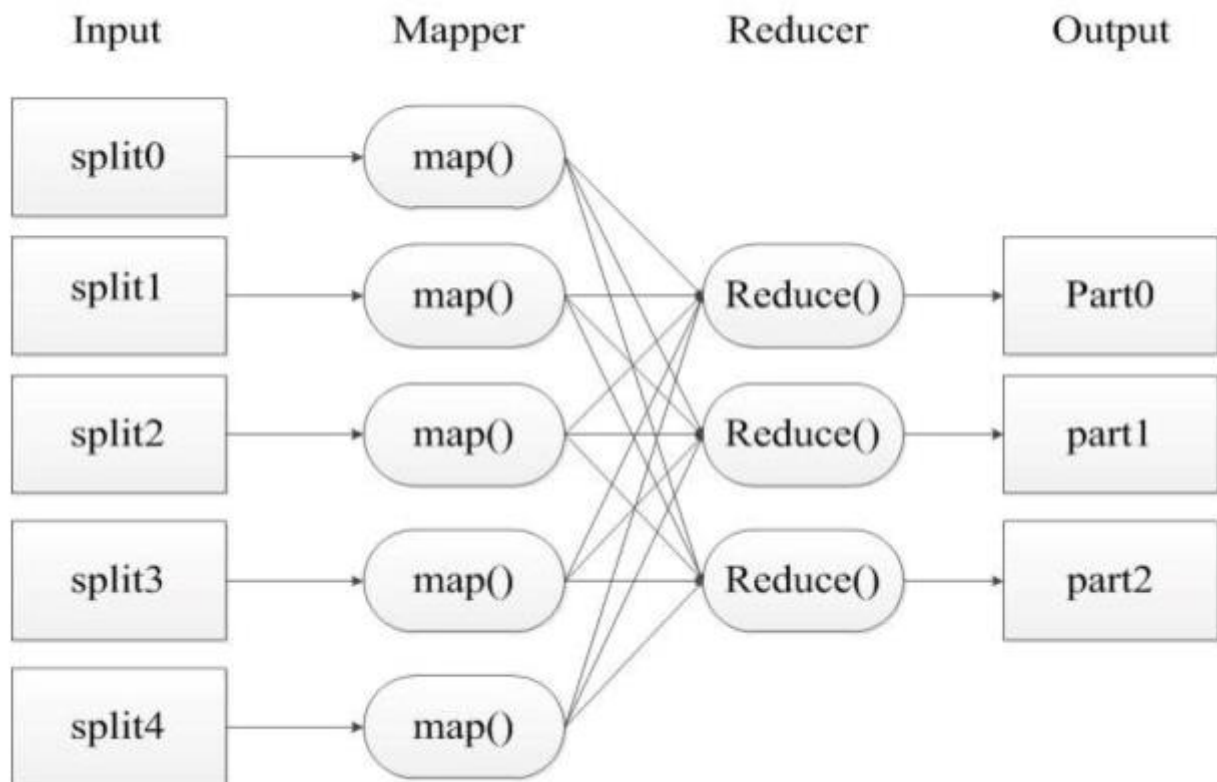
- Số lượng màu: Giảm từ 16 màu gốc xuống 3 màu đại diện.

CHƯƠNG 3: ỨNG DỤNG MAPREDUCE TRONG PHÂN CỤM ẢNH

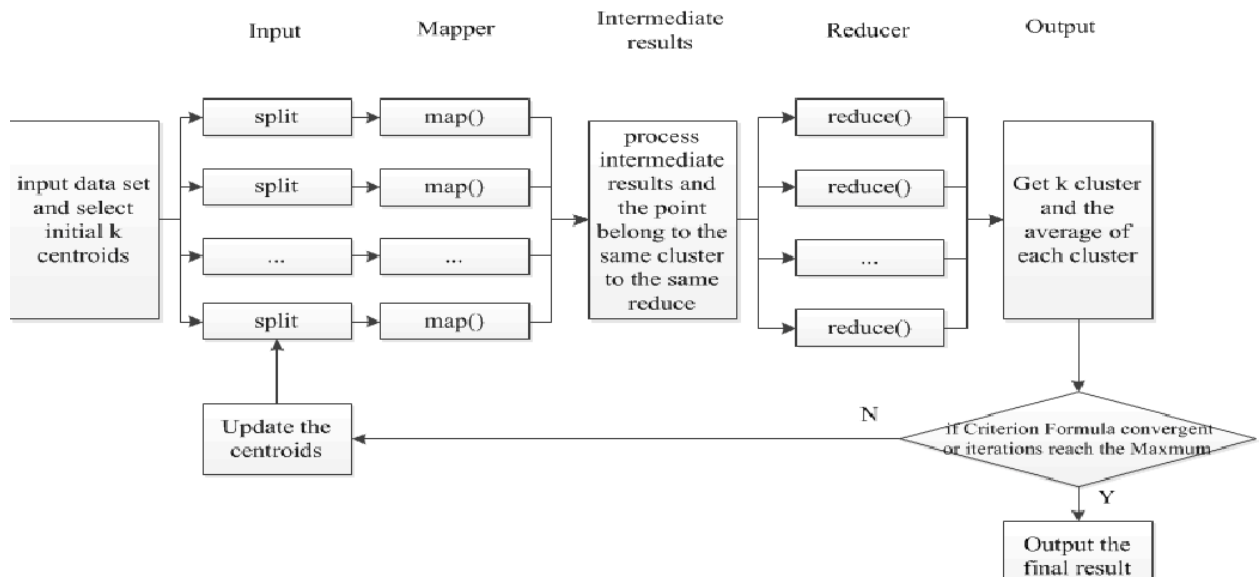
3.1 Ý tưởng tích hợp K-Means với MapReduce

Thuật toán K-Means truyền thống gặp phải hạn chế về hiệu năng khi xử lý các tập dữ liệu ảnh lớn. Việc tích hợp K-Means với MapReduce cho phép tận dụng khả năng xử lý song song và phân tán của MapReduce để giải quyết vấn đề này.

Ý tưởng chính là chia tập dữ liệu ảnh thành các phần nhỏ, mỗi phần được xử lý bởi một nút (node) trong hệ thống MapReduce. Các nút này sẽ thực hiện thuật toán K-Means một cách độc lập, sau đó kết quả được tổng hợp để cập nhật lại các centroid chung.



Hình 1.1 Quy trình MapReduce



Hình 2.2 Quy trình MapReduce K-Means

Quy trình Map và Reduce cho K-Means:

1. Giai đoạn Map:

- **Đầu vào:** Mỗi mapper nhận một phần của tập dữ liệu ảnh và danh sách các centroid hiện tại.
- **Xử lý:**
 - Chuyển đổi ảnh thành vector đặc trưng (ví dụ: sử dụng HOG, SIFT, SURF).
 - Tính toán khoảng cách từ mỗi điểm dữ liệu (vector đặc trưng của ảnh) đến các centroid.

▪ Công thức tính khoảng cách Euclidean:

$$d(x, y) = \sqrt{\sum (x_i - y_i)^2}$$

trong đó:

- x, y là hai vector.
- x_i, y_i là các thành phần tương ứng của vector x, y .
- Gán mỗi điểm dữ liệu vào cụm có centroid gần nhất.
- **Đầu ra:** Mỗi mapper tạo ra các cặp key-value, với key là centroid và value là danh sách các điểm dữ liệu thuộc về cụm đó.

2. Giai đoạn Reduce:

- **Đầu vào:** Mỗi reducer nhận một danh sách các điểm dữ liệu thuộc về một cụm.

- **Xử lý:**
 - Tính toán trung bình cộng của các điểm dữ liệu để cập nhật centroid mới cho cụm đó.
 - **Công thức tính trung bình cộng của các vector:**

$$c = \frac{\sum x_i}{n}$$

trong đó:

 - c là centroid mới.
 - x_i là các vector đặc trưng của ảnh trong cụm.
 - n là số lượng vector trong cụm.
- **Đầu ra:** Mỗi reducer tạo ra các cặp key-value, với key là nhãn cụm và value là centroid mới.

3.2 Mô hình và thuật toán chi tiết

Mô tả mapper:

- **Input:**
 - Key: Offset của dòng dữ liệu trong tập dữ liệu đầu vào.
 - Value: Dữ liệu ảnh (có thể ở dạng đường dẫn đến ảnh hoặc dữ liệu ảnh thô).
- **Xử lý:**
 1. Đọc dữ liệu ảnh và chuyển đổi thành vector đặc trưng.
 2. Đọc danh sách các centroid từ file clusters.txt (được lưu trong cache).
 3. Tính toán khoảng cách Euclidean từ vector đặc trưng của ảnh đến từng centroid.
 4. Xác định centroid gần nhất với vector đặc trưng của ảnh.
 5. Ghi ra cặp key-value với key là centroid gần nhất và value là vector đặc trưng của ảnh.
- **Output:**
 - Key: Centroid gần nhất.
 - Value: Vector đặc trưng của ảnh.

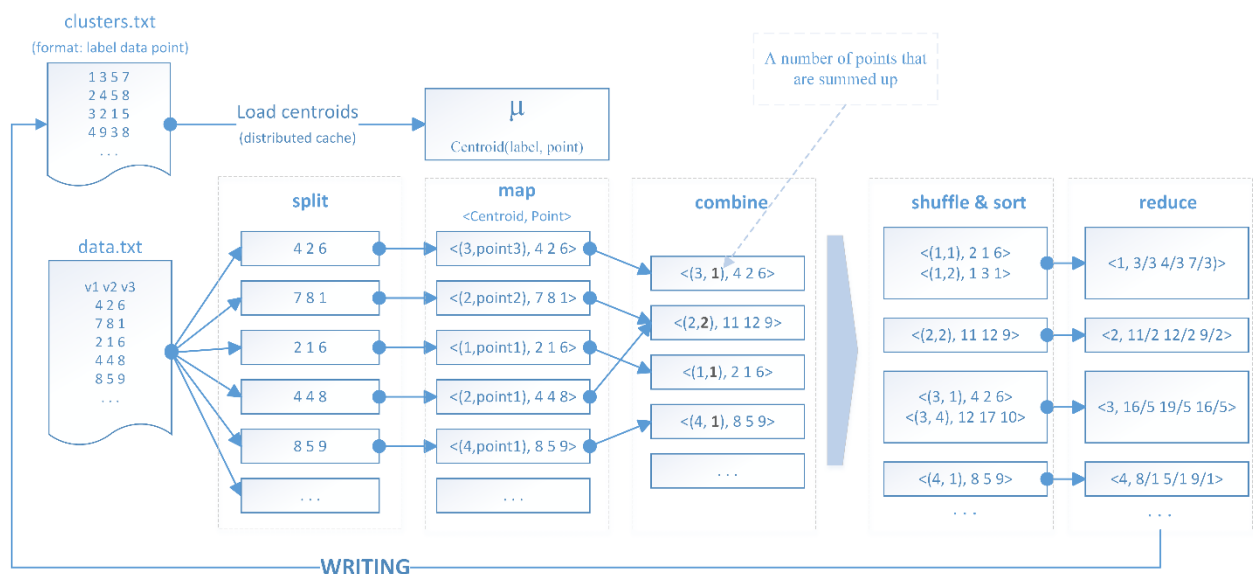
Mô tả Combiner:

- **Vai trò:** Giảm thiểu dữ liệu truyền giữa Mapper và Reducer bằng cách cộng gộp các điểm dữ liệu trên cùng một máy và ghi lại số lượng.
- **Input:** Tương tự như Reducer.

- **Xử lý:** Cộng các điểm dữ liệu và ghi lại số lượng vào biến Point.number.
- **Output:**
 - Key: Centroid.
 - Value: Điểm dữ liệu đã cộng gộp và số lượng.

Mô tả reducer:

- **Input:**
 - Key: Centroid
 - Value: Danh sách các vector đặc trưng của ảnh thuộc về cụm của centroid đó.
- **Xử lý:**
 1. Tính tổng các vector đặc trưng của ảnh trong danh sách.
 2. Chia tổng cho số lượng vector để tính trung bình cộng, từ đó cập nhật centroid mới.
 3. Ghi ra cặp key-value với key là nhãn cụm và value là centroid mới.
- **Output:**
 - Key: Nhãn cụm.
 - Value: Centroid mới.



Hình 2.3 Một lần lặp lại của chương trình MapReduce với dữ liệu

3.3 Áp dụng trên tập dữ liệu ảnh

- **Chuyển đổi dữ liệu ảnh thành vector:** Sử dụng các kỹ thuật trích xuất đặc trưng ảnh như HOG, SIFT, SURF hoặc các phương pháp nhúng ảnh (image embedding) để chuyển đổi ảnh thành vector số thực.
- **Các tiêu chí hội tụ:**
 - **Số lần lặp tối đa:** Giới hạn số lần lặp của thuật toán.
 - **Khoảng cách tối thiểu giữa các centroid:** Thuật toán kết thúc khi khoảng cách giữa các centroid ở lần lặp hiện tại và lần lặp trước đó nhỏ hơn một ngưỡng cho trước.
- **Xử lý dữ liệu phi cấu trúc:** Tùy thuộc vào tập dữ liệu ảnh, có thể cần xử lý các vấn đề như ảnh bị thiếu, ảnh nhiễu, hoặc ảnh có kích thước khác nhau.

3.4 Cải tiến K-Means

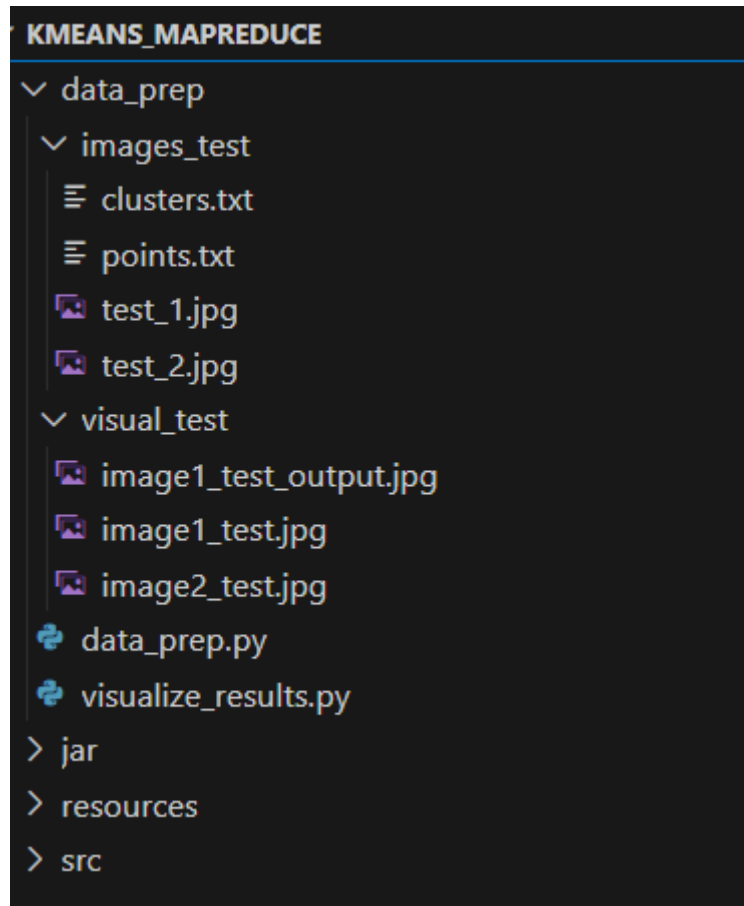
- **K-Means++:** Cải thiện việc khởi tạo centroid ban đầu để tăng tốc độ hội tụ và chất lượng phân cụm.
- **Fuzzy K-Means:** Cho phép một điểm dữ liệu thuộc về nhiều cụm với các mức độ khác nhau.
- **K-Means cải tiến với Artificial Bee Colony**
- **Các phương pháp tối ưu hóa:** Sử dụng các kỹ thuật tối ưu hóa để giảm thời gian tính toán và tăng hiệu năng.

3.5 Demo chương trình cài đặt

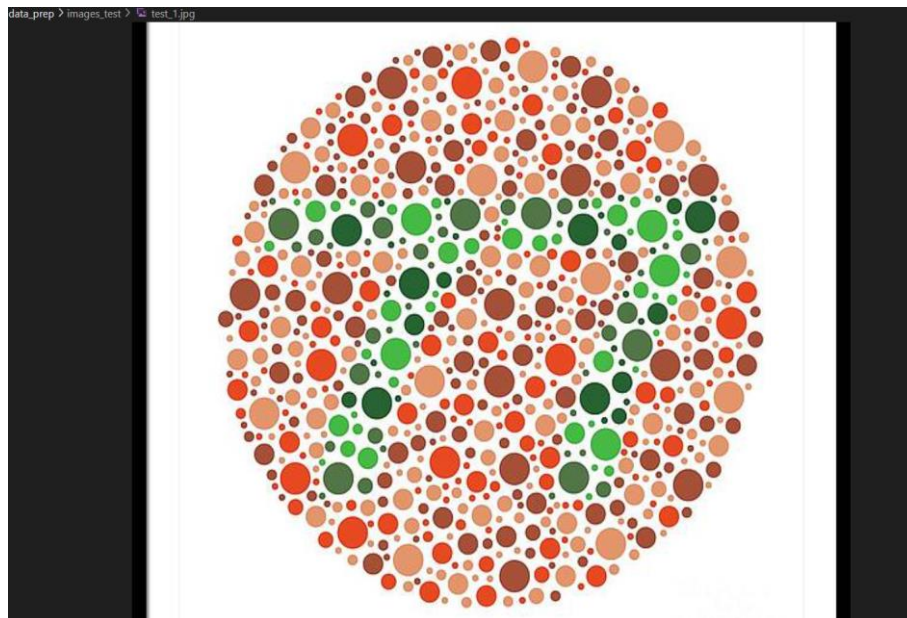
Hướng dẫn:

1. Chuẩn bị dữ liệu ảnh và chuyển đổi thành vector đặc trưng.
2. Lưu dữ liệu ảnh đã chuyển đổi vào file points.txt.
3. Khởi tạo ngẫu nhiên các centroid ban đầu và lưu vào file clusters.txt.
4. Chạy chương trình MapReduce để phân cụm ảnh.
5. Sử dụng script visualize_results.py để hiển thị kết quả phân cụm.

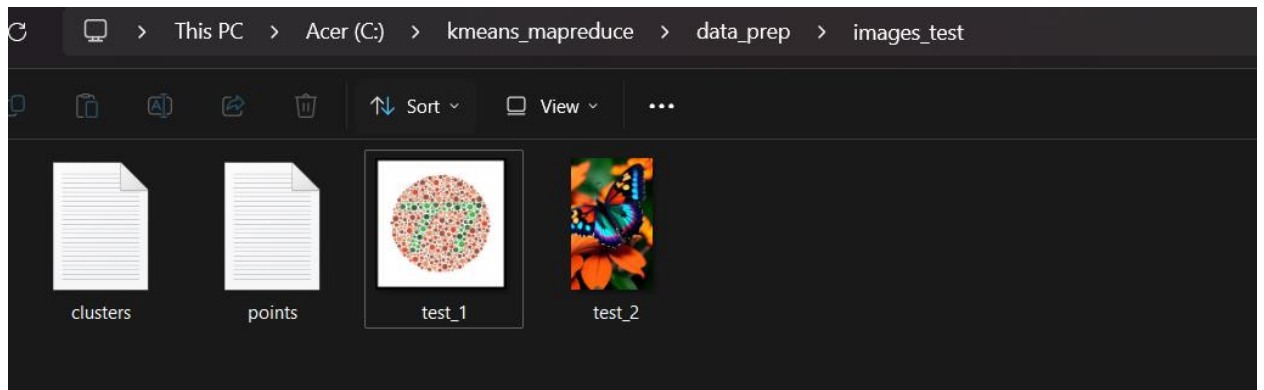
Chi tiết:



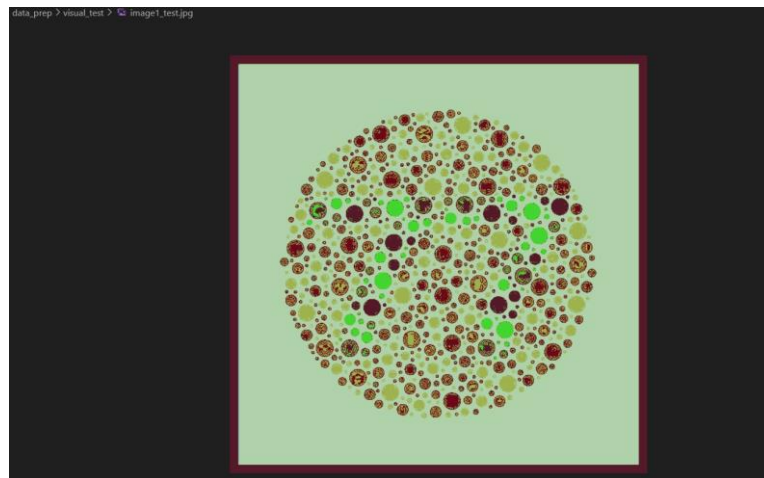
Hình 2.4 Cấu trúc thư mục xử lý ảnh đầu vào

























Hình 2.5 Ảnh test cluster



Hình 2.6. *data_prep.py* xử lí ảnh tạo *points.txt* và *clusters.txt*



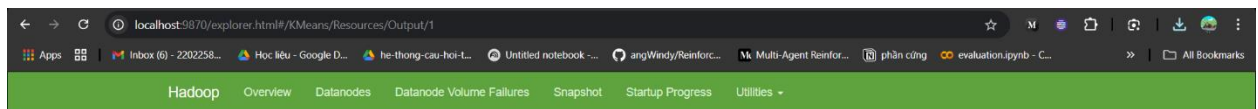
Hình 2.7. visualize ảnh test khi chưa cluster bằng kmean:

- ✓  Kmean-MapReduce_Image
 - >  JRE System Library [JavaSE-1.8]
 - ✓  src
 - ✓  (default package)
 - >  Algorithm.java
 - >  JobIterator.java
 - >  KMeansCombiner.java
 - >  KMeansMapper.java
 - >  KMeansPartitioner.java
 - >  KMeansReducer.java
 - >  Main.java
 - ✓  distances
 - >  Distance.java
 - >  EuclideanDistance.java
 - ✓  enums
 - >  KMeansCounter.java
 - ✓  iterators
 - >  CentroidIterator.java
 - ✓  writables
 - >  Centroid.java
 - >  Point.java
 - >  Referenced Libraries

Hình 2.8. Cấu trúc thư mục project mapreduce kmean

Quá trình chạy MapReduce:





```
Administrator: Command Prompt - hadoop jar "C:\jar\Kmean-Mapreduce.jar" --input /KMeans/Resources/Input/points.txt --state /KMeans/Resources/Input/clusters.txt --number 3 --output /KMeans/Resources/Output --delta 10000000.0 --max 10 --distance eucl...
C:\Windows\System32>hadoop jar "C:\jar\Kmean-Mapreduce.jar" --input /KMeans/Resources/Input/points.txt --state /KMeans/Resources/Input/clusters.txt --number 3 --output /KMeans/Resources/Output --delta 10000000.0
#####
input=/KMeans/Resources/Input/points.txt
state=/KMeans/Resources/Input/clusters.txt
output=/KMeans/Resources/Output
max=10
number=3
delta=10000000.0
distance=eucl
#####
First iteration adding /KMeans/Resources/Input/clusters.txt
2024-12-07 20:32:39,273 INFO client.DefaultHARFollowerProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-12-07 20:32:40,001 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/huyho/.staging/job_1733577519283_0001
2024-12-07 20:32:40,314 INFO input.FileInputFormat: Total input files to process : 1
2024-12-07 20:32:40,412 INFO mapreduce.JobSubmitter: number of splits:1
2024-12-07 20:32:40,579 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1733577519283_0001
2024-12-07 20:32:40,579 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-12-07 20:32:40,787 INFO conf.Configuration: resource-types.xml not found
2024-12-07 20:32:40,787 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-12-07 20:32:41,252 INFO impl.YarnClientImpl: Submitted application application_1733577519283_0001
2024-12-07 20:32:41,308 INFO mapreduce.Job: The url to track the job: http://maytinhcuahong:8088/proxy/application_1733577519283_0001/
2024-12-07 20:32:41,309 INFO mapreduce.Job: Running job: job_1733577519283_0001
2024-12-07 20:32:51,545 INFO mapreduce.job: Job job_1733577519283_0001 running in uber mode : false
2024-12-07 20:32:51,547 INFO mapreduce.job: map 0% reduce 0%
```



Browse Directory














/KMeans/Resources/Output/1

Go!



Show25entries

Search:

<input type="checkbox"/>	 Permission	 Owner	 Group	 Size	 Last Modified	 Replication	 Block Size	 Name	
<input type="checkbox"/>	-rw-r--r--	huyho	supergroup	0 B	Dec 07 21:47	1	128 MB	_SUCCESS	
<input type="checkbox"/>	-rw-r--r--	huyho	supergroup	113 B	Dec 07 21:47	1	128 MB	part-r-00000	
<input type="checkbox"/>	-rw-r--r--	huyho	supergroup	173 B	Dec 07 21:47	1	128 MB	part-r-00001	
<input type="checkbox"/>	-rw-r--r--	huyho	supergroup	57 B	Dec 07 21:47	1	128 MB	part-r-00002	

Showing 1 to 4 of 4 entries

Previous

1

Next

Hadoop, 2020.



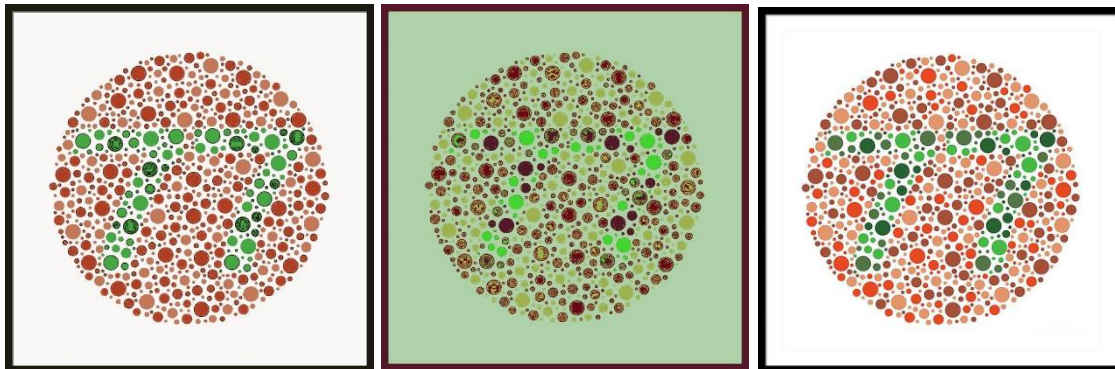
Hình 2. Quá trình Mapreduce chạy thành công(sử dụng 10 iterations và 3 reducers)

```

1 3 90.20912913243014 121.64695374975955 198.2799197559702
2 6 17.56349982933212 27.1616793719422 29.325201956991695
3 1 139.33814102564102 140.73397435897436 138.0096153846154
4 10 38.100141477517376 63.68518176785385 176.8189087777573
5 7 246.3543859785723 248.95894714712844 250.8753036626386
6 5 68.67252153352474 169.50109778753588 70.02499577774024

```

Hình 3. Kết quả output



Đánh giá chương trình

Chương trình chạy tốt trên tập dữ liệu ảnh mẫu. Tuy nhiên, nhóm chúng tôi thấy cần cải thiện thêm về hiệu năng khi xử lý dữ liệu lớn, cách khởi tạo centroid và cách chọn số lượng cụm.

Hướng phát triển trong tương lai

- **Cải thiện hiệu năng:** Tối ưu MapReduce, dùng MiniBatch K-Means.
- **Khởi tạo centroid:** Áp dụng K-means++.
- **Số lượng cụm:** Tìm phương pháp xác định số lượng cụm tự động.
- **Mở rộng:** Hỗ trợ nhiều loại khoảng cách, biến thể K-means.

CHƯƠNG 4: THUẬT TOÁN K-MEANS CẢI TIẾN VỚI ARTIFICIAL BEE COLONY

Thuật toán **Artificial Bee Colony (ABC)** lấy cảm hứng từ hành vi tìm kiếm mật hoa của ong. Nó sử dụng trí thông minh bầy đàn để tối ưu hóa các giải pháp trong một miền vấn đề cụ thể. Trong thuật toán, một đàn ong nhân tạo được tạo ra, mỗi con ong đại diện cho một giải pháp tiềm năng. Các con ong này mô phỏng cách ong thật tìm kiếm nguồn mật hoa tốt nhất.

4.1. Thuật toán ABC

1. Các loại ong:

- Ong thợ: Tìm kiếm các giải pháp mới (nguồn thức ăn) và chia sẻ thông tin với đàn ong.
- Ong quan sát : Chọn giải pháp từ các giải pháp hiện tại để cải thiện thêm.
- Ong trinh sát : Khám phá các giải pháp mới khi các giải pháp hiện tại không còn hiệu quả.

2. Các giai đoạn của thuật toán:

○ Giai đoạn khởi tạo:

- Tạo ngẫu nhiên quần thể ong (giải pháp ban đầu).
- Mỗi giải pháp nằm trong không gian giải pháp khả thi:

$$x_{i,j} = x_{j,min} + random(0,1) \cdot (x_{j,max} - x_{j,min})$$

○ Giai đoạn tìm kiếm của ong thợ:

- Tìm kiếm nguồn thức ăn mới gần nguồn hiện tại dựa trên công thức:

$$V_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj})$$

- $k \in \{1, 2, \dots, N\}$ và $j \in \{1, 2, \dots, D\}, k \neq i$
- $\varphi_{ij} \in [0,1]$

- Áp dụng phương pháp "chọn lọc tham lam" để giữ lại giải pháp tốt hơn.

So sánh x_{ij} với v_{ij} :

Nếu giải pháp mới tốt hơn, giữ lại giải pháp mới và đặt bộ đếm về 0.

Nếu giải pháp mới không tốt hơn, giữ lại giải pháp cũ và tăng bộ đếm lên 1.

- Giai đoạn tìm kiếm của ong quan sát:

- Chọn nguồn thức ăn dựa trên xác suất:

$$P_i = \frac{\text{fitness}_i}{\sum_{j=1}^N \text{fitness}_j}$$

- fitness_i : Chất lượng của nguồn thức ăn i.

$$\text{fitness}_i = \begin{cases} \frac{1}{1+f_i}, & f_i \geq 0 \\ 1 + |f_i|, & f_i < 0 \end{cases}, f_i: \text{ giá trị hàm mục tiêu}$$

- Tiếp tục cải thiện giải pháp được chọn dựa trên "chọn lọc tham lam".

- Giai đoạn tìm kiếm của ong trình sát:

- Nếu bộ đếm của một nguồn thức ăn vượt quá một ngưỡng xác định, nguồn thức ăn đó bị coi là không hiệu quả (bỏ đi).
- Ong dẫn đầu của nguồn thức ăn đó chuyển thành ong trình sát, tạo ra một nguồn thức ăn mới ngẫu nhiên theo công thức trong giai đoạn khởi tạo.

4.2. Thuật toán IK-ABC (Improved K-Means with Artificial Bee Colony)

4.2.1. Các điểm chính của IK-ABC

1. Nhược điểm của K-Means:
 - Dễ rơi vào cực trị cục bộ.
 - Phụ thuộc mạnh vào điểm khởi tạo ban đầu.
 - Hội tụ chậm khi xử lý dữ liệu lớn.
2. Vai trò của ABC:
 - Tối ưu hóa dựa trên thuật toán tìm kiếm đa mục tiêu, mô phỏng hành vi kiếm ăn của ong mật.
 - Cải thiện khởi tạo centroids và chất lượng phân cụm cuối cùng.
3. Nhược điểm ban đầu của ABC:
 - Hiệu suất thấp do khởi tạo ngẫu nhiên.
 - Hội tụ chậm do tìm kiếm một chiều.
4. Cải tiến trong IK-ABC:
 - Sử dụng khoảng cách lớn nhất và nhỏ nhất để giảm tính ngẫu nhiên trong khởi tạo.
 - Cải tiến hàm fitness để tăng tốc hội tụ.
 - Tăng hiệu quả tìm kiếm bằng công thức cập nhật vị trí toàn cục.

4.2.2. Các thành phần chính

1. Cải tiến trong khởi tạo ban đầu:

- Sử dụng phương pháp tích khoảng cách lớn nhất và nhỏ nhất để chọn vị trí khởi tạo centroids phù hợp, giảm phụ thuộc vào điểm ban đầu.
2. Hàm fitness:
- Đánh giá chất lượng cụm dựa trên số lượng điểm trong cụm (CM_i) và tổng khoảng cách giữa các điểm và centroid ($Dist_i$).

Công thức:

$$fitness_i = \frac{CM_i}{Dist_i} \quad i = 1, 2, \dots, N$$

$$C_i, Dist_i = \sum_{x_j \in C_i} d(x_j, C_k), Dist = \sum_{j=1}^k \sum_{x_i \in C_j} d(x_i, C_j)$$

3. Công thức cập nhật vị trí (Position Updating):
- Kết hợp khai thác cục bộ (exploitation) và tìm kiếm toàn cục (exploration):

$$v_{ij} = x_{ij} + r_{ij}(x_{mj} - x_{kj}) + \mu (x_{best,j} - x_{i,j})$$

- Trong đó:
 - $r_{ij} \in [-1, 1]$
 - $\mu \in [0, 1]$
 - $x_{best,j}$: Centroid tốt nhất hiện tại.
4. Phân tán với MapReduce:
- Mapper: Gán mỗi điểm dữ liệu vào cụm gần nhất.
 - Reducer: Tính lại centroid dựa trên điểm được gán.

4.2.3 Quy trình thuật toán IK-ABC

1. Khởi tạo quần thể ong (bee population):
 - Mỗi con ong đại diện cho một tập centroids ban đầu.
2. Phân đoạn dữ liệu và phân phối qua MapReduce:
 - Dữ liệu được chia nhỏ và phân bổ đến các worker nodes.
3. Thực thi K-Means trên từng phân đoạn:
 - Với centroids ban đầu do các "ong" cung cấp.
 - Tính toán giá trị fitness dựa trên chất lượng phân cụm (e.g., tổng lỗi bình phương).
4. Tối ưu hóa quần thể ong qua 3 giai đoạn:
 - Ong thợ: Cải thiện vị trí hiện tại bằng cách tìm kiếm cục bộ.
 - Ong quan sát: Chọn cụm tiềm năng dựa trên giá trị fitness.
 - Ong trình sát: Tìm kiếm giải pháp mới nếu cụm không cải thiện.
5. Kết hợp kết quả từ các worker nodes:
 - Trả dữ liệu về master node, nơi tổng hợp và cập nhật quần thể ong.
6. Lặp lại các bước trên cho đến khi hội tụ:

- Điều kiện dừng có thể là số vòng lặp tối đa hoặc độ hội tụ đạt yêu cầu.
7. Kết quả cuối cùng:
- Centroids tốt nhất từ "ong giỏi nhất" được dùng để phân cụm toàn bộ dữ liệu.

4.2.3. Kết quả

Giảm thời gian chạy: mô hình cũ khoảng 30' giảm xuống 20' ở mô hình mới

```
kmeans_traditional = KMeans(n_clusters=clusters.shape[0])
kmeans_traditional.fit(points)
```

✓ 30m 46.7s Python

```
# Thực hiện K-Means
kmeans = KMeans(n_clusters=clusters.shape[0])
kmeans.centroids = clusters[:, 1:] # Use centroids optimized by ABC
kmeans.fit(points)
```

✓ 18m 3.2s

Điểm hàm lỗi giảm xuống khoảng 4 lần

```
labels_traditional = kmeans_traditional.predict(points)
sse = calculate_sse(points, kmeans_traditional.centroids, labels_traditional)
print("SSE Traditional: ", sse)
```

✓ 17.6s

SSE Traditional: 826437817.263663

```
labels = kmeans.predict(points)
sse = calculate_sse(points, kmeans.centroids, labels)
print("SSE:", sse)
```

✓ 17.6s

SSE: 245498070.30738327

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Bài báo cáo nghiên cứu đã triển khai thành công thuật toán K-Means trong phân cụm dữ liệu ảnh, kết hợp với framework MapReduce trên Hadoop để xử lý hiệu quả các tập dữ liệu ảnh lớn. Những điểm nổi bật của nghiên cứu bao gồm:

1. Hiệu quả của K-Means trên dữ liệu lớn:

- Bằng cách tận dụng MapReduce, thuật toán K-Means đã được mở rộng khả năng xử lý phân tán, giúp giảm thiểu thời gian xử lý và tăng cường khả năng mở rộng.
- Việc sử dụng DistributedCache để lưu trữ centroid và Combiner để giảm thiểu dữ liệu truyền giữa các node đã góp phần cải thiện hiệu năng của chương trình.

2. Cải tiến K-Means với Artificial Bee Colony (ABC):

- Việc tích hợp ABC vào K-Means đã giải quyết một số hạn chế của thuật toán K-Means truyền thống như phụ thuộc vào khởi tạo ngẫu nhiên và dễ mắc kẹt tại cực tiểu cục bộ.
- Thuật toán cải tiến IK-ABC mang lại chất lượng cụm cao hơn thông qua việc tối ưu hóa liên tục các centroid trong quá trình chạy.

3. Kết quả thực nghiệm:

- Chương trình đạt độ chính xác chấp nhận được trong việc phân cụm dữ liệu ảnh mẫu.
- Tuy nhiên, khi áp dụng với tập dữ liệu lớn hơn, hiệu năng và tốc độ hội tụ cần được cải thiện thêm.
- Mặc dù đạt được nhiều thành công, nghiên cứu vẫn còn những hạn chế, bao gồm:
 - Hiệu năng chưa tối ưu đối với tập dữ liệu cực lớn.
 - Ảnh hưởng đáng kể của việc khởi tạo centroid ngẫu nhiên đến kết quả phân cụm.
 - Thiếu các phương pháp tự động xác định số cụm (k).

5.2 Hướng phát triển

Để khắc phục các hạn chế hiện tại và mở rộng tiềm năng ứng dụng của nghiên cứu, các hướng phát triển sau được đề xuất:

1. Cải thiện hiệu năng:

- Tối ưu hóa quy trình MapReduce bằng cách áp dụng các kỹ thuật như speculative execution và điều chỉnh tham số Hadoop.
- Sử dụng MiniBatch K-Means để xử lý từng lô dữ liệu nhỏ, giảm thiểu thời gian tính toán mà vẫn duy trì độ chính xác.

2. Tối ưu hóa khởi tạo centroid:

- Áp dụng kỹ thuật K-Means++ để khởi tạo centroid thông minh, giúp tăng tốc độ hội tụ và cải thiện chất lượng cụm.
- Nghiên cứu các phương pháp khởi tạo dựa trên phân tích đặc trưng dữ liệu để chọn các centroid khởi đầu tối ưu.

3. Tự động xác định số cụm:

- Sử dụng các phương pháp như Elbow method, Silhouette analysis hoặc Gap statistic để tự động xác định số lượng cụm phù hợp (k) với tập dữ liệu.

4. Mở rộng chương trình:

- Hỗ trợ nhiều loại khoảng cách khác nhau như Manhattan, Cosine similarity hoặc các biến thể của K-Means như K-Medoids và Kernel K-Means.
- Áp dụng các thuật toán phân cụm mạnh hơn như Expectation-Maximization (EM) để xử lý dữ liệu phức tạp hoặc có nhiễu.

5. Ứng dụng vào các loại dữ liệu khác:

- Nghiên cứu áp dụng thuật toán K-Means kết hợp MapReduce cho dữ liệu phi cấu trúc như video, âm thanh hoặc dữ liệu văn bản.
- Xây dựng các phương pháp trích xuất đặc trưng phù hợp cho từng loại dữ liệu để nâng cao hiệu quả phân cụm.

Bằng cách thực hiện các cải tiến và mở rộng như trên, nghiên cứu không chỉ tăng cường hiệu quả trong phân cụm dữ liệu ảnh mà còn mở ra các ứng dụng mới trong nhiều lĩnh vực khác, từ thị giác máy tính, xử lý ngôn ngữ tự nhiên đến phân tích âm thanh và video. Đây là bước tiến quan trọng trong việc ứng dụng công nghệ Big Data để giải quyết các bài toán phức tạp trong thực tế.

TÀI LIỆU THAM KHẢO

1. "MapReduce Algorithms for K-means Clustering" : [Stanford University](#)
2. "An Improved Parallel K-means Clustering Algorithm with MapReduce" : [IEEE Xplore](#)
3. "Optimized Big Data K-means Clustering Using MapReduce": [Springer Link](#)
4. "Parallel K-means Using MapReduce": [IEEE Xplore](#)
5. "Optimized Big Data K-Means Clustering Using MapReduce": [Springer Link](#)
6. "A MapReduce-Based K-Means Clustering Algorithm": [Springer Link](#)