Report del progetto di Big Data: Analisi delle visualizzazioni dei video YouTube di tendenza

Lorenzo Chiana - Mat. 0000880396

February 8, 2021

Contents

1 Introduzione

1.1 Descrizione del dataset

- Questo dataset è una raccolta giornaliera dei video di tendenza di YouTube di diverse nazioni. Secondo la rivista Veriety per determinare i video più di tendenza dell'anno, YouTube utilizza una combinazione di fattori tra cui il numero di visualizzazioni, condivisioni, commenti e "Mi piace".
- Link al sito dove è pubblicato il dataset (`https://www.kaggle.com/rsrishav/youtube-trending-video-dataset`).
- Link diretti per scaricati i singoli file:
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=BR_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=CA_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=DE_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=FR_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=GB_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=KR_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=MX_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=US_youtube_trending_data.csv`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=BR_category_id.json`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=CA_category_id.json`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=DE_category_id.json`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=FR_category_id.json`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=GB_category_id.json`
  - `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=KR_category_id.json`

- `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=MX_category_id.json`
- `https://www.kaggle.com/rsrishav/youtube-trending-video-dataset?select=US_category_id.json`

1.1.1 Descrizione dei file

1.1.2 File CSV

Ogni file csv contiene i video di tendenza del relativo paese e contiene di seguenti campi:

- **video_id**, identificato del video;
- **title**, titolo del video;
- **publishedAt**, data di pubblicazione del video;
- **channelId**, identificativo del canale;
- **channelTitle**, nome del canale;
- **categoryId**, identificativo della categoria;
- **trending_date**, data in cui il video è diventato virale;
- **tags**, elenco dei tag;
- **view_count**, numero di visualizzazioni;
- **likes**, numero dei "Mi piace";
- **dislikes**, numero dei "Non mi piace";
- **comment_count**, numero di commenti;
- **thumbnail_link**, link della thumbnail;
- **comments_disabled**, flag che indica se i commenti sono stati disattivati;
- **ratings_disabled**, flag che indica se è stata disattivata la possibilità di mettere "Mi piace" o "Non mi piace";
- **description**, descrizione del video.

1.1.3 File JSON

Ogni file json contiene la relazione tra l'identificativo del video e il relativo nome e ha la seguente struttura:
root{

- **kind**
- **etag**
- **items** [
  - **{**
    - **kind**
    - **etag**
    - **kind**
    - **id**
    - **snippet{**
      - **title**
      - **assignable**
      - **channelId**
    - **}**
  - **}**

}

2 Data preparation

File path on HDFS:

- `/user/lchiana/exam/dataset/BR_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/CA_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/DE_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/FR_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/KR_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/MX_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/US_youtube_trending_data.csv`
- `/user/lchiana/exam/dataset/Categoy_id_flat.json`

2.1 Pre-processing

Una delle esigenze per questo progetto è quella di sapere a che nome alfanumerico corrisponde ogni identificativo di categoria. Di conseguenza le sole informazioni che servono all'interno del file json sono relative ai campi *id* e *title*. Per facilitare la lettura di questi file per la fase implementativa si sono andati ad estrarre questi campi significativi in maniera automatizzata tramite comando jq. Ciò ha portato in evidenza che tutti i file json dei vari paesi contenevano le medesime informazioni nei relativi campi estratti ad eccezione di quello relativo agli Stati Uniti che conteneva un identificativo in più. Perciò, data l'uguaglianza dei vari file, si è scelto di tenerne solo uno e, nello specifico, quello risultatante dal file json relativo agli US.

```
jq -c '(.items[] | {id, category: .snippet.title})'
US_category_id.json > Category_id_flat.json
```

Ulteriore problematica si è presentata nella lettura dei file csv data la presenza di newlines all'interno di alcuni campi. Di conseguenza si è ricorsi ad un comando awk per andare a pulire i vari file dalla presenza di \n e \r.

```
awk 'BEGIN{FS=OFS=","} {for(i=16;i<=NF;i++){
gsub(/\\n/,"",$i); gsub(/\r/,"",$i); gsub(/\n/,"",$i)}}1'
BR_youtube_trending_data.csv > ./new/BR_youtube_trending_data.csv
```

3 Jobs

3.1 Job #1: Numero medio di visualizzazioni per ogni categoria di video

Con questo job si vuole andare a calcolare il numero medio di visualizzazioni dei video andati in tendenza suddiviso per categoria.

3.1.1 MapReduce

- File/Table in input:
  - `/user/lchiana/exam/dataset/BR_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/CA_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/DE_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/FR_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/KR_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/MX_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/US_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/Categoy_id_flat.json`
- File/Table in output:

  - `/user/lchiana/exam/outputs/mapreduce/output1`
  - `/user/lchiana/exam/outputs/mapreduce/output2`
  - `/user/lchiana/exam/outputs/mapreduce/output3`
- Tempo di esecuzione:
  - Job1:
    - Elapse: 1mins, 24sec
    - Average Map Time: 43sec
    - Average Shuffle Time: 7sec
    - Average Merge Time: 0sec
    - Average Reduce Time: 0sec
  - Job2:
    - Elapse: 1mins, 1sec
    - Average Map Time: 8sec
    - Average Shuffle Time: 8sec
    - Average Merge Time: 0sec
    - Average Reduce Time: 0sec
  - Job3:
    - Elapse: 51sec
    - Average Map Time: 8sec
    - Average Shuffle Time: 7sec
    - Average Merge Time: 0sec
    - Average Reduce Time: 0sec
- Quantità di risorse:
  - Job1:
    - File System Counters
      - FILE: Number of bytes read: 1201569
      - FILE: Number of bytes written: 6461080
      - HDFS: Number of bytes read: 332575241
      - HDFS: Number of bytes written: 156
      - HDFS: Number of read operations: 81
      - HDFS: Number of write operations: 40
  - Job2:
    - File System Counters
      - FILE: Number of bytes read: 1295
      - FILE: Number of bytes written: 6078508
      - HDFS: Number of bytes read: 7803
      - HDFS: Number of bytes written: 317

            - HDFS: Number of read operations: 123
            - HDFS: Number of write operations: 40
    - Job3:
        * File System Counters
            - FILE: Number of bytes read: 746
            - FILE: Number of bytes written: 5924693
            - HDFS: Number of bytes read: 3477
            - HDFS: Number of bytes written: 317
            - HDFS: Number of read operations: 120
            - HDFS: Number of write operations: 40

- YARN application history:
    - `http://isi-vclust0.csr.unibo.it:19888/jobhistory/job/job_1610205429022_0195`
    - `http://isi-vclust0.csr.unibo.it:19888/jobhistory/job/job_1610205429022_0196`
    - `http://isi-vclust0.csr.unibo.it:19888/jobhistory/job/job_1610205429022_0197`

- Descrizione dell'implementazione:
    - First Mapper
        * suddivide i record in input
        * prende i valori relativi a categoryId e view_count filtrando i valori che corrispondono al nome della tabella
        * versione schematica:
            - input: file.csv
            - output: categoryId →view_count
    - First Reducer
        * prende le varie visualizzazioni
        * calcola la media
        * versione schematica:
            - input: categoryId →[view_count, view_count, ...]
            - output: categoryId →avg_views
    - Second Mapper 1
        * lettura basilare dell'output del primo reducer come coppia chiave valore
        * versione schematica:
            - input: categoryId →avg_views
            - output: categoryId →avg_views

- Second Mapper 2
  - parsing del file json per ricavare i valori dell'id e del nome della categoria
  - aggiunge l'etichetta "*#join#*" prima del valore del nome in modo tale che, durante la fase di reduce, ci sia modo di distinguere i due tipi di valori in input.
  - versione schematica:
    - input: record json
    - output: id →#join#category
- Second Reducer
  - estrae i valori che gli arrivano in input dai due mapper
  - identifica da quale mapper arriva tale valore (grazie la presenza o meno dell'etichetta "*#join#*")
  - se il valore gli arriva da Second Mapper 2 lo imposta come chiave per l'output, altrimenti come valore
  - versione schematica:
    - input: categoryId →avg_views o id →#join#category
    - output: category →avg_views
- Third Mapper
  - inverte la chiave e il valore dell'output del secondo reducer utile per il sorting
  - versione schematica:
    - input: category →avg_view
    - output: avg_view →category
- Sorting
  - ordinamento decrescente delle chiavi dell'output del terzo mapper
- Third Reducer
  - inverte la chiave e il valore dell'output del sorting
  - versione schematica:
    - input: avg_view (ordinato) →category
    - output: category →avg_view (ordinato)

- Considerazioni sulle performance:
  - il numero dei mapper è scelto di default dal framework in base agli input split
  - il numero dei reducer è scelto di default dal framework ciò si riflette sulla struttura dell'output finale che viene salvato soddiviso, in questo caso specifico, in venti file (uno per ogni reducer che il framework ha creato).

- Output:
  - Music 3844296
  - Gaming 2404676
  - Science Technology 1935558
  - Entertainment 1346296
  - Film Animation 1300239
  - People Blogs 1237091
  - Sports 1109023
  - Comedy 1076787
  - News Politics 936546
  - Howto Style 833404
  - Pets Animals 832279
  - Nonprofits Activism 810967
  - Education 661902
  - Autos Vehicles 593843
  - Travel Events 487401

3.1.2 Spark

- File/Table in input:
  - `/user/lchiana/exam/dataset/BR_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/CA_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/DE_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/FR_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/KR_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/MX_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/US_youtube_trending_data.csv`
  - `/user/lchiana/exam/dataset/Categoy_id_flat.json`
- File/Table in output:
  - `/user/lchiana/exam/outputs/spark/output`
- Tempo di esecuzione: 1.1 min
- Quantità di risorse:
  - Input 317.2 MB
  - Shuffle Read 13.2 KB

  - Shuffle Write 5.9 KB

- YARN application history: `http://isi-vclust0.csr.unibo.it:18088/history/application_1610205429022_0194/jobs/`

- Descrizione dell'implementazione:
  - creazione di un RDD per i dati estratti dai file csv
  - creazione di un RDD per i dati estratti dal file json
  - trasformazione del primo RDD prima in uno che mappa categoryId e view_count e poi in uno contenente la media delle visalizzazioni per ogni id di categoria.
  - trasformazione del secondo RDD in uno che mappa l'id di categoria e il relativo nome.
  - join dei due RDD risultati sulla base dell'id di categoria e ordinamento sulla base delle visualizzazioni medie.

- Considerazioni sulle performance:
  - gli RDD vengono resi persistenti in cache nel momento in cui hanno trasformazioni significative.
  - il numero di partizioni di un RDD è deciso di default dal framework ciò si riflette sulla struttura dell'output finale che viene salvato soddiviso in più parti.

- Output:
  - (Music,3844296)
  - (Gaming,2404676)
  - (Science Technology,1935558)
  - (Entertainment,1346296)
  - (Film Animation,1300239)
  - (People Blogs,1237091)
  - (Sports,1109023)
  - (Comedy,1076787)
  - (News Politics,936546)
  - (Howto Style,833404)
  - (Pets Animals,832279)
  - (Nonprofits Activism,810967)
  - (Education,661902)
  - (Autos Vehicles,593843)
  - (Travel Events,487401)