

**UNDERGRADUATE PROJECT (CSD- 428) REPORT
ON TWITTER DATA ANALYSIS USING HADOOP
ECOSYSTEM**

B.Tech. Major Project Report

By

PRASHANT ARORA

(Registration Number: 15MI544)

MAYANK KUMAR

(Registration Number: 15MI538)

ASHUTOSH DUBEY

(Registration Number: 15MI537)

GOPAL KHATRI

(Registration Number: 15MI520)

ANUBHAV MARKANDAY

(Registration Number: 15MI521)



**DEPARTMENT OF COMPUTER SCIENCE
ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
HAMIRPUR-177005, HP (INDIA)**

MAY, 2019

TWITTER DATA ANALYSIS USING HADOOP ECOSYSTEM

A PROJECT REPORT

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

By

PRASHANT ARORA

(Registration Number: 15MI544)

MAYANK KUMAR

(Registration Number: 15MI538)

ASHUTOSH DUBEY

(Registration Number: 15MI537)

GOPAL KHATRI

(Registration Number: 15MI520)

ANUBHAV MARKANDAY

(Registration Number: 15MI521)

*Under the guidance
of*

Dr. PARDEEP SINGH



COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

NATIONAL INSTITUTE OF TECHNOLOGY

HAMIRPUR -177005 (INDIA)

May, 2019

Copyright © NIT HAMIRPUR (HP), INDIA, 2019

**NATIONAL INSTITUTE OF TECHNOLOGY
HAMIRPUR (HP)**

CANDIDATE DECLARATION

We hereby certify that the work which is being presented in the project titled “**Twitter Data Analysis using Hadoop Ecosystem**” in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology and submitted in the Computer Science and Engineering Department, National Institute of Technology, Hamirpur, is an authentic record of our own work carried out during a period from January 2019 to May 2019 under the supervision of **Dr. Pardeep Singh**, Associate Professor, Computer Science Engineering Department, National Institute of Technology Hamirpur.

The matter presented in this report has not been submitted by us for the award of any other degree of this or any other Institute/University.

Prashant Arora

Mayank Kumar

Ashutosh Dubey

Gopal Khatri

Anubhav Markanday

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

**Dr. Pardeep Singh, Associate Professor
National Institute of Technology, Hamirpur (HP)**

The project Viva-Voce Examination of Prashant Arora, Mayank Kumar, Ashutosh Dubey, Gopal Khatri, Anubhav Markanday has been held on

Signature of Supervisor

Signature of External Examiner

ACKNOWLEDGEMENT

It is our privilege and solemn duty to express our deepest sense of gratitude to **Dr. Pardeep Singh**, Assistant Professor at the Department of Computer Science and Engineering, under whose able guidance we carried out this work. We are indebted to him for his invaluable supervision, heart full cooperation, timely aid and advice till the completion of the report in spite of his pressing engagements.

We wish to record our sincere gratitude to respected Director Sir Dr. Vinod Yadava, for allowing our little mind to think in different directions and help us broaden horizons by making available all the necessary amenities needed in the working of our project. We take this opportunity to express our thanks to all those who helped us in the completion of our project work. We are very grateful to the author of various research papers, for helping us become aware of the research currently ongoing in the field.

We are very thankful to our parents for their constant support and love. Last, but not least, we would like to thank our classmates for their valuable comments, suggestions and support.

Prashant Arora

Mayank Kumar

Ashutosh Dubey

Gopal Khatri

Anubhav Markanday

ABSTRACT

In the era of the Internet, social media has become an integral part of modern society. People use social media to share their opinions and to have an up-to-date knowledge about the current trends on a daily basis. Throughout the last years ‘BIG DATA’ has been getting much importance in different industries, on a scale that has generated lots of data every day. Big Data is a term applied to data sets of very large size such that the traditional databases are unable to process their operations in a reasonable amount of time [2]. Here, the challenge is not only storing the data, but also accessing and analysing the required data in specified amount of time [1].

Twitter is one of the renowned social media that gets a huge number of tweets each day. This information can be used for economic, industrial, social or government approaches by arranging and analysing the tweets as per our demand. Since Twitter contains a huge volume of data, storing and processing this data is a complex problem. Hadoop is a big data storage and processing tool for analysing data with 3Vs, i.e. data with huge volume, variety and velocity [1]. In this project, tweets are being streamed into HDFS using Apache Flume and Twitter API. As an analysis, we have performed general analysis on tweets that includes finding top hashtags, rating per source user platform, user with most tweets and sentiment analysis on those tweets. The analysis is done using Hadoop ecosystem tools such as Apache Hive, Apache Flume, HDFS.

CONTENTS

Title	Page No.
CANDIDATE’S DECLARATION.....	i
ACKNOWLEDGEMENT.....	ii
ABSTRACT.....	iii
CONTENT.....	iv
LIST OF FIGURES.....	vi
 CHAPTER 1: INTRODUCTION	
1.1 Motivation.....	01
1.2 Objective.....	01
1.3 Big Data.....	03
1.4 Challenges.....	03
 CHAPTER 2: LITERATURE SURVEY	
2.1 Why Twitter?.....	05
2.2 Why Hadoop?.....	05
 CHAPTER 3: SYSTEM OVERVIEW	
3.1 Technologies used.....	07
3.1.1 Apache Hadoop.....	07
3.1.2 Apache Flume.....	08
3.1.3 Apache Hive.....	10
3.2 Hadoop Framework Architecture.....	12
3.2.1 Hadoop common utilities.....	12
3.2.2 MapReduce.....	12
3.2.3 SerDe.....	14
3.2.4 JSON Format.....	15

CHAPTER 4: IMPLEMENTATION AND CODING

4.1 Creating Twitter API.....	16
4.2 Starting HDFS.....	17
4.3 Configuring Flume to copy data into HDFS.....	18
4.4 Hive query for analysis over Twitter data.....	21
4.5 Finding Recent Trends and Most Popular hashtags.....	22
4.6 Finding user with the greatest number of tweets	23
4.7 Performing sentiment analysis on the tweet.....	24

CHAPTER 5: CONCLUSION AND RESULT

5.1 Conclusion.....	28
5.2 Result.....	28

CHAPTER 6: FUTURE SCOPE..... 29

BIODATA OF CANDIDATES..... 30

REFERENCES..... 31

LIST OF FIGURES

Sr. No.	Caption	Page No.
Fig 1.1	Design methodology of hadoop ecosystem	02
Fig 3.1	Layers of Hadoop	08
Fig 3.2	Basic architecture of Flume	09
Fig 3.3	Flume in action	10
Fig 3.4	Flowchart of functioning of hadoop ecosystem	11
Fig 3.5	Detailed view of functioning of MapReduce	13
Fig 4.1	Twitter API app details	17
Fig 4.2	HDFS Web UI	21
Fig 4.3	Most popular hashtags	23
Fig 4.4	User with maximum number of tweets	24
Fig 4.5	Tokenization of each word in tweets	26
Fig 4.6	Tweets classification	28

CHAPTER 1

INTRODUCTION

1.1 Motivation

With digitization, there has been a drastic increase in the usage of some of the popular social media sites such as Twitter, Facebook, Yahoo, and YouTube as well as e-commerce sites as in Flipkart and Amazon, which have resulted in the generation of large sets of data. If the data is of small size, it is very easy to extract useful information, but if the size of data is huge, then it is quite difficult to analyse what that data actually intends. Ultimately, processing and extracting only the useful information is a tedious job [1]. The advancement in technology has made it easy for people around the globe to express, share their views openly and every potential web user depends on this review and opinions to take decisions. Considering the popular micro-blogging forum Twitter, where every minute millions of tweets are being posted, which greatly varies with the field of topic, analysing these tweets, identifying what they express is tedious. In a situation where an organization is interested to know its customers' opinion regarding the product that it has manufactured, it is very difficult for organizations to keep track of each and every opinion. It turns out that the biggest challenge is in processing this type of data efficiently so that the information out of this data can be extracted for further survey. All the traditional data analysis techniques have gradually failed to perform analysis effectively on larger data sets. Recently, the powerful framework that has proved to be efficient in processing large sets of data is Hadoop, which is considered to be efficient for distributed processing as well as distributed storage of large sets of data.

1.2 Objective

The objective of this project is to do general analysis on the tweets that can help improve the user experience over a social network or system interface. This project provides a general analysis of twitter data to find out what are the top hash tags, users who have done most retweets and sentiment analysis of various tweets by users [3]. We have considered twitter API streaming data on elections 2019 which are stored in Hadoop's file system. Twitter data is extracted using Apache Flume

(<https://flume.apache.org/>). The data being extracted would be in JSON (JavaScript Object Notation) format. In JSON format, every data is represented in key/value pairs and separated by commas.

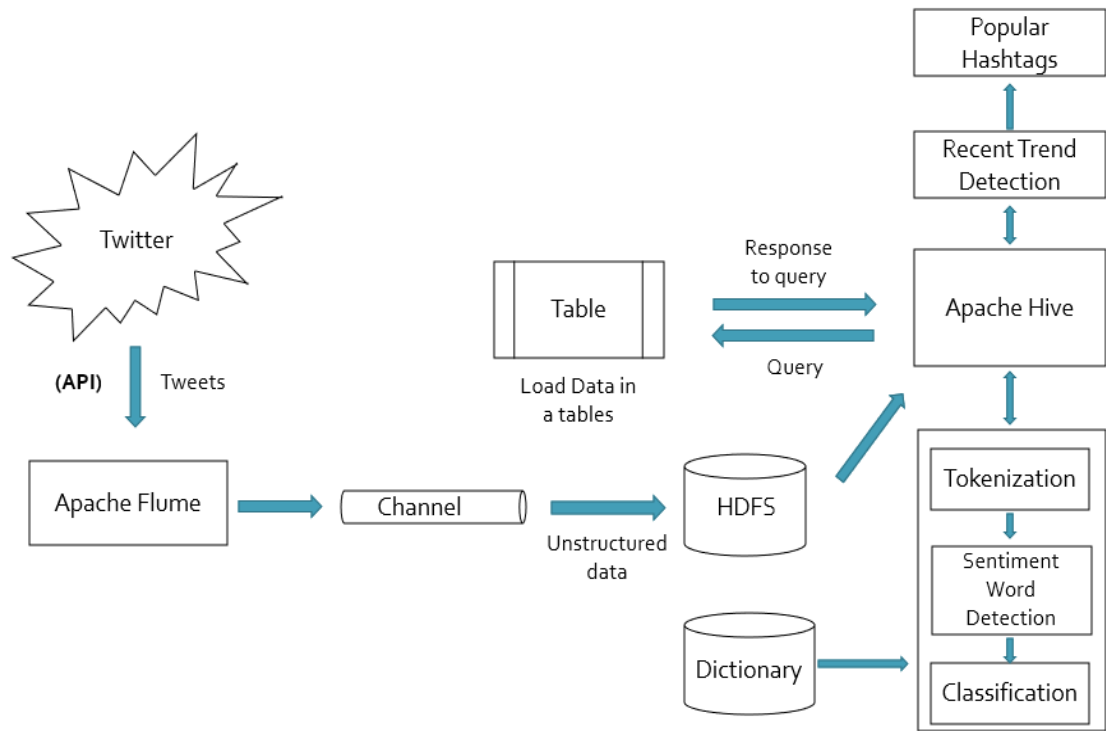


Fig 1.1: Design methodology of hadoop ecosystem

In text mining, Sentiment Analysis and Opinion Mining consists study of sentiments, attitudes, reactions, evaluation of the content of the text. Many times, while analyzing peoples' opinions, sentiments, evaluations, attitudes, reactions towards entities, such as services, products, organizations, individuals, events, topics, issues and their attributes Sentiment Analysis is also called Opinion Mining. Every minute opinion, reviews are expressed online and a potential user rely on these reviews, opinions, feedback given by various other users to make decisions with respect to purchasing an item or developing a software when it comes to an organization that provides services. Social media is popular with marketing teams. It is an effective tool for promotion because it helps companies to know about success of their campaigns [6].

1.3 Big data

Big data is a collection of large amounts of data so that it can be analyzed for making strategic business moves and decisions. The data that is gathered can be either structured or unstructured. Big data technologies are important in providing more accurate analysis, which may lead to more concrete decision-making resulting in greater operational efficiencies, cost reductions, and reduced risks for the business. To harness the power of big data, you would require an infrastructure that can manage and process huge volumes of structured and unstructured data in real time and can protect data privacy and security. There are various technologies in the market from different vendors including Amazon, IBM, Microsoft, etc., to handle big data [3].

The definition for big data was given by Doug Laney in early 2000s as 3 Vs: Volume, Velocity and Variety, Veracity:

Volume: The volume of data that is generated, which is being gathered for processing.

Velocity: The rate at which the data that is being generated is collected in storage.

Variety: The different type of data that is stored for processing [2].

1.4 Challenges

The world has entered the age of information, data sources are increasing at an unprecedented rate. Big data plays a large role in managing and analyzing the data from these sources but there are many challenges in harnessing the potential of big data:

- **Data Complexity:** Big data is a collection of various datatypes of highly complex data, which is a mixture of structured and unstructured data. Traditional data analysis and mining techniques, such as retrieval, semantic analysis and comparison do not suffice for analyzing big data. The complex datatypes, structure and complex patterns of big data make the representation, understanding, computation and analysis of big data very challenging.
- **Computational Complexity:** Big data has three key features: multi source, fast changing and huge volume, which make the computations by traditional computing methods on big data very challenging. New approaches are required

for analyzing big data which are free from traditional computational assumptions based on identical and independent data. New approaches will need to be big-data oriented, highly efficient in computing and provide innovative methods for analyzing big data.

- **System Complexity:** Analyzing big data requires high computational power since computation complexity is high and real time. Systems suitable to handle this kind of computational complexity are the key to process big data.

CHAPTER 2

LITERATURE SURVEY

2.1 Why Twitter?

- Twitter is a networking and microblogging website (currently 307 million monthly users).
- Each tweet can be of maximum 140 characters length.
- Different companies and organizations use twitter for branding and promotion.
- If an organization sends out good information through their tweets continuously, people will get to know their campaigns and brand's name and share their tweets.
- Tweets can help users to know about brands or competition, success of campaign, etc.

2.2 Why Hadoop?

- **Open-source:** Apache Hadoop is an open source project. It means its code can be modified according to business requirements.
- **Distributed Processing:** As data is stored in a distributed manner in HDFS across the cluster, data is processed in parallel on cluster of nodes.
- **Fault Tolerance:** By default, 3 replicas of each block are stored across the cluster in Hadoop and it can be changed also as per the requirement. So, if any node goes down, data on that node can be recovered from other nodes easily. Failures of nodes or tasks are recovered automatically by the framework. This is how Hadoop is fault tolerant.

- **Reliability:** Due to replication of data in the cluster, data is reliably stored on the cluster of machines despite machine failures. If your machine goes down, then also your data will be stored reliably.
- **High Availability:** Data is highly available and accessible despite hardware failure due to multiple copies of data. If a machine or few hardware crashes, then data will be accessed from other path.
- **Scalability:** Hadoop is highly scalable in the way new hardware can be easily added to the nodes. It also provides horizontal scalability which means new nodes can be added on the fly without any downtime. It is a very useful and one of the most important traits of Hadoop Ecosystem.
- **Economic:** Hadoop is not very expensive as it runs on cluster of commodity hardware. We do not need any specialized machine for it. Hadoop provides huge cost saving also as it is very easy to add more nodes on the fly here. So, if requirement increases, you can increase nodes as well without any downtime and without requiring much of pre-planning.
- **Easy to use:** No need of client to deal with distributed computing, framework takes care of all the things. So, it is easy to use.
- **Data Locality:** Hadoop works on data locality principle which states that move computation to data instead of data to computation. When client submits the algorithm, this algorithm is moved to data in the cluster rather than bringing data to the location where algorithm is submitted and then processing it.

CHAPTER 3

SYSTEM OVERVIEW

3.1 Technologies Used

3.1.1 Apache Hadoop

Hadoop, developed by Doug Cutting and Mike Camarilla in 2005, is an open source software and is a very popular tool used by major organizations and researchers to process and analyse big data. A major influence in the development of Hadoop was Google's architecture, Google File System and MapReduce. It is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage [6].

Due to the advent of new technologies, devices, and communication means like social networking sites, the amount of data produced by mankind is growing rapidly every year. The amount of data produced by us from the beginning of time till 2003 was 5 billion gigabytes. If you pile up the data in the form of disks it may fill an entire football field. The same amount was created in every two days in 2011, and in every ten minutes in 2013 [6]. This rate is still growing enormously. Though all this information produced is meaningful and can be useful when processed, it is being neglected. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

Two main components of Hadoop are storage and processing:

- **Storage:** HDFS (Hadoop Distributed File System) is the storage component used in Hadoop. The advantages of HDFS are that it provides fault tolerance and runs on commodity hardware. HDFS has master/slave architecture and can store data on thousands of servers. HDFS splits the files into blocks which makes the data more manageable.

- **Processing:** For processing, Hadoop uses MapReduce programming model which was introduced in 2004 by Google. The advantage of MapReduce is that it allows the programming of application which process large amount of data very easy [7]. The processing of data is done in parallel on large clusters of hardware and also supports fault tolerance. The two functions of MapReduce are Map and Reduce. Map function runs first and filter, transform or parse the data. The output of Map is passed on to Reduce function. Reduce function is used for summarizing the data produced by the Map function and is normally optional.

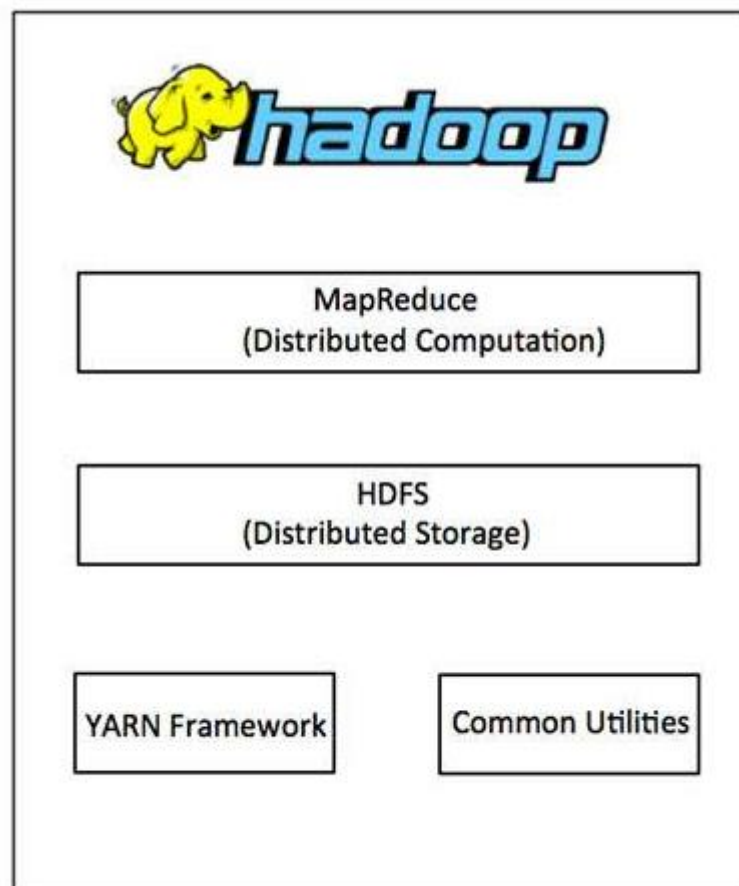


Fig 3.1: Layers of hadoop

3.1.2 Apache Flume

Apache Flume is developed by Cloudera and is an open source software program. It is used for aggregating and transferring large amount of data around a Hadoop cluster as soon as the data is produced or shortly after. Main function of Apache Flume is to gather

log files from different machines in the cluster and store them in a centralized store such as HDFS. There are sources and sinks, which are connected by chains of nodes so that data can flow from sources to sinks [7].

Flume is a standard, simple, robust, flexible, and extensible tool for data ingestion from various data producers (webservers) in hadoop. It is highly reliable, distributed, and configurable tool. It is principally designed to copy streaming data (log data) from various web servers to HDFS.

Apache Flume has three tier design: Agent tier, Collector tier and Storage tier.

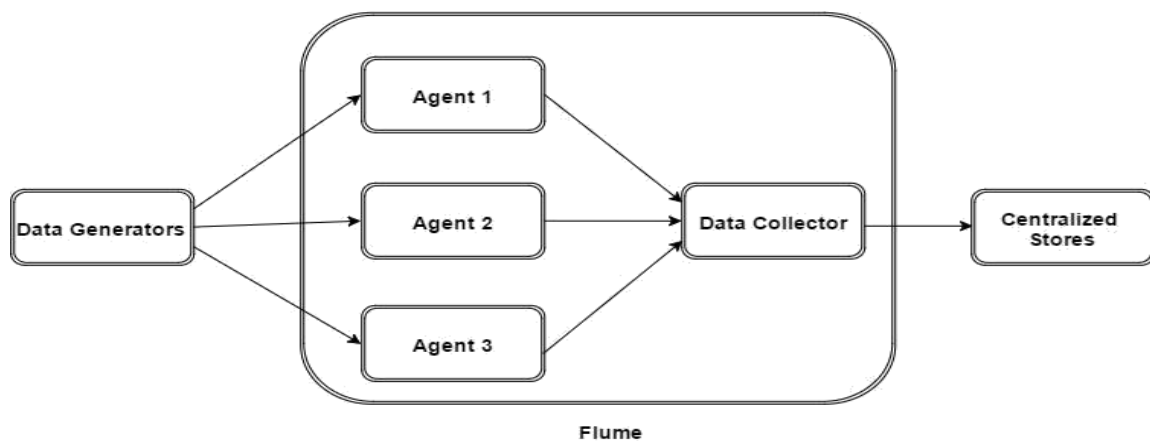


Fig 3.2: Basic architecture of Flume

- **Agent Tier:** An agent is a daemon process in Flume. The function of agent is to move data from clients or agents to its destination (agents or sink). Flume may have multiple agents to transfer data.
- **Collector Tier:** Collector tier consist of collectors which are connected to multiple agents. The function of a collector is to collect the data that is being sent by multiple agents and forward it to Storage tier.
- **Storage Tier:** Storage tier consists of different file systems to store and manage the big data. HDFS (Hadoop Distributed File System) and GFS (Google File System) are the file systems commonly used in Hadoop.

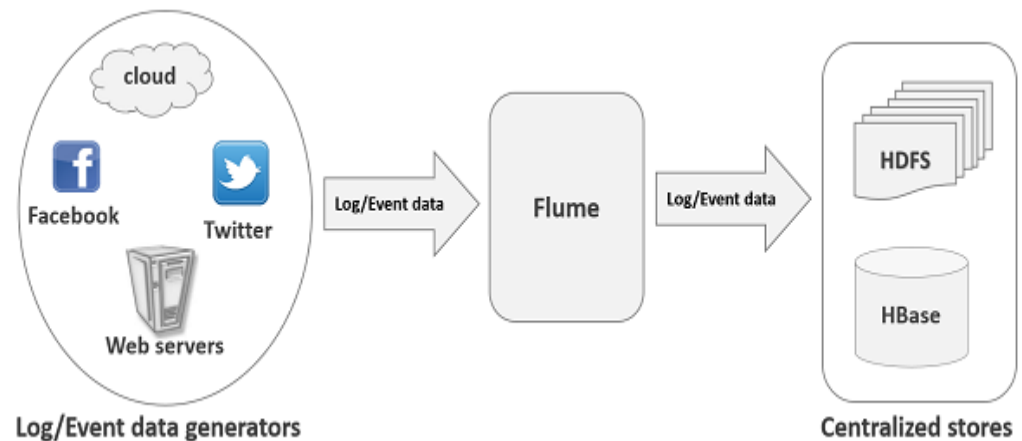


Fig 3.3: Flume in action

3.1.3 Apache Hive

Hive is developed by Facebook, which is used in Hadoop for turning the data collected into a complete data warehouse with an extension of SQL for querying and analysing data. SQL extension in hive is known as HiveQL and is a declarative language. In hive the description of flow of data isn't necessary as it can find the flow of data on the basis of result described. A schema has to be described in hive in order to use HiveQL [7].

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analysing easy. It is used by different companies. For example, Amazon uses it in Amazon Elastic MapReduce.

Hive is not relational database, it doesn't store the data itself but stores the data into HDFS. Hive does use a database to store Meta data. Hive runs on Hadoop which is batch- processing system, where jobs can have high latency. Hive is not suitable for real-time queries and is used for batch of large set of data, which is why it is used to query big data [6]. Hive converts unstructured data which is gathered by Apache Flume and converts it to structured data and stores it back to HDFS for further use and analysis. Hive architecture consists of:

- **Metastore:** The schema information, the structure in which the data is stored in HDFS is stored in Meta store.

- **HiveQL:** HiveQL is based on the SQL-92 specifications and is used to query data. HiveQL is similar to SQL and uses familiar relation database concepts like tables, schema, rows and columns. It converts queries into MapReduce jobs.
- **User Interface:** Hive has number of different user interfaces. It has WebUI and its own command line interface. For our project we used the command line interface of Hive.

Hive is not

- A relational database.
- A design for Online Transaction Processing (OLTP).
- A language for real-time queries and row-level updates.

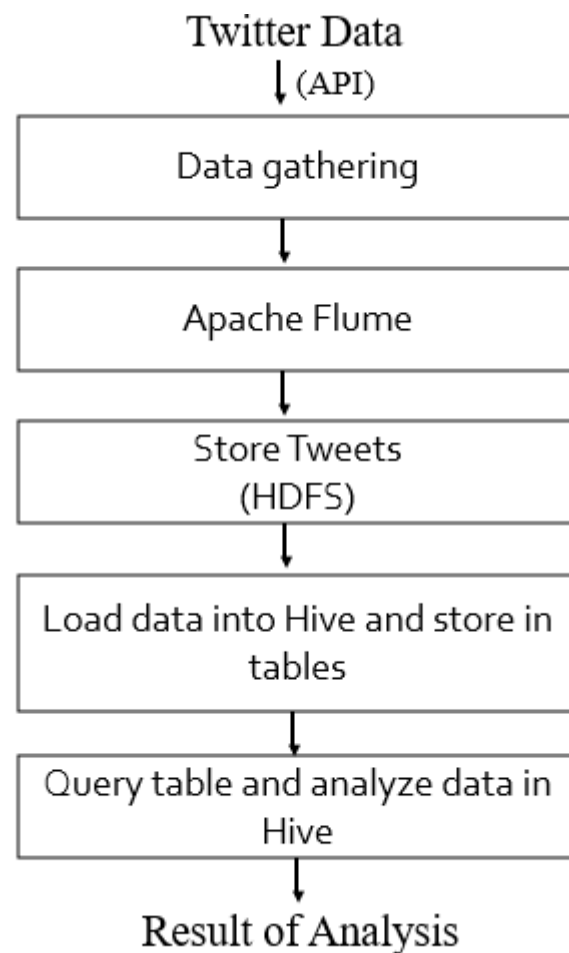


Fig 3.4: Flowchart of functioning of hadoop ecosystem

3.2 Hadoop Frame Architecture

3.2.1 Hadoop Common Utilities

Hadoop common utilities are required for the execution of Hadoop functions and are carried out by java files and scripts. Java libraries and utilities provide the support of operating system level and file system level abstractions, which are needed for Hadoop modules to execute [1].

- **Hadoop Yarn:** Hadoop Yarn carries out the operation of scheduling jobs and managing the resources to clusters. The conversion of unstructured data into structured data is also done by Hadoop Yarn.
- **HDFS:** HDFS is used to store the very large database of logs that are gathered by flumes. It provides easy and fast access to this data which results in high throughput.
- **MapReduce Paradigm:** It is the major component of Hadoop framework architecture. It enables the parallel processing of data, which is the key to process large database, perfect for processing big data.

3.2.2 MapReduce

The MapReduce paradigm enables the development of application very easy and effective and also the processing of large data sets is done in parallelism which makes processing big data by MapReduce very effective. The two functions defined in MapReduce are Map and Reduce [8]. The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into mappers and reducers is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change [8].

MapReduce is a programming model for writing applications that can process Big Data in parallel on multiple nodes. MapReduce provides analytical capabilities for analyzing huge volumes of complex data. Traditional Enterprise System normally have a centralized server to store and process data. The following illustrations= depicts a schematic view of modern enterprise system.

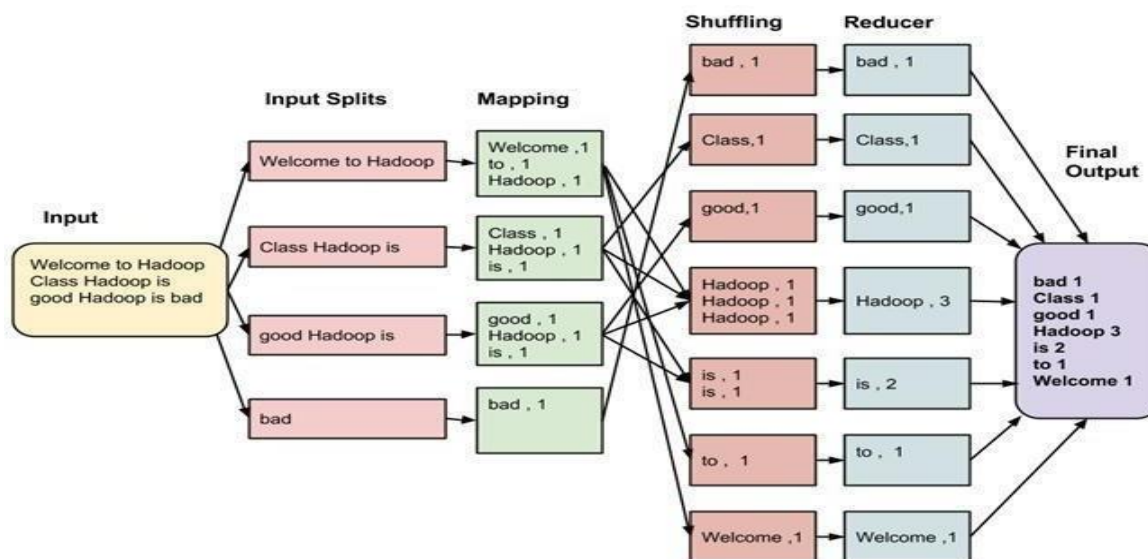


Fig 3.5: Detailed view of functioning of MapReduce

- **Map:** The first function that is performed by MapReduce is the Map function. In map function the data is captured and is divided pairs of data. This data is further divided into tuples to form a key/value pair. This data is then processed in parallel by different machines in the cluster.
- **Reduce:** The input to the Reduce is the output of the Map function. In Reduce function, all the divided tuples from Map function are combined to form smaller set of tuples. The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into mappers and reducers is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the MapReduce model [8].

How MapReduce works

The reduce task is always performed after the map job.

Phases in MapReduce and their significance

- **Input Phase** – Here we have a Record Reader that translates each record in an

input file and sends the parsed data to the mapper in the form of key-values pairs.

- **Map** – Map is a user-defined function, which takes a series of key-value pairs and processes each one of them to generate zero or more key-value pairs.
- **Intermediate keys** – The key-value pairs generated by the mapper are known as intermediate keys.
- **Combiner** – A combiner is a type of local Reducer that groups similar data from the map phase into identifiable sets. It takes the intermediate keys from the mapper as input and applies a user-defined code to aggregate the values in a small scope of one mapper. It is not a part of the main MapReduce algorithm; it is optional.
- **Shuffle and Sort** – The Reducer task starts with the shuffle and sort step. It downloads the grouped key-value pairs onto the local machine, where the reducer is running. The individual key-value pairs are sorted by key into a larger data list. The data list groups the equivalent keys together so that their values can be iterated easily in the Reducer task.
- **Reducer** – The reducer takes the grouped key-value paired data as input and runs a Reducer function on each one of them. Here, the data can be aggregated, filtered, and combined in a number of ways, and it requires a wide range of processing. Once the execution is over, it gives zero or more key-value pairs to the final step.
- **Output Phase** – In the output phase, we have an output formatter that translates the final key-value pairs from the Reducer function and writes them onto a file using a record writer.

3.2.3 SerDe

SerDe is short for Serializer/Deserializer. Hive uses the SerDe interface for IO. The interface handles both serialization and deserialization and also interpreting the results of serialization as individual fields for processing. The SerDe interface allows you to instruct Hive as to how a record should be processed. The Deserializer interface takes a string or binary representation of a record, and translates it into a Java object that Hive can manipulate. The Serializer, however, will take a Java object that Hive has been working with, and turn it into something that Hive can write to HDFS or another

supported system. Commonly, Deserializers are used at query time to execute 'select' statements, and Serializers are used when writing data, such as through an 'insert-select' statement.

3.2.4 JSON Format

JSON stands for JavaScript Object Notation. JSON is a syntax for storing and exchanging data. JSON is text, written with JavaScript object notation. Since the JSON format is text only, it can easily be sent to and from a server, and used as a data format by any programming language.

- JSON stands for JavaScript Object Notation
- JSON is a lightweight data-interchange format
- JSON is "self-describing" and easy to understand
- JSON is language independent

CHAPTER 4

IMPLEMENTATION AND CODING

For the development purpose twitter provides streaming API which allows the developer an access to 1% of tweets tweeted at that time bases on the particular keyword. The object about which we want to perform analysis is submitted to the twitter API's which does further mining and provides the tweets related to only that object. Twitter data is generally unstructured i.e. use of abbreviations is very high. Tweet messages also consist of a timestamp and the user name. This timestamp is useful for guessing the future trend application of our project. For doing twitter data analysis first data is collected using FLUME in local HDFS. Twitter data is generally unstructured i.e. use of abbreviations is very high. Tweet messages also consist of a timestamp and the user name. Tweets are preprocessed for removing noise, meaningless symbols and spams.

4.1 Creating Twitter API

Twitter API requires creation of twitter app and generation of consumer keys and access tokens. The steps to create Twitter API are as follows:

1. You need to have a working Twitter account for the authentication purposes. Twitter doesn't allow access to its data if one doesn't have twitter developer account. So, we need to fill out the details regarding the purpose that we need the data for.
2. Visit developer portal with and approved developer account to create a Twitter app and generate authentication keys.
3. These include:
Consumer' tokens – for app authentication and,
'Access' tokens – for user /account authentication.

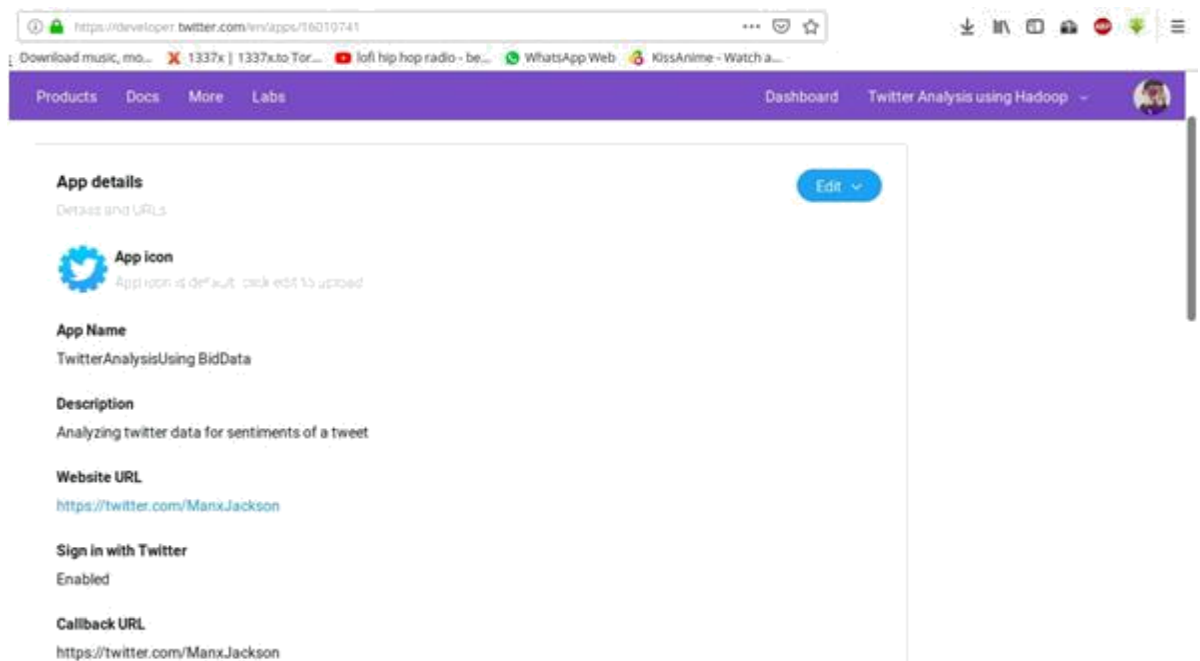


Fig 4.1: Twitter API app details

4.2 Starting HDFS

After installing HDFS, we need to verify its existence in the system. Firstly, we start hadoop and create a folder to store our tweets in it. Following are the steps to configure hadoop.

1. Verify Hadoop

If hadoop is already installed, verification of the installation is done using the following command:

```
$ hadoop version
```

If the system already has hadoop installed, the following output will be shown.

```
Hadoop 2.6.0
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r
compiled by Jenkins on 2014-11-13T21:10Z
compiled with protoc 2.5.0
```

2. *Starting HDFS*

Start Hadoop dfs (distributed file system) and yarn using following command:

```
$ start-dfs.sh
Localhost: starting namenode, logging to
/home/Hadoop/logs/hadoop-namenode-
localhost.localdomain.out
Localhost: starting datanode, logging to
/home/Hadoop/logs/hadoop-datanode-
localhost.localdomain.out
Starting secondary namenode [0.0.0.0]
Starting secondarynamenode, logging to
/home/Hadoop/logs/hadoop-secondarynamenode-
localhost.localdomain.out

$ start-yarn.sh
Starting yarn daemons
Starting resourcemanager, logging to
/home/hadoop/logs/hadoop-resourcemanager-
localhost.localdomain.out
localhost: starting nodemanager, logging to
/home/hadoop/logs/hadoop-nodemanager-
localhost.localdomain.out

$ jps
Namenode
Datanode
ResourceManager
NodeManager
```

3. *Create directory in HDFS*

In hadoop dfs, directories can be created using the command mkdir:

```
$ hdfs dfs -mkdir hdfs://localhost:9000/major
```

4.3 **Configure Flume to copy data to HDFS**

We have to configure the source, the channel, and the sink using the configuration file in the *conf* folder.

1. Create a file *flume.conf* inside `/usr/lib /apache-flume-1.6.0bin/conf` and add the following parameters in it:

```
# Naming the components on the current agent.
TwitterAgent.sources = Twitter
```

```

TwitterAgent.channels = MemChannel
TwitterAgent.sinks = HDFS

# Describing/Configuring the source
TwitterAgent.sources.Twitter.type =
org.apache.flume.source.twitter.TwitterSource
TwitterAgent.sources.Twitter.consumerKey = Your
consumer key
TwitterAgent.sources.Twitter.consumerSecret = Your
consumer secret key
TwitterAgent.sources.Twitter.accessToken = Your
consumer key access token
TwitterAgent.sources.Twitter.accessTokenSecret = Your
consumer key access token secret
TwitterAgent.sources.Twitter.keywords = elections2019

# Describing/Configuring the sink

TwitterAgent.sinks.HDFS.type = hdfs
TwitterAgent.sinks.HDFS.hdfs.path =
hdfs://localhost:9000/user/Hadoop/twitter_data/
TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream
TwitterAgent.sinks.HDFS.hdfs.writeFormat = Text
TwitterAgent.sinks.HDFS.hdfs.batchSize = 1000
TwitterAgent.sinks.HDFS.hdfs.rollSize = 0
TwitterAgent.sinks.HDFS.hdfs.rollCount = 10000

# Describing/Configuring the channel
TwitterAgent.channels.MemChannel.type = memory
TwitterAgent.channels.MemChannel.capacity = 10000
TwitterAgent.channels.MemChannel.transactionCapacity
= 100

# Binding the source and sink to the channel
TwitterAgent.sources.Twitter.channels = MemChannel
TwitterAgent.sinks.HDFS.channel = MemChannel

```

While configuring this source file, we need to provide values to some of the properties:

- Channel
- Source type: org.apache.flume.source.twitter.TwitterSource.
- Consumer key: Consumer key from twitter API.
- Access token: Access token from twitter API.
- Access token secret: Token secret from twitter API.
- batchSize: Maximum number of tweets that need to be in a twitter batch. Default value is 1000.
- batchSizeDurationMills: Time in milliseconds to wait for a batch is being is being closed. Default value is 1000.

a. *Channel*

We are using memory channel. We provide values to the type of channel.

- **type** – In our flume.conf file type of channel is **MemChannel**.
- **capacity** – The maximum number of events that can be stored in the channel is specified by capacity. Default value is 100.
- **transactionCapacity** – Number of events the channel can accept or send. Default value is 100.

b. *HDFS Sink*

HDFS Sink behaves like a database where all the data is dumped from twitter using flume. Following details need to be provided:

- **Channel**
- **Type** – hdfs
- **hdfs.path** – the path of directory where all the data will be stored.

Based on the scenario, we can add more optional values. Following are the optional properties of HDFS Sink that are being configured in this project.

- **fileType** – file format required in our HDFS. In our case, we use **DataStream**. The three available filetypes are **SequenceFile**, **DataStream** and **CompressedStream**.
- **writeFormat** – either writable or text.
- **batchSize** – The maximum number of events that is going to be written in a file before it is being flushed in HDFS. Default value is 100.
- **rollsize** – size of the file to trigger a roll. Default value is 100.
- **rollCount** – Maximum number of events that are written in the file before rolling. Default value is 10.

2. Start Flume to copy data to HDFS:

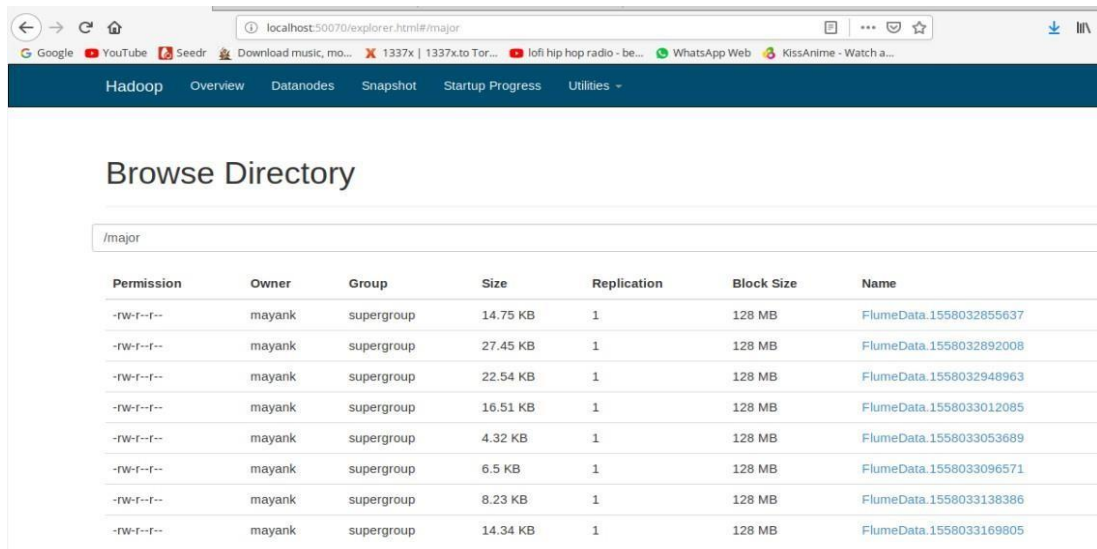
Now we browse to the flume's home directory and start the flume-ng command as shown below:

```
$ cd $FLUME_HOME
$ bin/flume-ng agent -conf ./conf/ -f
conf/flume.conf Dflume.root=DEBUG,console -n
TwitterAgent
```

If everything goes according to the plan, tweets will start streaming into HDFS. Now we can verify this by accessing Hadoop Administration Web UI using the URL:

<https://localhost:50070/>

The files that were extracted by Apache flume are seen in HDFS as follows:



Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	mayank	supergroup	14.75 KB	1	128 MB	FlumeData.1558032855637
-rw-r--r--	mayank	supergroup	27.45 KB	1	128 MB	FlumeData.1558032892008
-rw-r--r--	mayank	supergroup	22.54 KB	1	128 MB	FlumeData.1558032948963
-rw-r--r--	mayank	supergroup	16.51 KB	1	128 MB	FlumeData.1558033012085
-rw-r--r--	mayank	supergroup	4.32 KB	1	128 MB	FlumeData.1558033053689
-rw-r--r--	mayank	supergroup	6.5 KB	1	128 MB	FlumeData.1558033096571
-rw-r--r--	mayank	supergroup	8.23 KB	1	128 MB	FlumeData.1558033138386
-rw-r--r--	mayank	supergroup	14.34 KB	1	128 MB	FlumeData.1558033169805

Fig 4.2: HDFS Web UI

4.4 Hive query for analysis over Twitter data

After successful installation of Hive, it can be verified by using the following command:

```
$ cd $HIVE_HOME
$ bin/hive
```

If everything goes as planned, following message will be shown:

```
Logging initialized using configuration in
jar:file:/home/hadoop/hive-0.9.0/lib/hive-common-
0.9.0.jar!/hive-log4j.properties

Hive history
file=/home/hadoop/hive_job_log_hadoop_201312121621_14
94929084.txt
.....
hive>
```

Now, we need to extract the data from HDFS (unstructured format) into our hive tables (structured format) for further analysis.

```
hive> CREATE EXTERNAL TABLE raw_f (
    id BIGINT,
    created_at STRING,
    source STRING,
    favorited BOOLEAN,
    retweeted_status
    STRUCT<text:STRING, `user`:STRUCT
    <screen_name:STRING, name:STRING>,
    retweet_count:INT>, entities STRUCT<
    urls:ARRAY<STRUCT<expanded_url:STRING>
    >, user_mentions:ARRAY
    <STRUCT<screen_name:STRING, name:STRING
    >>,
    hashtags:ARRAY<STRUCT<text:STRING>>>,
    text STRING,
    `user` STRUCT< screen_name:STRING,
    name:STRING, friends_count:INT,
    followers_count:INT,
    statuses_count:INT, verified:BOOLEAN,
    utc_offset:INT, time_zone:STRING>,
    in_reply_to_screen_name STRING
)

ROW FORMAT SERDE
'org.openx.data.jsonserde.JsonSerDe' WITH
SERDEPROPERTIES
("ignore.malformed.json"="true") LOCATION
'hdfs://localhost:9000/major';
```

4.5 Finding Recent Trends and Most Popular hashtags

Recent trends from streamed tweets can also be found using Hive queries. Since tweets collected from twitter are in JSON format, we have to use JSON input format to load the tweets into Hive. We have used Cloudera Hive JsonSerDe for this purpose. This jar file has to be present in Hive to process the data [9]. It can be added using following command. This command is used in Hive interface.

```
hive> add jar /usr/lib/hive/lib/jsonserde-with-
dependencies.jar
```

Following steps are performed to find the recent trend:

a) Loading and Feature extraction

The tweets collected from twitter are stored in HDFS. In order to work with Data stored in HDFS using HiveQL, first an external table is created which creates the table definition in the Hive Metastore. This query not only creates a schema to store the tweets, but also extracts required fields like id and entities.

b) Extracting Hashtags

In order to extract actual hashtags from entities, we created another table which contains id and the list of hashtags. Since multiple hashtags are present in one tweet, we used UDTF (User Defined Table Generation Function) to extract each hashtag on the new row. The outcome of this phase is id and hashtag.

c) Counting hashtag

After performing all the above steps, we have id and hashtag text. A hive query is written to count the hashtags.

d) Query for top trends

For getting most popular hash tags or to find top trend we are use following hive query:

```
hive> CREATE VIEW L5 AS SELECT
      ht,
      count(ht) AS countht
    FROM raw_f LATERAL VIEW
      EXPLODE(entities.hashtags.text) dummy AS ht
    ORDER BY countht DESC limit 10;
```

Output of the query is:

```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 11.32 sec HDFS Read: 276861 HDFS Write: 324 SUCCESS
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 7.84 sec HDFS Read: 727 HDFS Write: 100 SUCCESS
Total MapReduce CPU Time Spent: 19 seconds 160 msec
OK
asusrog 3
MyCoolROG 3
ASUS_USA 3
ASUS_ROG 3
ASUSIndia 3
ASUS 3
disituKadangSayaMerasaSedih 1
asus 1
Time taken: 106.389 seconds, Fetched: 8 row(s)
hive>
```

Fig 4.3: Most popular hashtags

In the output we can see the popular hashtags are shown with the number of times they have occurred in our data set. For example, hashtag ASUS India has occurred 3 times in all of our tweets. The hashtags are arranged in decreasing order of their total occurrences.

4.6 Finding Users with the greatest Number of Tweet

During analysis, finding the users who are interested in the topic we are taking under

analysis might be required. This information can be used for purpose like targeting specific group of people for promotion and advertisement [9]. The query which will give us the user with maximum number of tweets is:

```
hive> CREATE VIEW L7 AS SELECT
      screen_name,
      count(id) AS cnt
    FROM tweets_simple
   GROUP BY screen_name
  HAVING screen_name IS NOT NULL
 ORDER BY cnt DESC LIMIT 10;
```

Output of the query is:



```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 7.73 sec HDFS Read: 276861 HDFS Write: 1511 SUCCESS
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 4.38 sec HDFS Read: 1913 HDFS Write: 145 SUCCESS
Total MapReduce CPU Time Spent: 12 seconds 110 msec
OK
ethan003      3
wineflakes    1
thomaslmbt    1
sparadra4288  1
seminudeman   1
saripmanfr    1
reichfrancois 1
poke_sokuhou  1
nikhilchandra 1
matteogauthier1 1
Time taken: 70.293 seconds, Fetched: 10 row(s)
hive>
```

Fig 4.4: User with maximum number of tweets

4.7 Performing sentiment analysis on the tweets

The tweets of trending topics stored in HDFS are used for sentiment analysis and processed using HiveQL [9]. The steps are discussed in the following section.

1. Removal of stopwords and cleaning

Removal of Stopwords and cleaning is used to remove unwanted or useless data in relation to our analysis process. There are Hive queries to remove the stopwords and also clean the data.

```
hive> CREATE VIEW tweets_simple AS SELECT
      id,
      `user`.screen_name, source,
```

```

retweeted_status.retweet_count,
entities.hashtags,
cast ( from_unixtime( unix_timestamp(concat(
'2016 ', substring(created_at,5,15)), 'yyyy
MMM dd hh:mm:ss')) as timestamp) ts,
text,
`user`.statuses_count,
`user`.friends_count,
`user`.followers_count,
`user`.time_zone
FROM raw_f;

```

2. *Tokenizing the tweets*

In order to find the sentiment words, the tweet is split into words using one of the Hive UDF functions. A Hive table is created to store the tweet id and the array of words present in each tweet. As multiple words are present in an array, we used some built in UDTF function to extract each word from an array and created a new row for each left outer join operation on a table that multiple words are present in an array, used some built in UDTF function to extract each word from an array and created a new row for each left outer join operation on a table that contains id, word and dictionary table if the word matches with the sentiment word in the dictionary.

Queries used for this purpose is:

```

hive> CREATE VIEW L1 AS SELECT
      id,
      words
    FROM raw_f LATERAL VIEW
      EXPLODE(sentences(lower(text))) dummy
    AS words;

hive> CREATE VIEW L2 AS SELECT
      id,
      word
    FROM L1 LATERAL VIEW
      EXPLODE(words) dummy AS word;

```

```

Total MapReduce CPU Time Spent: 7 seconds 310 msec
OK
1129097726690381824      rt
1129097726690381824      httweets
1129097726690381824      nathuram
1129097726690381824      godse
1129097726690381824      a
1129097726690381824      patriot
1129097726690381824      says
1129097726690381824      bjp
1129097726690381824      s
1129097726690381824      pragya
1129097726690381824      thakur
1129097726690381824      https
1129097726690381824      t.co
1129097726690381824      i8g1aub9ez
1129097726690381824      electionswithht
1129097726690381824      loksabhaelections2019
1129097726690381824      elect
1129097732537364482      perception
1129097732537364482      of
1129097732537364482      europeans
1129097732537364482      vis-a-vis
1129097732537364482      ewelections2019
1129097732537364482      makes
1129097732537364482      outcome
1129097732537364482      of
1129097732537364482      these
1129097732537364482      polls
1129097732537364482      worryingly
1129097732537364482      polarised
1129097732537364482      more
1129097732537364482      liberal
1129097732537364482      https
1129097732537364482      t.co
1129097732537364482      gemka2lqfr

```

Fig 4.5 Tokenization of each word in tweets.

3. Sentiment word detection

Sentiment analysis is done using dictionary-based method. A table is created to store the contents present in the dictionary. In order to rate the tokenized words, the tokenized words have to be mapped with the loaded dictionary. We performed left outer join operation on a table that contains id, word and dictionary table if the word matches with the sentiment word in the dictionary, then a rating is given to the matched word or else NULL value is assigned. A hive table is created to store id, word and then rating.

Creating external table for dictionary

```

hive> CREATE EXTERNAL TABLE dictionary (
        type STRING,
        length INT,
        word STRING,
        pos STRING,
        polarity STRING
    )
    ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' STORED AS
    TEXTFILE;

```

Load dictionary into HDFS

```

hadoop dfs -put /home/mayank/dictionary.csv

```

```
hdfs://localhost:9000/
```

```
LOAD DATA INPATH '/dictionary.csv' INTO TABLE dictionary
```

Converting polarity into numeric value

```
hive> CREATE VIEW L3 AS SELECT
      id,
      L2.word,
      CASE d.polarity
      WHEN 'negative' THEN 0
      WHEN 'positive' THEN 5
      ELSE 2.5 END AS polarity
      FROM L2 LEFT OUTER JOIN dictionary d on
      L2.word = d.word;
```

Classification of words into positive, negative or neutral

```
hive> CREATE TABLE tweets_sentiment AS SELECT
      id,
      CASE
      WHEN AVG (polarity) > 2.5 THEN 'positive'
      WHEN AVG (polarity) < 2.5 THEN 'negative'
      ELSE 'neutral' END AS sentiment
      FROM L3 GROUP BY id;
```

Classification of tweets

After performing all the above steps, we have id, word and rating. Then “group by” operation is performed on id to group all the words belonging to one tweet after which average operation is performed on the ratings given to each word in a tweet. Then “group by” operation is performed on id to group all the words belonging to one tweet after which average operation is performed on the ratings given to each word in a tweet. Based on the average ratings, tweets are classified into positive and negative.

```
hive> CREATE VIEW L8 AS SELECT
      sentiment,
      COUNT (id)
      FROM tweets_sentiment AS ts
      GROUP BY sentiment;
```

```
Total MapReduce CPU Time Spent: 4 seconds 918 msec
OK
negative      6
neutral 5
positive      7
Time taken: 48.73 seconds, Fetched: 3 row(s)
```

Fig 4.6: Tweets classification

CHAPTER 5

CONCLUSION AND RESULT

Conclusion

1. Generated insights can help in targeted marketing. During analysis, finding the users who are interested in the topic are calculated using the number of times they tweet on the particular topic. This information can be used for purpose like targeting specific group of people for promotion and advertisement.
2. Popular hashtags are shown with the number of times they have occurred in our data set. For example, hashtag ASUSIndia has occurred 3 times in all of our tweets.
3. General sentiment analysis: People use social media sites for spreading their sentiments about politics/celebrities/products/movies etc. So, we perform sentiment analysis on tweets/posts and compare the result we get by doing sentiment analysis one by one on each tweet.
4. Big data is the major problem nowadays in every organization due to ever growing data day-by-day [6]. This issue can be solved by using Hadoop paradigm in real time.

Result

1. Successful in calculating the top hashtags that were used in tweets which were related to our aspect.
2. Top users that were interested in our aspect tweeted about it and we analysed top 10 users who tweeted the most about the aspect. People are usually interested in talking about the politics and their views are equally important. Thus, analysis on their view is mandatory.
3. Sentiment analysis is done using dictionary-based method. Sentiments are required to determine the user point of view about some aspect. They do so by spreading their views over social media like Facebook, twitter etc.

CHAPTER 6

FUTURE SCOPE

1. The analysis of Twitter data reduces latency and price by using Big Data technology called as 'HADOOP' [5]. This system is very cost efficient and secure. So, Apache, the Open Source Product provides the platform that can have Twitter Analysis by using HADOOP the speed of the system will be increased.
2. Automated clustering of sentiment types and sentiment dependency rules need to be done.
3. Handling sarcasm and more discourse relations. More experiments need to be done to classify sarcastic tweets.
4. There are other platforms to deal with bigdata such as Microsoft's Azure HD Insights, Cloudera, Hortonworks etc. So, the same can be done in these platforms [5].
5. It can also be enhanced and applied to other types of big data such as Facebook, RFID's, cellular devices and different organizations where huge amounts of data is present and needs fast processing and analysing [5].

BIODATA OF CANDIDATES

The candidates, viz., Prashant Arora, Mayank Kumar, Ashutosh Dubey, Gopal Khatri and Anubhav Markanday are presently pursuing 8th semester with respect to the degree of Bachelor of Technology in Computer Science Engineering from National Institute of Technology Hamirpur, Anu, Hamirpur, Himachal Pradesh (177005).

Prashant Arora

Permanent Address: Shivalik estate, near Radha swami Satsang Bhawan Nalagarh, H.P.

Email: prashantarora873@gmail.com

Phone: (+91) 7018302276

Mayank Kumar

Permanent Address: JWO AK Singh, Air Force Station, Vill. Kharola, Rajpura, Punjab, 140602

Email: mkraone7@gmail.com

Phone: (+91) 9736952097

Ashutosh Dubey

Permanent Address: 377/7 Shastri Nagar, Kanpur

Email: ashul3297.ad@gmail.com

Phone: (+91) 7309523243

Gopal Khatri

Permanent Address: H. No. 144, Near Greenland Hotel, Bhajogi, The Manali, Kullu, 175131

Email: gkhatri090@gmail.com

Phone: (+91) 7018509684

Anubhav Markanday

Permanent Address: 68/12 Masanpura, Katcha tank, Nahan, Distt. Sirmour, H.P.

Email: anubhavmarkanday@gmail.com

Phone: (+91) 7018272334

REFERENCES

1. Anisha P. Rodrigues & Niranjana N. Chiplunkar, '**Real-time Twitter data analysis using Hadoop ecosystem. Data: A Literature Review**', IRJET, pISSN 2395-0072, 2018.
2. Yazala Ritika Siril Paul, Dilipkumar A. Borikar, '**Sentiment Analysis of Tweets at Sentence Level Using Hadoop**', Helix, 2017.
3. Abhinandan P Shirahatti , Neha Patil, Durgappa Kubasad , Arif Mujawar, '**Sentiment Analysis on Twitter Data using: Hadoop**', p- ISSN: 0976-1353 Volume 14 Issue 2 –APRIL 2015.
4. Hana Anber, Akram Shah Abd El-Aziz, '**A Literature review on Twitter Data Analysis**' , DOI: 10.17706/IJCEE.2016.8.3.241-249.
5. Aishwarya Kotwal, Dipali Jadhav, Priyanka Fulari Jaspreet kaur, Ratika Kad, '**Improvement in Sentiment Analysis of Twitter Data using Hadoop**', ISSN 0973-7529; ISBN 978-93-80544-20-5.
6. Kishore K. Pawar, Pukhraj P. Shrishrimal, R. R. Deshmukh., '**Twitter Sentiment Analysis: A Review**' IJCSIT-2016.
7. Atul Srivastav, Anuradha, Dimple Juneja Gupta, '**Social Network Analysis: Hardly Easy**', ICROIT 2014, India, Feb 6-8-2018.
8. A. Bhandarkar, Milind. (2010). '**MapReduce programming with apache Hadoop**'. 1. 10.1109/IPDPS.2010.5470377.
9. Saraswat, Stuti. (2018). '**Twitter Sentiment Analysis using Hive**'. International Journal for Research in Applied Science and Engineering Technology. 6. 4944-4949. 10.22214/ijraset.2018.4807.