K-Means & lập trình MapReduce hóa trong Phân cụm ảnh

Ứng dụng vào phân cụm trong ảnh chụp vệ tinh

Mục tiêu

Áp dụng hai công nghệ trên trong phân cụm ảnh, minh họa qua việc xử lý phân đoạn ảnh hàng không (semantic segmentation) nhằm nhận diện, phân biệt các đối tượng trong ảnh

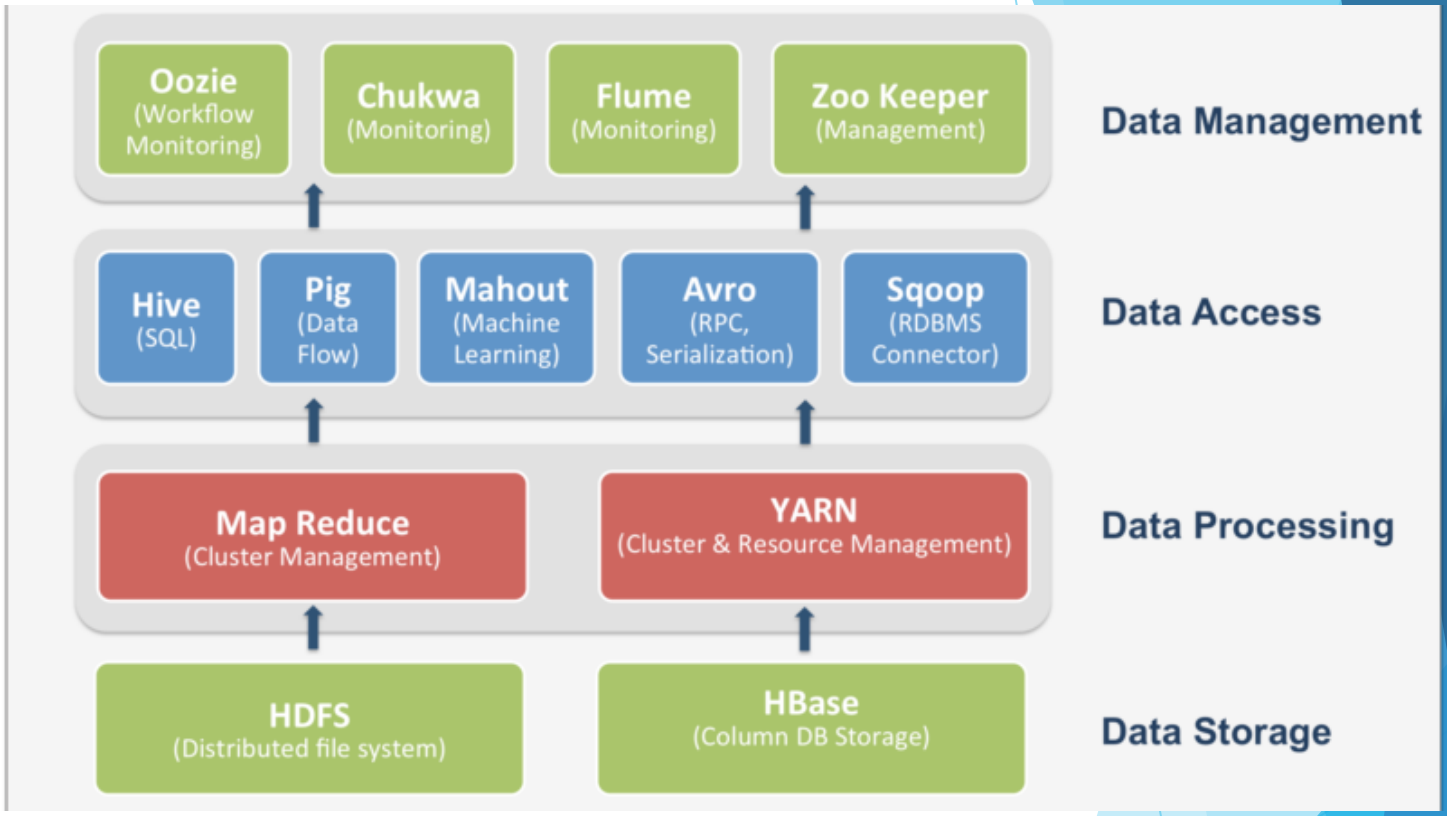
Mục lục

- 1. Tổng quan về Hadoop
- 2. Về kiến trúc MapReduce
- 3. Thuật toán KMeans và khởi tạo clusters bằng KMeans++
- 4. Triển khai trên MapReduce
- 5. Xử lý hình ảnh sau khi đã phân đoạn để phân cụm ảnh chụp vệ tinh
- 6. Đánh giá thuật toán.

1. Tổng quan về 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.

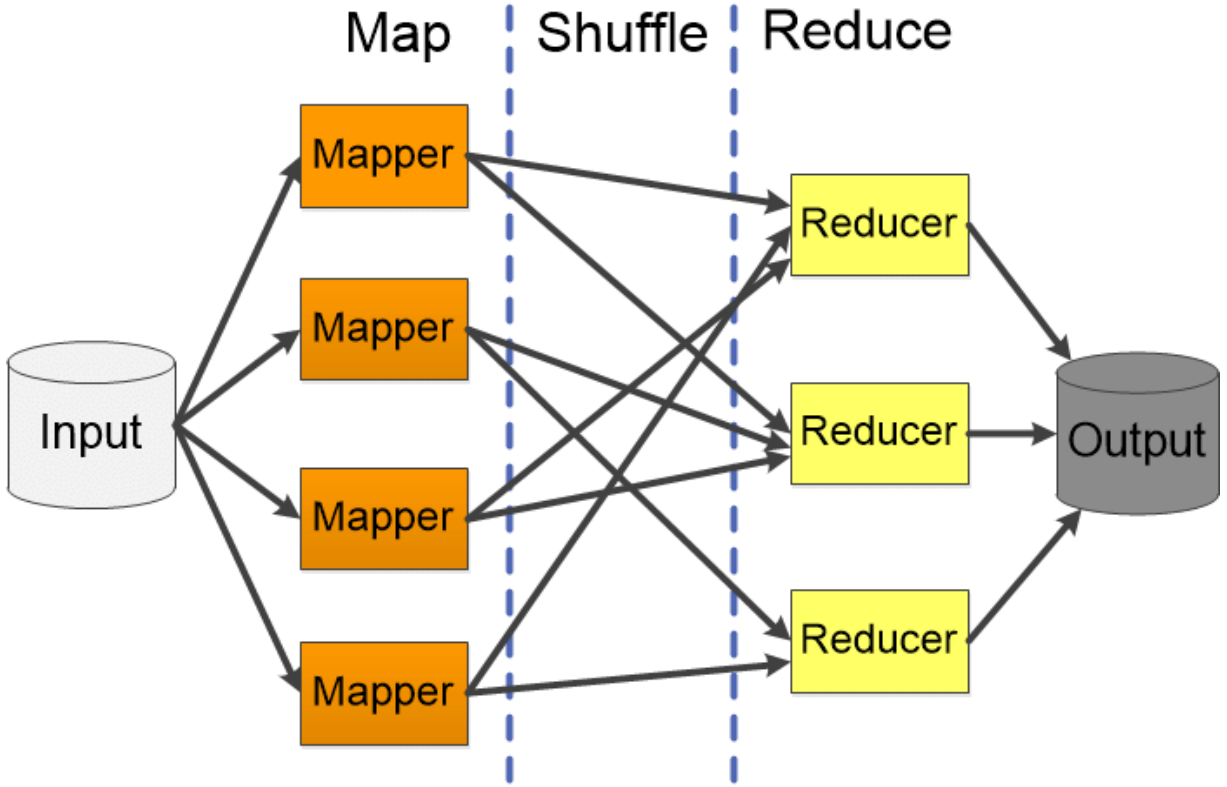
1. Tổng quan về Hadoop



- Hệ sinh thái Hadoop gồm nhiều thành phần khác nhau, nhưng trong chủ đề này chúng ta sẽ nói đến và thực hiện HDFS và Map Reduce

2. Về kiến trúc MapReduce

- **Theo Google**: MapReduce là mô hình dùng cho xử lý tính toán song song và phân tán trên hệ thống phân tán.


Map
Shuffle
Reduce
Input
Mapper
Mapper
Mapper
Mapper
Reducer
Reducer
Reducer
Output

2. Về MapReduce

- **Hàm Map** xử lý các mảnh dữ liệu đầu vào, trích xuất thông tin quan trọng từ từng phần tử để tạo ra kết quả trung gian.
- **Hàm Reduce** tổng hợp các kết quả trung gian và thực hiện các phép tính cần thiết để tạo ra kết quả cuối cùng.

3. Thuật toán KMeans và khởi tạo clusters với KMeans++

3.1 Thuật toán KMeans

- Cho một bộ dữ liệu gồm n điểm dữ liệu $(\overrightarrow{x_1}, \overrightarrow{x_2}, \dots, \overrightarrow{x_n})$, với mỗi điểm là một vector có d chiều

- **Mục tiêu:** chia n điểm dữ liệu đã cho thành $k\ (k \leq n)$ tập con $S = \{S_1, S_2, \dots, S_k\}$ nhằm giảm thiểu tổng bình phương bên trong cụm.

3.1 Thuật toán KMeans

- Mục tiêu chính sẽ phân cụm theo phương trình:

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\vec{x} \in S_i} \|\vec{x} - \overrightarrow{\mu_i}\|^2$$

- Với $\overrightarrow{\mu_i} = \frac{1}{|S_i|} \sum_{\vec{x} \in S_i} \vec{x}$ là kì vọng (centroid) và $|S_i|$ là kích cỡ của S_i

3.2 Các bước hoạt động của thuật toán KMeans

3.2 Thuật toán

- Khởi tạo các centroids ngẫu nhiên

$$\mathcal{C}^{(0)} = \left\{ \vec{\mu}_1^{(0)}, \vec{\mu}_2^{(0)}, \dots, \vec{\mu}_{\mathrm{k}}^{(0)} \right\}$$

3.2 Thuật toán

- Gán các điểm vào các clusters
- Với mỗi điểm dữ liệu, tính khoảng cách của nó tới centroids và gán vào centroids gần nhất và tạo thành cụm:

$$\mathcal{S}_i^{(t)} = \left\{ \vec{x}_p : \left\| \vec{x}_p - \vec{\mu}_i^{(t)} \right\|^2 \leq \left\| \vec{x}_p - \vec{\mu}_j^{(t)} \right\|^2 \right\}, \forall j, 1 \leq j \leq k$$

3.2 Thuật toán

- Cập nhật centroids

$$\vec{\mu}_i^{(t+1)} = \frac{1}{\left|\mathcal{S}_i^{(t)}\right|} \sum_{\vec{x} \in \mathcal{S}_i^{(t)}} \vec{x}_j$$

- Lặp lại thuật toán cho đến khi chấp nhận được.

Hạn chế của KMeans

- Trong trường hợp xấu nhất, độ phức tạp tính toán của K-Means có thể trở thành superpolynomial, đặc biệt khi số lượng điểm dữ liệu rất lớn và số cụm K cao. Tuy nhiên, với các trường hợp thực tế, thuật toán thường có độ phức tạp thời gian là O(K·n·t), trong đó:
  - K là số cụm,
  - n là số điểm dữ liệu,
  - t là số lần lặp.
- Việc khởi tạo các centroid ngẫu nhiên có thể khiến cho quá trình phân cụm trở nên khó khăn hơn. Các centroid ban đầu không hợp lý có thể dẫn đến kết quả phân cụm không

3.3 Cải thiện với thuật toán KMeans++

3.3 Thuật toán

- Thay vì khởi tạo k centroids một cách ngẫu nhiên, ta chỉ khởi tạo centroid đầu ngẫu nhiên
- Tìm $k - 1$ centroids còn lại bằng việc sử dụng xác suất

3.3 Thuật toán

- Với mỗi điểm dữ liệu, chúng ta tính khoảng cách từ $\vec{x}$ đến $\vec{\mu}$

$$\mathcal{D}_i(\vec{x}) = \min_{\vec{\mu} \in \{\vec{\mu}_1, \ldots, \vec{\mu}_i\}} \|\vec{x} - \vec{\mu}\|^2$$

- Tạo vector

$$\vec{\mathcal{D}}_{i+1} = [\mathcal{D}_1(\vec{x}_1) \quad \cdots \quad \mathcal{D}_1(\vec{x}_n)]$$

- Tính xác suất:

$$p(\vec{x}) = \frac{\mathcal{D}_i(\vec{x})}{\sum_{\vec{x'} \in X} \mathcal{D}_i\left(\vec{x'}\right)}, \qquad \forall\ \vec{x} \in X$$

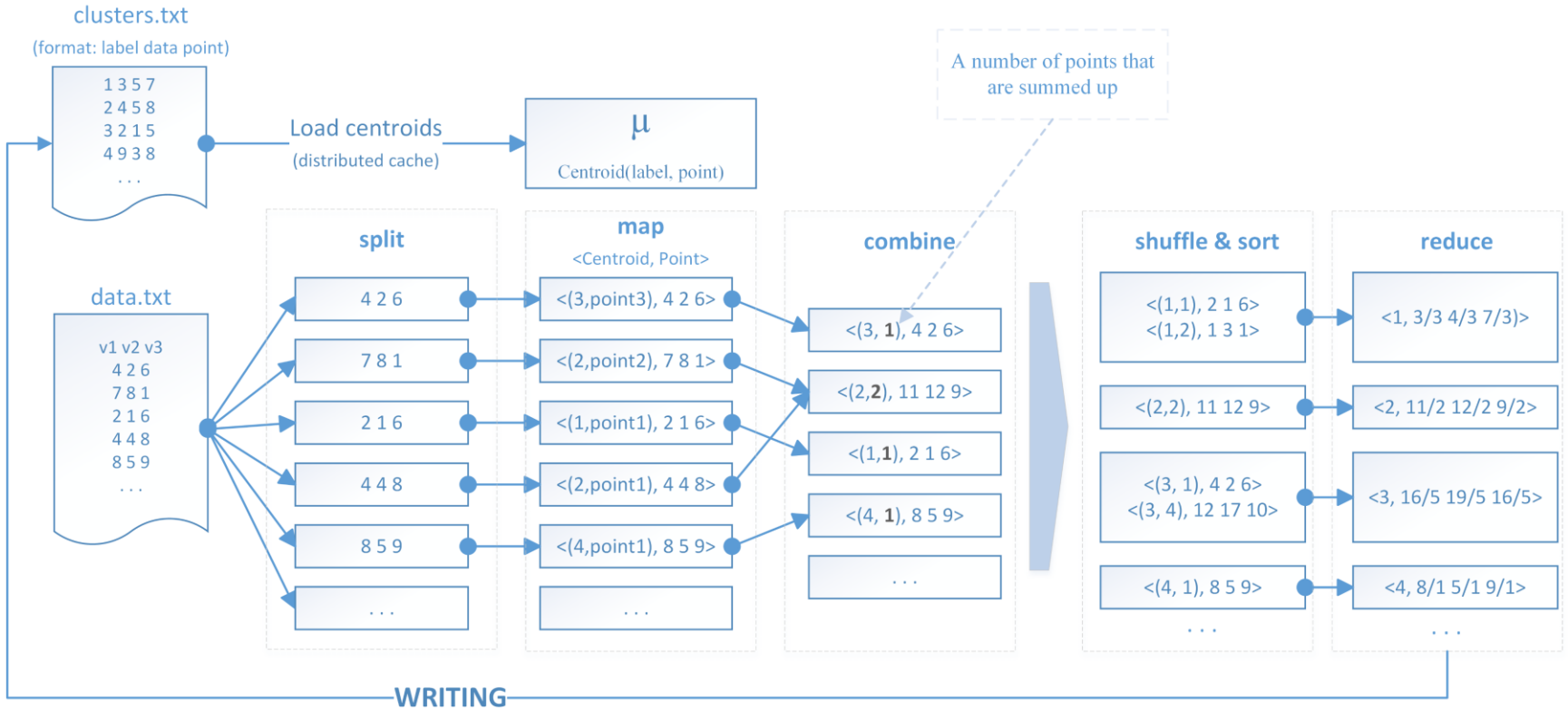
3.3 Thuật toán

- Tính hàm phân phối tích lũy:

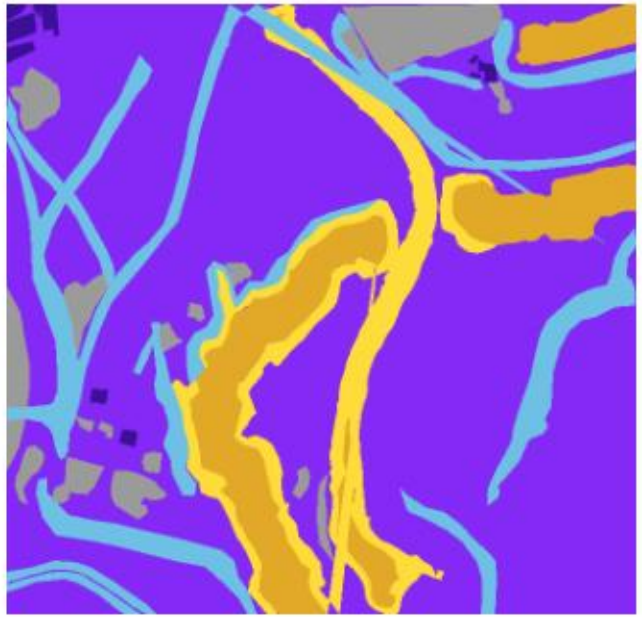
$$C(\vec{x}) = \sum_{i=1}^{j} \left(p(\vec{x}_i)\right), \qquad \vec{x}_j \leq \vec{x}$$

- Chọn ngẫu nhiên 1 điểm $r \in [0,1]$
- Dùng binary search để tìm khoảng $C(\vec{x})$ có chứa r
- Chọn $\vec{x}$ làm một centroid mới

4. Triển khai trên MapReduce


clusters.txt
(format: label data point)
1 3 5 7
2 4 5 8
3 2 1 5
4 9 3 8
. . .
Load centroids
(distributed cache)
μ
Centroid(label, point)
A number of points that are summed up
data.txt
v1 v2 v3
4 2 6
7 8 1
2 1 6
4 4 8
8 5 9
. . .
split
map
<Centroid, Point>
combine
shuffle & sort
reduce
4 2 6
7 8 1
2 1 6
4 4 8
8 5 9
. . .
<(3,point3), 4 2 6>
<(2,point2), 7 8 1>
<(1,point1), 2 1 6>
<(2,point1), 4 4 8>
<(4,point1), 8 5 9>
. . .
<(3, 1), 4 2 6>
<(2,2), 11 12 9>
<(1,1), 2 1 6>
<(4, 1), 8 5 9>
. . .
<(1,1), 2 1 6>
<(1,2), 1 3 1>
<(2,2), 11 12 9>
<(3, 1), 4 2 6>
<(3, 4), 12 17 10>
<(4, 1), 8 5 9>
. . .
<1, 3/3 4/3 7/3)>
<2, 11/2 12/2 9/2>
<3, 16/5 19/5 16/5>
<4, 8/1 5/1 9/1>
. . .
WRITING

5. Xử lý hình ảnh sau khi đã phân cụm

Ảnh gốc	Ảnh masking	Ảnh thu được
		

6. Đánh giá thuật toán

6.1 Dice

So sánh kết quả phân cụm với ảnh mask:

- Các ảnh phân đoạn được chuyển thành ảnh nhị phân (với vùng trắng đại diện cho khu vực có thể xây dựng).
- Sử dụng **Dice Similarity Coefficient (DSC)** để so sánh ảnh phân đoạn với ảnh mask, đánh giá mức độ chính xác của thuật toán phân cụm.
- Công thức tính Dice Similarity Coefficient là:

$$\text{Dice}(A, B) = \frac{2|A \cap B|}{|A| + |B|}$$

Trong đó:

- A là tập hợp các điểm ảnh trắng trong ảnh phân đoạn
- B là tập hợp các điểm ảnh trắng sau khi nhị phân hóa trong ảnh mask
- |A∩B| là số điểm ảnh chung giữa hai ảnh.
- |A| và |B| là tổng số điểm ảnh trong mỗi ảnh.

6.2 Kết quả

- **comparison_results.txt** chứa kết quả so sánh của 16 cặp ảnh. Kết quả phân tích cho thấy độ giống nhau cao nhất giữa ảnh phân đoạn và ảnh mask là 87.41%, độ giống nhau trung bình là 67.15%, chứng minh hiệu quả của thuật toán phân cụm KMeans trong việc xác định các khu vực có thể xây dựng từ ảnh vệ tinh.

7 Kết luận

- Kết hợp thuật toán K-Means với MapReduce giúp phân cụm ảnh vệ tinh hiệu quả, đạt độ tương đồng trung bình 67.15% so với ảnh mask thực tế.
- Mặc dù có kết quả khả quan, thuật toán gặp khó khăn với ảnh nhiễu và khu vực phức tạp, cần cải thiện phương pháp tiền xử lý hoặc thuật toán K-Means.
- Phương pháp này có tiềm năng ứng dụng trong phân tích ảnh vệ tinh quy mô lớn, nhờ khả năng mở rộng và hiệu suất cao từ MapReduce.