# Twitter review mining and analysis

Vismay Golwala
NC State University
vjgolwal@ncsu.edu

Dharmang Bhavsar
NC State University
dmbhavsa@ncsu.edu

Srija Ganguly
NC State University
sgangul2@ncsu.edu

Tushita Roychaudhury
NC State University
troycha@ncsu.edu

## ABSTRACT

Reviews, be it from peers or from the internet, is the first thing that a person looks for while making a choice between different places. Reviews are also helpful when you consider the organizational facade of businesses, because these reviews gives the management team a chance to look back and evaluate themselves and improve their services in the areas according to the feedback from the reviews. There are too many websites where a person can look for reviews of places. But, it is equally important nowadays to get a feel of how that place is perceived on social media. This project is an effort to do exactly the same by finding reviews about a particular location from Twitter and as an extended effort, we also recommend places surrounding the searched place having the most positive social media reviews.

## Keywords

Reviews, Tweets, Rating, Sentiment analysis, Aggregated reviews, Twitter, Google, API , NLP.

## 1. INTRODUCTION

People, wherever they go, look for reviews online and sometimes there are a lot of polarizing reviews about the same place and people just aren't sure what to do. During our research, we came across a strange positivity in reviews of places on portals such as Yelp and Zomato. To put things in perspective, about 68% of the reviews were 4 or 5 starred on Yelp [5]( See Figure 1 ). This data is from the day it was established till September 2017. This seemed eerily positive for a website which has most of the restaurants in the world listed; we doubted the fact if most of them were actually that good. Another thing that we came across during our research was that users were particularly more inclined to post negative reviews on social media than anywhere else [6]. That is the reason that we are trying to get reviews that are disguised as Tweets from Twitter.

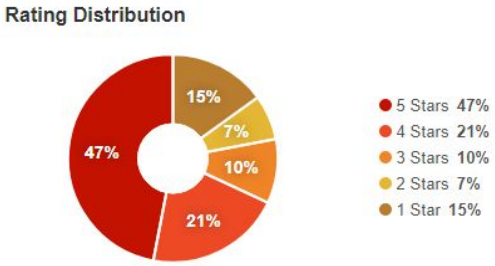



Figure 1: Yelp rating distribution

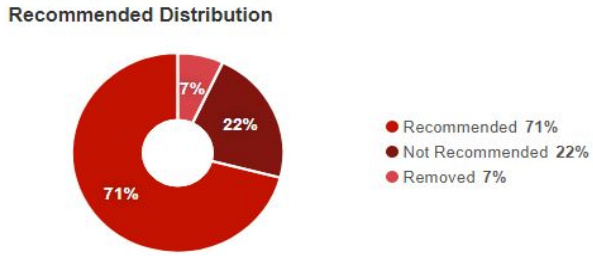



Figure 2: Yelp recommendation distribution

Reviews form an important part in the lives of people especially because they don't want to compromise on the quality and the reputation of a particular location [4]. It is also an important factor in their decision making process when they suggest places to their close ones. The reviews are a direct expression of the thoughts that people have about the place and categorize the rating into three parts - positive , negative and neutral - and it uses heavy involvement of users to filter data regarding the same and performs prediction analysis by cross-checking the current data with other users' data [4]. The overall rating is based on

an equation and takes into account all the categories of rating.

Following are the responses to certain questions asked to general users:

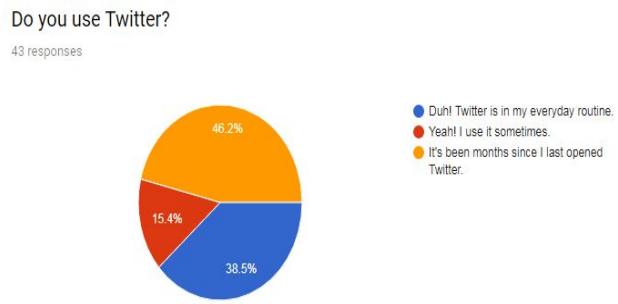



Figure 3: Twitter usage feedback

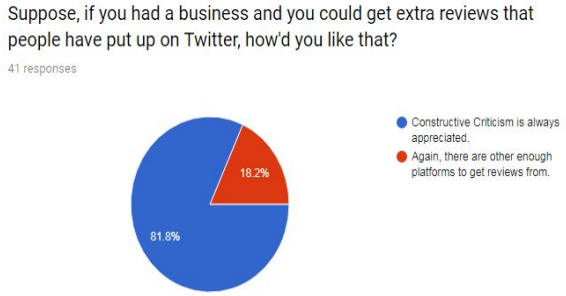



Figure 4: Need evaluation from feedback

Twitter being one of the most used social platform and the fact that Constructive feedback (including criticism) is always helpful to the customers; this idea can be effectively turned into a mechanism to aid in their decision process.

In addition, we used Google places to search for nearby places around a search term , that also displays the ratings of those places using Google reviews.

## 2. BACKGROUND LITERATURE

Substantial efforts have been spent on location identification and geo-tagging of documents and web pages. Social Networking on the other hand is still a very new field of computer science and little or no previous work has been done towards identifying the location of users based on their social activity.

The proposed method assumes that a dialog between users on Twitter has a constant topic and uses this model of social interaction in the Twitter network, along with content-derived location information, to employ a probabilistic framework to estimate the city-level location of a Twitter user [1]. This is done purely based on the content of the tweets, which may include reply-tweet information, without the use of any external information, such as a gazetteer, IP information etc.

Another work we read predicts a Twitter user's location based on his/her social network as opposed to the geo-tagging of tweets [2]. In the approach described in this paper, implicit attributes associated with the user in his/her social network are mined and user's location is predicted based on them. This paper does not estimate location based purely on content but uses additional knowledge to evaluate the user's location.

The work in geographic lexical variation [3] studies the variation of language usage on Twitter. Their multi-level generative model reasons jointly about latent topics and geographical regions allowing them to render each topic differently in each geographic region. This can be used to improve the accuracy of our location mining process.

These works can be used to augment our work in estimating if the user is really talking about the geographic location in consideration, thereby extracting the tweet content.

## 3. IMPLEMENTATION

### 3.1 Use Cases

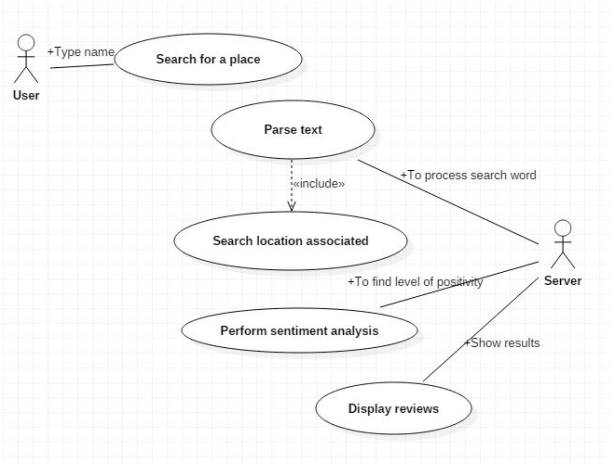



Figure 5: Use cases of application

### 3.1.1 Use case 1

| | |
|---|---|
| **Use Case Name** | Search for a place |
| **Actors** | User ( of the software ) |
| **Description** | The user will enter name of a place and hit search |
| **Trigger** | Use case is triggered when the user initiates the software to lookup a place and selects the search feature |
| **Preconditions** | 1. User has a PC<br>2. User has active internet connection |
| **Post conditions** | 1. User interface shows list of reviews or error |
| **Normal Flow** | 1. User starts software<br>2. User selects search feature and enters a name<br>3. User hits search button<br>4. Server takes up user's request |
| **Exceptions** | 1. Software doesn't start appropriately<br>- Restart software and try it again<br>2. Server remains unresponsive<br>- Restart server<br>- Try it after sometime<br>3. Interface is unresponsive<br>- Refresh page<br>- Try it after sometime |
| **Includes** | - |
| **Special Requirements** | The software must have a robust interface that can handle user input effectively and is easy for the user to understand it's functionality. |
| **Assumptions** | 1. User understands English |
| **Notes and Issues** | Interface is basic and provides a trigger to the main backend work of the software |

### 3.1.2 Use case 2

| | |
|---|---|
| **Use Case Name** | Parse text |
| **Actors** | Application server |
| **Description** | The server will parse the text requested by the user to find reviews related to the place |
| **Trigger** | When the user sends a request by hitting search button , the use case is triggered |
| **Preconditions** | - There should be a valid search request |
| **Post conditions** | - The server should be able to recognize the word effectively to search for location. |
| **Normal Flow** | 1. Search word is recognized by the server<br>2. Using certain method, parsing happens |
| **Exceptions** | 1. Software doesn't start appropriately<br>- Restart software and try it again<br>2. Server remains unresponsive<br>- Restart server<br>- Try it after sometime<br>3. Interface is unresponsive<br>- Refresh page<br>- Try it after sometime |
| **Includes** | Search location associated |
| **Special Requirements** | The software must have a robust interface that can handle user input effectively and is easy for the user to understand its functionality. |
| **Assumptions** | 1. Server is modeled to understand English |
| **Notes and Issues** | The first step towards contacting the server |

### 3.1.3 Use case 3

| | |
|---|---|
| **Use Case Name** | Search location associated |
| **Actors** | Application Server |
| **Description** | The server will parse the data and search the twitter feeds for any data related to the text |
| **Trigger** | Triggered right after server receives text to parse |
| **Preconditions** | 1. Text is valid<br>2. It is possible to search for the text |
| **Post conditions** | 1. State of software that has reviews available |
| **Normal Flow** | 1. Server takes parsed text<br>2. The twitter feeds are searched via the text<br>3. Finds data |
| **Exceptions** | 1. Software doesn't start appropriately<br>- Restart software and try it again<br>2. Server remains unresponsive<br>- Restart server<br>- Try it after sometime<br>3. Interface is unresponsive<br>- Refresh page<br>- Try it after sometime |
| **Includes** | - |
| **Special Requirements** | The software must have a robust interface that can handle user input effectively and is easy for the user to understand it's functionality. |
| **Assumptions** | 1. User understands English |
| **Notes and Issues** | Multiple locations can be returned based on the area being searched |

### 3.1.4 Use case 4

| | |
|---|---|
| **Use Case Name** | Perform sentiment analysis |
| **Actors** | Application server |
| **Description** | This module is dedicated to estimate the level of positivity in the feedbacks concerning the particular location |
| **Trigger** | This can be initiated upon searching for a place by the user |
| **Preconditions** | 1. A block of text related to a place on twitter<br>2. Software is still active |
| **Post conditions** | 1. Interface shows if the feedback sounds positive or not |
| **Normal Flow** | 1. Server finds data in Twitter<br>2. Using sentiment analysis, language is processed to judge the level of positivity<br>3. Results generated |
| **Exceptions** | 1. Software doesn't start appropriately<br>- Restart software and try it again<br>2. Server remains unresponsive<br>- Restart server<br>- Try it after sometime<br>3. Interface is unresponsive<br>- Refresh page<br>- Try it after sometime<br>4. Sentiments vary from person to person. |
| **Includes** | - |
| **Special Requirements** | The software must have a robust interface that can handle user input effectively and is easy for the user to understand it's functionality. |
| **Assumptions** | 1. A valid block of tweet is present that can be analyzed<br>2. Display monitor is active<br>3. English as a language can be analyzed by the system |
| **Notes and Issues** | The most sensitive part of the system that needs a lot of attention |

### 3.1.5 Use case 5

| | |
|---|---|
| **Use Case Name** | Display reviews |
| **Actors** | Application server |
| **Description** | After finding data related to the place,, server displays the reviews on the interface |
| **Trigger** | Server searches for data related to the location and state of the software changes to display |
| **Preconditions** | - Match for text found<br>- Software is still active |
| **Post conditions** | - Interface shows the reviews or error |
| **Normal Flow** | - Server finds data in Twitter<br>- Displays results |
| **Exceptions** | Software doesn't start appropriately<br>- Restart software and try it again<br><br>Server remains unresponsive<br>- Restart server<br>- Try it after sometime<br><br>Interface is unresponsive<br>- Refresh page<br>- Try it after sometime |
| **Includes** | - |
| **Special Requirements** | The software must have a robust interface that can handle user input effectively and is easy for the user to understand it's functionality. |
| **Assumptions** | - User understands English<br>- Display monitor is active |
| **Notes and Issues** | Interface is basic that tells us the effective usage of the system |

## 3.2 Data Flow Diagram

The data flow diagram of our application is represented as follows. The external entity is the user. The processes are Search process , Tweet filtering , Sentiment analysis of reviews , nearby place search via Google and Google reviews filtering . The two main stores used are Tweet reviews and Google reviews databases. All data flows are intuitive due to the labels.

The flow of data originates from the user when it gives a location name input to the application. This input string is then matched to the correct location and its coordinates in the Google API. Then with that search string and the coordinate data from Google a range query is made on the Twitter API to get tweets from that area. Then, searching of input string is done in the obtained tweets and the tweets that contain the searched keyword are saved. Then sentiment analysis is done on these tweets and the sentiment polarity of the tweets are converted into star rating ranging from 0 to 5. The, total rating of that location is found and displayed as well on the basis of the star ratings through sentiment analysis.

Also, a range search is made on the Google places API to find the nearby places from the main location searched. This list contains locations which are of the same type as the location searched. i.e. building searched will give buildings in recommended places.

All this data is gathered and displayed on a clean UI on the screen of the user.

Also as an added feature, Google Street View of the main searched location is also displayed to get the look and feel of the location.

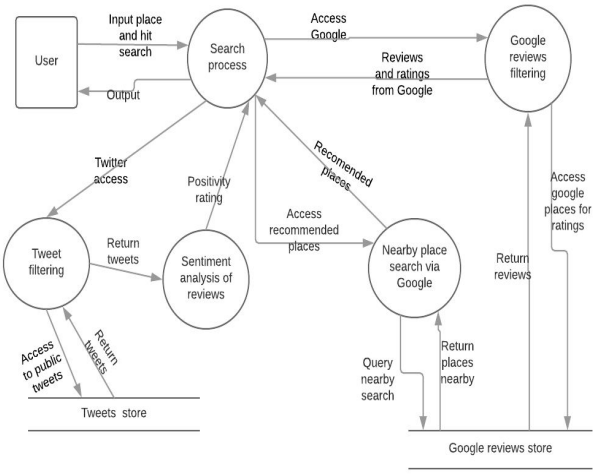



Figure 6: DFD of application

## 3.3 Screenshots of application


Figure 9 : Screenshot of homepage

The home page of our applications sports a Google Location API search box. As the user begins to type in the name of a place, google location suggestions are populated. The home page also provides quick-access icons of three popular locations.


Figure 10 :Eiffel Tower's street view upon search

Upon selection of a location on the home page, the user a redirected to another page that shows the locations street view, twitter reviews, google reviews and a button to fetch recommended places around.

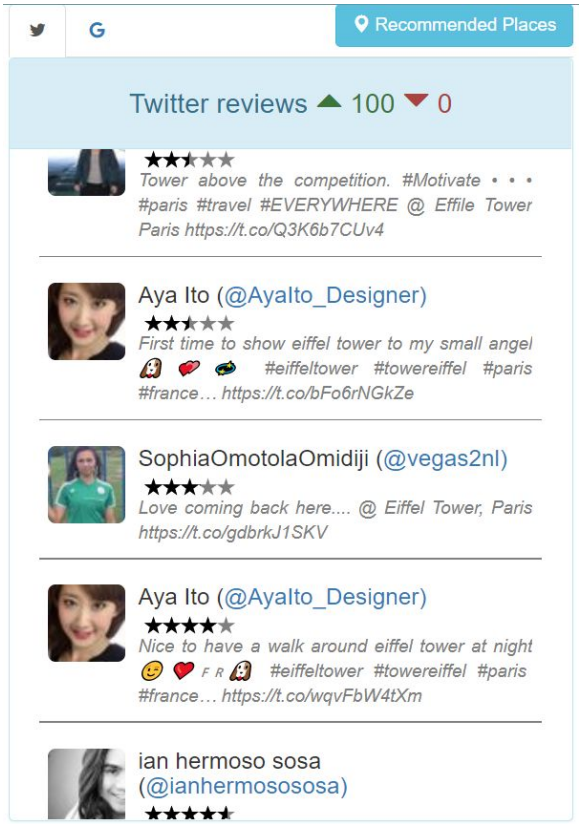



Figure 11 : Twitter reviews regarding Eiffel Tower

We used fuzzy N-word matching to first filter out the tweets that mention the location in consideration. We then use TextBlob sentiment analysis to assign a positive or negative scale to each tweet and correspondingly project it to a star rating.

The application also shows percentage positive and percentage negative reviews.

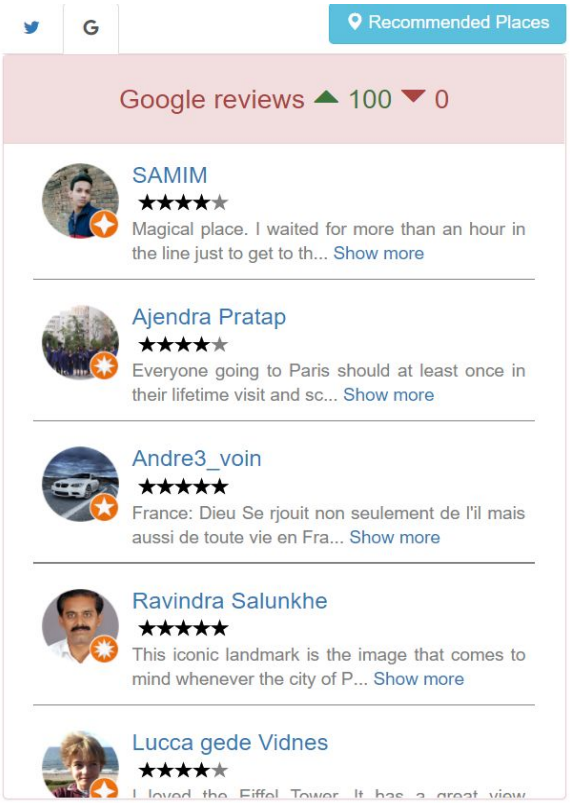



Figure 12 : Google reviews of Eiffel Tower

We use Google Places text-matching to retrieve google reviews that mention the location. The rating associated with the google user review is also queried and displayed as star rating.

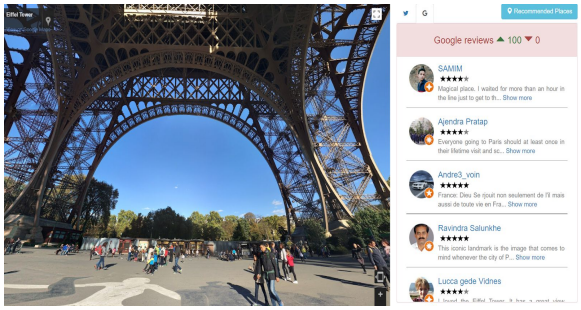


Figure13 : Overall look of the page after Search process is called

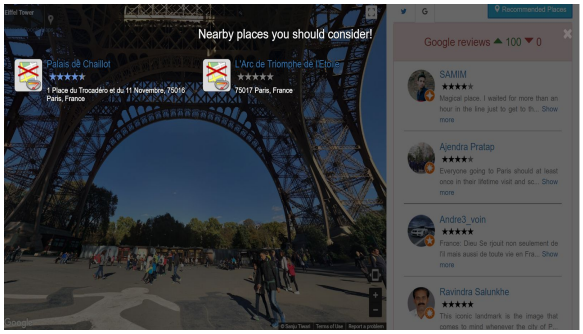



Figure 14 : Recommended place search

On hitting the “Recommended Places” button on top right of the page, a modal overlay is displayed that shows nearby similar places sorted in descending order of their rating. This is very useful when a user wants to check out highly rated places such as the one he’s currently at. This feature uses the Google places API, passing the radius of the search and the type of place to be be searched for, to obtain the results.

## 4. TECHNOLOGIES USED

Our system uses simple and easily available technologies. The description of the languages , APIs , libraries and frameworks used are as follows.

### 4.1 Languages

- **BackEnd** - Python, JavaScript.
  - Python is an easy to use language and provides support to data structures that our application uses.
  - JavaScript is also a robust and developer-friendly scripting language that could be easily integrated with our system.

- **FrontEnd** - JavaScript, HTML, CSS
  - The most common and usable web development languages.

- **Natural Language Processing (NLP)** - Python.
  - It is the most efficient and resourceful when it comes to NLP. There are several text processing libraries that can be used.

## 4.2 APIs

- Google Places API
- Google Maps API
- Twitter API

Google places , Google map and Twitter APIs are the very APIs on which our system is based. So choosing those were implicit. Google places lets us access information regarding a particular place; with Maps on the side, incorporating street view became easy.
Twitter API gave access to the relevant tweets that could be used for the sentiment analysis.

## 4.3 Libraries

The corresponding libraries that we considered were -

- Python Standard Libraries
- TextBlob
- Python-google-places
- Python-twitter
- Spacy

We chose the TextBlob library for sentiment analytics because of the cap limit of Microsoft and Google Analytics libraries. In addition to that, the tweets needed to be changed to a particular JSON structure for the Microsoft Azure TextAnalytics API causing an unnecessary overhead.
In any case, results of TextBlob and Microsoft Azure TextAnalytics are "almost" similar. Thus, these above details made us tilt towards using TextBlob.
Microsoft Azure TextAnalytics API had the following result when it tested on 2000 rows of data. It has a monthly limit of 5000 requests and a limit of 1000 rows at a time:

```
[2000 rows]
Accuracy 0.89863014257
Precision 0.907563210781
```

Figure 7 : Microsoft API efficiency result

Whereas , TextBlob had an almost similar result when it worked on more data. The reason for that might be that TextBlob has built in emoticon support while we did not find such support in the Microsoft Azure TextAnalytics API:

```
[6918 rows]
Accuracy 0.867236195432
Precision 0.894869638352
```

Figure 8 : TextBlob library efficiency result

Python libraries were an integral part of the development process. The usage of python-twitter and python-google-places libraries gave us access to several functionalities that were needed for achieving our goals.
Spacy was also used for some NLP tasks like Named Entity Recognition which was implemented to find the keyword from the tweets. The inbuilt N.E.R was not that effective in finding the keywords and tagging them because of the structure of the tweets. So another model of the N.E.R was made and trained with tweets collected from the python-twitter API. This model had a heavy overhead of this training mechanism and its asymptotic performance was not acceptable.

## 4.4 Frameworks

- **FrontEnd** - BootStrapJS, JQuery, RateYoJS , Django
  - The development of UI requires extensive use of the common web development languages - HTML, CSS and JavaScript. BootStrapJS and JQuery are known to provide quality support to development.
  - RateYoJS was used to show ratings in the form of stars.
  - Undoubtedly Django is the most commonly used framework for Python.
- **BackEnd** - Python, Django
  - The major functionalities of the system depends on the development of our backend which used Python as an obvious case and Django as the supporting framework.
  - Django in addition prevents cross site scripting.

# 6. TESTING AND EVALUATION

- **Natural Language Processing**

  The project was tested on 3 different NLP algorithms and in the end, analysis pointed out

that the use of the most simple algorithm was most justified. The three different algorithms used were:

1. Named Entity Recognition (N.E.R)
2. Fuzzy word matching
3. Regex Matching

For named entity recognition, a new model was trained using the tweet obtained by querying the Twitter API. The reason N.E.R was not used was because it was still missing many tweets because of the lack of structure in tweets and it was incredibly slow.

As for regex matching, it was giving performance but it was including tweets which did not involve the exact keyword we were looking for thus including extra tweets.

So finally, Fuzzy word matching was used. Its performance was not as great as regex matching, but better than N.E.R and unlike regex matching, this algorithm did not include more tweets than needed.

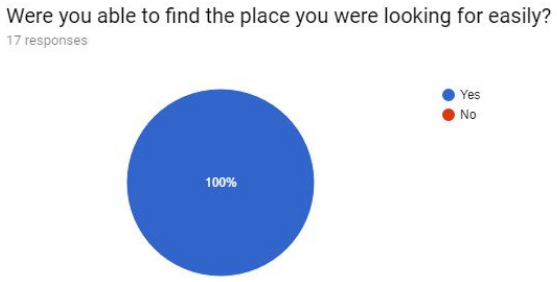
## 6. EVALUATION

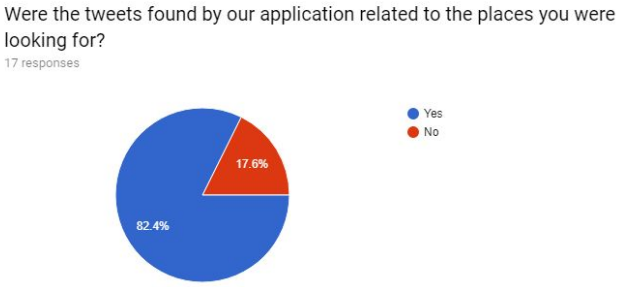
We developed a form and floated it that procured users' opinions based on three factors:

- Functionality of Search process : whether the search results are returning correct outputs.
- Accuracy of ratings of the places searched : This question is based on their perspective whether they think that the rating and reviews were approximately correct.
- Ease of use of the application : whether it helped the users understand the system easily and the flow of the application was intuitive.
- Importance of the street view search.
- Their opinion on whether the application was a good idea. This could in a way describe the "cool" factor.
- Further improvement ideas were gathered from them that could be looked forward to in the future scope.

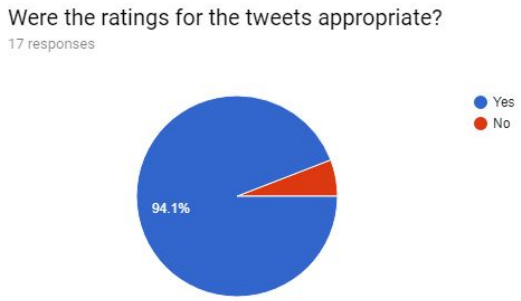
Graphs related to our evaluation form:

- The response for the functionality of the search process was highly positive.

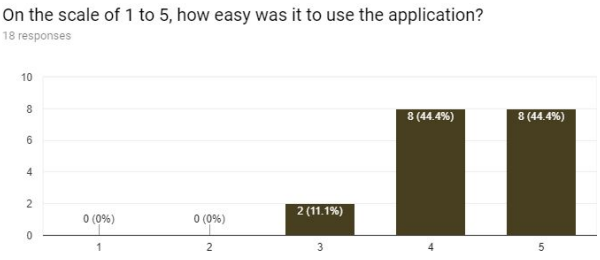


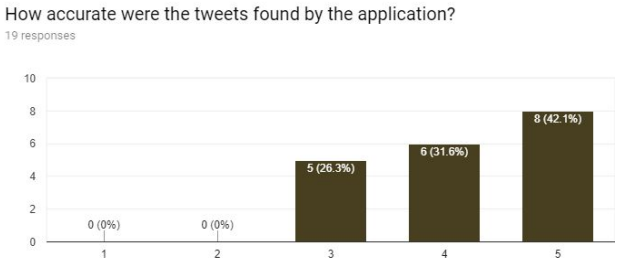

- Majority of the users were of the opinion that our system provided correct reviews.



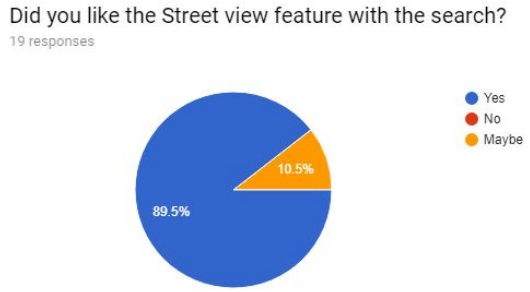

- More than 80 percent users were of the opinion that the application is easy to use and is intuitive.



- More than 40 percent users thought that the tweets generated are correctly corresponding to the searched places.




- More than 80 percent users said that the street view is a cool feature of the system.




- Looking at the responses to the a

## 7. RESULT

The results of out application were promising. As it is visible from the evaluation reports. Most of the users were satisfied with the functionalities of the project. The main issue that users said hindered the experience was lack of interactive UI. Some users were not able to find tweets but it might be the case that they were looking for a really obscure place and there might not be any tweets present about it.
Considering the feedback, it was a success although there is a lot of work is needed in brushing up the project.

## 8. CONCLUSION

A formal expectation of the project was that there are places reviews on the social media and work needs to be done to extract such reviews as this can help consumers as well as businesses. While the limitations of the APIs that we used did not let our application perform as good as it could have been, but it was able to fulfill the above criteria and was able to find reviews of places.

Also the tweets that were fetched from the Twitter API were the most recent ones and thus when these tweets are analysed, we get the most recent review and result of a place which is unprecedented even in websites specifically built for reviews. This was a big achievement of the app.lication

## 9. CHALLENGES

We faced challenges pertaining to the semantic complexities of the natural language used in tweets.

- Users did not use standard vocabulary in their tweets.
- Users referred to the same location using different names or different locations using the same name. For example, a user can refer to New York as NY or N.Y. or City of New York, etc. Or the user can say Washington for either the state or the city.
- Tweets sometimes mentioned the location but were not reviews. Tweets may not always contain location-specific reviews and thus such tweets will be considered noisy.
- Users use different languages to tweet since twitter allows the use of several languages.
- Due to API limitations, the number of reviews extracted from twitter was less and also it prevented from getting the exact location of popular searches.
- Also, there is a cap on the number of searches in the Google Places API for searching a particular place. This caused big problems while testing the application.
- Google Places API also has a cap on providing only top 5 most helpful reviews of a place. That is the reason that no place has more than 5 google places reviews in a single search.

## 10. FUTURE WORK.

- A mobile application for our application would be more convenient to use and can be developed.

- Advanced natural language processing that could specifically find the kind of object a word or name refers to.
- Reviews from other social media platforms can also be mined and aggregated in the application so as to make it a one-stop review engine
- Tweets were found of different languages and we were ignoring tweets with languages other than english because of a high overhead of translation. We can work on a sentiment analysis system of different languages.
- Finding tweets where people reference a place with a different or modified string. I.e. UPenn for University of Pennsylvania.

## 11. ACKNOWLEDGEMENTS

Our special thanks to our Prof. Timothy Menzies and Teaching Assistant, Amritanshu Agrawal for helping us in coming up with the idea and a concrete plan to successfully implement this project.

## 12. REFERENCES

[1] Chandra, Swarup, et al. "Estimating Twitter User Location Using Social Interactions--A Content Based Approach." 2011 IEEE Third Int'l Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third Int'l Conference on Social Computing, 2011.↩

[2] Abrol, Satyen, and Latifur Khan. "Tweethood: Agglomerative Clustering on Fuzzy k-Closest Friends with Variable Depth for Location Mining." 2010 IEEE Second International Conference on Social Computing, 2010. ↩

[3] Jacob Eisenstein , Brendan O'Connor , Noah A. Smith , Eric P. Xing, "A latent variable model for geographic lexical variation," Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Cambridge, Massachusetts, October 09-11, 2010, p.1277-1287 . ↩

[4] Sanjukta Saha and Dr. AK Santra, "Restaurant Rating Based on Textual Feedback", IEEE 2017.↩

[5] Yelp Inc, "Yelp Factsheet," September 2017. ↩

[6] Omega Management Corp, "Consumers increasingly turning to social media to share negative reviews" ↩