Location Prediction on Twitter

Rashmi Shivanna
329589544

Introduction

Social media services such as twitter and Facebook have gained popularity in providing abilities using which the users share their opinions. With the growing technology, social media services have come up with number of attractive features like check-in in Facebook, geotagging in twitter, etc. Using these features, users tag the place they are currently in and it has become a trend lately to have the place tagged in their posts. While these features are an attractive and fun way for the users to share their current locations to their followers/friends, geotagging has its own advantages. Using the locations tagged in, user's location can be tracked down. Even though this sounds like a violation of one's privacy, it helps with times when there is a need to know user's location like disaster response managers must know where to direct resources in order to effectively coordinate aid, and advertisers could benefit from tailoring advertisements to a user's location. Similarly, search results localization hinges on knowledge of a user's location.

Although many users use the tagging, not all the posts in social media services are geo tagged. So, in scenarios mentioned above, if we have a method to extract the location name by analyzing the text in the post, it will be a very good help to recover from such scenarios. This project is an attempt to achieve this. It uses data collected from social media service Twitter. In twitter, user posts are called tweets. Each tweet is of 140 characters or less. A tweet can also include #hashtags to indicate the topic of the tweet and @mentions to refer to another user. Despite the popularity of twitter and the large volume of tweets posted everyday, only 1% of the tweets are geo tagged, thus restricting the usability of many tweets for location-based services and studies. Due to this motivating factor, the geolocation prediction of tweets and Twitter users have garnered immense interest in recent years.

In this project, a geo location prediction algorithm for Twitter data, based on multinomial naive bases algorithm is proposed. The classifier is trained using text-based features that are picked up from the tweets. Effect of location indicative words and hashtags is analyzed and prediction is done on city level and country level and the results are compared. Project uses the data crawled from Twitter which has 180,909 tweets in it. This algorithm predicts the location just by analyzing the words used in the tweet. For e.g. if the tweet has text "Whatever the weather yoga always brings a ray of sunshine to the gloom. Here is @simi_melwani… https://t.co/W9TkQavmiY" the alogirhtm predicts the location as Plymouth. Not all the tweets are expected to be predicted correctly as it is hard to find such strong words which could help predict the right location. So, such inaccuracies will be discussed and few future works are proposed to improve the accuracy of the algorithm proposed.

Data Crawling

I crawled the twitter data for this project using the twitter API Tweepy, An easy-to-use Python library for accessing the Twitter API. Following packages of Tweepy were used for getting the live feed from twitter:

- **OAuthHandler:** This package is used for authentication of the API requests on twitter through OAuth. OAuth is an open standard for authentication adopted by twitter to provide access to the protected data. OAuth provides a safe method for authentication using a three way handshake. consumer_key, consumer_secret, access_token and access_secret keys were obtained for authenticating my connection to get the tweets.

- **Stream**: An instance of stream establishes a streaming session with twitter and routes messages to StreamListener. A stream object has a filter method using which we can get only tweets with specific attributes from twitter during the streaming process.

- **StreamListener**: Tweepy provides the capability to listen to the live feed and perform an action depending on the data received. StreamListener is a key component used for implementing this. We create an instance of this class to monitor the real time tweets and catch them. It has methods on_data, on_error and on_status which can be overwritten with required action. For e.g. in this project, I needed only the tweets with co-ordinates value not null. So whenever a tweet is received, I made a check on the tweet's coordinates attribute and if it is not None(null) then only I saved the tweet into a json file.

Dataset crawled has 180,909 tweets in it. Streaming feature of the API was used to crawl live tweets and the tweets were filtered on the geolocation co-ordinates with values between [-15.255119, -54.525961] for longitude and latitude respectively. These co-ordinates are for the countries in Europe region. Also, since the twitter geolocation prediction needs text processing, I filtered the tweets on the language and streamed only the tweets whose language was english. Tweets with all the attributes are saved in son format in the file "Tweets_Raw.json". For a better readability of the data and to have only the relevant features, I wrote the tweets from json file into a csv file named "tweets.csv". This csv file has only specific relevant attributes like tweetID, Tweet, Tweet_Url, Screename, TweetDate, TweetTime, Longitude, Latitude, TweetPlace, FT, RT, Lang, UserName, Bio, No. of followers, No. of friends, Lang mentioned in user's profile and Location mentioned in user profile. If the other attributes are needed, the code "Json_To_Csv.py" can be updated to write the needed attributes to csv and re-run to generate a new csv file.

For predicting the location, the object "place" in each tweet will be used which has an attribute called "name" which has the city which the user chooses while geotagging their tweet. This doesn't give the exact geographic location of the tweet. It is the place user selects when the tweet is written. To get the coordinates of this tagged place name, the attribute "coordinates" from object "geo" can be used. These coordinates are used for visualization of the tweets count on the world map.

Literature Survey:

There has been a lot of work done on twitter geolocation prediction. Probabilistic methods were adopted and different classification techniques were used to predict the geolocation. Even though there have been numerous papers, few of them were hand picked and following are such papers which closely relate to this project. Analysis and results obtained in these works will be used as reference for this project:

1. You Are Where You Tweet: A Content-Based Approach to Geo-locating Twitter Users: Using the open-source library twitter4j to access Twitter's open API from September 2009 to January 2010, base dataset of 1,074,375 user proles and 29,479,600 status updates was collected. Initial analysis was done based on on maximum likelihood estimation, the probabilistic distribution over cities for word w which can be formalized as pij(w) which identifies for each word w the likelihood that it was issued by a user located in city i. However, only 10.12% of the 5,119 users in the test set were geo-located within 100 miles to their real locations and the AvgErrDist was 1,773 miles, meaning that such a baseline content-based location estimator provides little value. So a different approach was used - Identifying Local Words in Tweets: Is there a subset of words which have a more compact geographical scope compared to other words in the dataset? And can these “local" words be discovered from the content of tweets? With local word alone, an accuracy of 0.498 was observed, which is almost five times as high as the accuracy got with the baseline approach that uses all words in the sampled Twitter dataset.

2. Geolocation Prediction in Twitter Using Location Indicative Words and Textual Features:

- A geolocation prediction algorithm for Twitter was presented based on a multinomial Naive Bayes classifier, using text-based features that are automatically learnt from tweets.
- The effects of using various feature sets including location indicative words, city/country names, #hashtags, @mentions and a combination of all of the above, was analyzed
- Twitter dataset comprised of 9.05 million tweets generated by 778K users and it was showed that this approach out-performs a state-of-the-art text-based classifier that solely uses location indicative words.

3. Text-Based Twitter User Geolocation Prediction

Advances the state-of-the-art of text-based geolocation prediction in a number of directions, and provides practical guidelines for the design of a text-based geolocation application. Following experiments were conducted in this research.

- A large-scale comparative evaluation of 12 feature selection methods for user geolocation
- Analysis of the impact of training on non-geotagged data
- A new set of experiments, and subsequent analysis, examining the influence of language
- Analysis of the utility of user-supplied metadata and ensemble learning
- More-detailed analysis of model generalization on temporal change including city-level meta-analysis.

Proposed Approach

The proposed approach for this project has 4 phases:
1. Data pre-processing
2. Features selection to select important words in the tweets
3. Training the multinomial Naive Bayes classifier using the features selected
4. Evaluation of prediction algorithm using the test data

Data Pre-Processing

Twitter data is highly noisy as people use many different kinds of information in it like urls, emoticons, special characters, etc. In order to extract words from tweets, we need to have pure english words in the tweets which could be meaningful. So, I followed the below processing steps on the tweets.

1. **Missing Values:** Since the main feature for this project is the text in the tweet and the tweet cannot be empty, we don't have any missing values. Also, this was verified programatically.
2. **Duplicate Values:** Since this a text dataset i.e. the feature values are text and also, since a tweet was streamed one at a time in a serial order, there were no duplicates found
3. **Data Cleaning:** Following steps were performed to clean the tweets before training the classifier:
    - Remove the urls present in the tweet
    - Tokenize the tweet
    - Remove the non-ascii characters since tweets contain emoticons and other special characters which are received from twitter as unicode
    - Convert the tweet to lower case
    - Remove stop words from the tweet
    - Remove the non-alpha characters

Location Indicative Words

Words that either explicitly (e.g., place names) or implicitly (e.g., dialectal words, slang or local references) encode geographical information are collectively referred to as “location indicative words” (LIWs); it is these words that we aim to automatically identify.

Examples of LIWs are
1. Local words that are used primarily in a single city, namely yinz (used in Pittsburgh to refer to the second-person plural pronoun), dippy (used in Pittsburgh to refer to a style of fried egg, or something that can be dipped in coffee) and hoagie (used primarily in Philadelphia, to refer to a kind of sandwich)
2. Semi-local words that refer to some feature of a relatively limited subset of cities, namely ferry (found, e.g., in Seattle, New York and Sydney), Chinatown (common in many of the largest cities in the US, Canada and Australia, but much less common in European and Asian cities), and tram (found, e.g., in Vienna, Melbourne and Prague). In addition to LIWs there are common

words (common) which aren't expected to have substantial regional frequency variation, namely twitter, iphone and today.

Features Selection

The most intuitive way to extract features from text is the bags of words representation:
1. Assign a fixed integer id to each word occurring in any document of the training set (for instance by building a dictionary from words to integer indices).
2. For each document #i, count the number of occurrences of each word w and store it in X[i, j] as the value of feature #j where j is the index of word w in the dictionary

Occurrence count has an issue: longer tweets will have higher average count values than shorter tweets, even though they might talk about the same topics. To avoid this potential discrepancies it suffices to divide the number of occurrences of each word in a tweet by the total number of words in the tweet: these new features are called tf for Term Frequencies. Another refinement on top of tf is to downscale weights for words that occur in many tweets and are therefore less informative than those that occur only in a smaller portion of the dataset. This downscaling is called tf–idf for "Term Frequency times Inverse Document Frequency".

Multinomial Naive Bayes

Multinomial Naïve Bayes algorithm implements the naive Bayes algorithm for multinomially distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). The distribution is parametrized by vectors $\Phi_y = (\Phi_{y1}, \Phi_{y2}, .., \Phi_{yn})$ for each class y, where n is the number of features (in text classification, the size of the vocabulary) and Φ_{yi} is the probability P(x_i|y) of feature i appearing in a sample belonging to class y

I trained multinomial naïve bayes classifier using the features selected using TF-IDF. I used country and city as 2 different labels and tried to analyze the accuracy with respect to both

If C is the set of all cities and T is the set of all tweets in the training set, each tweet t ∈ T with a city c ∈ C should be geotagged, such that the probability P(c|t) is maximized. Each tweet t gets a set of features f_i ∈ t (out of N total features) using tf-idf, where each feature f_i indicates the number of times (frequency count) that a feature word fi is used in a tweet t.

Thus, we have:

$$\arg\max_{c \in C} P(c|t) = \arg\max_{c \in C} P(c) \prod_{1<i<N} P(f_i|c)$$

Given that t_c is the set of all tweets that are posted in a specific city c and T is the set of all tweets, we can calculate the prior probability based on:

$$P(c) = \frac{|t_c|}{|T|}$$

Expected Results:

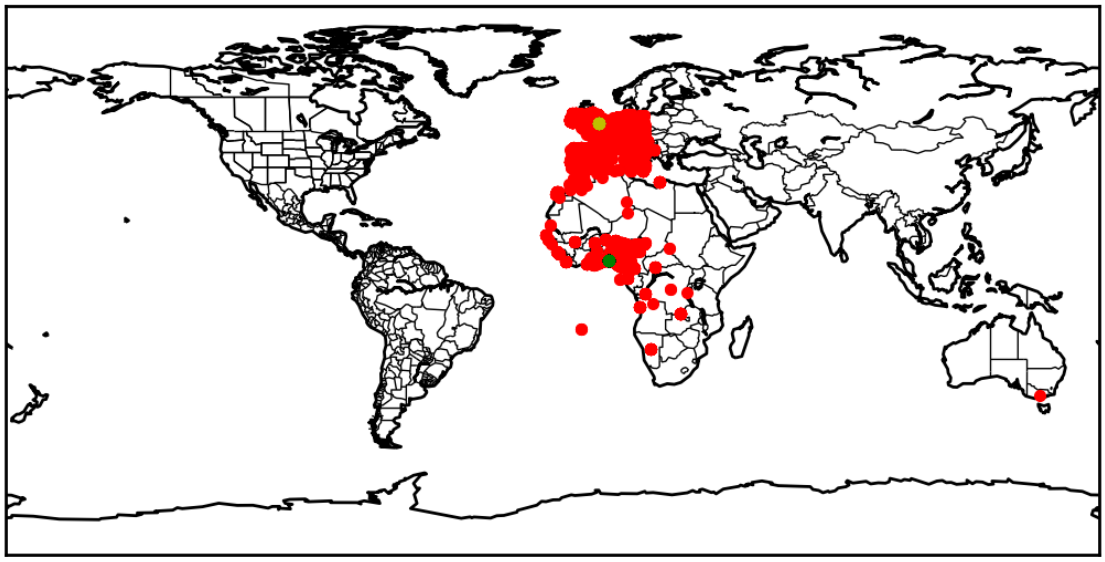

Fig 1: Test tweets' actual location

- The map above shows the distribution of tweets across the cities. These tweets are going to be used for testing. The tweets marked yellow are from London, green are from Lagos and red are from other cities. We expect the tweets to be classified to the right city or the closest possible with whom it shares the most similarity in the words present in the tweet.

- When we pass labels to the classifier, we have two options. Either we use the country name or the city name to which the tweet belongs. Using country as labels gives a better accuracy as the classifier will have less labels to classify the tweets into. This is because, there are only 196 countries to choose from. If we try to perform classification in city level, accuracy is expected to decrease as we have 7903 distinct cities to classify.

- If the tweets from the each city are equally distributed in number, there would be a good probability of predicting the correct location. However, as we can see from the map, the distribution is not uniform across the cities. So, the chances of picking words from the tweets which originate from the cities with most number of tweets will be more and the cities with less number of tweets will have a lesser probability of being classified correct

- Processing done on the tweet text affects the way features will be selected for training the model. Tweet being a very noisy data, the more we pre-process it, the lesser features we will be left with. For e.g., removing urls from the tweet text doesn't affect the results as the url shared in the tweet text is very rare to be a feature which would indicate a location. However, there will be words which are commonly used in a specific city and if we remove them, an important feature will be removed and affect the accuracy

Obtained Results

I experimented with the 4 aspects mentioned above and almost got what was expected. When I used the Country code in the tweet as the target variable an accuracy of 0.623 was obtained. Whereas, when city name was used as the target variable, an accuracy of 0.192549 was obtained. This was as expected as we have very few countries to choose from and more cities to choose from.

City-level prediction

Following map shows the distribution of tweets across the predicted cities. As we can see, only London and Lagos tweets were predicted right and the other tweets were not. When I looked at the predicted results, most of the tweets were either classified as London or Lagos. Hence, we see fewer red dots.

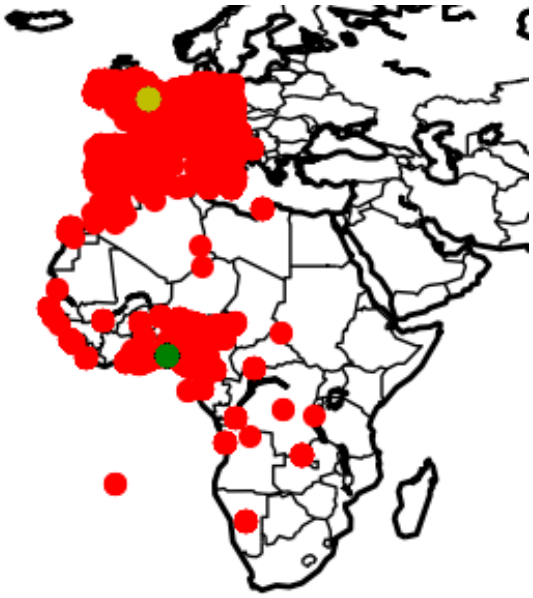

Fig 2: Test Tweets' actual location

Fig 3: Test tweets' predicted location

When I looked the distribution of tweet count, there were 32 cities with more than 1000 tweets whereas, the rest 7871 cities had very few tweets among which few cities had 1-2 tweets. So, as we expected, the probability of cities with lesser tweets have lesser chance to be predicted right. The table below(Table 1) shows the tweet count among the cities with London and Lagos having way more than the others and so, we see the red dots drastically reducing in the map with predicted cities on the right.

City	Tweet_Count
London	11450
Lagos	7422
Abuja	2981
Oxford	2707
Ikea	2456
Amsterdam	2417
Camden Town	2408
Paris	2357
South East	2256

Table 1: Tweet count in cities

Country-level prediction:

An accuracy of .623 was obtained when prediction was performed in country level. This is a lot higher than the accuracy obtained with city level prediction. Even though we noticed that the number of labels for classification is very less when countries are used compared to cities, there were few more attributes contributing to this increase in the accuracy. Out of 179286 tweets which were having the place name tagged, 86,730 tweets were from United Kingdom and 20,672 were from Nigeria. So, almost 60% of the tweets were from these 2 countries and the correctly predicted tweets were from these countries.

Hence, even though there were fewer labels when country was used, it looks like, the number of tweets from these two countries played a major role. So, to understand the behavior of classifier with respect to cities, as we have a better distributed data in city level, I tried few different steps while classifying using cities as labels and tried to understand the accuracy

Country	Tweet Count
United Kingdom	86730
Nigeria	20672
France	8225
Espa a	6052
Ireland	5955
Italia	5112
Germany	4444
Deutschland	4068
The Netherlands	3741

Table 2: Tweet count in countries

Removing words during data cleaning:

Initial data preprocessing involved removing urls, stop words and unicode characters from the tweet text. This gave an accuracy of 0.1925. I noticed that there were many tweets with temperature and speed mentioned. If we are concentrating on the words which give a good idea about which location they might belong to, these words might not be very useful. So, I added few more regular expressions to remove the words which had temperatures in Celsius, speed in mph, kmh, etc, and this gave an accuracy of 0.1744. It decreased accuracy.

Sampling:

Since the tweets from London and Lagos were relatively higher in number, I tried doing sampling and picked up following number of tweets from each cities

	Training	Testing
London	2000	1000
Lagos	1500	200

Table 3: Sampling without replacement

70% of the total tweets were used for training and the rest 30% for testing. This helped in improving the accuracy of prediction to 0.213. As discussed, giving each city a fair probability of having it's feature words being used in training, would give a better accuracy.

Fig 4: Actual location of test tweets

Fig 5: Predicted location of test tweets

As we can see in the map, most of the tweets marked red were predicted correct and the yellow and green ones from London and Lagos respectively are predicted correctly as before. This shows how the accuracy increased with sampling. The below 2 maps show the comparison of the map with sampled data and the data without sampling

Fig 6: Predicted location without sampling

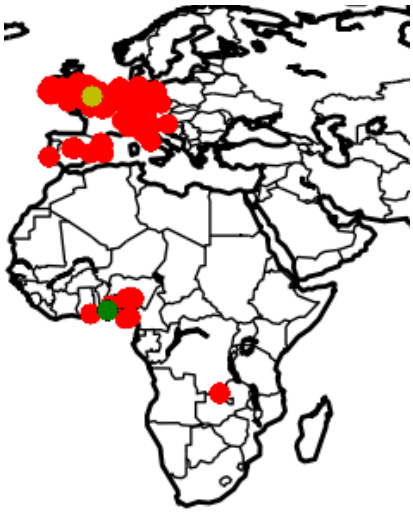

Fig 7: Predicted location after sampling

The right map has more red dots which indicates that the tweets from locations other than London and Lagos were predicted more accurately compared to the prediction done without sampling. London and Lagos prediction remains the same.

Hashtags:

Instead of using the whole text present in tweet was features, I tried to use hashtags as features as the hashtags are used by the users widely and there is a good chance that the location could be related to the hashtags. However, the accuracy dropped to 16.4% when hashtags were used as features. This is way too less compared to what we got by using all the words in the tweet.

Clustering:

I performed clustering of the words manually by counting each unique word from all the tweets followed by creating a vector. This word vector was passed to a kmeans algorithm to predict the words with same frequency count as single cluster. The idea was to use the cluster labels as the target in BNB algorithm and classify tweets into clusters. Then we could use a mean distance error to estimate the average distance between the cities predicted to be together in same cluster. I started with 10 clusters and use the cluster labels to classify the tweets. The classification accuracy was 0.98. Due to time constraint, this method wasn't investigated more.

Conclusion

Since the data which I used didn't have a good distribution of tweets across countries, a good prediction couldn't been done in country level. However, a fair prediction was made in city level as it had a better distribution of tweets

Choosing location indicative words has a very high impact on the accuracy of prediction. As we saw, removing the temperature, speed, etc. from tweets reduced the accuracy. So it seems like these words are adding value to the prediction. But these words cannot indicate anything specific

to a location unless the tweets were collected at the same time as it would mean that people were talking about weather of same location or talking about a sport event happening at that moment. But this cannot be generalized and such words cannot be used while selecting the features even though it helped in improving the accuracy in this case.

Using only hashtags instead of complete text from the tweet reduced the accuracy. Even though hashtags are useful in getting an idea of the location, it didn't help in prediction compared to the complete tweet text.

The amount of tweets from a location available in the dataset affects the prediction accuracy. As noticed, the country level prediction a very high accuracy but it turned out to be very obvious since the tweets were mostly from just 2 specific countries. So, in order to get a good prediction model, choosing a right proportion of tweets from each location in the dataset plays a very important role in getting an accurate prediction model

It is hard to get tweets with words which are location indicative. As people could be sitting at a location tweeting about and tagging a different place. So, getting accurate location using just the tweet text is complicated. however, as seen in many research papers, a better accuracy could be obtained by having more number of tweets and different feature selection methods.

Future

Since it is hard to get the location indicative words, Instead of predicting exact location of the tweet, an approach for predicting a boundary within which a tweet could be present, can be tried

Try to use other feature selection methods like chi square, Information gain ration/BNS and try to increase the accuracy of prediction when city is used as the target variable.

All the research papers had at least 9 billion tweets used and the accuracies obtained were pretty higher. So, crawl larger number of tweets to get a better accuracy. As more the number of tweets, a better quality of location prediction is expected.

There was no method used to handle the words which are not present in training tweets but in test tweets. So, techniques like Laplace smoothing can be used to cater such features. This will also help in increasing the accuracy of the prediction.

Sampling the tweets from each location helped in improving the accuracy of the prediction. Another technique to try would be to perform random sampling and see if the accuracy persists. This way, we could guarantee the accuracy obtained by the classifier used

References

- Zhiyuan Cheng, James Caverlee, and Kyumin Lee. 2010. You are where you tweet: a content-based approach to geo-−locating twitter users. CIKM '10. ACM, New York, NY, USA, 759-768. http://doi.acm.org/10.1145/1871437.1871535

- Lars Backstrom, Eric Sun, and Cameron Marlow. 2010. Find me if you can: improving geographical prediction with social and spatial proximity. WWW '10. ACM, New York, NY, USA, 61-70. http://doi.acm.org/10.1145/1772690.1772698

- Bo Han, Paul Cook, Timothy Baldwin. 2014 Text-based twitter user geolocation prediction https://dl.acm.org/citation.cfm?id=2655726

- Lianhua Chi†, Kwan Hui Lim‡†, Nebula Alam† and Christopher J. Butler† Geolocation Prediction in Twitter Using Location Indicative Words and Textual Features https://noisy-text.github.io/2016/pdf/WNUT30.pdf

- http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

- http://scikit-learn.org/stable/modules/naive_bayes.html