

Bùi Văn Khải
22022574

YOYTUBE DATA .ANALYSIS



GIỚI THIỆU

● Tại sao lại chọn đề tài ?

Trong thời đại công nghệ thông tin bùng nổ, dữ liệu lớn (Big Data) đã trở thành một tài sản quý giá đối với các doanh nghiệp và tổ chức. Việc khai thác và phân tích dữ liệu lớn không chỉ mang lại giá trị kinh tế mà còn đóng vai trò quan trọng trong các lĩnh vực như khoa học, giáo dục và giải trí. Một trong những thách thức lớn nhất khi làm việc với dữ liệu lớn là tìm ra các phương pháp hiệu quả để xử lý và phân loại thông tin trong khối lượng dữ liệu khổng lồ.

Chính vì lý do đó, em đã lựa chọn bộ dữ liệu YouTube để minh họa cho "Thuật toán ID3 & lập trình MapReduce hóa ID3 trong phân lớp dữ liệu"

Bộ dữ liệu YouTube, với hàng triệu bản ghi chứa thông tin về lượt xem, danh mục và tiêu đề video, là một minh chứng thực tiễn về quy mô và độ phức tạp của dữ liệu lớn.



Ý TƯỞNG

MapReduce

- Tính tổng lượt xem (views) cho từng categoryId.
- Đếm số lượng video thuộc mỗi categoryId.
- Kết nối dữ liệu JSON để thêm tên danh mục (title) vào kết quả.



```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import java.io.IOException;

public class CategoryMapper extends Mapper<Object, Text, Text,
IntWritable> {
    private final IntWritable views = new IntWritable();
    private final Text categoryId = new Text();
    private boolean isHeader = true; // Flag to track the header row

    @Override
    protected void map(Object key, Text value, Context context)
throws IOException, InterruptedException {
        if (isHeader) {
            isHeader = false;
            return;
        }

        String[] fields = value.toString().split(",");
        try {
            if (fields.length > 8) {
                categoryId.set(fields[5]); // category_id is the 6th
                views.set(Integer.parseInt(fields[8])); // views is
                context.write(categoryId, views);
            }
        } catch (NumberFormatException e) {
            System.err.println("Skipping malformed row: " +
value.toString());
        }
    }
}
```

Ý TƯỞNG

● Giai đoạn Map

- Đọc dữ liệu từ file CSV.
- Trích xuất các cặp categoryId và views từ từng video.
- Output của mapper là các cặp: (categoryId, views)

```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import java.io.IOException;

public class CategoryMapper extends Mapper<Object, Text, Text,
IntWritable> {
    private final IntWritable views = new IntWritable();
    private final Text categoryId = new Text();
    private boolean isHeader = true; // Flag to track the header row

    @Override
    protected void map(Object key, Text value, Context context)
throws IOException, InterruptedException {
        if (isHeader) {
            isHeader = false;
            return;
        }
        String[] fields = value.toString().split(",");
        try {
            if (fields.length > 8) {
                categoryId.set(fields[5]); // category_id is the 6th
                views.set(Integer.parseInt(fields[8])); // views is
                context.write(categoryId, views);
            }
        } catch (NumberFormatException e) {
            System.err.println("Skipping malformed row: " +
value.toString());
        }
    }
}
```

Ý TƯỞNG

● Giai đoạn Shuffle and Sort

- Gom nhóm tất cả các giá trị có cùng categoryId.
- Sắp xếp dữ liệu theo categoryId.

```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;

import java.io.File;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

public class CategoryReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
    private final IntWritable result = new IntWritable();
    private Map<String, String> categoryMap = new HashMap<>();

    @Override
    protected void setup(Context context) throws IOException {
        // Load category JSON from file
        String jsonFilePath =
context.getConfiguration().get("category.json.path");
        ObjectMapper mapper = new ObjectMapper();
        JsonNode root = mapper.readTree(new File(jsonFilePath));
        for (JsonNode item : root.get("items")) {
            String id = item.get("id").asText();
            String title = item.get("snippet").get("title").asText();
            categoryMap.put(id, title);
        }
    }

    @Override
    protected void reduce(Text key, Iterable<IntWritable> values,
Context context) throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
    }
}
```

Ý TƯỞNG

● Giai đoạn Reduce

- Nhận các cặp <category_id, [views1, views2, ...]> từ Mapper.
- Tính tổng views cho từng category_id.
- Ánh xạ category_id sang category_name (hoặc "Unknown" nếu không tìm thấy).



THUẬT TOÁN ID3

● Áp dụng

1. Tính toán Entropy:

- Entropy được tính dựa trên sự phân bố của các danh mục.

2. Tính Information Gain (Độ lợi thông tin):

- Xác định thuộc tính nào (tổng lượt xem hoặc số lượng video) giúp phân tách dữ liệu hiệu quả nhất.

Xây dựng cây quyết định:

- Chọn thuộc tính có Information Gain cao nhất làm nút gốc.
- Phân nhánh dữ liệu dựa trên giá trị của thuộc tính này.
- Lặp lại quá trình với các tập con dữ liệu cho đến khi đạt điều kiện dừng.

3. Kết hợp MapReduce và ID3



THUẬT TOÁN ID3

● Ứng dụng thực tiễn

- Phân tích xu hướng nội dung: Xác định các danh mục video phổ biến dựa trên lượt xem.
- Hỗ trợ chiến lược marketing: Tập trung vào các danh mục có tiềm năng cao.
- Cá nhân hóa nội dung: Gợi ý danh mục phù hợp cho người dùng hoặc nhà sáng tạo nội dung.



THANK YOU