PROJECT REPORT SENTIMENT ANALYSIS ON TWITTER USING APACHE SPARK

Technical Report · October 2017

DOI: 10.13140/RG.2.2.10737.79200

4 authors, including:

Deepesh Khaneja
Carleton University
8 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Khushboo Vyas
University of Ottawa
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Ranjit Singh Saini
Carleton University
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

SYSC 5807 - ADVANCED TOPICS IN COMPUTER SYSTEMS

PROJECT REPORT

SENTIMENT ANALYSIS ON TWITTER USING APACHE SPARK

Amandeep Kaur, Deepesh Khaneja, Khushboo Vyas, Ranjit Singh Saini

Submitted to Dr. Imran Ahmad
8th April, 2016
Carleton University

***Abstract*—Social media websites have emerged as one of the platforms to raise users' opinions and influence the way any business is commercialized. Opinion of people matters a lot to analyze how the propagation of information impacts the lives in a large-scale network like Twitter. Sentiment analysis of the tweets determine the polarity and inclination of vast population towards specific topic, item or entity. These days, the applications of such analysis can be easily observed during public elections, movie promotions, brand endorsements and many other fields. In this project, we exploited the fast and in memory computation framework 'Apache Spark' to extract live tweets and perform sentiment analysis. The primary aim is to provide a method for analyzing sentiment score in noisy twitter streams. This paper reports on the design of a sentiment analysis, extracting vast number of tweets. Results classify user's perception via tweets into positive and negative. Secondly, we discuss various techniques to carryout sentiment analysis on twitter data in detail.**

***Index Terms*— Apache Spark, Sentiment Analysis, Twitter, Opinion mining**

I. INTRODUCTION

As internet is growing bigger, its horizons are becoming wider. Social Media and Micro blogging platforms like Facebook, Twitter, Tumblr dominate in spreading encapsulated news and trending topics across the globe at a rapid pace. A topic becomes trending if more and more users are contributing their opinion and judgements, thereby making it a valuable source of online perception [3]. These topics generally intended to spread awareness or to promote public figures, political campaigns during elections, product endorsements and entertainment like movies, award shows. Large organizations and firms take advantage of people's feedback to improve their products and services which further help in enhancing marketing strategies. One such example can be leaking the pictures of upcoming iPhone to create a hype to extract people's emotions and market the product before its release. Thus, there is a huge potential of discovering and analysing interesting patterns from the infinite social media data for business-driven applications.

Sentiment analysis is the prediction of emotions in a word, sentence or corpus of documents. It is intended to serve as an application to understand the attitudes, opinions and emotions expressed within an online mention. The intention is to gain an overview of the wider public opinion behind certain topics. Precisely, it is a paradigm of categorizing conversations into positive, negative or neutral labels. Many people use social media sites for networking with other people and to stay up-to-date with news and current events. These sites (Twitter, Facebook, Instagram, google+) offer a platform to people to voice their opinions. For example, people quickly post their reviews online as soon as they watch a movie and then start a series of comments to discuss about the acting skills depicted in the movie. This kind of information forms a basis for people to evaluate, rate about the performance of not only any movie but about other products and to know about whether it will be a success or not. This type of vast information on these sites can used for marketing and social studies [1]. Therefore, sentiment analysis has wide applications and include emotion mining, polarity, classification and influence analysis.

Twitter is an online networking site driven by tweets which are 140 character limited messages. Thus, the character limit enforces the use of hashtags for text classification. Currently around 6500 tweets are published per second, which results in approximately 561.6 million tweets per day [1]. These streams of tweets are generally noisy reflecting multi topic, changing attitudes information in unfiltered and unstructured format. Twitter sentiment analysis involves the use of natural language processing to extract, identify to characterize the sentiment content. Sentiment Analysis is often carried out at two levels 1) coarse level and 2) fine level. In coarse level, the analysis of entire documents is done while in fine level, the analysis of attributes is done [3]. The sentiments present in the text are of two types: Direct and Comparative. In comparative sentiments, the comparison of objects in the same sentence is involved while in direct sentiments, objects are independent of one another in the same sentence.

However, doing the analysis of tweets expressed in not an easy job. A lot of challenges are involved in terms of tonality, polarity, lexicon and grammar of the tweets. They tend to be highly unstructured and non-grammatical. It gets difficult to interpret their meaning. Moreover, extensive usage of slang words, acronyms and out of vocabulary words are quite common while tweeting online. The categorization of such words per polarity gets tough for natural processors involved. This project uses Apache Spark's fast processing capabilities to analyze sentiment from such high velocity real-time tweets.

The rest of this project report is structured as follows. In Section II, we detailed some related work of our project by highlighting important features. Next, Section III gives brief details about the technologies used. Section IV cover details of methodology & implementation of the project. The problems we came across and the challenges we resolved during implementation are specified in section V. Further, in Section VI, future work is discussed. Finally, Section VII concludes the report.

II. LITERATURE REVIEW

This section summarizes some of the scholarly and research works in the field of Machine Learning and data mining to analyze sentiments on the Twitter and preparing prediction model for various applications. As the available social platforms are shooting up, the information is becoming vast and can be extracted to turn into business objectives, social campaigns, marketing and other promotional strategies as explained in [4]. The benefit of social media to know public opinions and extract their emotions are considered by authors in [2] and explained how twitter gives advantage politically during elections. Further, the concept of the hashtag is used for text classification as it conveys emotion in few words. They suggested how previous research work suffered from lack of training set and misses some features of target data. They opted two stage approach for their framework- first preparing training data from twitter using mining conveying relevant features and then propounding the Supervised Learning Model to predict the results of Elections held in USA in 2016. After collecting and preprocessing the tweets, training data set was created first by manual labelling of hashtags and forming clusters, next by using online Sentimental Analyzer VADER which outputs the polarity in percentage. This approach reduced the number of tweets or training set and further they applied Support Vector Machine and Naive Bayes classification algorithm to determine the polarity of tweets. Multistage Classification approach was used where an entity classifier receives general class of tweets and categorise them with respect to individual candidates for comparison. The metric they used to determine the winner was the "PvT ratio" which is Positive number of tweets to total count of tweets for respective candidate.

Sentiment Analysis by researchers Imran et al. [1] exploited the technology 'Apache Spark' for fast streaming of tweets and presented the approach StreamSensing to handle real time data in unstructured and noisy form. They conducted the approach on twitter data to find some useful and interesting trends which further can be generalized to any real-time text stream. Unsupervised learning approach is used to locate interesting patterns and trends from tweets processed on Apache Spark. Inspired by the approach described by Zhu et al. [7] and Li et al. [8] for mining data by selecting time window, authors [1] opted for sliding window method for capturing the live streams of tweets. The common approach found in almost all relevant research works constitutes data collection using Twitter API, preprocessing of data, filtering of data then approaches in feature extraction, classification and pattern analysis makes the distinction. Authors used sliding window of 5 minutes during data collection and further created Term Document Matrix(TDM) for feature extraction. The pattern analysis was carried out by using the score of TF-IDF for finding most important keywords as explained in [9] by Wu et al. The trending topic or hashtag is fed and tweets relevant to it are filtered to form TDM and computing the weights of TF-IDF to find most important words is the key idea of this sentiment analysis. Parallel computation of TDM, TF-IDF score and determining top 5 keywords generated from TDM in each minute as the sliding window moves are one of the highlighting features of this research work. Thus, it leverages the fast computation power of Apache Spark.

In another work [5] of Sentiment Analysis and Influence Tracking on Twitter, authors also predicted the polarity – positive, negative or neutral of tweets by creating a classifier. In addition, they used multiple algorithms and methods to determine the influence of active entity on the tweet patterns of users exhibiting certain emotions. They mined tweets only at the entity level i.e. brand, product, celebrity elements present in tweets rather than the whole sentence in the tweets posted by users. The approach they followed using algorithms to extract features and track the impact and influence made their work different from rest of the literature. The feature extraction process after preprocessing included constructing n grams along with POS taggers taking care of negation part and improving accuracy of classification. For further analysis and measuring influence, they opted two algorithms – People Rank Algorithm inspired by Page Rank Algorithm [6] used by Google. The main idea behind this algorithm is more the value of People Rank, the more central is the node in the graph means its importance on twitter in terms of followers, retweets and mentions. The other algorithm is Twitter Rank algorithm, an extension to page Rank to determine the influence of users by considering the similarity between users and the structure of nodes i.e. other users they are linked to. They addressed the shortcomings of Page Rank and developed this approach. The influence measure is considered by following the idea that popular/influential people follow you and they act as medium to broadcast specific topic. Following some mathematical computation of ratio of followers/following, retweets, mentions like parameters, they determine the weights and finally derived a mathematical formula to track influence of specific entity. The approach they proposed have potential to determine influence personalities/entities on twitter and can be used for promotional and branding purposes.

III. TWITTER SENTIMENT ANALYSIS

A) Introduction to Problem

Every day massive amount of data is generated by social media users which can be used to analyze their opinion about any event, movie, product or politics. Conventional tools like Apache Storm analyze stream in micro-batch whereas novel tools like Apache Spark process data in real

time making analyzing and processing of real time data possible.

B) Platform and Technologies

There are different technologies and tools implemented in the project. These are introduced below.

Apache Spark: It is an open source lightning fast cluster computing platform to retrieve streaming data and forwarding to storage system like HDFS, Database Server. It is built on top of Map Reduce and can integrate well with other Apache software. Apache spark is an in-memory fast processing system used for large scale data processing. It has come up as an advanced version of Hadoop. Though it implements the MapReduce technology but it processes data even 100 times faster by partitioning on memory and 10 times faster on disk across different nodes. Its structure is based on Resilient Distributed datasets(RDD) which is read only, multi sets of data partitioned and distributed across different node, to ensure fault intolerance and scalability factors. It overcomes the limitation of MapReduce in which data after reducing was stored into a disk by implementing iterative algorithms who fetch data from multiple datasets in a loop thereby implementing repeated database-style querying of data. In this way, the latency involved is reduced thereby making it faster. RDD is basically an abstraction feature which before data processing lays down the execution plan and then later depicts computation using Direct Acyclic Graph(DAG).The generated DAG acts as a framework to carry out the pattern analysis and processing and task segregation. Further, it has a better edge over other technologies as it is quite easy to implement due to multiple available APIs. Also, the other benefits include high level libraries. This inbuilt feature can deliver support to SQL, machine learning, graph processing and for streaming data. It can access data from different storage sources like HDFS, CASSANDRA, HBase, S3.

Scala: It is not only a High Level Functional but also supports Object Oriented Programming language model. This provides it an edge over Java which require more code to be written for the same task as compared to Scala. The major success of Scala is that Apache Spark is itself implemented in Scala. There are vast number of packages available in Scala language for Apache Spark. Thus, we proceeded with implementation in Scala as compared to Python or Java.

Twitter: It is an online social media platform which is suitable for our use case due to number of factors. Firstly, the amount of relevant data is much larger for twitter as compared to blogs or review websites. Secondly, response on twitter is general and prompt. Other social media giants like Facebook does not provide much data so using their public API was not considered. Finally, most twitter users voice their opinion about other people like actors, products: in case they bought a new phone or unsatisfied with customer service behaviour as opposed to other social media where users post most status and pictures of themselves. These factors make twitter a logical choice for real time data analysis.

IntelliJ Idea: It is an Integrated Development Environment to build, run and test code. It is closed source but community edition of the software is provided free of cost. It offers support for SBT plugin which is used to import Apache Spark dependencies and build the project. Intellij Idea professional edition is used along with SBT plugin which is a build tool, an alternative for maven build tool. SBT makes it easy to define dependencies and import libraries and dependencies.

IV. CASE STUDY

Our project involves the usage of Apache Spark to analyze real time tweets. The objective of our case study is to find the polarity of the words (in tweets) retrieved.

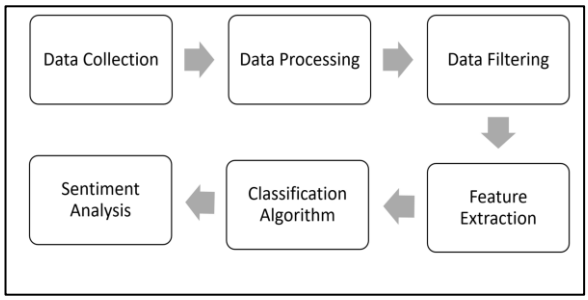

Fig. 2 Framework for Twitter Analysis

Each step in the framework involves several sub-tasks.

1. Data collection:

Data in the form of raw tweets is retrieved by using the Scala library "Twitter4j" which provides a package for real time twitter streaming API. The API requires us to register a developer account with Twitter and fill in parameters such as consumerKey, consumerSecret, accessTokenaccess, and TokenSecret. This API allows to get all random tweets or filter data by using keywords. Filters supports to retrieve tweets which match a specific criterion defined by the developer. We used this to retrieve tweets related to specific keywords which are taken as input from users. Initially, we set at least set an application name and mode. We execute the program in local mode instead of cluster. Then, input array of keywords is provided as an argument to Streaming Context "ssc" using "sc" where "sc" is spark context.

For example, on inputting multiple keywords like, 'Canada', 'Trump', 'Toronto', the output we obtained from 15 seconds' window time was the live stream of tweets associated with these keywords. Only caveat of using filters is that famous keywords like "India" have more

tweets compared to niche words like “Focusrite” which makes it difficult to get data for niche specific keywords.

2. Data Processing:

Data processing involves Tokenization which is the process of splitting the tweets into individual words called tokens. Tokens can be split using whitespace or punctuation characters. It can be unigram or bigram depending on the classification model used. The bag-of-words model is one of the most extensively used model for classification. It is based on the fact of assuming text to be classified as a bag or collection of individual words with no link or interdependence. The simplest way to incorporate this model in our project is by using unigrams as features. It is just a collection of individual words in the text to be classified, so, we split each tweet using whitespace.

For example, the tweet “Met aziz today !!” is split from each whitespace as follows.

{
Met
Aziz
!!”
}

The next step in data processing is normalization by conversion of tweet into lowercase. Tweets are normalized by converting it to lowercase which makes its comparison with an dictionary easier. The following function is used as shown in fig. 3.

```
def toLowercase(sentence: Sentence): Sentence =
  sentence.map(_.toLowerCase)
```

Fig. 3 code snippet for lowercase

3. Data Filtering:

A tweet acquired after data processing still has a portion of raw information in it which we may or may not find useful for our application. Thus, these tweets are further filtered by removing stop words, numbers and punctuations.

Stop words: For example, tweets contain stop words which are extremely common words like “is”, “am”, “are” and holds no additional information. These words serve no purpose and this feature is implemented using a list stored in stopfile.dat. We then compare each word in a tweet with this list and delete the words matching the stop list as shown in fig.

```
def keepMeaningfulWords(sentence: Sentence, uselessWords: Set[String]): Sentence =
  sentence.filterNot(word => uselessWords.contains(word))
```

Fig. 4 Code snippet for stop words removal

Removing non-alphabetical characters: Symbols such as “#@” and numbers hold no relevance in case of sentiment analysis and are removed using pattern matching. Regular expressions are used to match alphabetical characters only and rest are ignored.

```
def keepActualWords(sentence: Sentence): Sentence =
  sentence.filter(_.matches("[a-z]+"))
```

Fig. 5 Code snippet for removing non-alphabets

This helps to reduce the clutter from the twitter stream.

Stemming: It is the process of reducing derived words to their roots. Example includes words like “fish” which has same roots as “fishing” and “fishes”. The library to use stemming is Stanford NLP which also provides various algorithms such as porter stemming. In our case, we have not employed any stemming algorithm due to time constraints.

4. Feature Extraction:

TF-IDF is a feature vectorization method used in text mining to find the importance of a term to a document in the corpus. Feature extraction involves “mlib” library of Apache Spark. The recommended API is the Data Frame based API. This feature is useful for a case where we need to find trending topics or to create word clouds. However, this project is more focused towards finding sentiment in twitter streams so TF-IDF is not implemented.

5. Sentiment Analysis

Sentiment analysis is done by using custom algorithm which finds polarity as below.

Finding polarity: For discovering the polarity, we used a simple algorithm of counting positive and negative words in a tweet. For both, positive and negative words, different lists were made. Next step is to compare every word in a tweet against both these list. If the current word matches a word in positive list, then a score of 1 is incremented and if a negative word is found then it is decremented. More positive words lead to higher sentiment score as shown in fig. 6. However, Standford NLP can be used to predict accurate sentiment analysis which provide complex algorithms to predict it.

Sentiment Analysis output: The output contains a list of tweets in real time along with their sentiment score on the left-hand side. The first tweet has score of -2 which is due to two negative keywords. Next two tweets are positive as they contain keywords like “good” and “great. Both these words are in the positive words list. It is to be noted that if a tweet has a score of 0, then it is ignored from final output. The problem with neutral tweets is that they serve no purpose as they don’t convey any sentiment towards the product.

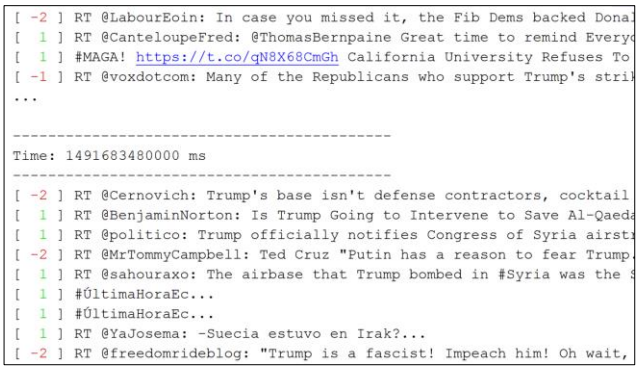

Fig. 6 Sentiment analysis result

The last tweet is most negative tweet with sentiment score of -2 which contains some abuse word not shown. Negative tweets indicate hate and dislike towards a product or public figure. The result here indicate that People don't hate Donald Trump as portrayed in media and news as general sentiment regarding trump is positive as indicated by the results.

V. DISCUSSION

Developing the project proved to be a lot more challenging than expected due to the relative inexperience we had with Apache Spark and Scala.

A) Project Limitation & challenges

Following challenges were faced during implementation.

Apache Spark Memory error: Apache spark has a setting related to allotted memory for processing the program and the default value was less than what our application needed. The solution was to change settings in VM options in IntelliJ Idea settings by adding following parameters.

```
-Xms128m -Xmx512m -XX:MaxPermSize=300m -ea
```

Accessing Country Specific Tweets: There was no parameter in twitter API to restrict tweets to a specific country. This prevented us to retrieve tweets from a specific region to analyze which could be a future work.

Library dependencies: There were some initial challenges in building the application using SBT tools due to incompatible versions of Scala and Scala SDK as we had limited knowledge about the technologies we were using. Moreover, the given examples used outdated libraries which we update to latest by comparing the given version against maven repository.

VI. FUTURE WORK

From future perspective, we would like to extend this project by implementing some machine learning algorithms for applications like election results, product ratings, movies' outcomes and running the project on clusters to expand its functionalities. Moreover, we would like to make a web application for users to input keywords and get analyzed results. In this project, we have worked only with unigram models, but we would like to extend it to bigram and further which will increase linkage between the data and provide accurate sentiment analysis results.

Computation of overall tweet score can be done for a single keyword which can provide an overall sentiment of public regarding a topic.

VII. CONCLUSION

Twitter is a source of vast unstructured and noisy data sets that can be processed to locate interesting patterns and trends. Apache Spark proved prolific in extracting live streams of data and has further capability to store batches of data in HDFS and other major conventional storages. The processing capabilities of Spark makes the project flexible to further extend to multiple nodes, thereby supporting distributed computing. Real time data analysis makes it possible for business organizations to keep track of their services and generates opportunities to promote, advertise and improve from time to time.

Our heartfelt appreciation goes to Professor Imran Ahmad with regards to his feedback across the course of project from the initial proposal up to the conclusion and for the valuable lessons learned along the way including collaboration within a group and the challenges involved in a large-scale software development efforts.

REFERENCES

[1] Dr. Khalid N. Alhayyan & Dr. Imran Ahmad "Discovering and Analyzing Important Real-Time Trends in Noisy Twitter Stream" n.p

[2] J. Ramteke, S. Shah, D. Godhia, and A. Shaikh, "Election result prediction using Twitter sentiment analysis," in Inventive Computation Technologies (ICICT), International Conference on, 2016, vol. 1, pp. 1–5.

[3] M. Desai and M. Mehta, "Techniques for sentiment analysis of Twitter data: A comprehensive survey", 2016 International Conference on Computing, Communication and Automation (ICCCA), 2016.

[4] Alexander Pak and Patrick Paroubek. "Twitter as a corpus for sentiment analysis and opinion mining". In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, may 2010.

[5] R. Mehta, D. Mehta, D. Chheda, C. Shah, and P. M. Chawan, "Sentiment analysis and influence tracking using twitter," *International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE)*, vol. 1, no. 2, p. pp–72, 2012.

[6] Mtibaa, M. May, C. Diot and M. Ammar, "PeopleRank: Social Opportunistic Forwarding", 2010 Proceedings IEEE INFOCOM, 2010.

[7] Zhu, Y. and Shasha, D. (2002). Statstream: Statistical monitoring of thousands of data streams in real time. In Proceedings of the 28th Very Large Data Base Conference. 358–369

[8] Li, H.-F. and Lee, S.-Y. (2009). Mining frequent itemsets over data streams using efficient window sliding techniques. Expert Syst. Appl. 36, 2, 1466–1477.
[9] H. Wu and R. Luk and K. Wong and K. Kwok. "Interpreting TF-IDF term weights as making relevance decisions". ACM Transactions on Information Systems, 26 (3). 2010