Real-time Twitter Trends Report and Sentiment Analysis

Huy Tran [htn279]

Ann Malavet

May 5, 2022

I. Abstract

In the modern days, social media has become an integral part of our lives. We use social media to keep ourselves up to date on the latest trends and share our own opinions. Among the most widely used social media, we chose Twitter for this research due to its property of usually having an easy-to-recognize and process keywords (hashtags). Twitter received 350,000 per minute, translated to roughly over 5 hundred million tweets a day - making it a prime candidate for goverment, economics, culture or business-related analytics.

Due to Twitter's sheer amount of data, the Hadoop framework is chosen to carry out this analytics. We used Apache Flume for data ingestion and Apache Hive and MapReduce for our analytics.

The results we have have successfully met our needs. It detected the top trends on Twitter during the time of the analytics and was able to return accurate sentiment analysis on chosen keyword(s).

II. **Introduction**

This research was a part of our effort to deepen our understanding and get some hands-on experience with the Hadoop ecosystem along with the Processing Big Data for Analytis Applications course at NYU.

User interactions on social media tell us a lot about what is happening in the real world. By understanding these interactions, we can recognize and even predict important trends.

Motivations

What sorts of insights can we derived from these analytics? Suppose that we have a product and a large amount of customer are complaining about our product. Suppose that one financial influencer just tweets a status about acquiring a company or regarding a new crypto (yes, Musk, I am looking at you). In all cases, some questions that we might want answers to are when did the trend first start and how long it has been going on. Some other questions might include the significance of the trends or more crucialy, how big is the trend compared to other past trends on Twitter. With understanding these trends and being able to quantify the changes, one can develop quantitative model to meaningfully predict future behavior.

After reading this study, you will have some insights on how to build a system to identify and compare changes that arise from social media interactions.

Proposed method

This section describes the method we used to capture and analyzed tweets real time. First, we collected the tweets in real-time using Twitter API and Flume, then the data is saved in HDFS. After that, we used Hive to do our data profiling, MapReduce to find recent trends and a Lexicon-based approach to do sentiment analysis on a chosen keyword.

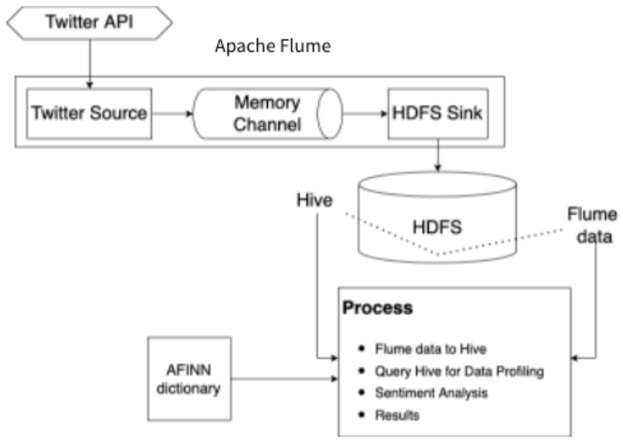


Figure 1. Design and dataflow

The study is divided into two main parts: finding trends and sentiment analysis. It is worth noting that even though we implemented the trends detection part using MapReduce, it is better and much less tedious if done correctly using Hive. The authors were more comfortable using MapReduce at the time of carrying out this study.

III. Related works

In this section, we disussed systems used in other studies to fetch Tweets from Twitter and sentiment analysis of its data.

Fetching tweets

As mentioned above, Twitter generates a huge amount of data daily. Collecting this kind of data manually proved to be a tedious and time-consuming task. However, the Twitter Application Programming Interface (API) enabled progarmmatic access to Twitter in unique and advanced ways. It provided us with a streaming API to stream real-time data. *Uzunkya, Ensari, Kavurucu (2015)* and *Gonzalez-Ibanez et al. (2011)* analyzed collecting tweets using Twitter4j while *Sheela (2016)* used the Twitter Streaming API. However, Twitter has a few restrictions on the number of tweets one can stream during a period of time. Therefore, we need a more efficient and scalable tool to acquire data from Twitter. *Vissamsetti, Prasanth, Jacob (2020)* and *Keskar, Palwe, Gupta (2020)* has proved that Apache Flume is the tool to do so.

Sentiment analysis

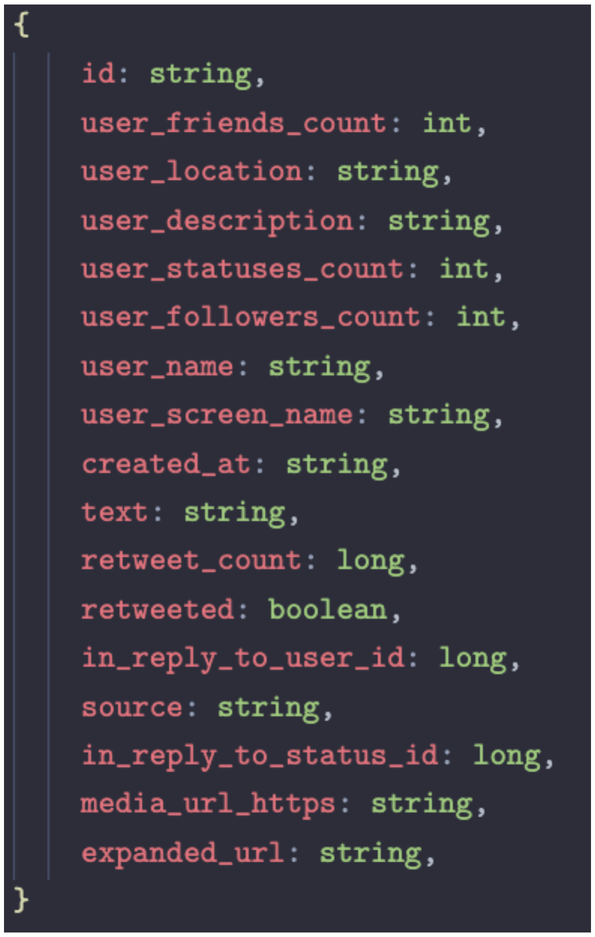
Sentiment analysis is not a new topic in natural language processing (NLP) or computer science in general. *Das, Behera et al. (2018)* has implemented a model using the Stanford's CoreNLP API an Recurrent Neural Network (RNN) to do sentiment analysis and predict stock prices. Meanwhile, *Keskar, Palwe and Gupta (2020)* discussed the use of N-Gram Analysis and Decision Tree Machine Learning technique to classfiy fake news. Other researchers like *Shang et al. (2017)* also proposed a model to classify public opinion on big data using the Hadoop Mahout text mining algorithms.

IV. Datasets

The dataset discussed in this paper is retrieved real-time using Apache Flume.

Flume is responsible for communicating with the source (Twitter streaming API) and retrieve tweets that match certain keywords (can be configured in Flume configuration file) or criterial (language, locations, etc). The data retrieved are in AVRO format which are then put into HDFS. The AVRO file consisted of tweets in JSON format which can later be loaded into Apache Hive. Our dataset was streamed for around an hour and a half on Wednesday, April 27 at around 5:00 PM. It is roughly 200 MB in size, consisting of around 2 millions records. Each record is a JSON data point with the following schema:

Figure 2. Tweet schema

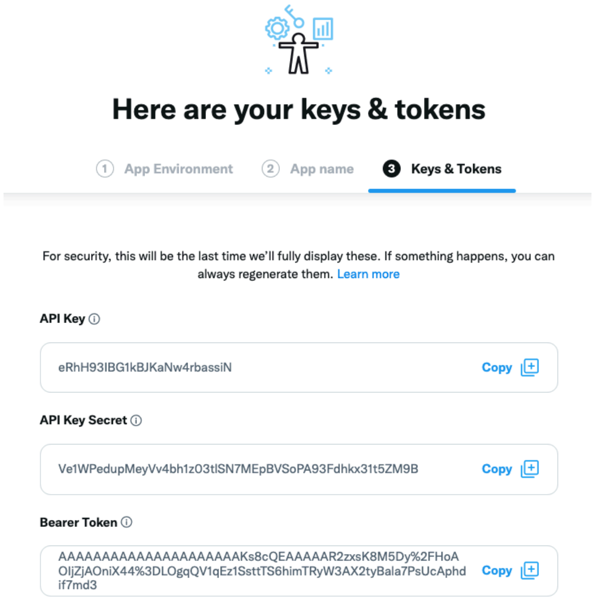


```
{
    id: string,
    user_friends_count: int,
    user_location: string,
    user_description: string,
    user_statuses_count: int,
    user_followers_count: int,
    user_name: string,
    user_screen_name: string,
    created_at: string,
    text: string,
    retweet_count: long,
    retweeted: boolean,
    in_reply_to_user_id: long,
    source: string,
    in_reply_to_status_id: long,
    media_url_https: string,
    expanded_url: string,
}
```

V. Data ingestion and analysis

Creating Twitter application

The first step in our data ingestion process would be to go to Twitter Developer portal (TDP) and create an account. After creating our account, we then moved on to creating our app (or project). After setting up the app environment and choosing an app name, we will be given an API key, secret and Bearer Token which will later be used with Flume.



Here are your keys & tokens

For security, this will be the last time we'll fully display these. If something happens, you can always regenerate them. Learn more

API Key
eRhH93IBG1kBJKaNw4rbassiN

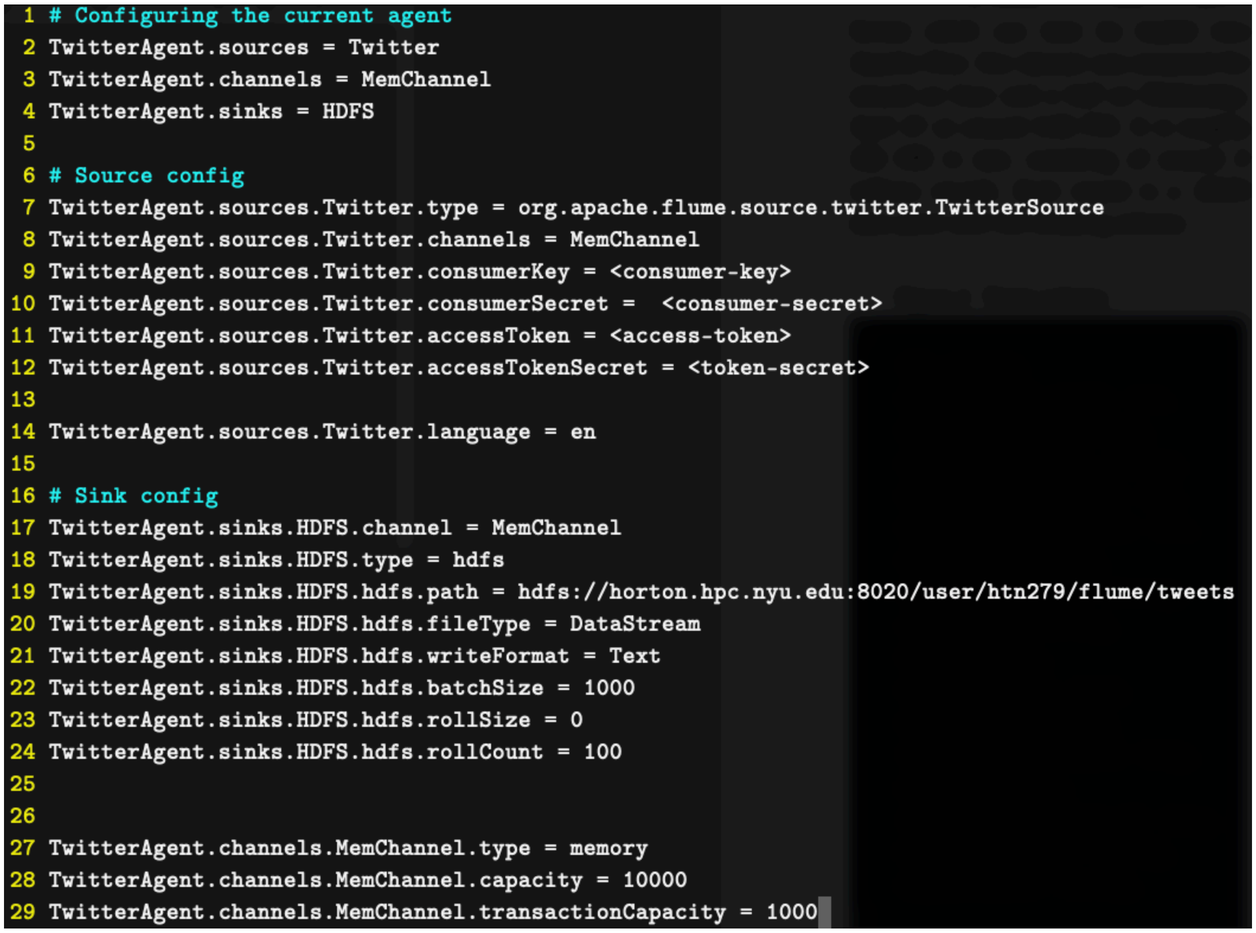
API Key Secret
Ve1WPedupMeyVv4bh1z03tlSN7MEpBVSoPA93Fdhkx31t5ZM9B

Bearer Token
AAAAAAAAAAAAAAAAAAAAAKs8cQEAAAAAR2zxsK8M5Dy%2FHoAOljZjAOniX44%3DLOgqQV1qEz1SsttTS6himTRyW3AX2tyBala7PsUcAphdif7md3

Figure 3. API keys and tokens sample

Getting data with Flume

After having our API keys and secrets, we created a configuration file for Flume. In this file, we have to configure our Twitter agent, our source, pipeline and sink.



```
# Configuring the current agent
TwitterAgent.sources = Twitter
TwitterAgent.channels = MemChannel
TwitterAgent.sinks = HDFS

# Source config
TwitterAgent.sources.Twitter.type = org.apache.flume.source.twitter.TwitterSource
TwitterAgent.sources.Twitter.channels = MemChannel
TwitterAgent.sources.Twitter.consumerKey = <consumer-key>
TwitterAgent.sources.Twitter.consumerSecret =  <consumer-secret>
TwitterAgent.sources.Twitter.accessToken = <access-token>
TwitterAgent.sources.Twitter.accessTokenSecret = <token-secret>

TwitterAgent.sources.Twitter.language = en

# Sink config
TwitterAgent.sinks.HDFS.channel = MemChannel
TwitterAgent.sinks.HDFS.type = hdfs
TwitterAgent.sinks.HDFS.hdfs.path = hdfs://horton.hpc.nyu.edu:8020/user/htn279/flume/tweets
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 = 100


TwitterAgent.channels.MemChannel.type = memory
TwitterAgent.channels.MemChannel.capacity = 10000
TwitterAgent.channels.MemChannel.transactionCapacity = 1000
```

Figure 4. Sample flume config

It is also worth noting that the source we were using is Twitter 1% firehose Source - which was highly experimental. The limitation to this soruce is that it can only stream 1% of the total number of tweets in the current version.

Querying using HiveQL

After running Flume with the above configuration, our Twitter data would be safely stored in the HDFS path we specified above. After this, we wanted to create a formatted table with the data above. For this, we would be using a custom serde in order for our Hive to create a table in the correct format. We then used Hive to see the schema of our data and profile our data.

Trends Detection

As mentioned above, finding trends on Pig/Hive would probably be much less trouble, but since we needed practice on MapReduce, we decided to go for the MapReduce approach anyway.

The algorithm we implemented was the "Hello World" program of MapReduce: the WordCount algorithm. We tweaked it a little bit to just count keywords with hashtag using RegEx (which would not detect trends that do not involve the use of a hashtag, but that is a story for another day). After counting the hashtag, we got our count as `<tweet, count>` pair, sorted using the key which is `tweet` (MapReduce default sorting). However, we needed to sort it from most used to least used. In order to do that, we wrote another simple MapReduce job that swapped the key and value into `<count, tweet>` and return the results.

Sentiment analysis

To carry out our sentiment analysis, we used a lexicon-based approach with AFINN dictionary - a dictionary that consists of 2500 words rated from +5 to -5 depending on the meaning. For example, `breathtaking` will be +5 while `terrible` will be -3.

Our dictionary will be required for every MapReduce jobs so that reading the file everytime from HDFS will increases seek time, thus increase latency. In order to overcome this, we had to implement distributed caching for our dictionary dataset.

In our job, we first created a `HashMap`, read the dictionary file and stored the word and its value as key and value.

After that, in our map method, we parse the JSON record into string by using `jsonParser` and get `id_str` as `id` and `text` as `content`. Next, we split our tweet's `content` into words and look up the rating of each words and add it to the `sum`. Then we return `id` and `text` as the key in the `context` and the `sum` as value.

VI. Results

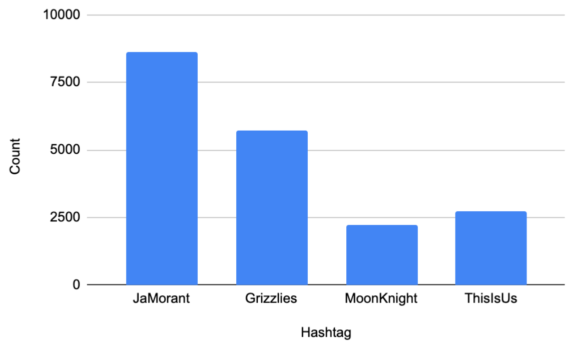


Figure 5. Sample of popular hashtags

After analysis of two keywords of our choice, `#ThisIsUs` and `#RipTwitter`, `#ThisIsUs` received an average scored of 6.2, meaning that people were mostly saying good things about the show while `#RipTwitter` got an average of -1.0, which means people were probably have doubts in Musk's decision of acquiring Twitter.

VII. Challenges

While doing our research, we noticed a few things. For example, this sentiment analysis did not account for sarcasm in people's tone (which is worth a whole independent research on its own). Furthermore, in the recent years, with the increasing use of memes and lately, short videos called reels, sentiment analysis have become harder than ever. The dataset has become more complicated, increased in size and will required a different system to approach.¶

VIII. Conclusion

There are different ways to approach the problem of sentiment analysis. However, the Hadoop ecosystem has proved itself to be a useful and beginner-friendly to carry-ing out these operations while not giving up performance and scalability.

REFERENCES

Sheela, L. Jaba. "A review of sentiment analysis in twitter data using Hadoop." International Journal of Database Theory and Application 9.1 (2016): 77-86.

Vissamsetti, Mohan Manoj, Yalamandala Prasanth, and T. Prem Jacob. Twitter Data Analysis for Live Streaming by Using Flume Technology. No. 2915. EasyChair, 2020.

Keskar, Devyani, Sushila Palwe, and Ayushi Gupta. "Fake news classification on twitter using flume, n-gram analysis, and decision tree machine learning technique." Proceeding of International Conference on Computational Science and Applications. Springer, Singapore, 2020.

Das, Sushree, Ranjan Kumar Behera, and Santanu Kumar Rath. "Real-time sentiment analysis of twitter streaming data for stock prediction." Procedia computer science 132 (2018): 956-964.

Sharma, Dipty. "Study of sentiment analysis using hadoop." Big Data Analytics. Springer, Singapore, 2018. 363-376.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying Sarcasm in Twitter: A Closer Look. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 581–586, Portland, Oregon, USA. Association for Computational Linguistics.

Uzunkaya, Can et al. "Hadoop Ecosystem and Its Analysis on Tweets." Procedia - Social and Behavioral Sciences 195 (2015): 1890-1897.