North Carolina State University

CSC 591: Advanced Algorithms

Capstone Project Report

Cricket Sentiment Analysis in Space & Time using Twitter

Submitted by:
Abhishek Agrawal {akagrawa}
Nirmesh Khandelwal {nbkhande}
Nisarg Gandhi {ndgandh2}
Rohit Arora {rarora4}

May 1, 2015

Table of Contents

1. Problem Statement

In this project our aim was to perform Cricket Sentiment Analysis in space and time using Twitter tweets. For this task, which is not frequently occurring, we have persisted tweets from Cricket World Cup semifinal match between India vs Australia 2015 match (as it is not round-the-clock event) for real-time event emulation.

2. Learning Objectives

From this project we built an in-depth understanding of constructing an end-to-end real-time sentiment analysis system using cutting edge technologies such as MongoDB (for stream emulation), Apache-Storm streamparse framework, Redis and NodeJS. We have also put into use the probabilistic data structures (such as count-min etc.) to answer many questions (such as top trending players, most talked batsmen, most talked topic) in real-time and to detect anomalous behavior as well.

We also explored various JavaScript visualization libraries that can provide a simplistic understanding of project outcomes in space and time. In the temporal analysis we have displayed the changes in the sentiment of the people as the match progresses. For spatial hotspot analysis, we have captured the tweet sentiments and country-wise changes in trend with respect to tweet counts. We are also displaying to the users trending top 50 words (using word cloud) during the match and how these varies.

Given the scope and size of project and the components installation is definitely a big challenge, we also tried to minimize this complexity by writing an automation script that makes it easier for a user to setup the system to perform this analysis.

3. Description of algorithm(s) used

We have used CountMin Sketches, also known as sublinear space data structure used for summarizing data stream [7], to implement TopK (where we have fixed K=50). A Top-K query determines the 'k' most common items in a data stream: $\{i_1,,\ldots,_k\}$ such that $ci_1 \geq ci_2 \geq \cdots ci_k \geq c(j \neq i_1,\ldots,i_k)$, where '$i_x$' is the item 'x' and '$ci_x$' is the count of the item i_x. We are using Top-K to find top 50 trending words tweeted during the cricket match, these keeps on updating on the user's dashboard.

In order to determine the sentiment of tweets in time, we also perform binning of data into time slots and then perform sentiment analysis using *textBlob* package of python. Similarly we divide the data based on country code and detect its sentiment in order to perform space based analysis. Details of the implementations can be found in successive sections.

4. Implementation & Experiment Design

In the system architecture diagram shown in figure 1 below, we have shown the frameworks that we have used and the workflow between the modules that aids our analysis on the streaming data. Below are the module wise details of the components.

a. Twitter Stream API : We have used twitter public stream API[1] to capture the cricket match related track words such as cwc15, worldcup, indVsAus etc. These tweets were captured via NodeJS, our Event Receiver, using ntwitter package. [**Please note:** In order to avoid a bias we ensured that track words are neutral and not hinting towards any of the contending teams].

b. Event Receiver: NodeJS script which is our event receiver for tweet. The received tweets were then pre-processed to capture the useful data (time, geo location and tweet texts) for the further processing.

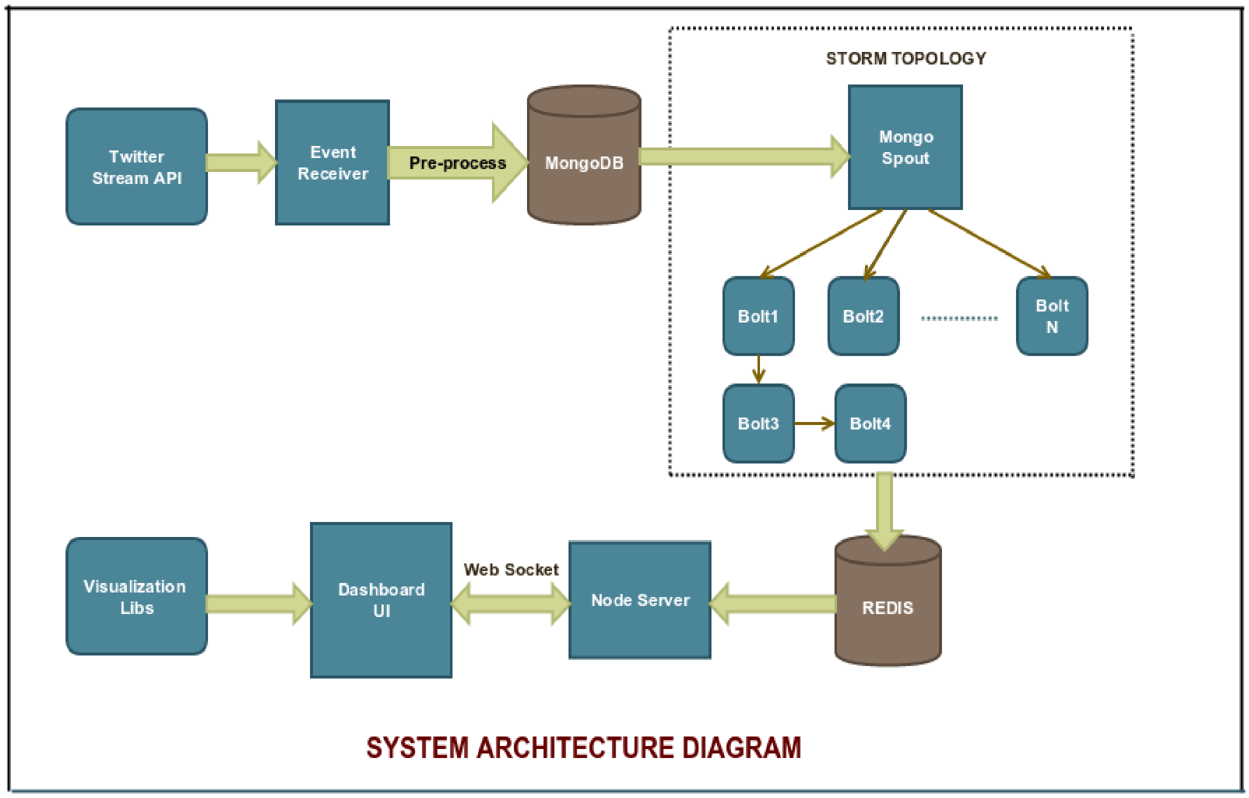
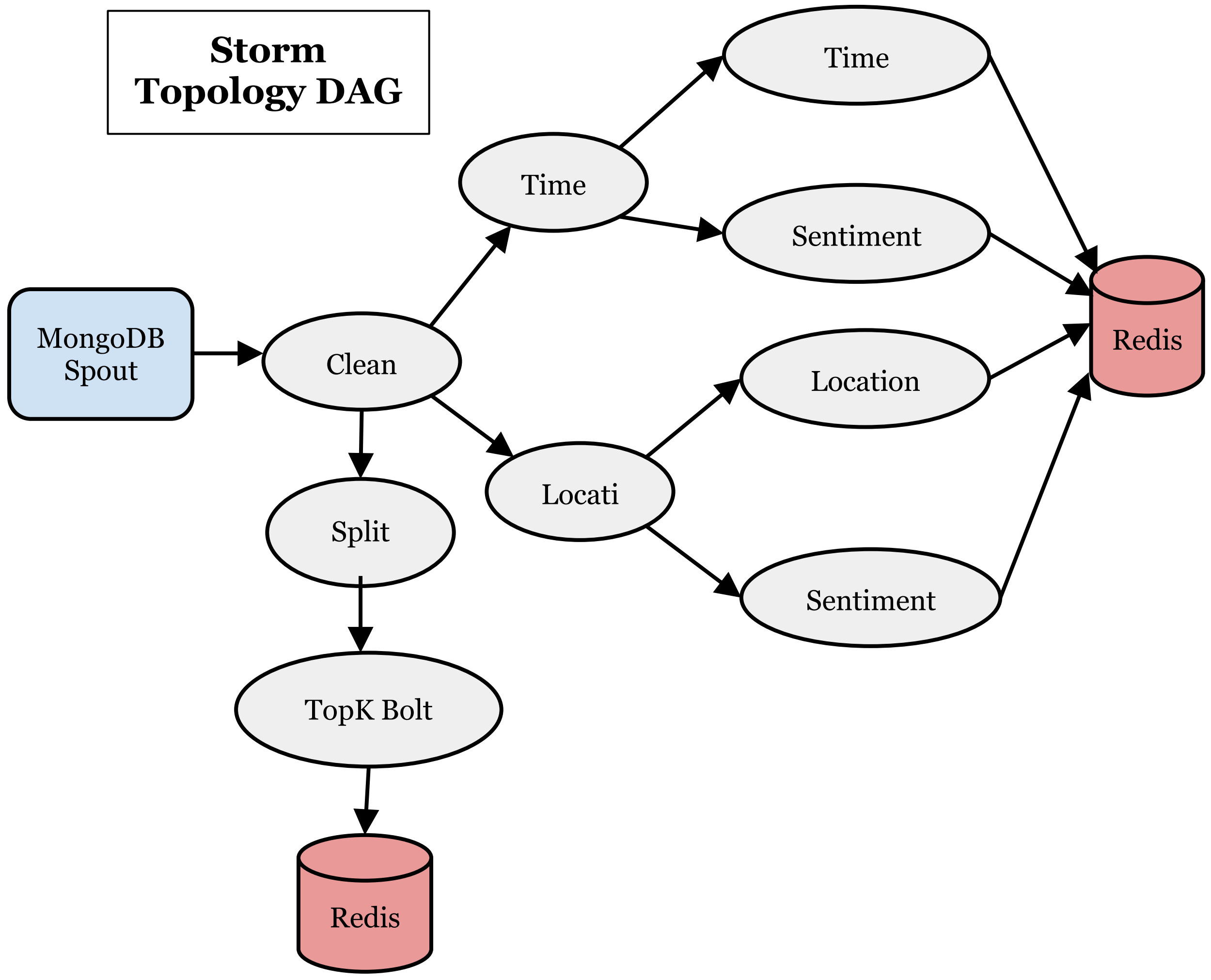


Figure 1: System Architecture

c. MongoDB: We then used mongodb server to persist around 640K pre-processed tweets during the match hours and used indexing on timestamp and tweet id to store unique and timestamp sorted tweets in the database. We then access the persisted tweets using mongodb queries to emulate real-time streaming for our space and time analysis.

d. Storm Topology: We have used a project management package for Apache storm called streamparse[5], it enables to write spouts and bolts in Python and provides a single command setup of storm project. This is how our directory structure is laid:

- Spouts are located inside *src/spouts/ folder*.
- Bolts are located inside *src/bolts/ folder*.
- Some Extra helper files are located inside *src/helper/ folder*.
- Topologies are defined in form of DAG using a clojure file in topologies/*tweetAnalysis.clg*.



- **MongoDB Spout:** Reads data from MongoDB and emits the data in the form of a stream.
- **Clean Up Bolt:** Performs basic text cleanup operations on tweet text.
- **Split and Filter Bolt:** Split the tweet text into words and remove the stop words.
- **TopK Bolt:** Maintain a *CountMin* Sketch in order to calculate Top-k words.
- **Time Slot Creation Bolt:** Performs Binning on timestamp in order to perform time series analysis.
- **Location Filter Bolt:** Filter the tweets based on country code. In a large proportion of data, country was not available. We have extracted the location using time zone in such cases.
- **Time based Tweet Count Bolt:** Count the number of tweets received in particular time slot bin.
- **Sentiment Analysis Bolt:** To get the sentiment of tweet text, we have used "textBlob" python package to get the sentiment polarity. If the polarity is positive, it corresponds to positive sentiment and if it is negative then it indicates the negative sentiment else it is a neutral sentiment.

- **Location Based tweet Count:** Count the number of tweets received in particular country.

e. Redis Database: We have used Redis Server as an in-memory database to persist the outputs of the storm topologies for the further aggregated analysis. Redis has supported hands on features for real-time updates on the aggregated results and simultaneously leveraging the APIs to provide the current status of the stored key-value pairs. It has also serves as an intermediary layer between our data processing using Storm and dashboard application layer, from which the dashboard (implemented using NodeJS) fetches the most updated values independent of storm processing.

f. NodeJS : For implementing real time dashboard. NodeJs polls redis database for any updates and displays result in web browser. We have used Jade and express node_modules for the Dashboard UI. Similarly we have used web-socket protocol based modules for client-server communication.

g. Dashboard and Visualization Libraries: We have used twitter bootstrap based UI theme to display different analytics charts. We have explored many visualization libraries such as d3.js, viz.js, wordcloud2.js, datamap.js etc. and selected a few of them to display our space, time and text data analytics.

5. Experiment Results

In the experiment we conducted using twitter tweets for the cricket match event, we have obtained the following results as part of our multi-dimensional analysis of the data.

Time Series Analysis: We have persisted our data in the time sorted manner and kept the timestamp with each tweet we collected. This facilitated us to perform any kind of time-resolution time-series analysis. To capture the time peaks wrt. to the number of tweets per time unit for the entire match, we have kept our time-slot to 10 minutes. In each 10 minute time-slot we have captured the total number of tweets, positive sentiment tweets, negative sentiment tweets and neutral sentiment tweets pertaining to that time slot. We have used vis.min.js[2] visualization library to visualize our time-series data. This library provides the user an easy interface for analyzing time-series tweets at different level of time resolution (or granularity) with mouse scroll.

For the sake of capturing entire match, the resolution of the time-series plot shown in figure 2 is set for an hour interval. We have only projected the total tweets, total positive and negative sentiment tweets in the time-series plot below. In the plot above we can observe several time peaks through the match event.

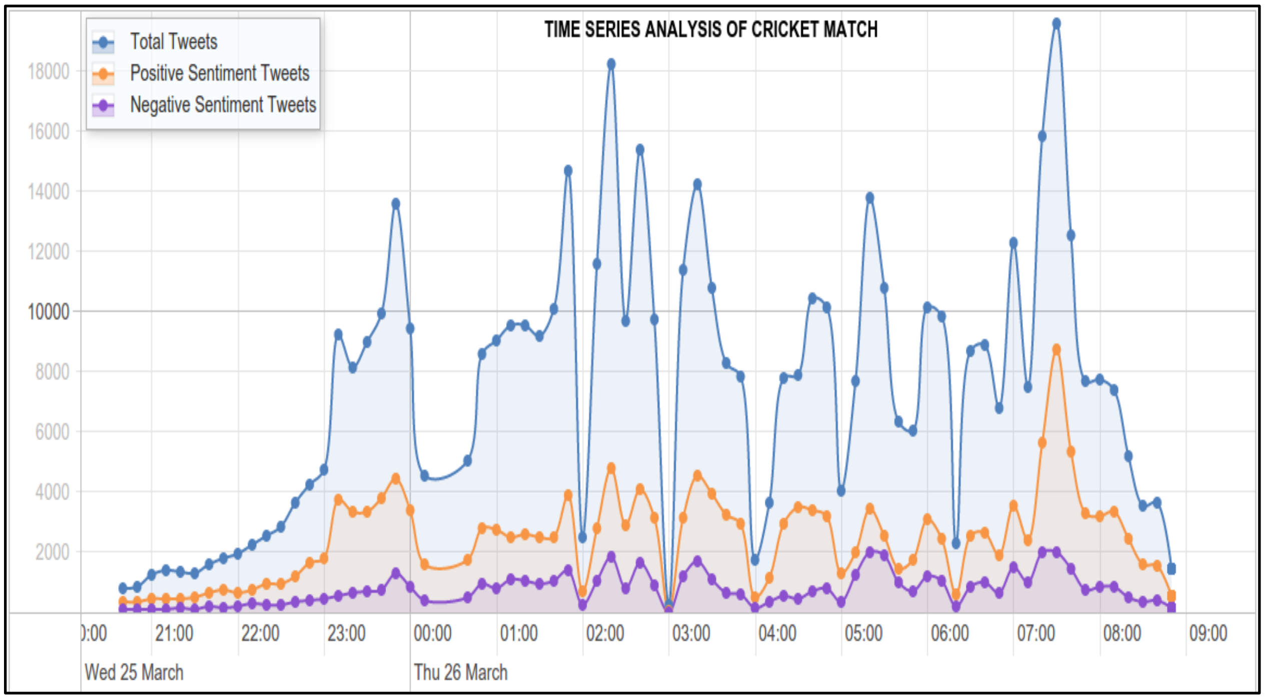


Figure 2: Time Series Sentiment Analysis of Ind vs Aus Cricket Match

Geo-Spatial Analysis: We have also analyzed the tweets wrt to its geographic locations. To capture the geo-location we have persisted the geo-locations of the tweets consisting of geo-attributes such as country, country-code, city, latitude, longitude etc. But these fields are present only on the tweets whose users (who posted tweets) have enabled their location sharing settings on. But we found that the timezone field coming in the tweets capture the user's exact timezone and we were able to successfully map it backwards to the user's country. For the visualization purpose we have used datamap.js[4] which uses d3.js[5] library for projecting world heat map. In the plot below figure 3, we have captured the heat map of the tweet count aggregating for all the countries. Here the level of heat map which indicates the tweets frequency is decided based on the log of number of tweets per country. Simultaneously, we have also analyzed the country-wise sentiment of the tweets to capture country wise changing sentiment trends of the users on country basis during the match.

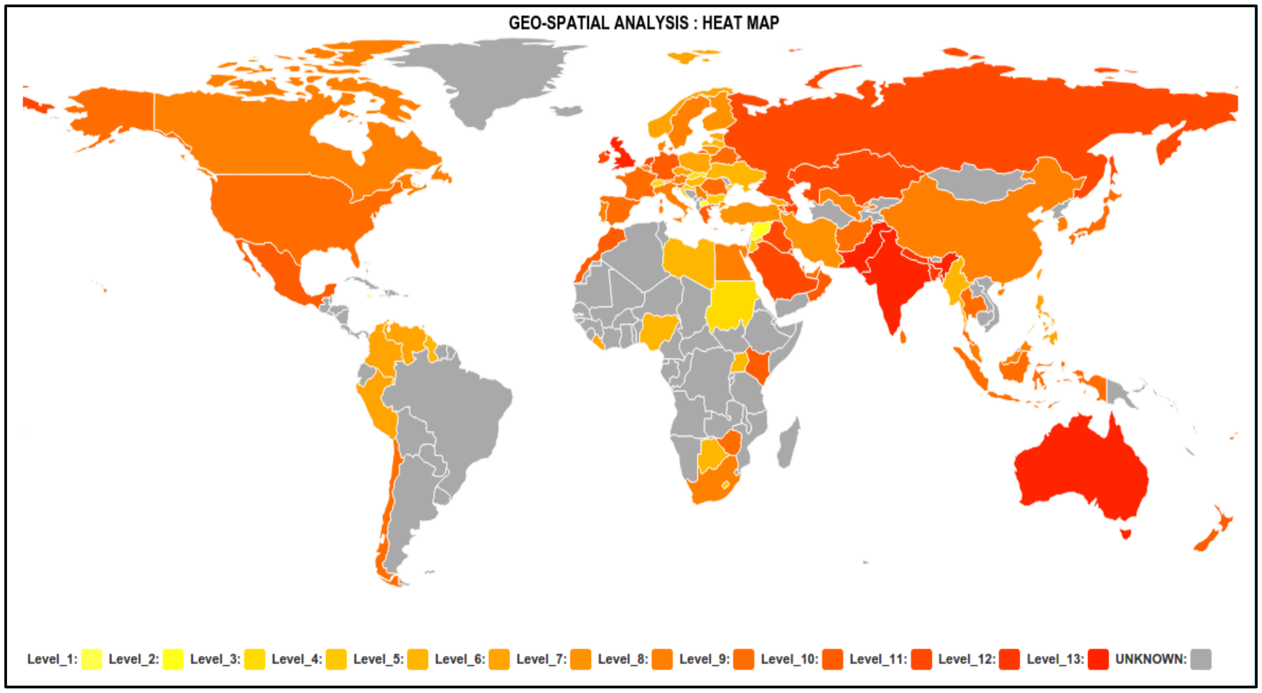

GEO-SPATIAL ANALYSIS : HEAT MAP
Level_1: Level_2: Level_3: Level_4: Level_5: Level_6: Level_7: Level_8: Level_9: Level_10: Level_11: Level_12: Level_13: UNKNOWN:

Figure 3: Geo-Spatial Sentiment Analysis of Ind vs Aus Cricket Match

Text Analytics: We have also analyzed the tweet's text to capture the top-k (k=50) words which reflects the most trending words during the cricket match. For the visualization purpose we have used wordcloud2.js library to project the top-50 most trending words in the form of word-cloud. We have assigned the square-root of the word-frequency to give the weights to the word in the word-cloud so that the words can be projected in proportion to their frequency. We have used count-min sketch inside storm bolts to capture the top-k words in the streaming data.

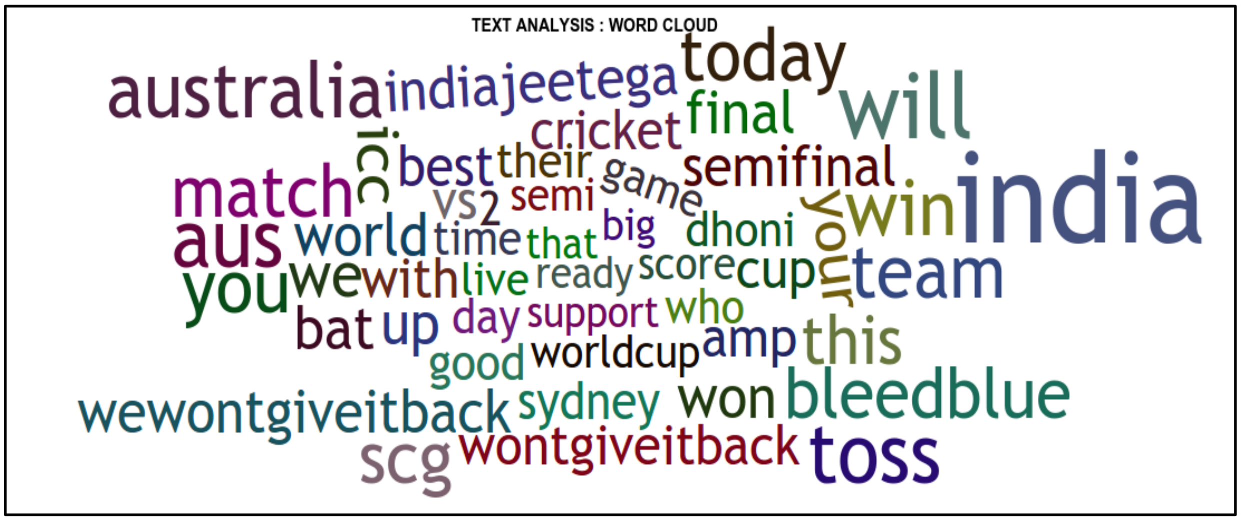


Figure 4: Text Analysis of Ind vs Aus Cricket Match

6. Analysis and Result Interpretations

After performing the multi-dimensional analysis of the cricket match data, we have drawn below conclusions based on visual analytics.

Time Series trends: From the time-series plot below, we have inferred that the peaks with respect to the tweet count correspond to the actual big moments in the match. For eg. we have identified that the first peak in time-series associates with the match start time and the very next peak associates with the first wicket down moment of Australia, which can be verified against the ground truth data, available online. Similarly the last peak in the plot corresponds to the end of the match moment. These time anomalies/peaks are the resultant of exceptionally high values of tweets wrt. neighborhood time slots.

So as shown in figure 5, we also observed more peaks in the second half of the plots, indicating the falls of the Indian team wickets. We can also observe the positive and negative sentiment trend in the time-series data as proportion to the total tweets. During the start of the match there is a peak in the positive sentiments. Whenever there is a fall of a wicket, we can see a raise in the negative sentiment tweets (tweets showing anger, frustration). During the end of the match, there was a clear peak in the positive tweets as well as the negative tweets which is as a result of an exciting yet unexpected match result.

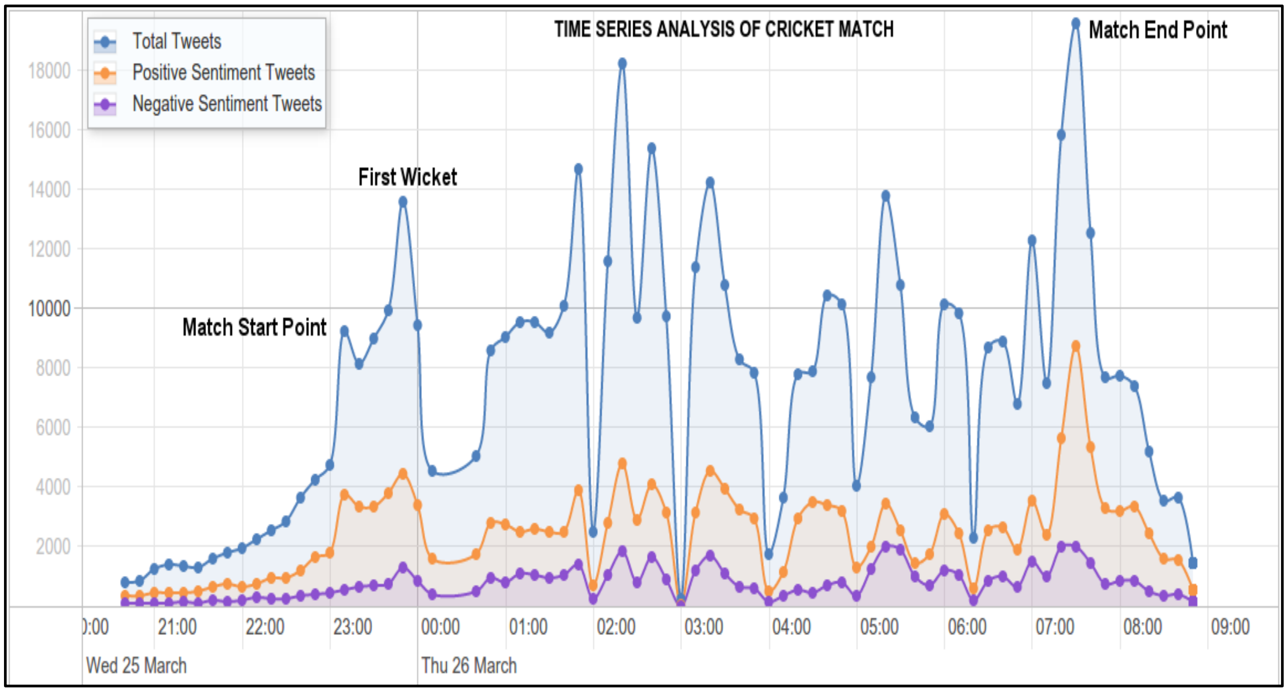


Figure 5: Time Series Sentiment Analysis with validation against ground truth

Geo-Spatial trends: For the geospatial hotspot analysis we have identified the countries with top tweet-counts in the heat-map. From the plot below, we can see that countries in Asia, and the Australian continent have been most active throughout the event. The heat map clearly reveals the world-wide fan following of the cricket match and specifically projecting the countries such as India, Australia, Pakistan, England, South Africa, New Zealand, Bangladesh, Sri-lanka etc. in high heat map levels which in this context are the countries that actually participated in the Cricket World Cup 2015.

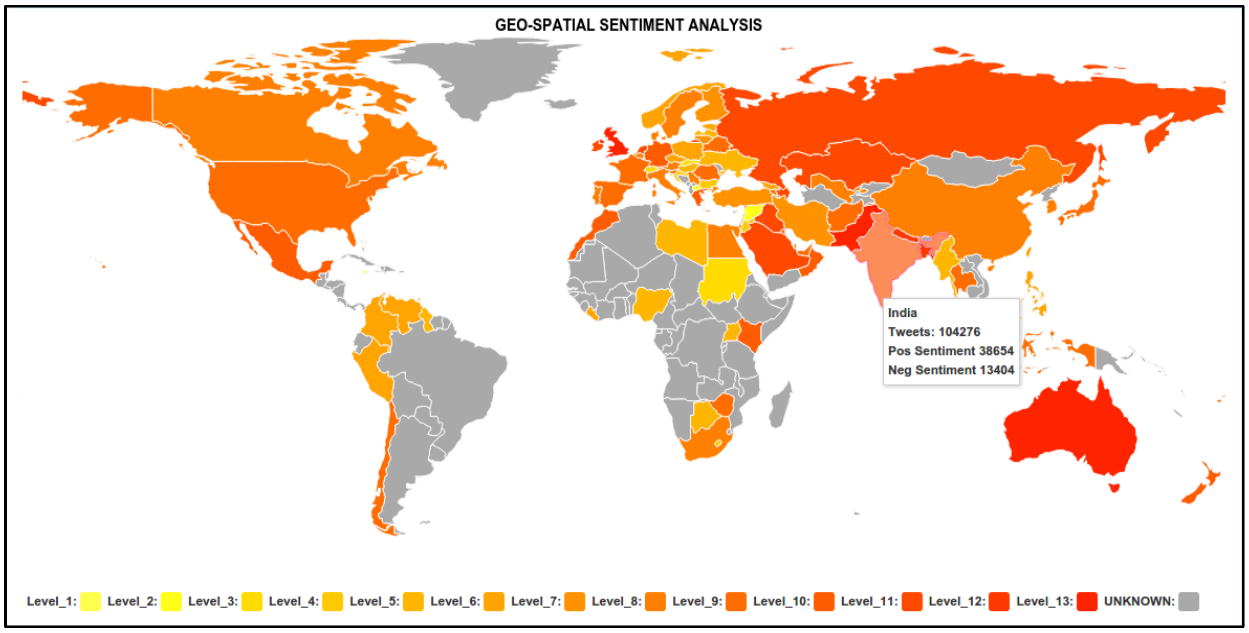


Figure 6: Geo Spatial Sentiment Analysis with validation against ground truth

We also observe less or null tweets from the countries in West part of South America and East Africa indicating very low influence of cricket as a sport in these regions. Along with tweet counts, we also have analyzed country-wise sentiment count which is also in proportion to the actual tweet counts and the ratio of positive to negative tweets is relatively high for countries like India and Australia which is interesting as they were playing the match against each other.

Top trending words: The word-cloud, figure 4, generated from the top 50 trending words clearly depicts some of the most expected trending words like India, Aus, cwc15, team, semifinal, etc. All these words are closely related to the Cricket World Cup and the semi-final match between India and Australia. Moreover this word cloud was captured during the time when the Coin Toss took place. So we can observe some of the keywords like toss, dhoni, bat, etc. Also we observed that there was a hashtag #WeWontGiveItBack captured in the word cloud. It was really popular on Twitter during the entire world cup.

Although the analysis and results that we have shown here are specific towards the time-sensitive cricket match event, but the tools, technologies and the system design we have adopted are capable of performing any such spatial, temporal, sentiment and textual analytics for any kind of streaming data. In the future, we may extend this application robust enough to capture any user defined track words to analyze spatio-temporal and hot topic trends of the current events in real-time. With this capstone project, we have strived to cover most of the topics covered as part of our curriculum to provide analytics on streaming data.

7. Lessons Learnt

We achieved our goals successfully by implementing an end-to-end real-time sentiment analysis system. In doing so we were able to integrate MongoDB, Apache-Storm streamparse framework, Redis and NodeJS. On top of this integrated framework we utilized probabilistic data structure, count-min sketch to provide to user's top trending K words (TopK) in real-time.

We also utilized many JavaScript Visualization libraries and Twitter Bootstrap that helped us in designing an intuitive User Interface. And, finally we devised our own Automation Script that aids setting up this huge end-to-end system seamlessly and provides analysis.

8. References

[1] "Twitter Public Stream APIs" https://dev.twitter.com/streaming/public

[2] "Time series analysis using vis.js" http://visjs.org/

[3] "Geo-spatial analysis using datamap.js" http://datamaps.github.io/

[4] "Text analytics with word cloud using wordcloud2.js" https://github.com/timdream/wordcloud2.js/blob/master/API.md

[5] "Stream Parse for easy integration on Python with Storm." https://github.com/Parsely/streamparse

[6] "Virtual Env tool to create isolate environment for python project." https://virtualenv.pypa.io/en/latest/

[7] Graham Cormode , S. Muthukrishnan, An improved data stream summary: the count-min sketch and its applications, Journal of Algorithms, v.55 n.1, p.58-75, April 2005