Statistical analysis of Youtube data using Hadoop and MapReduce

ID2221 - Data intensive Computing

Project report

Raghunath Vairamuthu (rvai@kth.se)

Vaikunth Srinivasan A (vaikunth@kth.se)

Abstract

A solution to compute and analyse the statistics of youtube data is proposed here. We use Hadoop system as a reliable shared means of storage for our dataset and a platform to aid the analysis of our application. The storage is provided by HDFS (Hadoop Distributed File System) and analysis by MapReduce. Using these infrastructure, we compute various metrics on the dataset for Youtube and visualise them with ChartJS, a tool from Google. Our application reads the data, and produces the graphs in a webpage as a final result. In this paper, we solve two problem statements using the YouTube dataset – top 5 most demanded video categories (genres) with the maximum average views per video, and top 5 video uploaders on YouTube community. The goal here is to demonstrate by using hadoop concepts how data generated from YouTube can be mined and utilized to make targeted and informed decisions.

Background

With the rapid growth in the number of internet companies and the internet savvy population, today's advanced systems are generating huge volumes of data in various formats (video, audio, images, sensor data). This has given birth to a new type of data called "Big Data" which can be unstructured and also unpredictable, given the real-time nature. YouTube has billions of users and creators upload over 1.2 millions videos to YouTube everyday. Analysing and understanding the activity on such a scale would need a massively parallel and distributed system like hadoop.

Experimental Setup

We used Apache Hadoop software library as a framework for our distributed processing of Youtube data. Since Youtube dataset is unstructured, we ruled out using Hive, as it deals mainly with structured and semi-structured data. MapReduce can be implemented using plain Java, and can be very efficient than most systems if developed well. Hence, we chose MapReduce for this project. To visualise the results, we use Chart.js, a product of Google. The results can be viewed from a web browser.

Dataset:

Youtube data is available for free from Google. The unstructured dataset consists of data from approximately 3000 videos and contains 10 columns in total. The type of dat each column holds is defined below,

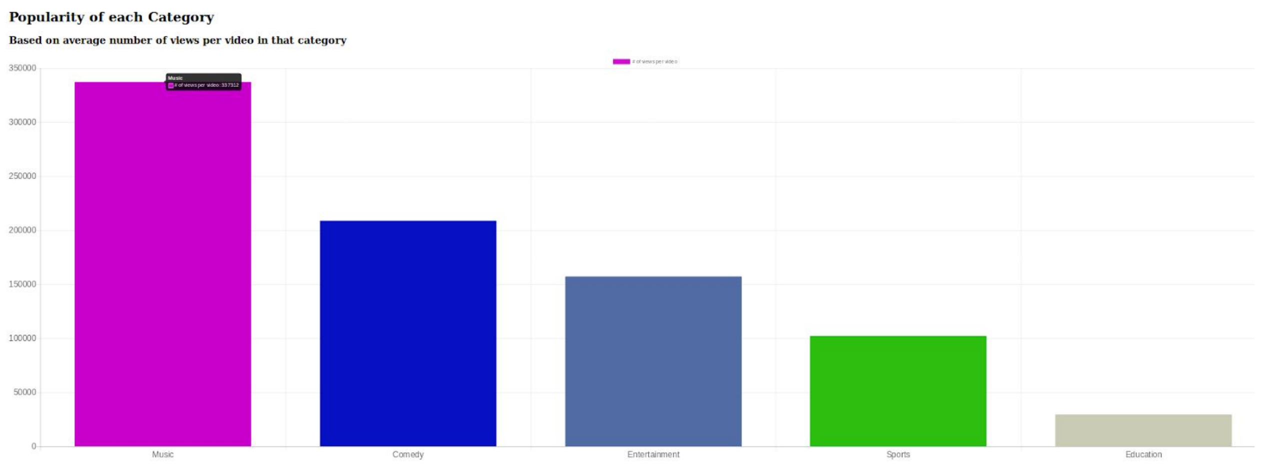
Column 1	A 11 character video ID
Column 2	Name of the uploader
Column 3	Time when the video was uploaded
Column 4	Video Category
Column 5	Length of the video
Column 6	Number of views for the video
Column 7	Rating on the video
Column 8	Number of ratings given for the video
Column 9	Number of comments on the videos
Column 10	Related video ids

Implementation

Problem 1: Determine top 5 most demanded video categories (genres) with the maximum average views per video

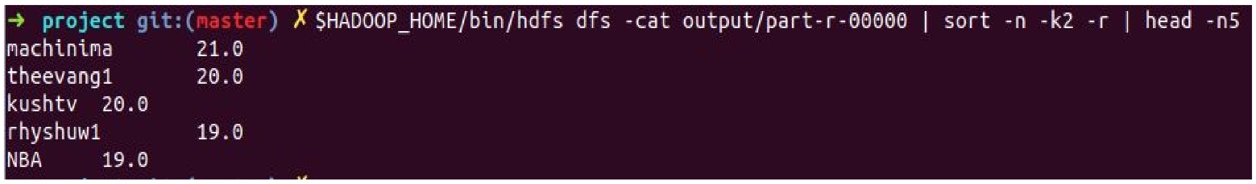
Our metric to measure the demand for a video category is by the average number of views a video in that category gets. The higher the average views per video in that category, more demanded the category is in the Youtube community.

To solve our problem, we created a class called AvgViews. The Mapper algorithm basically processes the data and maps it into key-value pairs. The key here being the category of video (Column 4) and the value being the number of views (Column 6) for that video. This key-value pair will be the output of the map method. The Reducer algorithm accepts the output from the mapper and processes it to find the sum-average of all the view counts in a particular category. The output of the Reducer is a key value pair containing the key as the name of the category, and the value being the average views per video in that category. The output is read by a bash script which then dumps a Javascript file containing a bar chart representation of the statistics of each category.



Problem 2: Determine top 5 uploaders in Youtube community.

The problem is very similar to that of the first one, but we the hashing processing of values is quite different. The output of the mapper function here is the hash that contains a key which is the name of the uploader(Column 2) and a value of 'one'. The reducer, in turn, receives these key-value pairs and sums all the values with the same key and writes a final key-value pair with that summed up value. The results for this dataset is shown below.



```
➜ project git:(master) ✗ $HADOOP_HOME/bin/hdfs dfs -cat output/part-r-00000 | sort -n -k2 -r | head -n5
machinima       21.0
theevang1       20.0
kushtv  20.0
rhyshuw1        19.0
NBA     19.0
```

Conclusion

As seen from the implementation, our application reads the data from the Youtube dataset and reports results to have better understanding of Youtube statistics, more specifically, the most demanded category and top uploaders. This is just a sample of what can be achieved by running our MapReduce algorithm on Youtube data. Our approach is scalable and also expandable to determine other useful metrics that can be used to better understand the video streaming platform. The details of the project and how to run the applications can be found in our github repository at the link below.
https://github.com/raghunath-v/data-intensive-computing/tree/master/project

References

1. MapReduce: Simplified Data Processing on Large Clusters, Jeffrey Dean and Sanjay Ghemawat, Google, Inc.

2. Youtube dataset: https://drive.google.com/file/d/0ByJLBTmJojjzR2x0MzVpc2Z6enM/view
3. Big Data analysis on Youtube using Hadoop and Mapreduce, S. Hota, IJCERT Vol. 5, Iss.4, 2018.
4. Data-Intensive Text Processing with MapReduce, Jimmy Lin et al., Morgan & Claypool Publishers, 2010. (Ch. 2-3)