YOUTUBE DATA ANALYSIS USING HADOOP

A Project

Presented to the faculty of the Department of Computer Science

California State University, Sacramento

Submitted in partial satisfaction of the
requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

by

Charu Khosla

FALL
2016

YOUTUBE DATA ANALYSIS USING HADOOP

A Project

by

Charu Khosla

Approved by:

_____, Committee Chair
Dr. V. Gordon Scott

_____, Second Reader
Devin Cook

_____
Date

Student: Charu Khosla

I certify that this student has met the requirements for format contained in the University format manual, and that this project is suitable for shelving in the Library and credit is to be awarded for the project.

_____, Graduate Coordinator _____
Dr. Ying Jin Date

Department of Computer Science

Abstract

of

YOUTUBE DATA ANALYSIS USING HADOOP

by

Charu Khosla

Analysis of structured data has seen tremendous success in the past. However, analysis of large scale unstructured data in the form of video format remains a challenging area. YouTube, a Google company, has over a billion users and generates billions of views. Since YouTube data is getting created in a very huge amount and with an equally great speed, there is a huge demand to store, process and carefully study this large amount of data to make it usable

The main objective of this project is to demonstrate by using Hadoop concepts, how data generated from YouTube can be mined and utilized to make targeted, real time and informed decisions.

The project utilizes the YouTube Data API (Application Programming Interface) that allows the applications/websites to incorporate functions that are used by YouTube application to fetch and view information. The Google Developers Console is used to generate a unique access key which is further required to fetch YouTube public channel data. Once the API key is generated, a .Net(C#) based console application is designed to use the YouTube API for fetching video(s) information based on a search criteria. The text file output generated from the console application is then loaded from HDFS

(Hadoop Distributed File System) file into HIVE database. Hive uses a SQL-like interface to query data stored in various databases and file systems that integrate with Hadoop. HDFS (Hadoop Distributed File System) is a primary Hadoop application and a user can directly interact with HDFS using various shell-like commands supported by Hadoop. This project uses SQL like queries that are later run on Big Data using HIVE to extract the meaningful output which can be used by the management for analysis.

_____, Committee Chair
Dr. V. Scott Gordon

_____
Date

ACKNOWLEDGEMENTS

I would first like to thank Dr. Scott Gordon for allowing me to work on such an interesting and fun project. He was instrumental to my success in this project. He was very supportive and understanding provided me an extra push and confidence to succeed in this venture. He took an extra effort to review the report and provide his invaluable feedback.

I would like to acknowledge and thank Dr. Meiliu Lu and Prof. Devin Cook for their mechanical expertise which was invaluable in this project.

In addition, I would like to thank the Department of Computer Science at California State University, Sacramento for providing an opportunity to pursue my Master's degree and guiding me all the way to become a successful student.

Last but not the least, I am thankful to my husband Vivek Trehan and my family for their constant support and belief in me, their words of wisdom and moral support helped me to overcome all the challenges. With their guidance, I was able to successfully complete my project and earn my Master's Degree.

TABLE OF CONTENTS

List of Figures

Figures

Chapter 1

INTRODUCTION

With rapid innovations and surge of internet companies like Google, Yahoo, Amazon, eBay and a rapidly growing internet savvy population, today's advanced systems and enterprises are generating data in a very huge volume with great velocity and in a multi-structured formats including videos, images, sensor data, weblogs etc. from different sources. This has given birth to a new type of data called Big Data which is unstructured sometime semi structured and also unpredictable in nature. This data is mostly generated in real time from social media websites which is increasing exponentially on a daily basis.

According to Wikipedia, “Big Data is a term for data sets that are so large or complex that traditional data processing applications are inadequate to deal with them. Analysis of data sets can find new correlations to spot business trends, prevent diseases, combat crime and so on." [1] With millions of people using Twitter to tweet about their most recent brand experience or hundreds of thousands of check-ins on Yelp, thousands of people talking about a recently released movie on Facebook and millions of views on YouTube for a recently released movie trailer, we are at a stage wherein we are heading into a social media data explosion. Companies are already facing challenges getting useful information from the transactional data from their customers (for e.g. data captured by the companies when a customer opens a new account or sign up for a credit card or a service). This type of data is structural in nature and still manageable. However, social

media data is primarily unstructured in nature. The very unstructured nature of the data makes it very hard to analyze and very interesting at the same time.

Whereas RDBMS is designed to handle structured data and that to only certain limit, RDBMS fails to handle this kind of unstructured and huge amount of data called Big Data. This inability of RDBMS has given birth to new database management system called NoSQL management system.

Some of the key concepts used in Big Data Analysis are

1. Data Mining: Data mining is incorporation of quantitative methods. Using powerful mathematical techniques applied to analyze data and how to process that data. It is used to extract data and find actionable information which is used to increase productivity and efficiency.
2. Data Warehousing: A data warehouse is a database as the name implies. It is a kind of central repository for collecting relevant information. It has centralized logic which reduces the need for manual data integration.
3. MapReduce: MapReduce is a data processing paradigm for condensing large volumes of data into useful aggregated results. Suppose we have a large volume of data for particular users or employees etc. to handle. For that we need MapReduce function to get the aggregated result as per the query.
4. Hadoop: Anyone holding a web application would be aware of the problem of storing and retrieving data every minute. The adaptive solution created for the same was the use of Hadoop including Hadoop Distributed File System or HDFS

for performing operations of storing and retrieving data. Hadoop framework has a scalable and highly accessible architecture.

5. Hive: Hive is a data warehouse system for Hadoop that facilitates ad hoc queries and analysis of large data sets stored in Hadoop.
6. HQL: Hive uses a SQL like language called Hive. HQL is a popular choice for Hadoop analytics.

YouTube is one of the most popular and engaging social media tool and an amazing platform that reveals the community feedback through comments for published videos, number of likes, dislikes, number of subscribers for a particular channel. YouTube collects a wide variety of traditional data points including View Counts, Likes, Votes, and Comments. The analysis of the above listed data points constitutes a very interesting data source to mine for obtaining implicit knowledge about users, videos, categories and community interests.

Most of the companies are uploading their product launch on YouTube and they anxiously await their subscribers' reviews. Major production houses launch movie trailers and people provide their first reaction and reviews about the trailers. This further creates a buzz and excitement about the product. Hence the above listed data points become very critical for the companies so that they can do the analysis and understand the customers' sentiments about their product/services.

1. This project will help people in understanding how to fetch a specific channel’s YouTube data using YouTube API.

2. This project requires access to Google Developers Console and generate a unique access key. That unique key is required to fetch YouTube public channel data. With the help of the unique access key, the required data is fetched from YouTube using a .Net(C#) console application.
3. The extracted data is first stored in HDFS file and then the data is loaded into HIVE database. The queries are run into HIVE database so that the YouTube data can be mined intelligently and the findings can be shared with the management.

Chapter 2

BACKGROUND

2.1 Purpose and Scope

"YouTube has over a billion users and every day people watch hundreds of millions of hours on YouTube and generate billions of views”[2] . "Every day, people across the world are uploading 1.2 million videos to YouTube, or over 100 hours per minute and this number is ever increasing [3]. To analyze and understand the activity occurring on such a massive scale, a relational SQL database is not enough. Such kind of data is well suited to a massively parallel and distributed system like Hadoop.

The main objective of this project is to focus on how data generated from YouTube can be mined and utilized by different companies to make targeted, real time and informed decisions about their product that can increase their market share. This can be done by using Hadoop concepts. The given project will focus on how data generated from YouTube can be mined and utilized. There are multiple applications of this project. Companies can use this project to understand how effective and penetrative their marketing programs are. In addition to the view counts, subscribers and shares, audience retention count, companies can also evaluate views according to date range. This can tell the companies when is the slow period or spike in viewership and attribute the same to certain marketing campaign. Applications for YouTube data can be endless. For example, Companies can analyze how much a product is liked by people. This project can also help in analyzing new emerging trends and knowing about people's changing behavior with time. Also people in different countries have different preferences. By analyzing the

comments/feedbacks/likes/view counts etc. of the videos uploaded, companies can understand what are the likes/dislikes of people around the world and work on their preferences accordingly.

This project uses following concepts and tools throughout its lifecycle.

1. Java API
2. Hadoop
3. Hive
4. Unix
5. Data warehousing (Extraction Transformation Loading)
6. Data Mining
7. Business Intelligence

2.2 Overview of the Project

In this project we fetch a specific channel's YouTube data using YouTube API. We will use Google Developers Console and generate a unique access key which is required to fetch YouTube public channel data. Once the API key is generated, a .Net(C#) based console application is designed to use the YouTube API for fetching video(s) information based on a search criteria. The text file output generated from the console application is then loaded from HDFS file into HIVE database. HDFS is a primary Hadoop application and a user can directly interact with HDFS using various shell-like commands supported by Hadoop. Then we run queries on Big Data using HIVE to extract the meaningful output which can be used by the management for analysis.

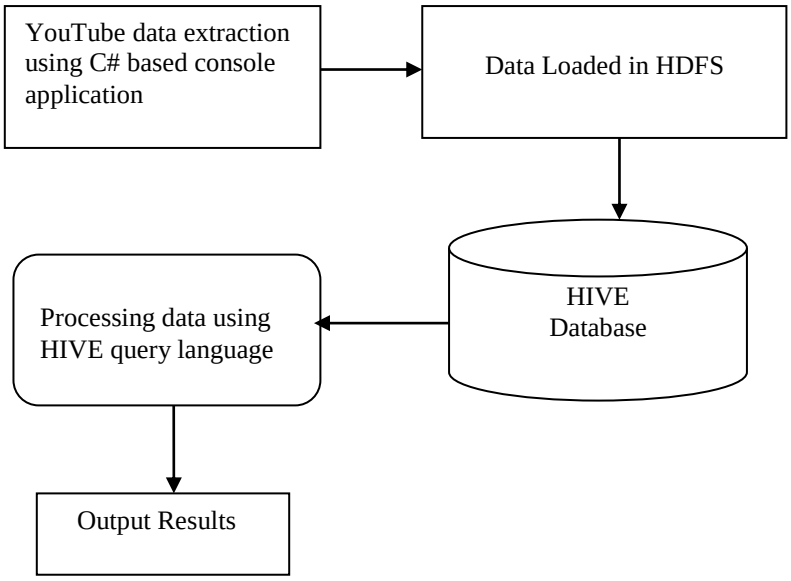

Figure 1 High Level Flow Diagram

Chapter 3

RELATED PROJECTS

There are many projects which have used Hadoop to analyze the Big Data. Some of them are listed below herewith.

3.1 Using Hadoop to Analyze Stock Market Data

Since Stock Markets generate wide variety of unstructured data, this type of data can be analyzed using Hadoop framework. A stock market data analysis project was conducted by taking sample 'New York Stock Exchange' data set. Using Hadoop Framework, the covariance for this stock data was calculated and aimed to solve both storage and processing problems related to a huge volume of data. [4].

The dataset used in this project was a comma separated file (CSV) that contains the stock information such as daily quotes, stock opening price, stock highest price, etc. on the New York Stock Exchange. Using Hive commands, a Hive Table was created. Once the table was created, the CSV data was loaded into the Hive Table. By using the Hive select queries, Covariance for the provided stock dataset for the inputted year was calculated. From the covariance results, stock brokers provided key recommendations including the possibility of stock prices moving in the upward direction or inverse direction.

3.2 Sentiment Analysis of Twitter Data Using Hadoop

Sentiment analysis or opinion mining is defined as categorizing opinions expressed on a social media platform about a given subject. This project was undertaken

to understand the comment writer's attitude towards a particular product or a given subject. Using Sentiment Analysis, it can be determined if the general attitude of people is positive, negative or neutral towards a specific subject. [10]

The core objective of this project was to analyze the twitter data and classify/categorize it based on the polarity of the words. The data was collected from Twitter using java Twitter streaming API. This data was then stored in HDFS (Hadoop Distributed File System) in a certain format. This data was further fed to mapper in MapReduce programming approach. The data collected from Twitter over a period of time was processed by using java and distributed processing software framework and using MapReduce programming model and Apache hive frame work. The output obtained from reducer phase was further analyzed and represented to the management in the form of pie-charts. The final data outcome showed in this project was in the form of Positive, Negative and Neutral tweets. This helped companies in identifying if the marketing campaign for a specific product was success or a failure.

3.3 Weather Data Analysis Using Hadoop to Mitigate Event Planning Disasters

Many government organizations and private companies are closely monitoring the global temperature changes and weather patterns. The collection of data files is both data and compute intensive. Hence it was decided to use MapReduce programming to analyze this data over other traditional methods. The weather data was analyzed using Hadoop Distributed System, which was used to plan any outdoor events. The proposed event planning system decided appropriate days for outdoor events and activities per month for

different attractive cities based on the analysis of historical weather data. All collected data was stored at HDFS, i.e., Hadoop Distributed File System, and then the data was processed and analyzed by using MapReduce programming. As a result, useful information about event planning was discovered, such as locations (city), time and statistical data.

3.4 Airline Analysis Project

As travelling by airplane has become more common there are many challenges that passenger face. Every year approximately 20% of airline flights are delayed or cancelled, resulting in significant costs to both travelers and airlines. Using Hadoop and MapReduce Programming, a model was built that can predict the airline delay from historical flight data and weather information. The historical airline delay dataset was available to start with. Using PIG and Python a feature matrix was designed from the given data set. Hadoop was employed to perform various types of data pre-processing and multiple iterations were performed. With each iteration, the input data was enriched resulting in predictive features such as temperature, snow conditions, wind turbulence etc.

As part of analysis, this project focused on possible delays and provided the output based upon historical information fed into the system and answered following questions:

- Are there any airlines which have significantly less delays?
- Which airport within the same metro area offers the least delay to passengers?
- How much does weather play a role in flight delays?

The output for the research was that it matters which airlines you travel by for example certain airlines performed better than other airlines. Also, it was found that snowfall had a significant amount of impact in flight delays.

Chapter 4

BIG DATA AND HADOOP

4.1 What is Big Data?

Wikipedia defines Big Data as "a collection of data sets so large and complex that it becomes difficult to process using the available database management tools. The challenges include how to capture, curate, store, search, share, analyze and visualize Big Data" [1]. In today's environment, we have access to more types of data. These data sources include online transactions, social networking activities, mobile device services, internet gaming etc.

Big Data is a collection of data sets that are large and complex in nature. They constitute both structured and unstructured data that grow large so fast that they are not manageable by traditional relational database systems or conventional statistical tools. Big Data is defined as any kind of data source that has at least three shared characteristics:

- Extremely large Volumes of data
- Extremely high Velocity of data
- Extremely wide Variety of data

According to Big Data: Concepts, Methodologies, Tools, and Applications, Volume I by Information Resources Management Association (IRMA), "organizations today are at the tipping point in terms of managing data. Data sources are ever expanding. Data from Facebook, Twitter, YouTube, Google etc., are to grow 50X in the next 10

years. Over 2.5 exabytes of data is generated every day. Some of the sources of huge volume of data are:

1. A typical large stock exchange captures more than 1 TB of data every day.
2. There are over 5 billion mobile phones in the world which are producing enormous amount of data on daily basis.
3. YouTube users upload more than 48 hours of video every minute.
4. Large social networks such as Twitter and Facebook capture more than 10 TB of data daily.
5. There are more than 30 million networked sensors in the world which further produces TBs of data every day. " [11]

Structured and semi-structured formats have some limitations with respect to handling large quantities of data. Hence, in order to manage the data in the Big Data world, new emerging approaches are required, including document, graph, columnar, and geospatial database architectures. Collectively, these are referred to as NoSQL, or not only SQL, databases. In essence the data architectures need to be mapped to the types of transactions. Doing so will help to ensure the right data is available when you need it.

4.2 What is Hadoop?

As organizations are getting flooded with massive amount of raw data, the challenge here is that traditional tools are poorly equipped to deal with the scale and complexity of such kind of data. That's where Hadoop comes in. Hadoop is well suited to

meet many Big Data challenges, especially with high volumes of data and data with a variety of structures.

At its core, Hadoop is a framework for storing data on large clusters of commodity hardware — everyday computer hardware that is affordable and easily available — and running applications against that data. A cluster is a group of interconnected computers (known as nodes) that can work together on the same problem. Using networks of affordable compute resources to acquire business insight is the key value proposition of Hadoop.

Hadoop consists of two main components

1. A distributed processing framework named MapReduce (which is now supported by a component called YARN(Yet Another Resource Negotiator) and
2. A distributed file system known as the Hadoop Distributed File System, or HDFS.

In Hadoop you can do any kind any kind of aggregation of data whether it is one-month old data or one-year-old data. Hadoop provides a mechanism called MapReduce model to do distributed processing of large data which internally takes care of data even if one machine goes down.

4.3 Hadoop Ecosystem

Hadoop is a shared nothing system where each node acts independently throughout the system. A framework where a piece of work is divided among several parallel MapReduce task. Each task operated independently on cheap commodity servers. This enables businesses to generate values from data that was previously considered too

expensive to be stored and processed in a traditional data warehouse or OLTP (Online Transaction Processing) environment. In the old paradigm, companies would use a traditional enterprise data warehouse system and would buy the biggest data warehouse they could afford and store the data on a single machine. However, with the increasing amount of data, this approach is no longer affordable nor practical.

Some of the components of Hadoop ecosystem are HDFS (Hadoop Distributed File System), MapReduce, Yarn, Hive and Hbase. Hadoop has two core components. 'Storage' part to store the data and 'Processing' part to process the data. The storage part is called 'HDFS' and the processing part is called as 'YARN'.

4.4 Sketching Out the HDFS Architecture

4.4.1 Storage Component: Hadoop Distributed File System (HDFS)

As stated above, the Hadoop Distributed File System (HDFS) is the storage component of the core Hadoop Infrastructure. HDFS provides a distributed architecture for extremely large scale storage, which can easily be extended by scaling out. It is important to mention the difference between scale up and scale out. In its initial days, Google was facing challenges to store and process not only all the pages on the internet but also its users' web log data. At that time, Google was using scale up architecture model where you can increase the system capacity by adding CPU cores, RAM etc to the existing server. But such kind of model had was not only expensive but also had structural limitations. So instead, Google engineers implemented Scale out architecture model by using cluster of smaller servers which can be further scaled out if they require

more power and capacity. Google File System (GFS) was developed based on this architectural model. HDFS is designed based on similar concept.

The core concept of HDFS is that it can be made up of dozens, hundreds, or even thousands of individual computers, where the system's files are stored in directly attached disk drives. Each of these individual computers is a self-contained server with its own memory, CPU, disk storage, and installed operating system (typically Linux, though Windows is also supported). Technically speaking, HDFS is a user-space-level file system because it lives on top of the file systems that are installed on all individual computers that make up the Hadoop cluster.

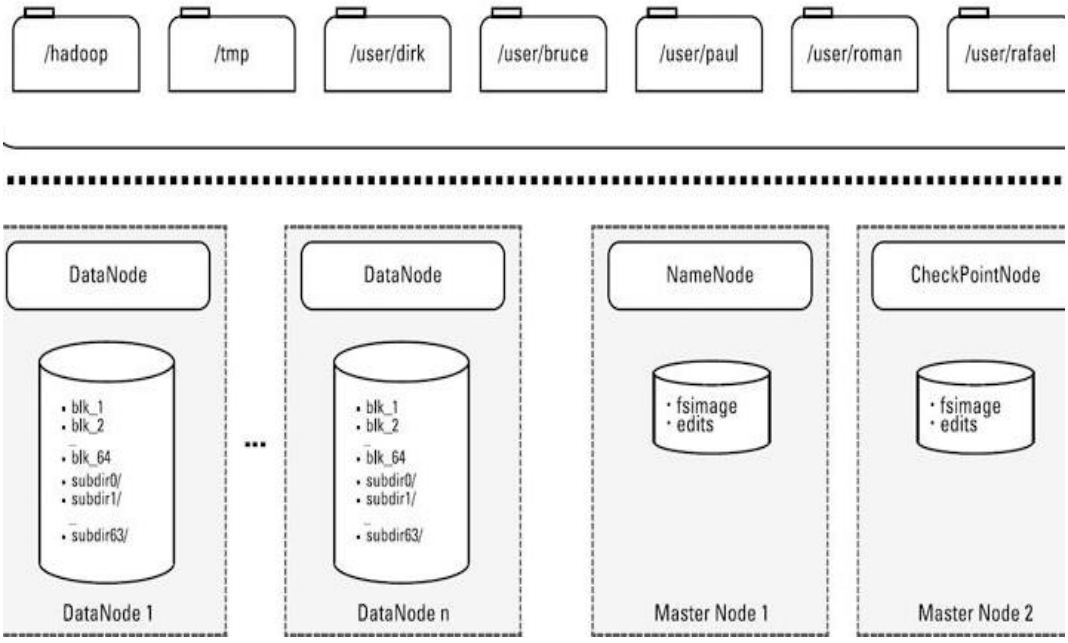

Figure 2 HDFS as a user-space-level file system [12]

The above figure [12] shows that a Hadoop cluster is made up of two classes of servers: slave nodes, where the data is stored and processed and master nodes, which

govern the management of the Hadoop cluster. On each of the master nodes and slave nodes, HDFS runs special services and stores raw data to capture the state of the file system. In the case of the slave nodes, the raw data consists of the blocks stored on the node, and with the master nodes, the raw data consists of metadata that maps data blocks to the files stored in HDFS.

HDFS is a system that allows multiple commodity machines to store data from a single source. HDFS consists of a NameNode and a DataNode. HDFS operates as master slave architecture as opposed to peer to peer architecture. NameNode serves as the master component while the DataNode serves as a slave component. NameNode comprises of only the Meta data information of HDFS that is the blocks of data that are present on the Data Node

- How many times the data file has been replicated?
- When does the NameNode start?
- How many DataNodes constitute a NameNode, capacity of the NameNode and space utilization?
- The DataNode comprises of data processing, all the processing data that is stored on the DataNode and deployed on each machine.
- The actual storage of the files being processed and serving read and write request for the clients

In the earlier versions of Hadoop there was only one NameNode attached to the DataNode which was a single point of failure. Hadoop version 2.x provides multiple NameNode where secondary NameNode can take over in the event of a primary

NameNode failure. Secondary NameNode is responsible for performing periodic check points in the event of a primary NameNode failure. You can start secondary NameNode by providing checkpoints that provide high availability within HDFS.

Let's take look at a data warehouse structure example where we have one machine and with HDFS we can distribute the data into more than one machine. Let's say we have 100 GB of file that takes 20 minutes to process on a machine with a given number of channel and hard drive. If you add four machines of exactly the same configuration on a Hadoop cluster, the processing time reduces to approximately one fourth of the original processing time or about 5 minutes.

But what happens if one of these four machines fails? HDFS creates a self-healing architecture by replicating the same data across multiple nodes. So it can process the data in a high availability environment. For example, if we have three DataNodes and one NameNode, the data is transferred from the client environment into HDFS DataNode. The replication factor defines the number of times a data block is replicated in a clustered environment. Let's say we have a file that is split into two data blocks across three DataNodes. If we are processing these files to a three DataNode cluster and we set the replication factor to three. If one of the nodes fails, the data from the failed nodes is redistributed among the remaining active nodes and the other nodes will complete the processing function.

4.4.2 Processing Component: Yet Another Resource Negotiator (YARN)

YARN (Yet Another Resource Negotiator) is a resource manager that identifies on which machine a particular task is going to be executed. The actual processing of the task or program will be done by Node Manager. In Hadoop 2.2, YARN augments the MapReduce platform and serves as the Hadoop operating system. Hadoop 2.2 separates the resource management function from data processing allowing greater flexibility. This way MapReduce only performs data processing while resource management is isolated in YARN. Being the primary resource manager in HDFS, YARN enables enterprises to store data in a single place and interact with it in multiple ways with consistent levels of service. In Hadoop 1.0 the NameNode used job tracker and the DataNode used task tracker to manage resources. In Hadoop 2.x, YARN splits up into two major functionalities of the job tracker - the resource management and job scheduling. The client reports to the resource manager and the resource manager allocates resources to jobs using the resource container, Node Manager and app master. The resource container splits memory, CPU, network bandwidth among other hardware constraints into a single unit. The Node Manager receives updates from the resource containers which communicate with the app master. The Node Manager is the framework for containers, resource monitoring and for reporting data to the resource manager and scheduler.

4.4.3 Hadoop Framework

Hadoop Framework comprises of Hadoop Distributed File System and the MapReduce framework. The Hadoop framework divides the data into smaller chunks and

stores divides that data into smaller chucks and stores each part of the data on a separate node within the cluster. For example, if we have 4 terabytes of data, the HDFS divides this data into 4 parts of 1TB each. By doing this, the time taken to store the data onto the disk is significantly reduced. The total time to store this entire data onto the disk is equal to storing 1 part of the data as it will store all the parts of the data simultaneously on different machines.

In order to provide high availability what Hadoop does is replicate each part of the data onto other machines that are present within the cluster. The number of copies it will replicate depends on the replication factor. By default the replication factor is 3, in such a case there will be 3 copies of each part of the data on three different machines. In order to reduce the bandwidth and latency time, it will store two copies on the same rack and third copy on a different rack. For example, in the above example, NODE 1 and NODE 2 are on rack one and NODE 3 and NODE 4 are on rack two. Then the first two copies of part 1 will be stored on NODE 1 and third copy will be stored either on NODE 3 or NODE 4. Similar process is followed in storing remaining parts of the data. The HDFS takes care of the networking required by these nodes in order to communicate.

Chapter 5

YOUTUBE DATA ANALYSIS USING HADOOP

YouTube, owned by Google, is a video sharing website, where users can upload, watch and share videos with others. YouTube provides a forum for people to connect, inform, and inspire others across the globe and acts as a distribution platform for original content creators and advertisers large and small.

> According to Statista.com (The Statistics Portal), "As of July 2015, more than 400 hours of video content were uploaded to YouTube every minute, a fourfold increase compared to only two years prior. The platform, which was created in 2005, has slowly become one of the most visited websites in the world and a global phenomenon. In the first quarter of 2015, more than 80 percent of global internet users had visited YouTube in the last month. In the United States, it is the second largest social media website after Facebook, accounting for over 22 percent of social media traffic. The rise in Smartphone and other mobile devices usage has also helped increase the consumption of YouTube videos on the go. As of mid 2015, approximately half of U.S. mobile users accessed YouTube via a mobile device, whether Smartphone or tablet computer." [9]

While companies, musicians or film distributers might use YouTube as a form of free direct advertisement, YouTube has also become a launch pad for various products/services wherein large corporations reveal the first look of the product on YouTube, generate a buzz about their product, assess the market demand based upon likes and view counts and improve their product based upon customers' feedback. Hence the data points including View Counts, Likes, Votes, and Comments etc. become very critical for the companies so that they can do the analysis and understand the customers' sentiments about their product/services.

The main objective of this project is to show how companies can analyze YouTube data using YouTube API to make targeted real time and informed decisions. This project will help in understanding changing trends among people by analyzing YouTube data and fetching meaningful results. For example, when companies like Disney launch their new movie trailers on YouTube, this application can help Disney in analyzing the reaction of people towards a specific movie trailer. This application can analyze home many people liked the trailers, in which country the trailer was liked the most, whether the comments posted on YouTube are generally positive, negative or neutral etc. This way management can take executive decisions how to spend their marketing budget in order to maximize their returns.

Since YouTube data is getting created in a very huge amount and with an equally great speed, there is a huge demand to store, process and carefully study this large amount of data to make it usable. Hadoop is definitely the preferred framework to analyze the data of this magnitude.

5.1 Hadoop Data Analysis Technologies

While Hadoop provides the ability to collect data on HDFS (Hadoop Distributed File System), there are many applications available in the market (like MapReduce, Pig and Hive) that can be used to analyze the data.

Let us first take a closer look at all three applications and then analyze which application is better suited for YouTube Data Analysis project.

5.1.1 MapReduce

MapReduce is a set of Java classes run on YARN with the purpose of processing massive amounts of data and reducing this data into output files. HDFS works with MapReduce to divide the data in parallel fashion on local or parallel machines. Parallel structure requires that the data is immutable and cannot be updated. It begins with the input files where the data is initially stored typically residing in HDFS. These input files are then split up into input format which selects the files, defines the input splits, breaks the file into tasks and provides a place for record reader objects. The input format defines the list of tasks that makes up the map phase. The tasks are then assigned to the nodes in the system based on where the input files chunks are physically resident. The input split describes the unit of work that comprises a single map task in a MapReduce program. The record reader loads the data and converts it into key value pairs that can be read by the Mapper. The Mapper performs the first phase of the MapReduce program. Given a key and a value the mappers export key and value pairs and send these values to the reducers. The process of moving mapped outputs to the reducers is known as shuffling. Partitions are the inputs to reduce tasks, the partitioner determines which key and value pair will be stored and reduced. The set of intermediate keys are automatically stored before they are sent to the reduce function. A reducer instance is created for each reduced task to create an output format. The output format governs the way objects are written, the output format provided by Hadoop writes the files to HDFS.

5.1.2 Hive

Hive provides the ability to store large amounts of data in HDFS. Hive was designed to appeal to a community comfortable with SQL. Hive uses an SQL like language known as HIVEQL. Its philosophy is that we don't need yet another scripting language. Hive supports maps and reduced transform scripts in the language of the user's choice which can be embedded with SQL. Hive is widely used in Facebook, with analyst comfortable with SQL as well as data miners programming in Python.

Supporting SQL syntax also means that it is possible to integrate with existing tools like. Hive has an ODBC (Open Database Connectivity JDBC (Java Database Connectivity) driver that allows and facilitates easy queries. It also adds support for indexes which allows support for queries common in such environment. Hive is a framework for performing analytical queries. Currently Hive can be used to query data stored in HBase which is a key value store like those found in the gods of most RDBMS's (Relational database management system) and the Hadoop database project uses Hive query RDBMS tier.

5.1.3 Pig

Pig comes from the language Pig Latin. Pig Latin is a procedural programming language and fits very naturally in the pipeline paradigm. When queries become complex with most of joins and filters then Pig is strongly recommended. Pig Latin allows pipeline developers to decide where to checkpoint data in the pipeline. That is storing data in between operations has the advantage of check pointing data in the pipeline. This ensures

the whole pipeline does not has to be rerun in the event of a failure. Pig Latin allows users to store data at any point in the pipeline without disturbing the pipeline execution.

The advantage that Pig Latin provides is that pipelines developers decide where appropriate checkpoints are in the pipeline rather than being forced to checkpoint wherever the schematics of SQL impose it. Pig Latin supports splits in the pipeline. Common features of data pipelines is that they are often graphics and not linear pipelines since disk's read and write scan time and intermediate results usually dominate processing of large datasets reducing the number of times data must be written to and read from disk is crucial for good performance.

Pig Latin allows developers to insert their own code almost anywhere in the data pipeline which is useful for pipeline development. This is accomplished through a user defined functions UDFS (User Defined Functions). UDFS allows user to specify how data is loaded, how data is stored and how data is processed. Streaming allows users to include executables at any point in the data flow. Pipeline also often includes user defined columns transformation functions and user defined aggregations. Pig Latin supports writing both of these types of functions in java.

5.2 Analysis of Which Technology to Use

The following table [4] shows features and comparison of leading Hadoop Data Analysis technologies available in the market.

Featured	MapReduce	Pig	Hive
Language	Algorithm of Map and Reduce Functions (Can be implemented in C, Python, Java)	PigLatin (Scripting Language)	SQL-like
Schemas/Types	No	Yes (implicit)	Yes(explicit)
Partitions	No	No	Yes
Server	No	No	Optional (Thrift)
Lines of code	More lines of code	Fewer (Around 10 lines of PIG = 200 lines of Java)	Fewer than MapReduce and Pig due to SQL Like nature
Development Time	More development effort	Rapid development	Rapid development
Abstraction	Lower level of abstraction (Rigid Procedural Structure)	Higher level of abstraction (Scripts)	Higher level of abstraction (SQL like)
Joins	Hard to achieve join functionality	Joins can be easily written	Easy for joins
Structured vs Semi-Structured Vs Unstructured data	Can handle all these kind of data types	Works on all these kind of data types	Deal mostly with structured and semi-structured data
Complex business logic	More control for writing complex business logic	Less control for writing complex business logic	Less control for writing complex business logic
Performance	Fully tuned MapReduce program would be faster than Pig/Hive	Slower than fully tuned MapReduce program, but faster than badly written MapReduce code	Slower than fully tuned MapReduce program, but faster than bad written MapReduce code

Figure 3 Features and Comparisons of MapReduce, Pig and Hive [4]

The YouTube sample dataset collected using .NET console application (see Section 7) has the following properties:

- There is a structured format of data
- It would require joins to capture country wise count statistics
- The collected data set can be organized into schema

For this project, dataset is relatively smaller. The dataset is around 2000 records for any given search criteria across multiple countries. However, in real environment, with the extensive information available on YouTube the data size can be much larger. Given a video ID, the application first extracts information from the YouTube API, which contains all the meta-data. The application then scrapes the video's webpage to obtain the remaining information. The following information of YouTube video is recorded in order; they are divided by '\t' in the data file.

Channel ID	Channel ID is a 11-character string key that is used to uniquely identify YouTube video. This ID is case sensitive. The individual characters come from a set of 64 possibilities (A-Za-z0-9_-)
Uploader Name	Uploader name is a string of the video uploader's username. YouTube uploader names are limited to 20, case insensitive, alphanumeric characters. Minimum length of a channel name is 6 characters
Published at	Published at is in date format. The date that video was published. The value is specified in (YYYY-MM-DD) format.
Category	A video category resource identifies a category that has been or could be associated with uploaded videos. YouTube provides 16 video categories.
Duration	Duration is the length of the video. The duration is in the format H#M#S where H represents hours, M represents minutes, S represent seconds. The range can vary from a single video frame to 11 hours.
Views	View count is unsigned long integer. View count is number of times the video has been viewed. Maximum number of views possible are 9,223,372,036,854,775,808 (as defined by YouTube)
Comments	Comment count is unsigned long integer. Comment count is the number of comments for the video. There is no limit on maximum number of comments. YouTube restricts comment length to 500 characters
Likes	Like count is unsigned long integer. Like count is an integer number of the likes. There is no limit on maximum number of likes
Country	A 2-digit string of the country from where the video was uploaded. Country code is a string value and any country with a valid ISO country code can be searched upon.

After extracting the sample dataset from YouTube API (Figure 21 represents the snapshot of extracted dataset), this dataset can be fed into various Hadoop Technologies and meaningful results can be extracted and analyzed.

Following feature comparison analysis is performed in order to analyze which Hadoop Technology is suitable for YouTube Data Analysis project.

1. If MapReduce is to be used for YouTube Data analysis project, then we need to write complex business logic in order to successfully execute the join queries. We would have to think from map and reduce view of what is important and what is not important and which particular code little piece will go into map and which one will go into reduce side. Programmatically, this effort will become quite challenging as lot of custom code is required to successfully execute the business logic even for simplest tasks. Also, it may be difficult to map the data into schema format and lot of development effort may go in to deciding how map and reduce joins can function efficiently.
2. Pig is a procedural data flow language. A procedural language is a step by step approach defined by the programmers. Pig requires a learning curve since the syntax is new and different from SQL. Also, Pig requires more maintenance. The values of variables may not be retained; instead, the query needs to rerun in order to get the values from a variable. Moreover, Pig is a scripting language that is more suitable for prototyping and rapidly developing MapReduce based jobs. The data schema is not enforced explicitly in Pig and hence it becomes difficult to

map the data into schema format. Also, the error that Pig produces is not very user friendly. It just gives exec error even if the problem is related to syntax or type error. Pig development may require more time than hive but is purely based on the developer's familiarity with the pig code.

3. Hive provides a familiar programming model. It operates on query data with a SQL-based language. It is comparatively faster with better interactive response times, even over huge datasets. As data variety and volume grows, more commodity machines can be added without reducing the performance. Hence, Hive is scalable and extensible. Hive is very compatible and works with traditional data integration and data analytics tool. If we apply Hive to analyze the YouTube data, then we would be able to leverage the SQL capabilities of Hive-QL as well as data can be managed in a particular schema. Also, by using Hive, the development time can be significantly reduced.

After looking at the pros and cons, Hive becomes the obvious choice for this YouTube Data Analysis project.

Chapter 6

HADOOP SETUP

The following section explains a step by step approach of completing Hadoop setup on local machine.

6.1 Installing a Virtual Machine

Step 1 Create a virtual box on your operating system using the link below

http://www.oracle.com/technetwork/serverstorage/virtualbox/downloads/index.html

Step 2 Setup Hadoop on your virtual box using the link below

http://share.edureka.co/pydio/data/public/hadoop

Step 3 Import the file downloaded from “STEP 2” on your virtual machine

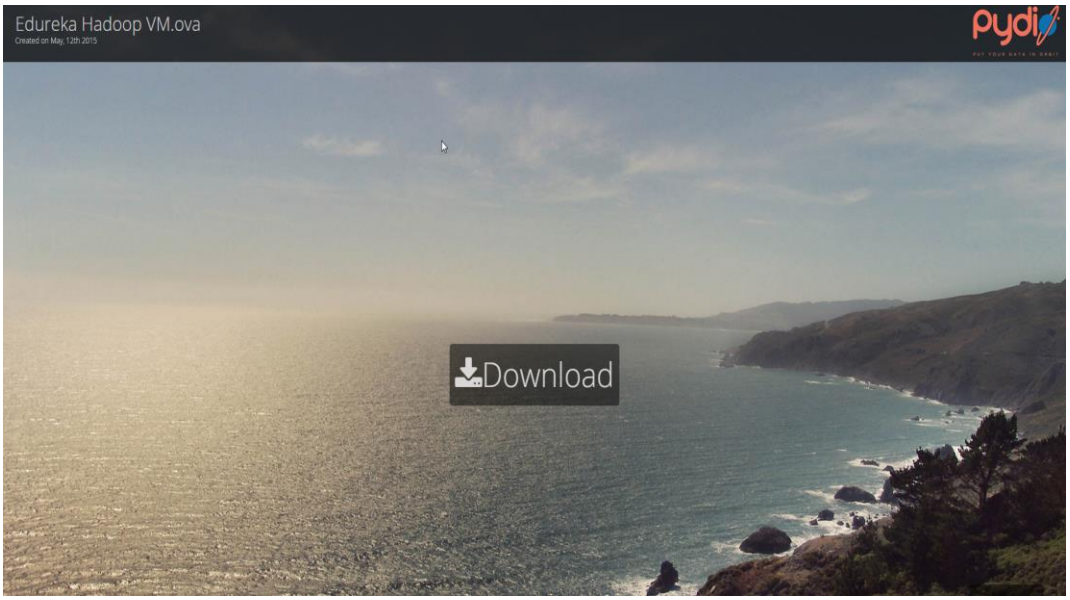

Figure 4 Hadoop VM.ova [13]

6.2 Running MapReduce

Step 1 Type command ‘sudo jps’ to check all the demons

- NameNode
- ResourceManager
- DataNode
- Jps
- NodeManger

Let us assume we are in the home directory of a Hadoop user

(e.g. /home/hadoop).

Step 2 To compile and execute a program in Hadoop use the following command.

This command is used to create a directory to store the compiled java classes.

```
$ mkdir units
```

Step 3 Download Hadoop-core-1.2.1.jar, which is used to compile and execute

MapReduce program using the link below

http://mvnrepository.com/artifact/org.apache.hadoop/hadoop-core/1.2.1

The following commands are used for compiling the ProcessUnits.java program

and creating a jar for the program.

```
$ javac -classpath hadoop-core-1.2.1.jar -d units
ProcessUnits.java
$ jar -cvf units.jar -C units/.
```

Step 4 Copy your YouTube Input from local machine into HDFS using

```
Hadoop dfs -copyFromLocal ‘your_destination_path’
hdfs:/
```

Step 5 Now run your jar file in Hadoop environment run command

```
Hadoop jar wc.jar hdfs:/datal.txt hdfs: out2'
```

Chapter 7

FETCHING YOUTUBE DATA USING API

The project utilizes the YouTube Data API that allows the applications/websites to incorporate functions that are used by YouTube application to fetch and view information. The aforementioned data can be used in numerous ways. To retrieve information from YouTube using their Data API our application needs to be authenticated. Once the application is authorized we can fetch data that can be used to analyze and represent.

7.1 Generate API Key to Fetch YouTube Data

To communicate with YouTube API an Application program interface Key is required, Google Developer allows you to create a unique key to connect to YouTube.

Step 1 Log into https://developers.Google.com/ with existing credentials.

Step 2 To create the unique API key for retrieving data, a new project needs to be created from the Google provided developer's console.

Step 3 Go to https://console.developers.Google.com/project

Step 4 Click create project.

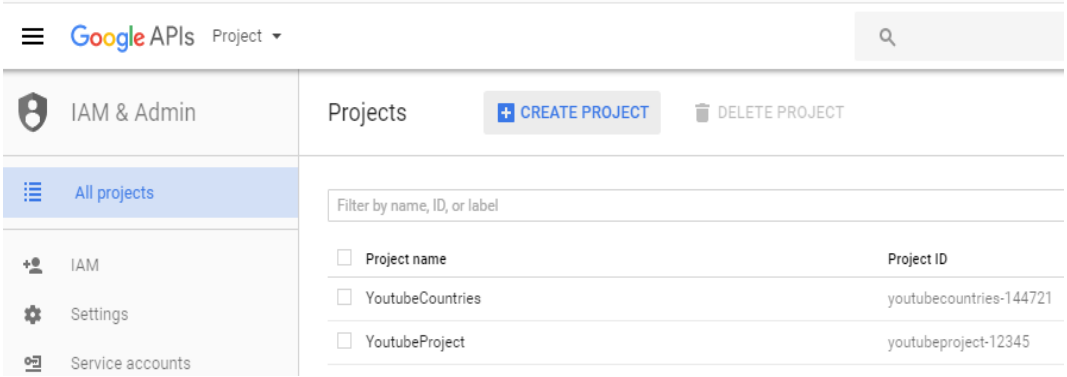

Figure 5 Creating A Project

Step 5 A new project needs to be created. Provide a name for the project

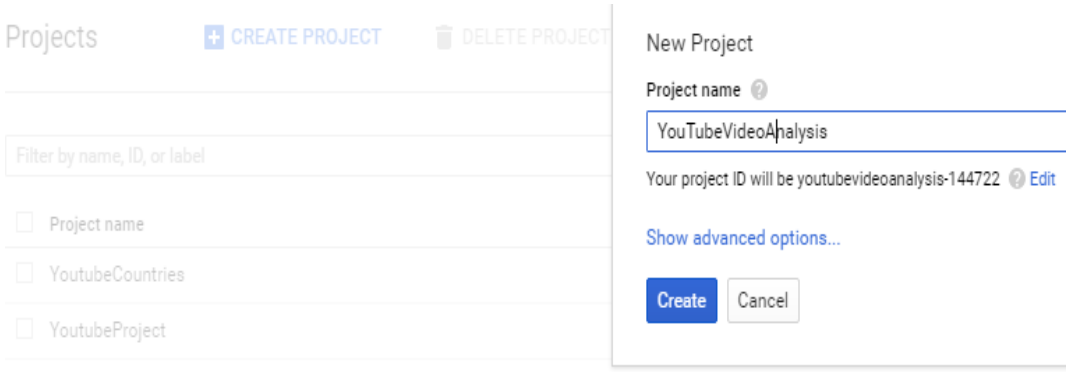

Figure 6 Providing a Name for Project

Step 6 To create a new API key Google provides the YouTube Data API that is available under the developer tools.

Library > YouTube APIs > YouTube Data API

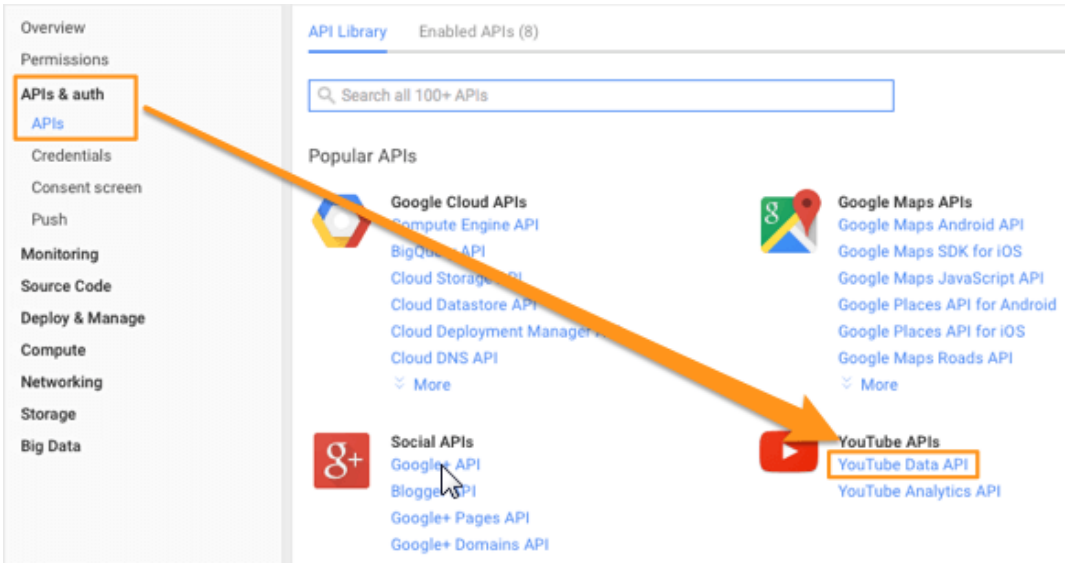

Figure 7 Getting the YouTube API Key [14]

Step 7 To utilize the YouTube Data API, it needs to be enabled under the logged in credentials. Click "Enable" under the YouTube Data API v3.

Dashboard > YouTube Data API v3: Enable

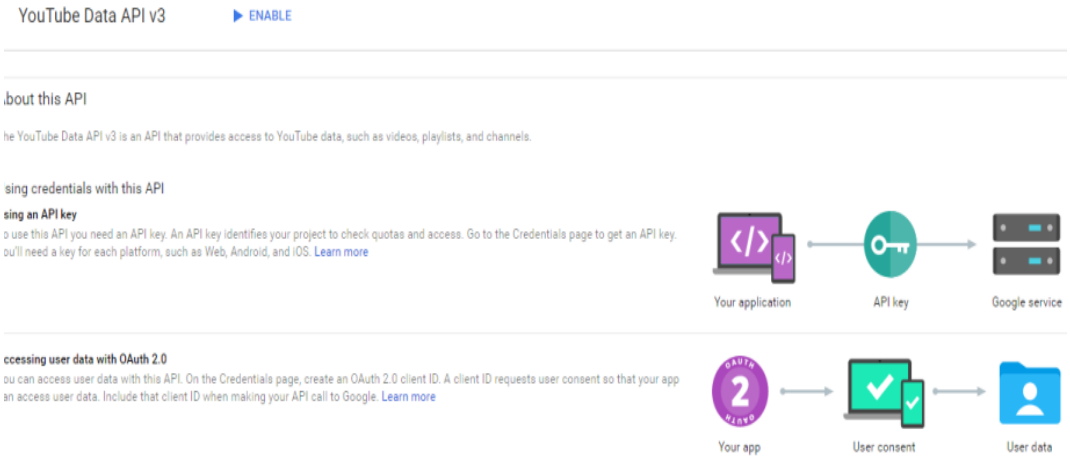

Figure 8 Enabling YouTube API Key

Step 8 Once the YouTube data API is enabled, create credentials in order to utilize the API. To create credentials

Dashboard > Go to Credentials Button

Step 9 Add credentials to the project. YouTube provides three options for creating an API Key.

- API key
- client ID
- service account

The project utilizes the "client ID" option

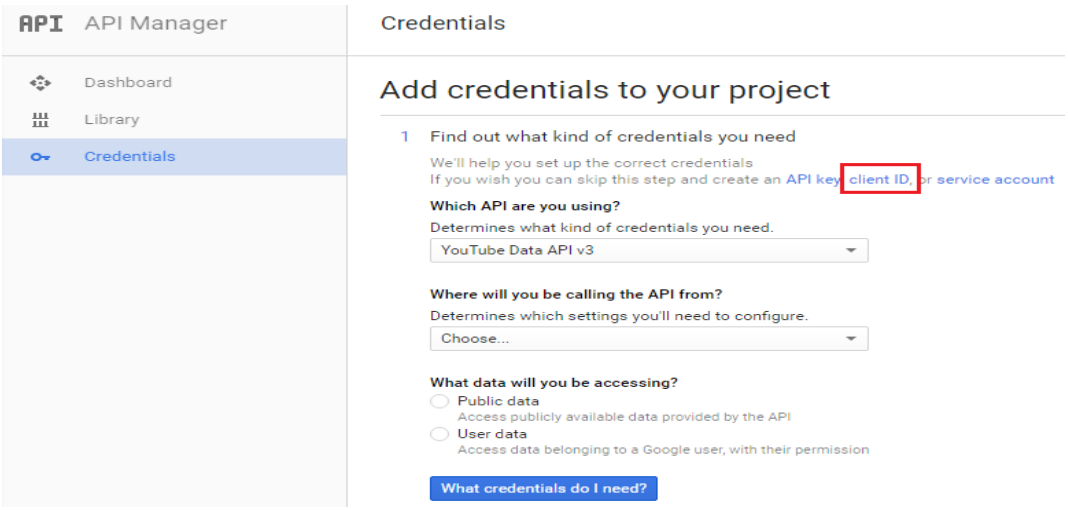

Figure 9 Creating a 'Client ID' API key

Step 10 *Create Client ID:* To create a JSON file to fetch data, we need to select the application that will be using the data. YouTube Data API offers the following options

- Web application
- Android
- Chrome App
- iOS
- PlayStation 4
- Other

The project utilizes the "**Other**" option

Step 11 Provide a name for the Client ID

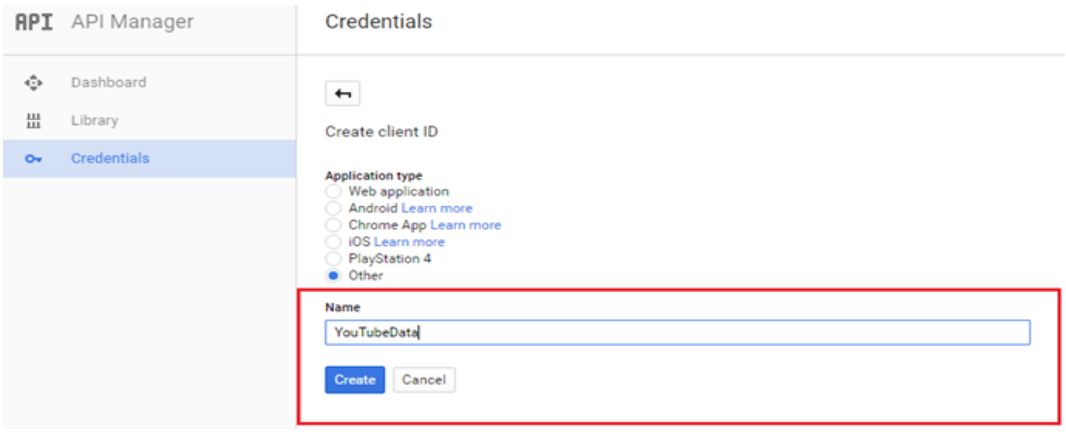

Figure 10 Providing Name for the Application

Step 12 YouTube creates the Client ID for the project to utilize and provides the API Key

Figure 11 Fetching the API key

Step 13 Once the OAuth 2.0 client IDs are created for the project the **"client_secret JSON"** file needs to be downloaded to be added to the project.

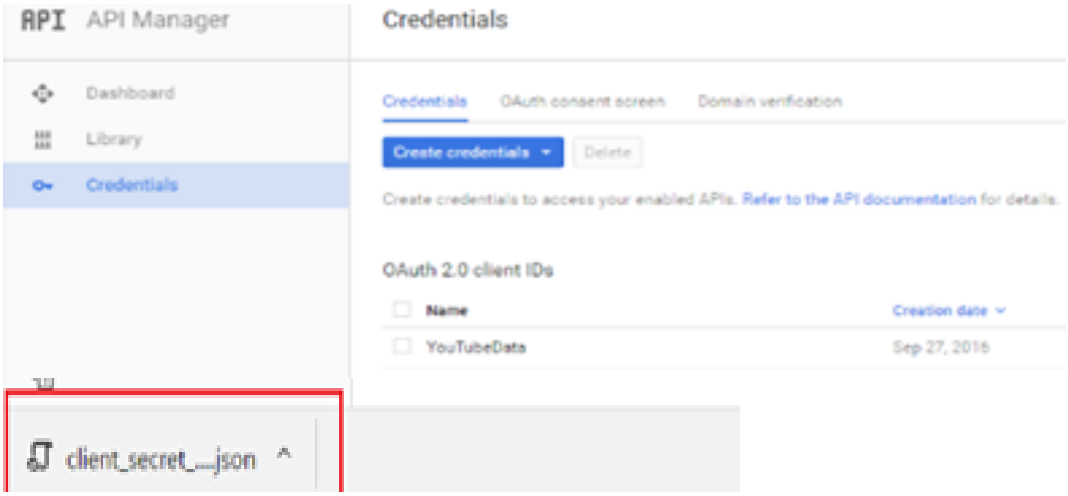

Figure 12 Fetching "client_secret JSON" file

7.2 Creating a .Net(C#) Console Application to Use the YouTube API

Step 1 Create a new c# console project

Step 2 Next step is to add the required DLLs to the .NET project. The Dlls can be downloaded from Google's NuGet Packages.

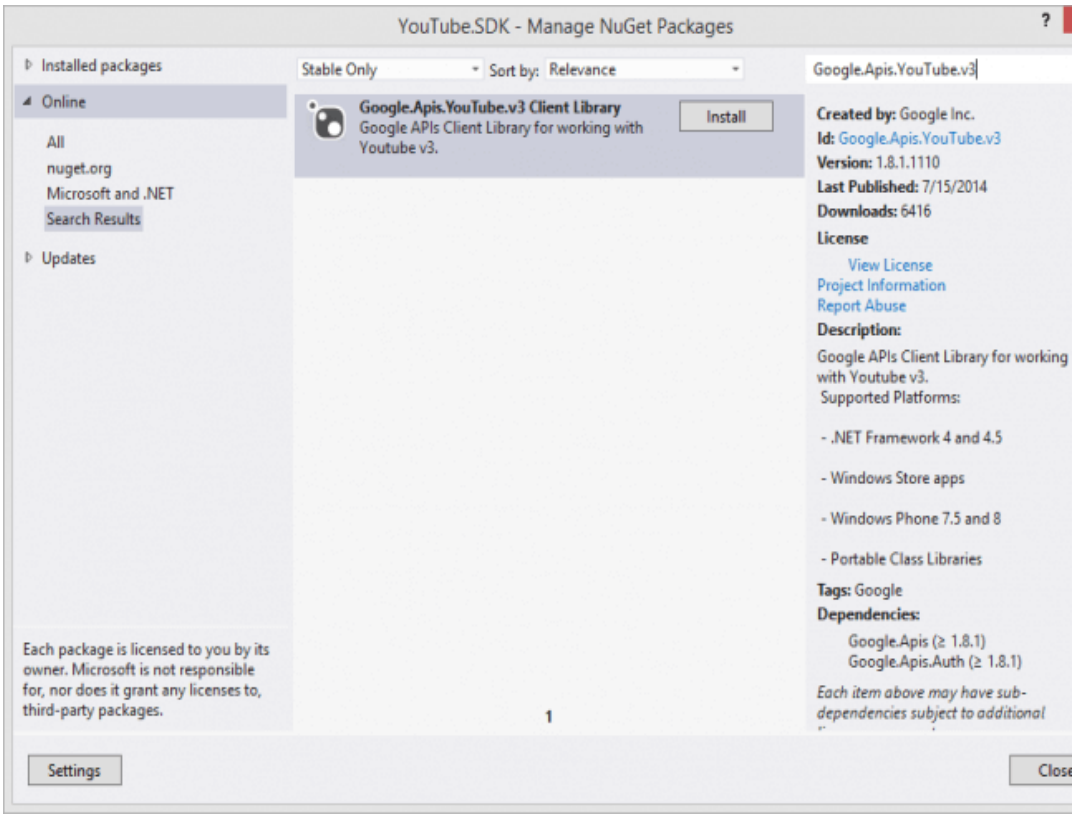

Figure 13 Adding the required DLLs To the Project

Open NuGet Console and search for "Google.Apis.YouTube.v3".

Step 3 Once all packages are installed, create a new class- **YouTubeInterfaceClass** and add the reference that will be used to fetch data.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using Google.Apis.Auth.OAuth2;
using Google.Apis.Services;
using Google.Apis.Util.Store;
using Google.Apis.YouTube.v3;
using Google.Apis.YouTube.v3.Data;
using System.IO;
using System.Net.Mime;
using System.Text;
using System.Xml;
```

Figure 14 References used in the Class For Fetching YouTube Data

Step 4 As mentioned above, Google's YouTube service needs to be authorized. We need to add the downloaded authentication JSON file to the project. To utilize the JSON file an Authorize function needs to be created.

```
private static YouTubeService ytService = Auth();


private static YouTubeService Auth()
{
UserCredential creds;
string appDirectory =
AppDomain.CurrentDomain.BaseDirectory.Replace(@"bin\Debug\", "");
using (var stream = new FileStream(appDirectory + "YouTube_Client.json",
FileMode.Open, FileAccess.Read))
{
creds = GoogleWebAuthorizationBroker.AuthorizeAsync(
GoogleClientSecrets.Load(stream).Secrets,
new[] { YouTubeService.Scope.YouTubeReadonly },
"user", System.Threading.CancellationToken.None, new
FileDataStore("YouTubeKey")).Result;
}
var service = new YouTubeService(new BaseClientService.Initializer()
{
HttpClientInitializer = creds,
ApplicationName = "YouTubeKey"
});

return service;
}
```

Figure 15 Creating an Authorization Function

The **Auth** function gets the user credentials from "YouTube_Client.json" and uses the credentials to establish service connection. **AppDirectory** variable gets the path of **"YouTube_Client.json"** file that has user credentials.

Step 5 Create an array to set a list of countries. The list of counties codes from taken from https://countrycode.org/

```
string [] countryCodes = new[] { "US", "IN", "CA", "BR" };
```

Figure 16 Array To Set Country Codes

Step 6 Create the header for the export header file

```
sbuilBuilder.Append("ID" + "\t" + "Channel Title"
                    + "\t" + "Published At"
                    + "\t" + "Category Name"
                    + "\t" + "Duration"
                    + "\t" + "View Count"
                    + "\t" + "Content Rating"
                    + "\t" + "Comment Count"
                    + "\t" + "Like Count"
                    + "\t" + "Country code"
```

Figure 17 Creating a Header

Step 7 Using YouTube services loop through the countries to get a list of records based on the search criteria.

```
var searchListRequest = YtService.Search.List("snippet");
        searchListRequest.Q = search; // Replace with your search term.
        searchListRequest.MaxResults = 50;
        searchListRequest.RegionCode = countryCode;
```

Figure 18 Getting List of records for a given Country Code

Step 8 Based on the search criteria list item. Loop through each video id to obtain additional information. The following modules were used to fetch the video information

- Snippet
- ContentDetails
- Localizations
- Statistics
- Status

Step 9 Populate the export file with the information about the video.

```
foreach (var searchListResponseElement in searchListResponse.Items)
{
if (searchListResponseElement.Id != null && searchListResponseElement.Id.VideoId
!= null)
{
//Fetching information for a given video ID
var videoRequest =
YtService.Videos.List("snippet,contentDetails,localizations,statistics,status");
videoRequest.Id = searchListResponseElement.Id.VideoId;
var singleReponse = videoRequest.Execute();

foreach (var obj in singleReponse.Items)
{
var cateName = "";
if (!string.IsNullOrEmpty(obj.Snippet.CategoryId) &&
categoryDictionary.ContainsKey(Convert.ToInt16(obj.Snippet.CategoryId)))
{
string value = categoryDictionary[Convert.ToInt16(obj.Snippet.CategoryId)];
cateName = value;
}
sbuilBuilder.Append(obj.Id
+ "\t" + obj.Snippet.ChannelTitle
+ "\t" + (obj.Snippet.PublishedAt.HasValue
? obj.Snippet.PublishedAt.Value.ToShortDateString()
: "null")
+ "\t" + cateName
+ "\t" + ReturTime(obj.ContentDetails.Duration)
+ "\t" + obj.Statistics.ViewCount
+ "\t" + (obj.ContentDetails.ContentRating != null ?
obj.ContentDetails.ContentRating.YtRating : null)
+ "\t" + obj.Statistics.CommentCount
+ "\t" + obj.Statistics.LikeCount
+ "\t" + countryCode
);
```

Figure 19 Populating the Export File

Step 10 Export the data into results.txt file

```
string appDirectory =
AppDomain.CurrentDomain.BaseDirectory.Replace(@"bin\Debug\", "");
File.WriteAllText(appDirectory + "Results.txt", sbuilBuilder.ToString());
```

Figure 20 Fetching The Final Data

Channel ID	Uploader Name	Punbilshed At	Category	Duration	View Count	Comment count	Like count	Country code
UB-jhT6Gg4o	DAAMS	1/8/2013	Music	1:33:05	4590995	1128	9554	US
vvjW90XlHHg	Pedro Angeles	7/30/2015	People & Blogs	0:56:24	1535133	230	5838	US
btrgg4jSiVs	Rclbeauty101	8/5/2016	Comedy	0:07:26	18080148	68864	335987	US
iAmI1ExVqt4	Fresh Movie Trailers	9/15/2016	Film & Animation	0:06:08	3236088	2022	15397	US
ZWpD9AgkT2s	Samuel Hjelm	1/9/2014	Film & Animation	0:48:49	3614437	1501	8864	US
ncqXZp3XTMQ	LewToons	10/1/2016	Film & Animation	0:18:26	25267	459	1865	US
Z47tn5d3IBQ	Screen Rant	10/2/2016	Entertainment	0:05:24	22719	221	1561	US
hNSUnH_Uj-A	Screen Rant	9/30/2016	Entertainment	0:06:37	66905	117	2365	US
x6q-NnxSZqw	Azuliso	5/7/2016	Entertainment	0:18:01	34988170	56418	241701	US
eINu9HyCe0w	DisneyToysFan	9/30/2016	Entertainment	0:08:22	579490	705	2978	US
0NWbAcu9_VU	Rclbeauty101	4/15/2016	Comedy	0:06:59	96463505	31078	464667	US
QbE0ISC4yEI	Disney	9/23/2016	Entertainment	0:02:25	932052	216	3150	US
TuZgFzkQIbo	FaZe Rain	9/30/2016	Entertainment	0:06:21	495925	4606	101193	US
UB-jhT6Gg4o	DAAMS	1/8/2013	Music	1:33:05	4590995	1128	9554	IN
vvjW90XlHHg	Pedro Angeles	7/30/2015	People & Blogs	0:56:24	1535133	230	5838	IN
btrgg4jSiVs	Rclbeauty101	8/5/2016	Comedy	0:07:26	18080148	68864	335987	IN
iAmI1ExVqt4	Fresh Movie Trailers	9/15/2016	Film & Animation	0:06:08	3236088	2022	15397	IN
ZWpD9AgkT2s	Samuel Hjelm	1/9/2014	Film & Animation	0:48:49	3614437	1501	8864	IN
ncqXZp3XTMQ	LewToons	10/1/2016	Film & Animation	0:18:26	25267	459	1865	IN
Z47tn5d3IBQ	Screen Rant	10/2/2016	Entertainment	0:05:24	22719	221	1561	IN
hNSUnH_Uj-A	Screen Rant	9/30/2016	Entertainment	0:06:37	66905	117	2365	IN
x6q-NnxSZqw	Azuliso	5/7/2016	Entertainment	0:18:01	34988170	56418	241701	IN
eINu9HyCe0w	DisneyToysFan	9/30/2016	Entertainment	0:08:22	579490	705	2978	IN
0NWbAcu9_VU	Rclbeauty101	4/15/2016	Comedy	0:06:59	96463505	31078	464667	IN
UGU5VmU5RKY	JOOGSQUAD PPJT	9/30/2016	Entertainment	0:17:19	104195	373	4222	IN
Fbf8wZPO2pk	DisneyChannelUK	10/2/2016	Entertainment	0:01:31	5356	12	275	IN
Hiw4W7PFQVc	pstoyreviews	10/1/2016	Entertainment	0:14:22	30484	218	718	IN
pPMNd-4fXMI	Avant et Après	9/4/2016	Entertainment	0:16:36	3035907	4833	15555	IN
Wtj8zf6bAX0	Los Juguetes de Titi	9/29/2016	Entertainment	0:18:28	418794	1030	7605	IN

Figure 21 Dataset Retrieved Through the C# Console Application

7.3 Solution Extraction Using HIVE

Step 1 Use the following command to 'Create a Table' in HIVE

```
Hive> create table YouTube_data_table (vedioid STRING,
uploader STRING, published_on STRING, category
STRING, length INT, noofviews INT, no_of_comments
INT,no_of_likes INT,country STRING,)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE;
```

This command will create a Hive table named ‘YouTube_data_table’ in which rows will be delimited and rows fields will be terminated by commas**.**

Step 2 Load YouTube data into the Hive Table

Use the command given below to load YouTube data into the Hive table created in

```
Hive> load data local
inpath'/home/MyYouTubeProject/YouTubeDataset.txt'
overwrite into table YouTube_data_table;
```

This command will load the YouTube dataset from the given path to the table (YouTube_data_table) created in Hive so that meaningful processing of the dataset can be done using the Hive queries.

7.4 Output Result

Step 1 Calculate top 10 channels with maximum number of likes

We can extract the top 10 channels with maximum number of likes using the following Hive query. The Hive select query will trigger the following MapReduce job:

```
hive> select vedioid, uploader, no_of_likes FROM
YouTube_data_table ORDER BY no_of_likes DESC LIMIT 10;
```

```
hive> select vedioid, uploader, no_of_likes FROM youtube_data_table ORDER BY no_of_likes DESC LIMIT 10;
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1475448366255_0001, Tracking URL = http://localhost:8088/proxy/application_1475448366255_0001/
Kill Command = /usr/lib/hadoop-2.2.0/bin/hadoop job  -kill job_1475448366255_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2016-10-03 04:18:21,308 Stage-1 map = 0%,  reduce = 0%
2016-10-03 04:18:25,593 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.67 sec
2016-10-03 04:18:30,808 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.33 sec
MapReduce Total cumulative CPU time: 1 seconds 330 msec
Ended Job = job_1475448366255_0001
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative CPU: 1.33 sec   HDFS Read: 14167 HDFS Write: 276 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 330 msec
```

Figure 22 Hive select Query triggering MapReduce Job

Output Result

```
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative CPU: 1.72 sec   HDFS Read: 278161 HDFS Write: 409 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 720 msec
OK
12Z3J1uzd0Q     kaejane         6534
4DC4Rb9quKk     ChrisBrownTV    3375
LU8DDYz68kM     Jason275        2772
kHmvkRoEowc     itschriscrocker 1823
Md6rURKhZmA     TPainVideos     1814
EwTZ2xpQwpA     TayZonday       1684
A2f3cuUXXRs     RKellyTV        1303
rZBA0SKmQy8     TownIdiot25     1100
irp8CNj9qBI     Frozentoast     1017
ZCYaw5tGYAs     nozzle49        8944
Time taken: 17.882 seconds, Fetched: 10 row(s)
```

Figure 23 Output of Top 10 channels with maximum number of likes

The output result describes that for a specific video id, how many likes were received. The number of likes -- or "thumbs-up" -- a video had has a direct significance to the YouTube video's ranking, according to YouTube Analytics. So if a company posts its video on YouTube, then the number of YouTube likes the company has could determine whether the company or its competitors appear more prominently in YouTube search results. The output result shows number of likes for "Disney" channel videos.

Step 2 Calculate top 5 categories with maximum number of comments

```
hive > select category, max(no_of_comments) as
max_no_of_comments from YouTube_data_table GROUP ORDER
BY max_no_of_comments DESC LIMIT 5;
```

Hive select query will trigger the following MapReduce job:

```
hive> select category, max(no_of_comments) as max_no_of_comments from youtube_data_table GROUP  ORDER BY max_no_of_comments D
BY (category) ORDER BY max_no_of_comments DESC LIMIT 5;
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1475448366255_0002, Tracking URL = http://localhost:8088/proxy/application_1475448366255_0002/
Kill Command = /usr/lib/hadoop-2.2.0/bin/hadoop job  -kill job_1475448366255_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2016-10-03 04:40:59,893 Stage-1 map = 0%,  reduce = 0%
2016-10-03 04:41:05,399 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.63 sec
2016-10-03 04:41:10,648 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.33 sec
MapReduce Total cumulative CPU time: 1 seconds 330 msec
Ended Job = job_1475448366255_0002
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1475448366255_0003, Tracking URL = http://localhost:8088/proxy/application_1475448366255_0003/
Kill Command = /usr/lib/hadoop-2.2.0/bin/hadoop job  -kill job_1475448366255_0003
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2016-10-03 04:41:23,172 Stage-2 map = 0%,  reduce = 0%
2016-10-03 04:41:27,401 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 0.65 sec
```

Figure 24 Hive select Query triggering MapReduce Job

Output Results

```
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative CPU: 1.33 sec   HDFS Read: 14167 HDFS Write: 381 SUCCESS
Job 1: Map: 1  Reduce: 1   Cumulative CPU: 1.38 sec   HDFS Read: 747 HDFS Write: 77 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 710 msec
OK
Entertainment   97
Film & Animation        806
Education       7429
Comedy  68864
Music   6591
Time taken: 39.75 seconds, Fetched: 5 row(s)
hive>
```

Figure 25 Output of top 5 categories with maximum number of comments

According to data gathered in the dataset, the output results show 'Entertainment' and 'Film & Animation' are some of the bigger categories. On the other hand, number of comments generated were most for the videos tagged under 'Comedy' category. This output shows if there is a pattern of affinity of interests for certain category and for which category, a meaningful discussion can be triggered. For e.g., if a company falls under 'Entertainment' section, they would be better off launching their new

product/service on YouTube. On the other hand, if the company falls under 'Comedy' or 'Education' category, a meaningful discussion in the form of comments can be triggered on YouTube. A comment analysis can further be conducted to understand the attitude of people towards the specific video.

Step 3 Calculate Country wise analysis of top 15 videos with maximum number of views

Hive select query triggered the following MapReduce job

```
hive> select vedioid, uploader, noofviews FROM youtube_data_table WHERE country
= 'US' ORDER BY noofviews  DESC LIMIT 10;
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1475453414176_0001, Tracking URL = http://localhost:8088/prox
y/application_1475453414176_0001/
Kill Command = /usr/lib/hadoop-2.2.0/bin/hadoop job  -kill job_1475453414176_000
1
```

Figure 26 Hive select Query triggering MapReduce Job

Output Results

```
Total MapReduce CPU Time Spent: 2 seconds 100 msec
OK
w7zW6m7Fq8      Clevver News    99962
NWbAcu9_VU      Rclbeauty101    96463505
N6K05_bO6k      Disney Television Animation News        9324
bE0ISC4yEI      Disney  932052
r3qoezFYoY      DisneyToysFan   898419
LMeDUBKmBs      Voctave 8391025
jik3zsGNF4      Comicbook.com   7201395
NSUnH_Uj-A      Screen Rant     66905
x0eO9DPpas      DisneyToysFan   634743
-KP9FoROZc      Screen Rant     6281227
Time taken: 24.226 seconds, Fetched: 10 row(s)
```

Figure 27 Output of Country wise analysis of top 15 videos with number of views

The above output displays the Top 15 videos with maximum number of views for a specific channel in a specific country. This type of analysis can be used for multi-national companies who have a strong YouTube footprint. YouTube has a very large viewer and subscriber base all over the world. When companies like Disney launch their movie trailers, an analysis can be performed to identify which trailer was liked the most in which country. Based upon that analysis, a global company like Disney can distribute the marketing budgets at country level.

Chapter 8

CONCLUSION

8.1 Conclusion

The task of big data analysis is not only important but also a necessity. In fact many organizations that have implemented Big Data are realizing significant competitive advantage compared to other organizations with no Big Data efforts. The project is intended to analyze the YouTube Big Data and come up with significant insights which cannot be determined otherwise.

The output results of YouTube data analysis project show key insights that can be extrapolated to other use cases as well. One of the output results describes that for a specific video id, how many likes were received. The number of likes -- or "thumbs-up" -- a video had has a direct significance to the YouTube video's ranking, according to YouTube Analytics. So if a company posts its video on YouTube, then the number of YouTube likes the company has could determine whether the company or its competitors appear more prominently in YouTube search results.

Another output result gives us insights on if there is a pattern of affinity of interests for certain video category. This can be done by analyzing the comments count. For e.g., if the company falls under 'Comedy' or 'Education' category, a meaningful discussion in the form of comments can be triggered on YouTube. A comment analysis can further be conducted to understand the attitude of people towards the specific video.

8.2 Future Work

The future work would include extending the analysis of YouTube data using other Big Data analysis Technologies like Pig and MapReduce and do a feature comparison analysis. It would be interesting to see which technology fares better as compared to the other ones.

One feature that is not added in the project is to represent the output in a Graphical User Interface (GUI). The current project displays a very simplistic output which does not warrant a GUI interface. However, if the output is too large and complex, the output can be interfaced in a GUI format to display the results. The data can then be presented in different format including pie-charts and graphs for better user experience.

Another possible extension of this project could be the YouTube Comment Analysis project. The current scope of the project includes analyzing the statistics for a channel/category including view counts, likes, dislikes, country wise view etc. By identifying classifying/categorizing the polarity of the words, sentiment analysis or opinion minding can be performed for a specific video. This would tell us writer's attitude towards a particular product or a given subject. Using Sentiment Analysis, we can determine if the general attitude of people is positive, negative or neutral towards a specific subject/video.

APPENDIX

Source Code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using Google.Apis.Auth.OAuth2;
using Google.Apis.Services;
using Google.Apis.Util.Store;
using Google.Apis.YouTube.v3;
using Google.Apis.YouTube.v3.Data;
using System.IO;
using System.Net.Mime;
using System.Text;
using System.Xml;

namespace YouTubeTagExtraction
{
public class YouTubeInterfaceClass
{
private static readonly YouTubeService YtService = Auth();

private static YouTubeService Auth()
{
//What Auth Function does ?
//Gets the UserCredentials from "YouTube_Client.json"
//Uses the UserCredentials to establish service connection with YouTube data.
//return the service connection.

UserCredential creds;

//Getting user creds from saved "YouTube_Client.json" file.
//Getting file path and file name

//appDirectory gets the path of the "YouTube_Client.json" file that has the UserCredential.
string appDirectory = AppDomain.CurrentDomain.BaseDirectory.Replace(@"bin\Debug\", "");
using (var stream = new FileStream(appDirectory + "YouTube_Client.json", FileMode.Open,
FileAccess.Read))
{
creds = GoogleWebAuthorizationBroker.AuthorizeAsync(
```

```
GoogleClientSecrets.Load(stream).Secrets,
new[] { YouTubeService.Scope.YouTubeReadonly },
"user", System.Threading.CancellationToken.None, new FileDataStore("YouTubeKey")).Result;
}

//Passing the variable to set connection to get data from YouTube
var service = new YouTubeService(new BaseClientService.Initializer()
{
HttpClientInitializer = creds,
ApplicationName = "YouTubeKey"
});

return service;
}

public static void GetVideoInfoBySearchCriteria(string search)
{
//Get the list of category id and name
var categoryDictionary = GetCategoryDictionary();

//Setting list of countries to get videos by Region
string[] countryCodes = new[] { "US", "IN", "CA", "BR" };
////List of country codes https://countrycode.org/


//Create Stringbuilder to hold the data that will be exported to text file
StringBuilder sbuilBuilder = new StringBuilder();

//Create header for export text file

//------------------- Comment/Remove this part if the header is not needed ------------------
sbuilBuilder.Append("ID" + "\t" + "Channel Title"
+ "\t" + "Published At"
+ "\t" + "Category Name"
+ "\t" + "Duration"
+ "\t" + "View Count"
+ "\t" + "Content Rating"
+ "\t" + "Comment Count"
+ "\t" + "Like Count"
+ "\t" + "Country code"
);
sbuilBuilder.Append(Environment.NewLine);
//------------------- END Comment/Remove this part if the header is not needed ---------------
```

```
//First foreach looop will run for all the country codes in countryCodes[Array]
// It executes the YouTube services's Search module that allows us to fetch data for the given
search criteria and get the snippet information for 50 record. It filters it by country code

//Run the code for each country code provided in countryCodes variable
foreach (var countryCode in countryCodes)
{
//Using the search function in YouTube services
var searchListRequest = YtService.Search.List("snippet");
//Adding Parameter
searchListRequest.Q = search; // Replace with your search term.
searchListRequest.MaxResults = 50;
searchListRequest.RegionCode = countryCode;

//We got 50 records based on our search criteria and we got their video IDs

var searchListResponse = searchListRequest.Execute();


foreach (var searchListResponseElement in searchListResponse.Items)
{

if (searchListResponseElement.Id != null && searchListResponseElement.Id.VideoId != null)
{
//Fetching information for a given video ID
var videoRequest = YtService.Videos.List("snippet,contentDetails,localizations,statistics,status");
videoRequest.Id = searchListResponseElement.Id.VideoId;
var singleReponse = videoRequest.Execute();

foreach (var obj in singleReponse.Items)
{
var cateName = "";
if (!string.IsNullOrEmpty(obj.Snippet.CategoryId) &&
categoryDictionary.ContainsKey(Convert.ToInt16(obj.Snippet.CategoryId)))
{
string value = categoryDictionary[Convert.ToInt16(obj.Snippet.CategoryId)];
cateName = value;
}


sbuilBuilder.Append(obj.Id
```

```
+ "\t" + obj.Snippet.ChannelTitle
+ "\t" + (obj.Snippet.PublishedAt.HasValue
? obj.Snippet.PublishedAt.Value.ToShortDateString()
: "null")
+ "\t" + cateName
+ "\t" + ReturTime(obj.ContentDetails.Duration)
+ "\t" + obj.Statistics.ViewCount
+ "\t" + (obj.ContentDetails.ContentRating != null ? obj.ContentDetails.ContentRating.YtRating :
null)
+ "\t" + obj.Statistics.CommentCount
+ "\t" + obj.Statistics.LikeCount
+ "\t" + countryCode
);
}
sbuilBuilder.Append(Environment.NewLine);
}

}
}

string appDirectory = AppDomain.CurrentDomain.BaseDirectory.Replace(@"bin\Debug\", "");
File.WriteAllText(appDirectory + "Results.txt", sbuilBuilder.ToString());
}

public static string ReturTime(string duration)
{
//PT 2H 1M 15S
TimeSpan ts = XmlConvert.ToTimeSpan(duration);
return ts.ToString();
}

public static Dictionary<int, string> GetCategoryDictionary()
{
//Dictionary(Key,Value) variable created to get video category
Dictionary<int, string> dictionary = new Dictionary<int, string>();
dictionary.Add(1, "Film & Animation");
dictionary.Add(2, "Autos & Vehicles");
dictionary.Add(10, "Music");
dictionary.Add(15, "Pets & Animals");
dictionary.Add(17, "Sports");
dictionary.Add(19, "Travel & Events");
dictionary.Add(20, "Gaming");
dictionary.Add(22, "People & Blogs");
```

```
dictionary.Add(23, "Comedy");
dictionary.Add(24, "Entertainment");
dictionary.Add(25, "News & Politics");
dictionary.Add(26, "Howto & Style");
dictionary.Add(27, "Education");
dictionary.Add(28, "Science & Technology");
dictionary.Add(29, "Nonprofits & Activism");

return dictionary;
}

}
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using YouTubeTagExtraction;
namespace YouTubeTagExtraction
{
class Program
{
static void Main(string[] args)
{
//First function that will be executed.
//search Criteria  to perform search on YouTube API
string searchCriteria = "Music";

//passing the searchCriteria variable to get video list by tag.
YouTubeInterfaceClass.GetVideoInfoBySearchCriteria(searchCriteria);

}
}
}
```

BIBLIOGRAPHY

[1] **Wikipedia.org. 2016**. Big Data. *https://en.wikipedia.org/wiki/Big_data*. [Online] February 2016. https://en.wikipedia.org/wiki/Big_data.

[2] **Youtube.com. 2016.** YouTube for media. *https://www.YouTube.com/yt/press/statistics.html*. [Online] March 2016. https://www.YouTube.com/yt/press/statistics.html.

[3] **Datanami.com. 2016.** Mining for YouTube Gold with Hadoop and Friends *https://www.datanami.com* [Online] July 2016. https://www.datanami.com/2014/11/12/mining-YouTube-gold-hadoop-friends/.

[4] **3pillarglobal.com. 2016.** How To Analyze Big Data With Hadoop Technologies *http://www.3pillarglobal.com/*. [Online] June 2016. http://www.3pillarglobal.com/insights/analyze-big-data-hadoop-technologies

[5] **Cs.ubc.ca. 2016.** University of British Columbia, Department of Computer Science. Brief Introduction to Database Systems. *https://www.cs.ubc.ca/*. [Online] August 2016. http://www.cs.ubc.ca/nest/dbsl/intro.html.

[6] **H. Garcia-Molina, J. D. Ullman and J. Widom. 2009.** *Database System Implementation: The complete book*, 2nd edition. New Jersey: Prentice-Hall, Inc. 2009

[7] **R. Elmaseri and S. Navathe**. **2006.** *Fundamentals of Database Systems*, 5th edition. Boston: Pearson / Addison Wesley. 2006

[8] **Oracle.com. 2016.** Java swings and concepts. *http://docs.oracle.com/*. [Online] August 2016. http://docs.oracle.com/javase/tutorial/uiswing/

[9] **Statista.com. 2016.** Statistics and facts about YouTube. *https://www.statista.com/* [Online] August 2016. https://www.statista.com/topics/2019/

[10] **Ajinkya Ingle, Anjali Kante, Shriya Samak, Anita Kumari. 2005.** Sentiment Analysis of Twitter Data Using Hadoop. *http://www.pnrsolution.org/*. [Online]. May 2016. http://www.pnrsolution.org/Datacenter/Vol3/Issue6/18.pdf

[11] **Resources Management Association (IRMA). 2016.** Information. "Chapter 1 - Big Data Overview". Big Data: Concepts, Methodologies, Tools, and Applications, Volume I. IGI Global. http://common.books24x7.com/toc.aspx?bookid=114046 (accessed September 2016)

[12] **deRoos et al., Dirk. 2014.** "Chapter 4 - Storing Data in Hadoop—The Hadoop Distributed File System". Hadoop for Dummies. John Wiley & Sons. http://common.books24x7.com/toc.aspx?bookid=62641 (accessed August, 2016)

[13] **Share.edureka.co. 2016.** Edureka Hadoop VM Download. *http://share.edureka.co/* [Online] February 2016. http://share.edureka.co/pydio/data/public/hadoop

[14] **Appcoda.com. 2016.** Building a Video Search App with YouTube. *https://www.appcoda.com*. [Online] August 2016. https://www.appcoda.com/YouTube-api-ios-tutorial/