Helping prospective Youtubers choose video content

Team: suppala2
sboddire

ABSTRACT

YouTube is one of the most popular platforms for making money online. The biggest challenge that every youtuber face is “which videos grab more user’s attention(views)?” more specifically, what should be the video content so that the video can obtain more user views. In order to answer this question, we have made big data analytics on Kaggle dataset (YouTube’s trending video statistics) alternatively, we can also collect data in real-time using YouTube data API. Our Kaggle dataset contains information such as video_id, trending_date, views, likes, dislikes, title, channel, tags, category, etc. we provide a complete solution about choosing actual video content using both tags column (example: NBA| "Basketball"| "Sports") and views column of dataset as input and running an algorithm similar to word count algorithm in MR where mappers split the tags by ‘|’ character, associate view count as initial counter and reducers reduce the similar tags. This algorithm outputs tag view counts and when visualized though tools like tableau youtubers can decide which video content to choose. We use spark framework to run this algorithm because spark provides faster data processing, minimal implementation code compared to Hadoop. Also, we can perform real-time data processing using spark.

Problem Statement

Youtubers earn money from advertisers based on their video views and youtubers often ponder with this question “what type of videos should I make in order to get more user views?”. youtubers should always be aware of the current video trend and analysing YouTube’s big data helps youtubers cope up with the current trend and earn money through video views.

However, YouTube contains large amounts of real-time data (big data), and its difficult to analyse this large data with normal analytical tools so

we chose spark for data analytics because spark offers real time data streaming and faster data processing.

Data Crawling

The first step in solving this problem is data crawling. Trending YouTube videos data can be crawled through

- YouTube Data API
- Datasets available in online resources like Kaggle.

We collected big data set from Kaggle which contains 6 months of trending videos data on daily basis. We have also crawled small data set from YouTube API due to time constraint of this project.

Data source

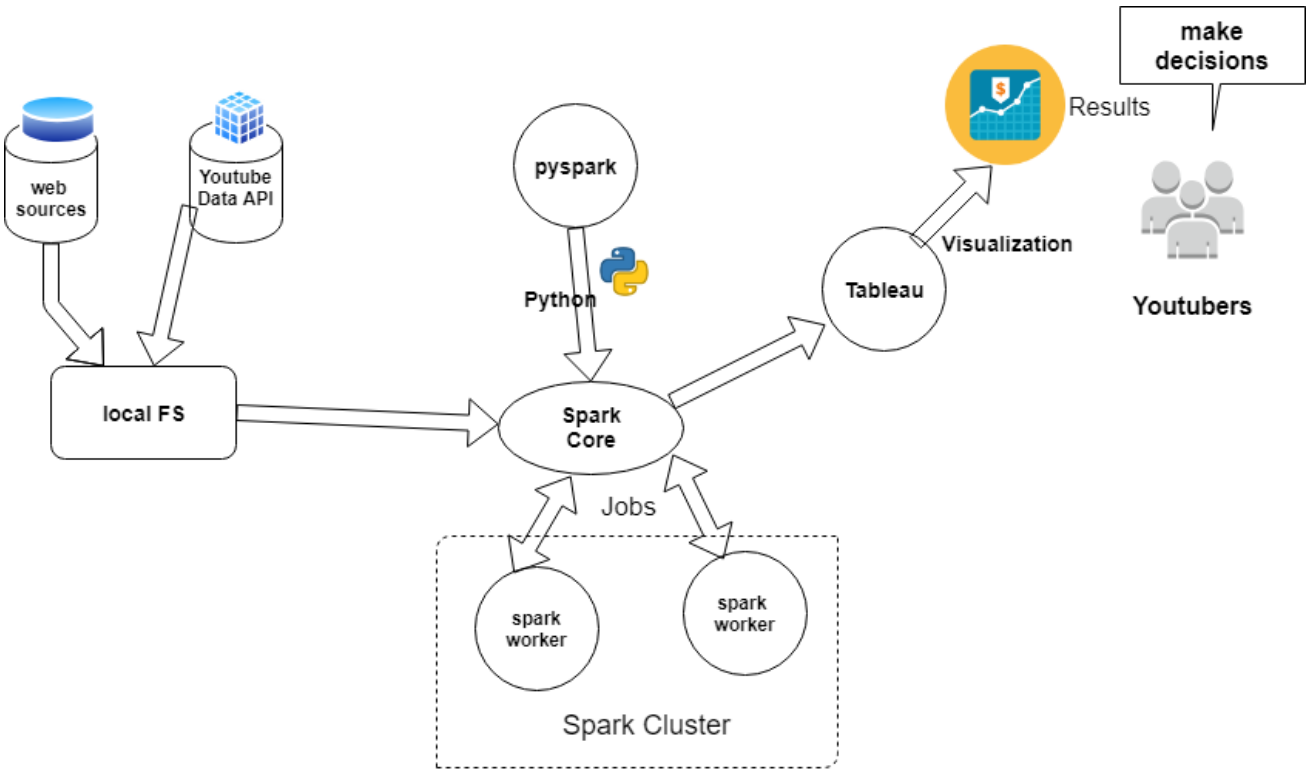
https://www.kaggle.com/datasnaek/youtube-new#USvideos.csv---trending

Number of rows : 50,000
Columns : video_id, trending_date, title, channel_title, category_id, publish_time, **tags, views**, likes, dislikes, comment_count, thumbnail_link, comments_disabled, ratings_disabled, video_error_or_removed, description

Programming Model

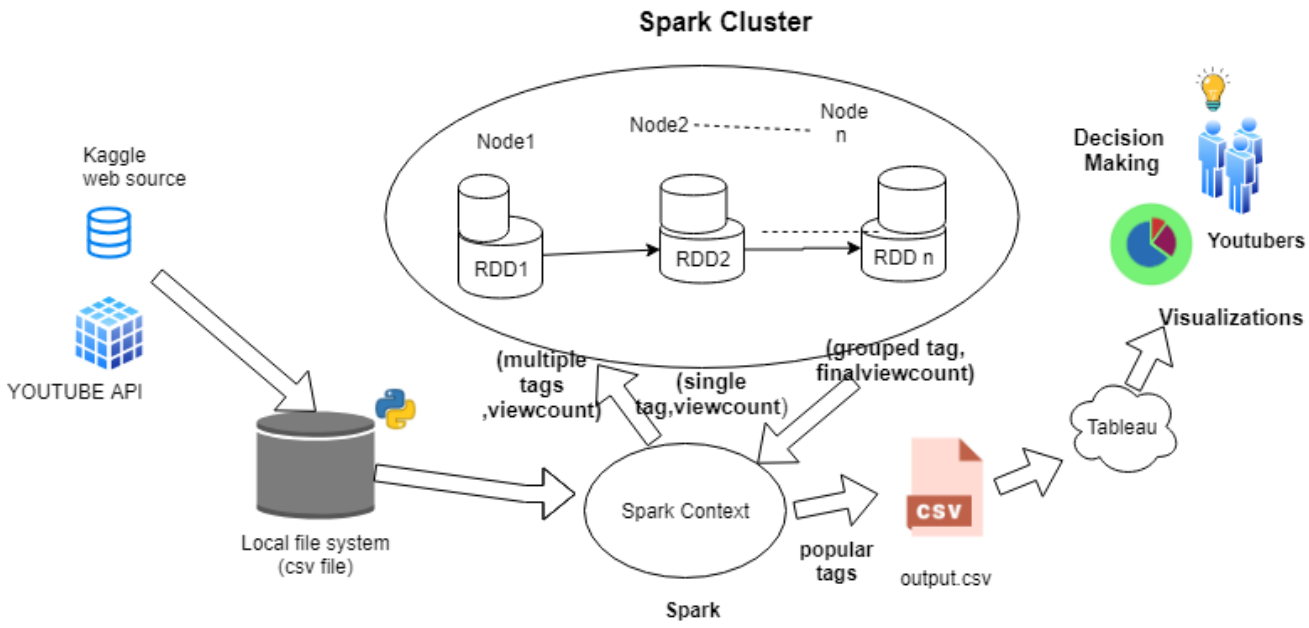
We used python as our programming language because of its simplicity and ease of use. In order to code in python, we used pyspark API provided by Apache spark.

Architecture diagram



- ✓ The data extracted from YouTube API and Kaggle dataset are saved to local File system.
- ✓ Data from local file system is loaded to spark engine using pyspark API (python).
- ✓ Spark sends data and tasks (map, reduce) to worker nodes to do parallel processing. Map and reduce tasks for our use case is like wordcount algorithm.
- ✓ Finally spark sends output back to local system where output is fed to tableau for visualization and the results are used for decision making.

Data Pipeline



- Data collected from different sources are saved locally and fed to spark.
- Spark transmits data into RDDS to its worker nodes. All the workers nodes execute their tasks and sends the sorted output back to local fs.
- Outputs are visualised in tableau for decision making.

Algorithm

1. Load dataset to RDD in spark.
2. Splits tags column by “|” and associate view count for each tag.
3. Apply log on view count and reduce will group similar tags by adding their respective view counts.
4. Sort data by view count and write top 35 tags and counts to disk as csv.

CODE (implemented by our self)

```
import findspark
findspark.init()
from pyspark.context import SparkContext
from pyspark.sql.session import SparkSession
from operator import add
import math
#Creating Spark Context
sc = SparkContext.getOrCreate()
#Creating Spark Session
spark = SparkSession(sc)
#Reading data set(csv file),dropping null values and selecting only tabs and views
column
tags_views = spark.read.csv("file:///C:\\Users\\saikr\\Desktop\\DIC\\USvideos.csv",
inferSchema = True, header = True).dropna().select("tags","views")
#Mapping by splitting tags with "|" character,
def tags_split(x):
    tags=x["tags"].split("|")
    result=[]
    for every in tags:
        if not x["views"].isdigit() or every==None:
            continue
#Associating view count as counter and applying log on view count because views
count could be large
        result.append((every.strip("\"").lower(),math.log(int(x["views"]))))
    return tuple(result)
rdd1=tags_views.rdd.flatMap(tags_split).reduceByKey(add)#reduce by similar tags and
adding its view count
#Top Tags are queried from RDD by Sorting RDD's in descending order of view count
toptags=rdd1.takeOrdered(35, key = lambda x: -x[1])
df=spark.createDataFrame(toptags)
#Writing back to Disk
df.repartition(1).write.csv(path="file:///C:\\Users\\saikr\\Desktop\\DIC\\trending.c
sv")
#Stopping Spark session
spark.stop()
```

- This code is implemented with python and well commented.
- We read data from local file system and selected only “tags” and “views” column into RDD.
- We then split tags by “|” character, associate each tag with view count and reduce by same key. We apply log on views because they could sum to large values, so we minimise them with log.
- Finally, we sort the top trending tags by view count and write back to disk.

VISUALIZATION

- This visualization is implemented with tableau.
- Top 35 video tags are sorted in spark and fed to tableau.

Conclusions:

- From this word cloud visualization, it is evident that during the time data is collected “comedy and funny” videos grabbed more user views.
- Any youtuber can now choose video content with the help of trend visualized.
- As time passes, the trend changes so it is always better to do real-time data processing with spark streaming API to cope up with the trend.

Summary:

- ❖ Spark took less than 5 minutes to process 50000 records which is incredibly fast compared to other tools.
- ❖ We can analyse both real-time and periodic data using spark.
- ❖ Spark provides various inbuilt programming capabilities; in our use case we use pyspark.
- ❖ Data analysed through spark can be fed to visualization tools like tableau for gaining knowledge and decision making.

References:

1. https://medium.com/@dvainrub/how-to-install-apache-spark-2-x-in-your-pc-e2047246ffc3
2. https://www.youtube.com/watch?v=639JCua-bQU
3. https://spark.apache.org/docs/1.2.0/programming-guide.html
4. https://www.codementor.io/jadianes/spark-python-rdd-basics-du107x2ra