CSE 5095 Big Data Analysis

Course Project Report

Statistics Analyzer of YouTube Videos with Spark

Yanyuan Qin and Renjie Chen

# 1 Introduction

YouTube is the most successful Internet website sofar. It is an American video-sharing website headquartered in San Bruno, California. The service was created by three former PayPal employees—Chad Hurley, Steve Chen, and Jawed Karim—in February 2005. YouTube has a great impact on Internet traffic nowadays, yet itself is suffering from a severe problem of scalability. The site allows users to upload, view, rate, share, add to favorites, report and comment on videos, subscribe to other users, and it makes use of WebM, H.264/MPEG-4 AVC, and Adobe Flash Video technology to display a wide variety of user-generated and corporate media videos. Available content includes video clips, TV show clips, music videos, short and documentary films, audio recordings, movie trailers and other content such as video blogging, short original videos, and educational videos. Most of the content on YouTube has been uploaded by individuals, but media corporations including CBS, the BBC, Vevo, and Hu lu offer some of their material via YouTube as part of the YouTube partnership program. Therefore, understanding the characteristics of YouTube and similar sites is essential to network traffic engineering and to their sustainable development.

In viewing of its history, since the domain name www.youtube.com was activated on February 14, 2005, and the website was developed over the subsequent months, the first YouTube video, titled Me at the zoo, was uploaded on April 23, 2005. The site grew rapidly, it reached 8 million views a day by the time it launched officially on December 15, 2005. And in July 2006 the company announced that more than 65,000 new videos were being uploaded every day, and that the site was receiving 100 million video views per day. YouTube has become the dominant provider of online video in the United States, with a market share of around 43% and more than 14 billion views of videos in May 2010. In May 2011, 48 hours of new videos were uploaded to the site every minute,which increased to 60 hours every minute in January 2012, 100 hours every minute in May 2013, 300 hours every minute in November 2014,and 400 hours every minute in February 2017. In May 2010, YouTube videos were watched more than two billion times per day. This increased to three billion in May 2011, and four billion in January 2012. In February 2017, one billion hours of YouTube was watched every day. The site has 800 million unique users a month.It is estimated that in 2007 YouTube consumed as much bandwidth as the entire Internet in 2000. According to third-party web analytics providers, Alexa and SimilarWeb, YouTube is the second-most visited website in the world, as of December 2016; SimilarWeb also lists YouTube as the top TV and video website globally, attracting more than 15 billion visitors per month.

We are wondering what is inside this “giant”, we want to answer the following questions:

1. What is the statistics of YouTube videos?
2. What is the relationship between user like or dislike vs video length, video category and video comments?
3. How to get the video info?
4. How to analysis the data via the “state-of-art” technique?

In this work, we have the following contributions:

1. Build a platform with YouTube API to collect the data from YouTube server.
2. Analysis the data via spark.
3. Got statistics features of YouTube video and user behaviors.
4. Analyze the relationship between YouTube metrics. We found that the linear relation among like, dislike and comments.

To this end, we have crawled the YouTube site during one week, with more than 500,000 YouTube video’s data collected. Based on the data, we use Apache Spark to perform reasonable and in-hand statistical analysis in order to see the meaningful results inside of the data.

# 2 Apache Spark

Apache Spark is open-source and useful for cluster-computing, which is originally developed at the University of California, Berkeley's AMPLab. Apache Spark provides programmers with an application programming interface centered on a data structure called the resilient distributed dataset (RDD), a read-only multiset of data items distributed over a cluster of machines, that is maintained in a fault-tolerant way. It is Spark’s main abstraction.

It was developed in response to limitations in the MapReduce cluster computing paradigm, which forces a particular linear dataflow structure on distributed programs: MapReduce programs read input data from disk, map a function across the data, reduce the results of the map, and store reduction results on disk. See Fig. 1. Spark's RDDs function as a working set for distributed programs that offers a (deliberately) restricted form of distributed shared memory.

Since its particular structure, it can be reused efficiently across parallel operations. The availability of RDDs facilitates the implementation of both iterative algorithms, that visit their dataset multiple times in a loop, and interactive/exploratory data analysis. It can even automatically recover from node failures.

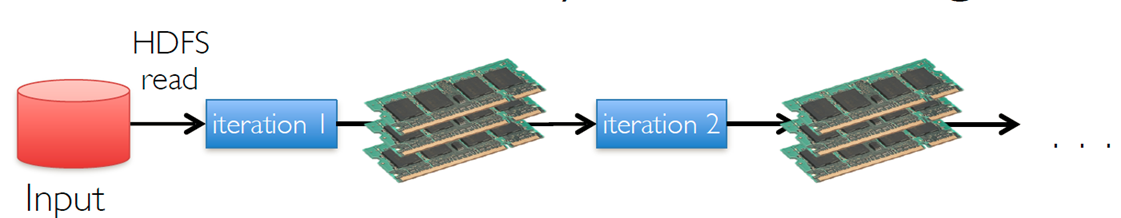


Fig1. Spark

Another abstraction in Spark provides shared variables that can be used in parallel operations. When Spark runs a function in parallel as a set of tasks on different nodes, it ships a copy of each variable used in the function to each task. It supports two types of shared variables: broadcast variables, which can be used to cache a value in memory on all nodes, and accumulators, which are variables that are only “added” to, such as counters and sums. It requires a cluster manager and a distributed storage system. Spark also supports a pseudo-distributed local mode, usually used only for development or testing purposes, where distributed storage is not required and the local file system can be used instead; in such a scenario, Spark is run on a single machine with one executor per CPU core. As our dataset is not that huge, here we are actually using the pseudo-distributed local mode.

Spark has several cores. Spark Core is the foundation of the overall project. It provides distributed task dispatching, scheduling, and basic I/O functionalities, exposed through an application programming interface (for Java, Python, Scala, and R) centered on the RDD abstraction. It mirrors a functional model of programming. Spark is a very powerful software, not only because it has multiple cores for use, it contains several powerful program invokes parallel operations. These operations, like map, filter, reduce or join work on RDD and produce new RDDs. RDDs can contain any type of Python, Java, or Scala objects.

Spark even supports Spark SQL and DataFrames, Spark streaming, build-in machine-learning, graph methods and some other third party projects.

### 2.1 Spark SQL

Spark SQL is useful for working with structured data. It can query structured data inside Spark programs, using either SQL or a familiar DataFrame API. It is usable in Java, Scala, Python and R and it contains a data abstraction called DataFrames, which provides support for structured and semi-structured data. DataFrames and SQL also provide a common way to access a variety of data sources, including Hive, Avro, Parquet, ORC, JSON, and JDBC. It can even join data across these sources. SQL of Spark SQL allows command-lines. After running SQL from within another programming language the results will be returned as a Dataset/DataFrame.

In Spark SQL, it contains dataset and dataframes. A Dataset is a distributed collection of data. It provides the benefits of RDDs and the benefits of Spark SQL’s optimized execution engine. It can be constructed from JVM objects and then manipulated using functional transformations (map, filter, etc.). A DataFrame is a Dataset organized into named columns. It can be constructed from a wide array of sources: structured data files, tables in Hive, external databases, or existing RDDs. More generally, Spark SQL can run SQL Queries programmatically and contains rich library of functions including string manipulation, date arithmetic, common math operations and more.

### 2.2 Spark Machine Learning Library Guide

Spark Machine Learning Library Guide(MLlib) is the build-in package of Spark. It is a distributed machine learning framework on top of Spark Core. Due in large part to the distributed memory-based Spark architecture, it is as much as nine times as fast as the disk-based implementation. Its goal is to make practical machine learning scalable and easy. Many common machine learning and statistical algorithms have been implemented and are shipped with MLlib which simplifies large scale machine learning pipelines:

1. Descriptive statistics, hypothesis testing, sampling, simulation;
2. ML Algorithms: common algorithm like support vector machines, logistic regression, linear regression, decision trees, naive Bayes classification, k-means, and Latent Dirichlet Allocation (LDA) clustering;
3. Feature extraction, transformation, dimensionality reduction( singular value decomposition (SVD), and principal component analysis (PCA)), and selection.

There are two different MLlib, the RDD-based APIs and the DataFrame-based API. In the future, the RDD-based APIs may be deprecated and the DataFrames-based API will be extended. From the trend, it is easy to find out from both Spark SQL and MLlib that DataFrame-based API is the type of data in Spark easier to analysis the data.

# 3 YouTube Data Collection

In order to get the video data, we will develop a crawler to fetch the data automatically. Since we are interact with YouTube, we will use their API to build the crawler.

### 3.1 YouTube API

YouTube API includes YouTube Data API, YouTube Analytics API, and YouTube Live Streaming API. Here in order to get the raw data, we are going to use the YouTube Data API. The meta data we collected is shown in Table1.

|  |  |
| --- | --- |
| VideoID | 72UO0v5ESUo |
| Upload Date | 4445 |
| ChannelID | UCLp8RBhQHu9wSsq62j\_Md6A |
| Video Duration (s) | 229 |
| Category ID | 10 (Music) |
| View Count | 88,218,239 |
| Comment Count | 240,216 |
| Like Count | 2,470,987 |
| Dislike Count | 295,415 |
| Related Video | HFI0CwQFStY, kJQP7kiw5Fk... |

Table 1. Meta data for each YouTube video

The Related Video ID can be accessed via youtube.search().list(). The other video information is collected by parse the response of youtube.videos().list().

It worth to mention that when users build the application based on YouTube API, they should go to Google Developer Console to (1) create a new application (2) enable YouTube API (3) generate the Authentication Keys for the application.

The data collection platform is shown in Fig. 2. The client run the crawler and send request to YouTube server, parse the response and store the result into Database.

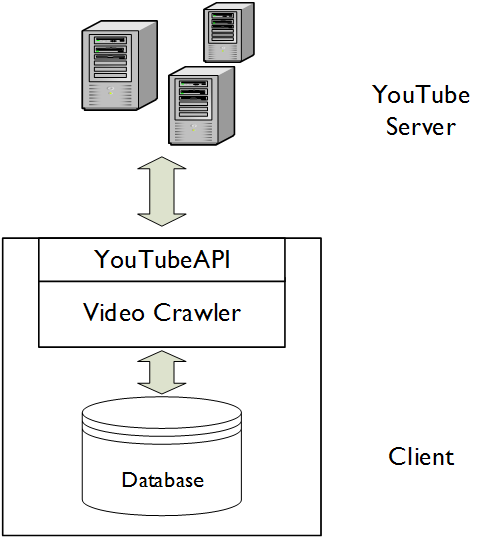


Fig. 2. Data collection platform

### 3.2 Sampling

On the high level, we only collect part of the data about YouTube. So the challenge is how to sampling the YouTube videos and get reasonable amount of data to do analysis. Because we care about the user’s rating and engagement. Assume that in most time, the popular video can reflect some behaviors of user and the highly rated video will affect people more. Also, we want the see the trend of YouTube videos at different time of period. Finally, we sampling the data in the following way: collect the top 50 videos of “Most viewed” every year and top 50 videos of “top rated” every year. Take them as seeds and get their related videos by BFS (depth = 2). In total, we can get more than 500,000 video data. We will use Spark to analyze the data in the next section.

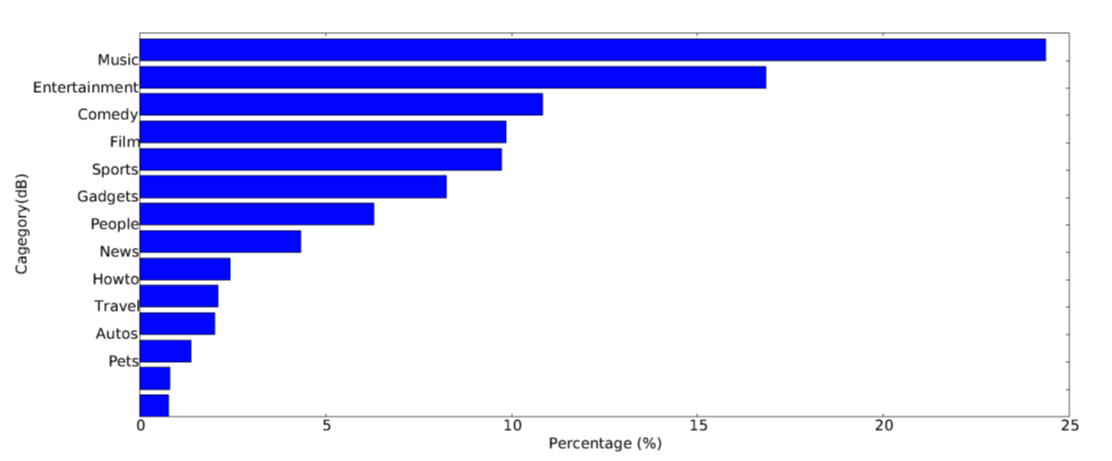
3.3 Existing Data Set

There are one YouTube data set available (<http://netsg.cs.sfu.ca/youtubedata/>). The data is collected at 2007 (two years after the establishment of YouTube). We want to use is as a comparison for our data. In other words, we can find the changes and non-changes of YouTube video by comparing our latest data set and the case of twelve years ago.

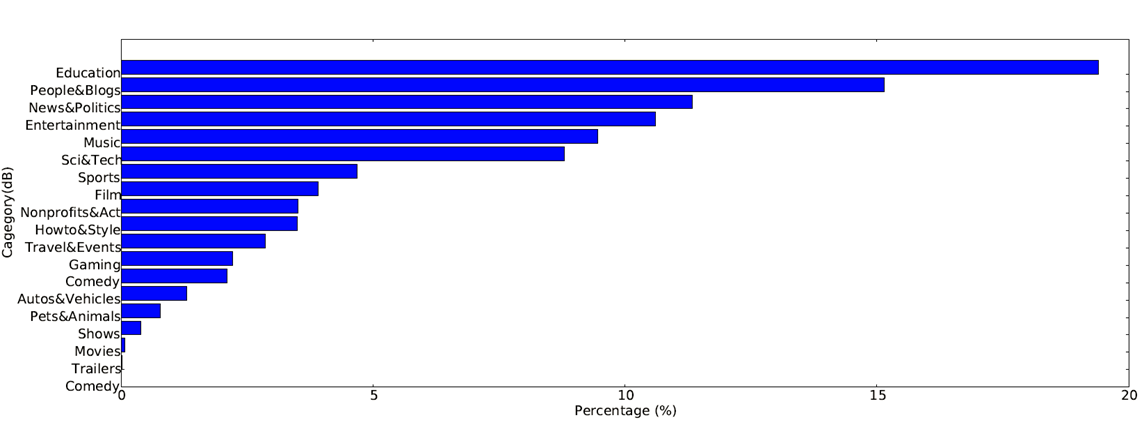
# 4. Data Analysis

## A Descriptive Statistics

### 4.1 Video Category



(a) 2007 data set

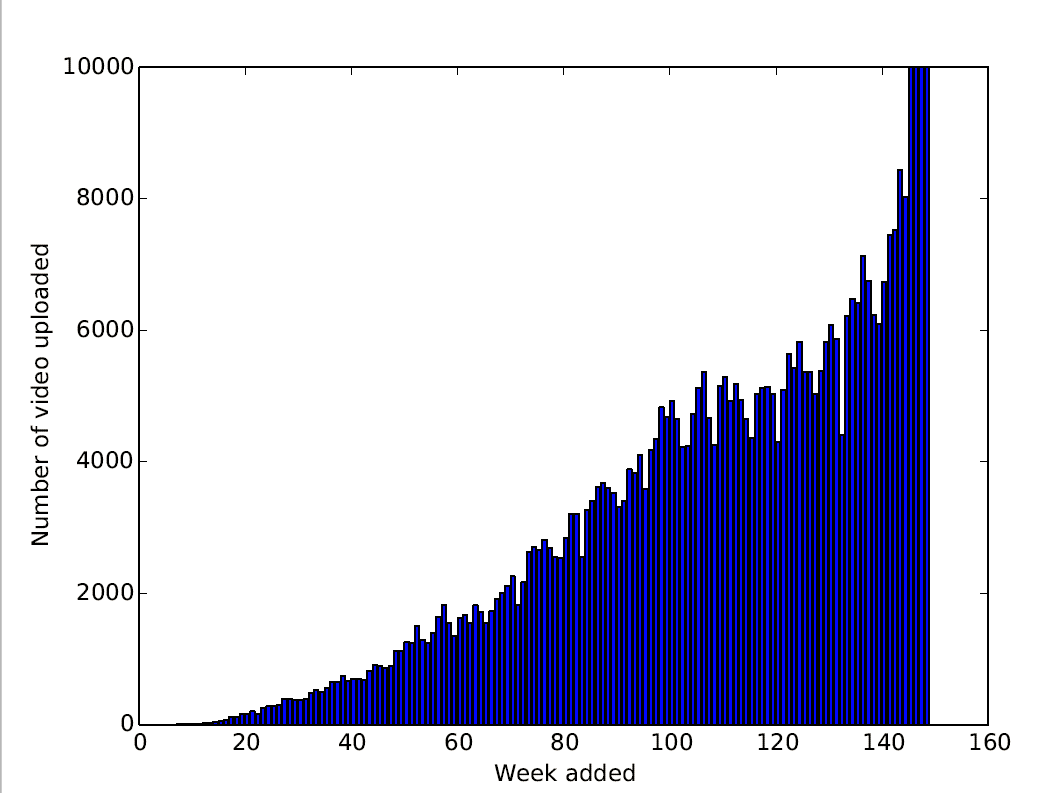
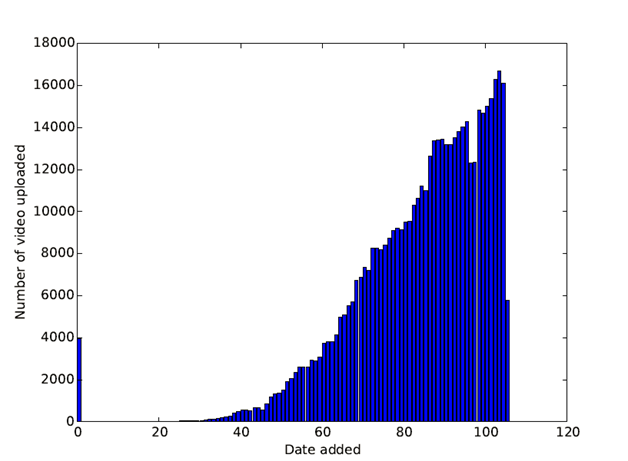


(b) 2017 data set

Fig.3 Distribution of YouTube video categories

From Figure 3, we can know that at 2007, the video only have 12 categories and the Music is the most popular category is “Music” and Entertainment. However, nowadays, the YouTube have more categories. Music and Entertainment is among the most popular categories all the time. Nowadays, Education, People & Blogs and News & Politics is the most popular categories. Which shows that people are more likely to use YouTube to learn and upload their own recorded video in their life. YouTube is also a big platform for news.

### 4.2 Upload trend follows the power curve

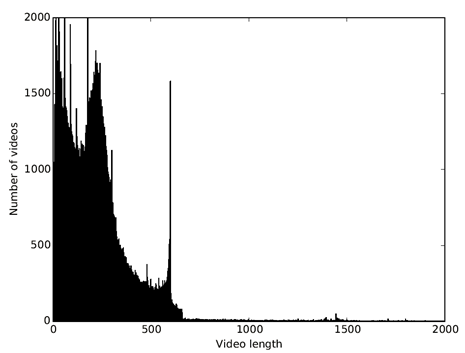
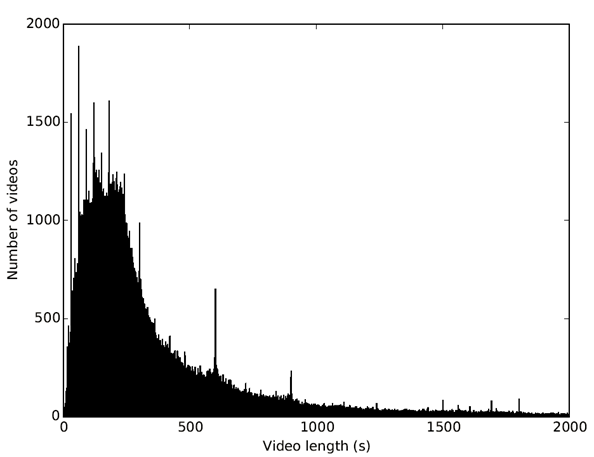


(a) 2007 data set (b) 2017 data set

Fig. 4 Uploading Trend of YouTube Videos

We also record the age of each video. In Fig. 4, we can see that both data set shows that the upload trend follows the power curve. Which means the uploading of video is increasing in the speed of power.

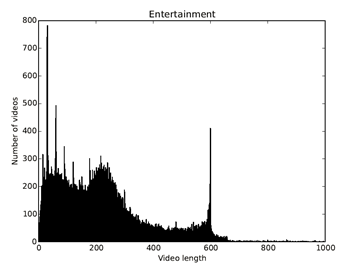
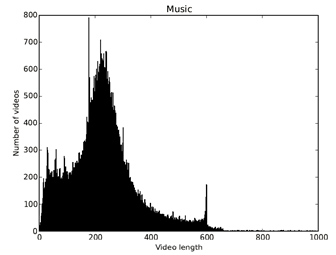
### 4.3 Video Length

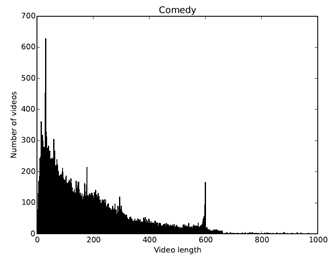
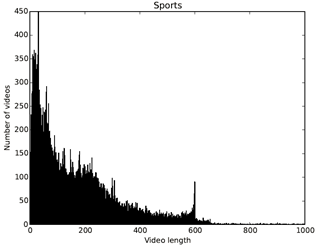
 

(a) 2007 data set (b) 2017 data set

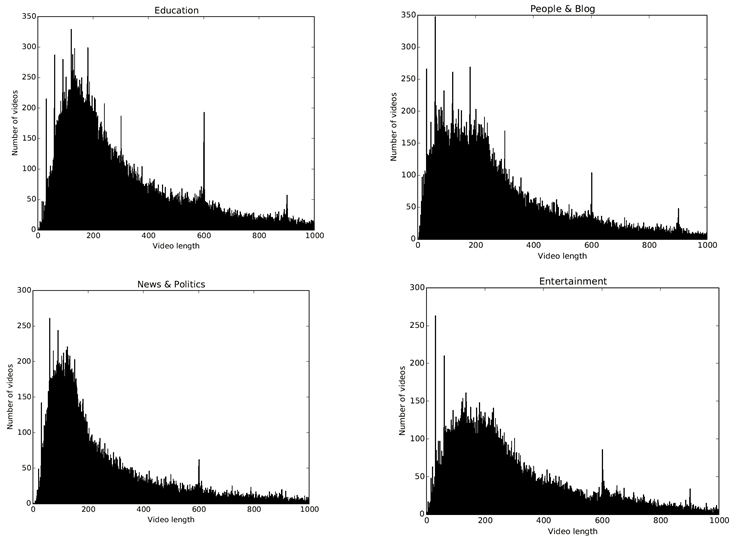
Fig. 5 Distribution of YouTube video length

From Fig. 5 we can see that majority of video is within 10 min in both 2007 and 2017.But in 2007, the video is rarely longer than 10 mins and there are also a number of videos with length smaller than 1 mins. On the other hand, in 2017, the video is more evenly distributed and more videos are larger than 10 minutes.



(a) 2007 data set

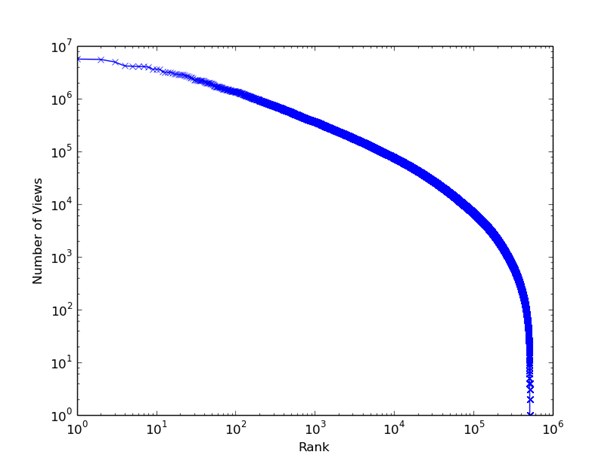
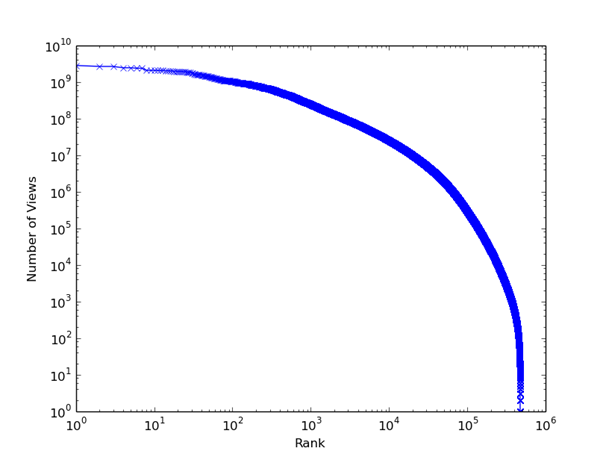


(b) 2017 data set

Fig. 6 Distribution of YouTube video length for four most popular categories

We plot the distribution of YouTube video length for four most popular categories. In 2007, Music videos have a peak at 3~4 minutes because of typical music video length. Entertainment videos have a cutoff around 10 mins due to being divided into multiple clips. As for Comedy and Sports, there are more videos within 2 minutes. In 2017, Rayleigh distribution for all four most popular categories, but their variances are different.It make sense because that some news & politics are usually shorter than videos like Education. Moreover, Entertainment videos’ cutoff around 10 mins is not obvious.

### 4.4 Video Popularity and User Access Pattern

(a) 2007 data set (b) 2017 data set

Fig. 7 YouTube video rank ordered by popularity

YouTube is more popular as years growing, the most popular video in YouTube in 2017 have been played around 0.5 billion. However, the one in 2007 only have 8 million views. We can see that growing influence of YouTube video.

## B Statistical Analysis

In general, from the correlation matrix (Table 2), it is known that the upload time does not show any high correlation with other variables. More specifically, upload time does not have correlation with video duration, which means uploader’s hobby do not significantly change over time. The plots from Fig. 9 also support it.

On the contrary, view counts, comment counts, like counts and also dislike counts have relative high correlation to each other, especially pairs of view counts and like counts and also comment counts and dislike counts. It seems interesting as if people like the videos they would like to view it more often. Also if they dislike the video, they are more likely to leave some comments.

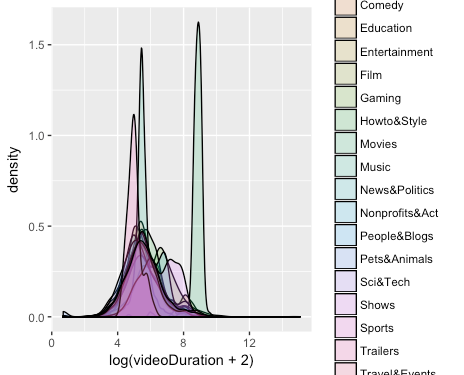
On the other hand, from Fig. 8, it shows the difference and similarities between categories. It is straightforward to see that the “movies” has significantly different distributions of both video duration and view counts from other categories, which means “Movies” is usually significantly longer than others and has less view counts. It seems reasonable. Also the third plot from Fig.9 shows there is no significantly difference between categories, which is also reasonable.

What is more, from Fig. 9, which is similar to Fig.8, it shows the difference and similarities between upload time. The groups comes from the the upload time div 500 into 8 and the lighter color of the plots mean the larger number and newer the video. There are no significant group show out, which means all the variables do not have significant change over time. It has some interesting results: 1) It means the 10 minutes limit rule change from Youtube does not affect the hobby of uploader significantly; 2) view counts and also comments do not significantly change over time neither. However, more interestingly, the Music, Sports, Entertainment and Gaming video have significantly both view counts and comments increase, which means not only recently much more people start to look at the Youtube and they are more likely to watch those categories. Also, the counts of the large values of view counts and comments of Movies( right side of the histogram), increasing significantly which means top popular movies also have highly increasing customers.

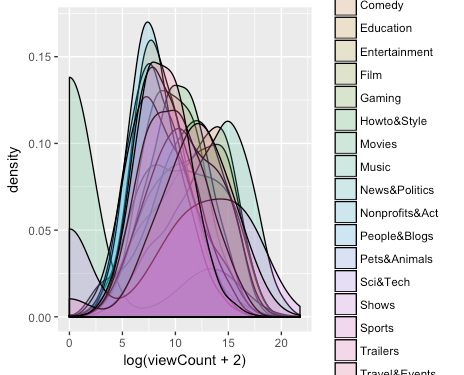
The last part Fig. 10 linear relation is straightforward and it again support the highly correlated shown from the correlation matrix.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Upload Time | Video Duration | View  Count | Comment | Like  Count | Dislike  Count |
| Upload Time | 1 | 0.013770 | -0.02329 | -0.009954 | -0.00372 | -0.00103 |
| Video Duration | 0.0137702 | 1 | -0.002253 | -0.001564 | -0.002915 | -0.000945 |
| View Count | -0.023294 | -0.002253 | 1 | **0.5994780** | **0.9038008** | **0.5636737** |
| Comment | -0.009954 | -0.001564 | 0.5994780 | 1 | **0.6754886** | **0.7984239** |
| Like Count | -0.003729 | -0.002915 | 0.9038008 | 0.6754886 | 1 | **0.5231864** |
| Dislike Count | -0.001030 | -0.000945 | 0.5636737 | 0.79842398 | 0.5231864 | 1 |

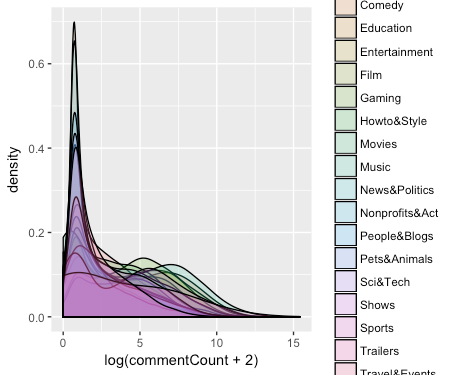
Table 2 Correlation Matrix of YouTube metrics



(a) Density plot of Video Duration by category



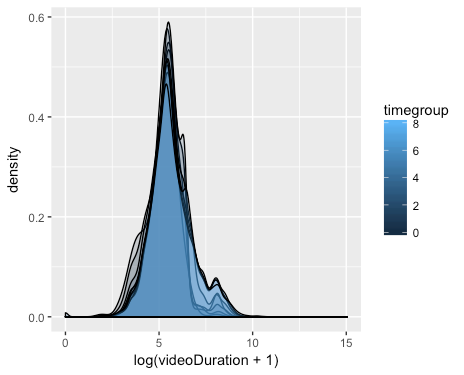
(b) Density plot of View counts by category



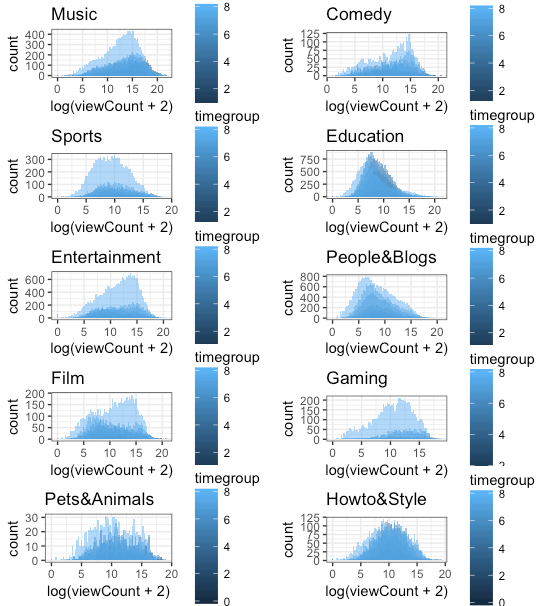
(c) Density plot of comment counts by category

Fig. 8 Density of video duration, view counts and comment counts by category

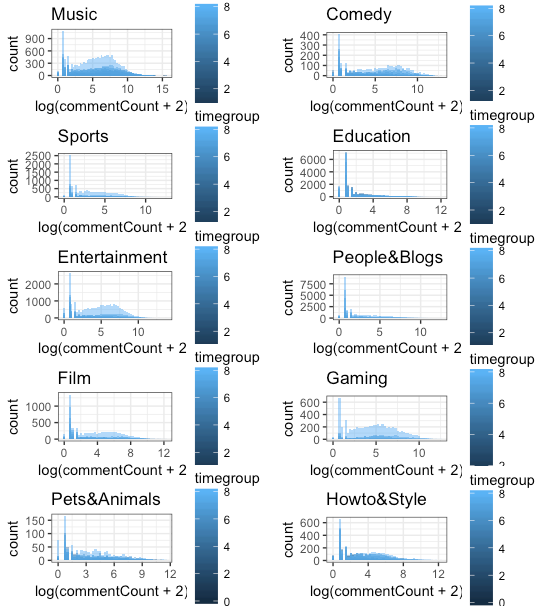
### 



(a) Density plot of Vide Duration by uploadtime

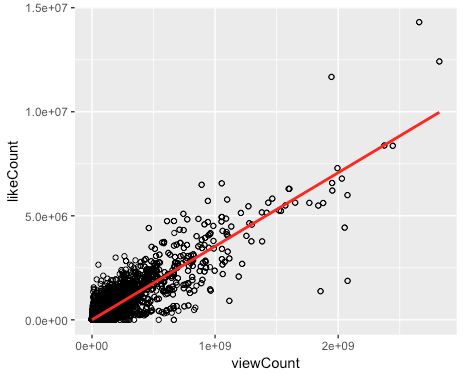


(b) Density plot of view counts by uploadtime group for categories

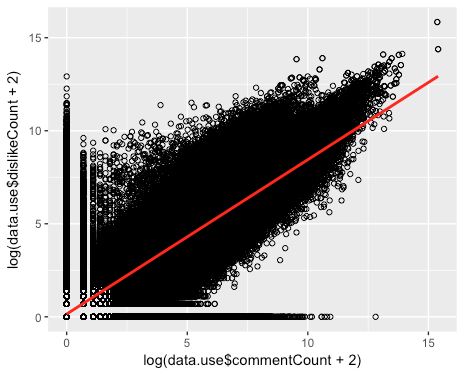


(c)Density plot of comment counts by Upload Time group for categories

Fig. 9 Density of video duration, view counts and comment counts by upload time group for categories



1. view count V.S. like count



(b) log(comment counts) V.S. dislike counts

Fig. 10 Linear Relation

# 5. Conclusion

In this work, we build a platform with YouTube API to collect the data from YouTube server. Next we analysis the data via spark. Then we got statistics features of YouTube video and user behaviors. Finally, we analysis the relationship between YouTube metrics. We found that the linear relation among like, dislike and comments.

# References:

[1] <http://spark.apache.org/>

[2] <https://developers.google.com/youtube/v3/getting-started>

[3] <http://netsg.cs.sfu.ca/youtubedata/>

[4] Cheng, Xu, Cameron Dale, and Jiangchuan Liu. "Statistics and social network of youtube videos." *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*. IEEE, 2008.