Heuristic and Learning based approaches for Twitter Geolocation
Notebook for Geolocation Prediction in Twitter – SS 2020

Vishal Khanna
Mat. 120333, CS4DM

Sneha Mohanty
Mat. 120799, CS4DM

Sagar Nagaraj Simha
Mat. 120797, CS4DM

vishal.khanna@uni-weimar.de
sneha.mohanty@uni-weimar.de
sagar.nagaraj.simha@uni-weimar.de

Geographic location information can add great value to users' data on online social platforms and can be used to address a variety of high impact problems such as regional sentiment analysis and local event detection. In this report, we discuss a clustering and text classification based solution for the shared task on Twitter Geolocation Prediction and add novel enhancements to improve the existing model's performance for (i) coordinates estimation and (ii) city-name prediction. We evaluate our proposed approaches across various feature spaces such as tf-idf and word vectors, classification models such as Logistic Regression, Naive Bayes and Support Vector Classifier coupled with parameter tuning to achieve a classification accuracy of 10.5% and median and mean error distances of 2540 and 5022 respectively. Our results successfully beat the baseline and are comparable to several official submissions for the shared task.

1 Introduction

In the age of web and social media, internet's global connectivity and penetration is the highest it has ever been. This has lead to a plethora of user data being recorded daily and a sizeable chunk of it being made available in the public domain. As stated by Han et al. 2016, such user data is a goldmine of valuable insights into their behaviour and preferences and can help analyze various aspects of the society at large. Some hot topics of research involving web-based social data analysis include 'detecting and mitigating bias in social systems', 'analysis and detection of hate speech and misinformation online' and 'understanding the trajectory of a trending topic's popularity on social media'.

In this report, we propose simple enhancements to an existing approach for the shared task on Twitter Geolocation Prediction(Han et al., 2016) and evaluate each of them in several experimental conditions. The shared task addresses the problem of inferring users' tweets' geographic location(in terms of their coordinates and city names) based on their textual content and meta data of the tweet.

Geographical information brings great value to users' data and can be leveraged for tasks such as disaster management, localized event detection and demographic analysis. However, often times such geographic information is missing or unreliable, which calls for effective geolocation approaches.

2 Related Work

Han et al. 2012 device a text based geolocation technique based on extracting Location Indicative Words(LIWs) through feature selection. They experiment with LIWs extraction using Inverse City Frequency, Information Gain Ratio(IGR) and Maximum Entropy as metrics for feature selection and show that for a Multinomial Naive Bayes classifier, the LIWs based feature space outperforms a traditional unigram based one with the best results achieved using IGR. Rahimi et al. 2016 propose (i) a text based classification approach based on clustering the data points and learning them over word frequencies and (ii) a network based regression approach which models a user's '@-mentions' on Twitter to estimate her location. Miura et al. 2016 employ feature engineering approaches based on word embeddings extracted from Twitter metadata such as timezone and user profile description and achieve one of the best results for the task.

We chose to not consider approaches such as Huang et al. 2019 involving deep learning architectures and attention based mechanisms despite their high effectiveness for this task due to our limited prior background in these areas and to narrow down our exploration towards approaches more in line with our learning goals for this course.

Application of geolocation towards addressing a diverse set of problems has been extensively covered in the literature. Dredze et al. 2013 design a geolocation system for tweets named Carmen and use it to improve tracking of contagious disease outbreaks. Carmen leverages the user's profile information and twitter's geotagging feature for tweets to infer location information of varying granularity(country, city etc.). However, their methodology does not leverage the statistical properties of the tweets' text for the task - something we claim could have further improved the system's accuracy and rendered it robust against inconsistent and missing data.

3 Data

3.1 Sourcing the data

The training and test datasets have been provided alongside the task's specifications. The training data consists of 12827165 tweets from 1 million users. The test dataset consists of additional 10000 tweets separate from the training set. Both datasets have been sourced from Twitter's streaming API. Due to Twitter's terms of service, the contents of the tweets are not included in the training dataset and have to be downloaded separately using the tweet ids provided. The test dataset contains the text content for the tweets but the tweet id and user id fields have been hashed.

To download the tweets' text content, the organizers of the task have provided a python based downloader script. However, we faced a few challenges in successfully using it for our work. One, the script was written in Python 2.7 which has since become obsolete and running it in Python 3.4 causes minor runtime and compiletime errors. Second, the Twitter API restricts the number of requests to 900 per account within a 15 minute window; for a free developer account(Twitter). Exceeding this limit causes a $rate\text{-}limit\text{-}exceeded$ error which the script has no handling mechanism for.

Because of the above mentioned issues, we decided to write a separate script to download the required data using a Python wrapper for Twitter's API(Tweepy). The latest version of the library also includes support to handle the $rate\text{-}limit\text{-}exceeded$ error with the $wait\_on\_rate\_limit$ parameter. Downloading tweets' content was not possible for a sizeable portion of the tweet ids in the training set due to either the tweet or its associated user's account having been deleted/banned or the user having restricted their tweets to only their followers. In the end, we were left with 7582780 and 10000 entries in the training and test sets respectively.

3.2 Descriptive statistics for the training dataset

The final training data comprises of 7582780 tweets from 632474 users in 38 languages. The most common languages are English, Indonesian and Portuguese whereas the least common ones are Georgian, Greek and Simplified-Chinese. The second most prevalent language tag with 472248 tweets is 'und'(undefined). The tweets are geographically distributed across 3262 cities, 930 districts/countys and 174 countries with the most number of tweets from Jakarta, London and Los Angeles and the least from 110 cities having 1 tweet each. As illustrated in figure 1, the distribution of cities in the training dataset is heavily skewed with 1000 and 100 of the most frequent cities comprising of 94 and 40 percent of the total tweets respectively.

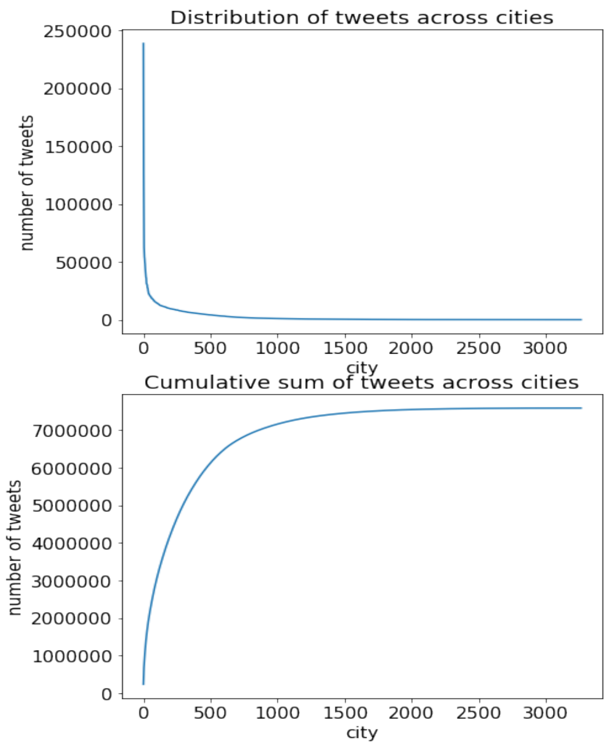

Figure 1: (a) Number of tweets distributed across cities (b) Cumulative sum of tweets across cities

The distribution of the number of tweets across users approximates a normal distribution significantly skewed towards the right of the mean. The largest fraction of users have 10 tweets in the training set and most users have 10 or more tweets. 97

and 3 percent of the tweets had the geolocation setting enabled and disabled by their users respectively. 99.5 and 0.5 percent of the tweets belonged to users with unverified and verified accounts respectively.

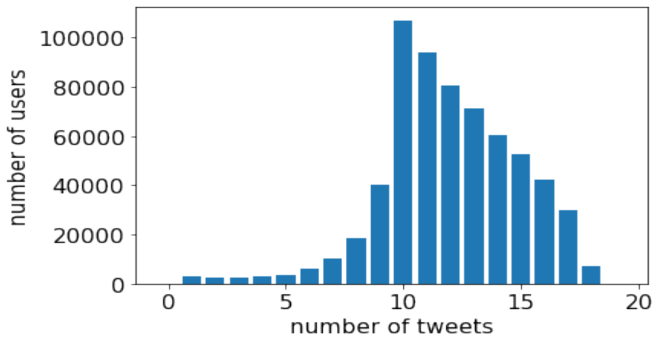

Figure 2: Number of tweets vs number of users having that many tweets in the training set

4 Method

The shared task on geolocation prediction, as specified by Han et al. 2016, can be divided into two related sub-problems:

1. Regression problem of estimating the coordinates for a tweet(latitude and longitude), given its textual content
2. Multi-class classification problem of predicting a tweet's city from its textual content

Moreover, each of the two sub-problems are to be addressed in two different settings:

1. Tweet level geolocation
2. User level geolocation

We narrow down the scope of our experiments to only the tweet level setting for the purpose of this project.

The evaluation criteria laid out by the organizers includes three metrics:

1. Classification accuracy for city name prediction
2. Median error distance between predicted and true location w.r.t the position coordinates
3. Mean error distance between predicted and true location w.r.t the position coordinates

Our solution for the geolocation task builds on the model proposed by Rahimi et al. 2016 which formed the basis of their python based geotagging tool named pigeo. We extend their approach with more textual features and various commonly used classification models, compare their effectiveness for the each of the two subproblems of the Tweet level geolocation task separately and perform prelimnary parameter tuning to achieve the best possible results across the three evaluation metrics.

4.1 Estimating latitude and longitude for tweets

We convert the regression problem of coordinates prediction into a classification problem through domain discretization. This is achieved by representing the data points in a 2 dimensional space with longitude as X-axis(range -180 to 180) and latitude as Y-axis(range -90 to 90). Plotting the data points on a scatter plot gives a rough visual illustration of their geographical distribution as shown in figure 4.

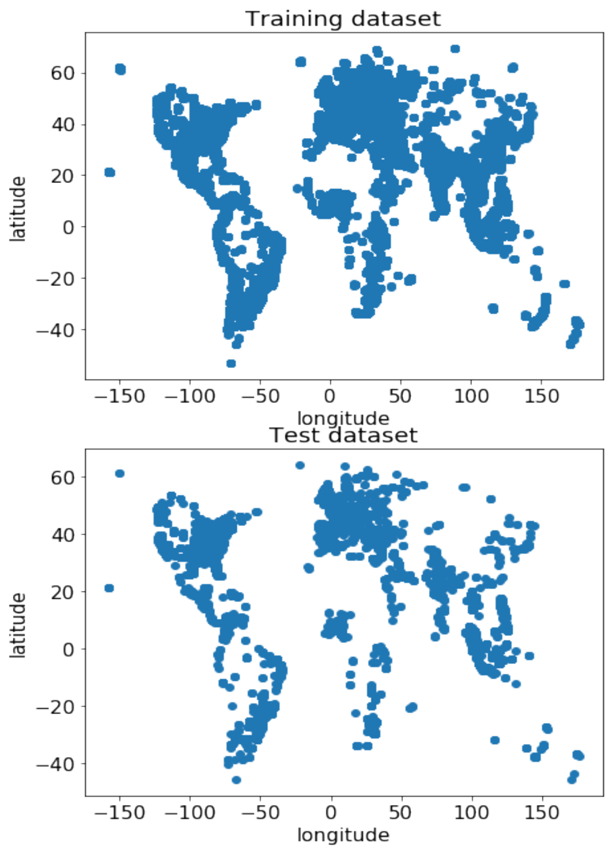

Figure 3: Scatter plot of (a) training and (b) test datasets in a 2-dimensional coordinate space

We group together geographically proximate points into clusters and assign them arbitrary class labels, which serve as output labels for classification. We employ the k-Means clustering algorithm and experiment with different values for the number of clusters parameter. Doing this, we effectively transition to a classification problem of predicting the cluster for each data point from its corresponding tweet.

Having obtained the output labels for our classification task, we now look at useful feature models, which can be derived from the tweets' textual content, to serve as input to our model. As an initial step, we preprocess the tweets by performing the following steps:

1. Removing URLs and special characters using regular expressions

2. Tokenizing and removing 'uninformative' tweets containing less than 3 tokens from the training dataset

Once the tweets are preprocessed, we consider two feature models for the classification task:

1. **Bag-of-words with tf-idf** - We consider each unique token in the training dataset's tweets as a feature and for each data point, the token's tf-idf as its value. The tf-idf for token W with absolute frequency $f(W,T)$ in tweet T of dataset D is the product of its tf and idf values where:

$$tf(W,T) = \frac{f(W,T)}{|T|}$$
$$idf(W,D) = \log \frac{|D|}{|\{t{:}t \in D \wedge W \in t|\}}$$

   The tf-idf value represents the relative frequency of a token in a tweet normalized by the inverse of the tweet-frequency for the token in the dataset. With this feature space, the learning algorithm attempts to model the correlation between the distribution of tokens in a tweet and its coarse-grained geolocation class obtain by clustering the data points. Due to the large number of features in the BoWs model, we had to decrease the training set size to 100000 entries for our experiments because of memory and computing resources limitations

2. **Pre-trained word embeddings** - Word embeddings create a mapping between a token and its representative vector of real numbers in a multi-dimensional semantic space(Wikipedia).

   For our experiments, we use fastText's pre-trained 300 dimensional word vectors for one million English language words, trained on Wikipedia and two additional corpora. We use the word embeddings model to obtain tweet-level embeddings by computing the average for all tokens in a tweet weighted by their respective tf-idf values. We, thereby obtain a 300 dimensional semantic feature space to train our classifiers on.

For each of the feature models, the multi-class classification task involves assigning a cluster to each tweet in the test set. Then, we estimate the tweet's coordinates to be that of its cluster's mean/centroid. Hence, all tweets belonging to a cluster will share the centroid's coordinates. We try three clustering settings with 10, 100 and 500 clusters. Furthermore, we experiment with three standard models for the classification problem - Logistic Regression, Naive Bayes and Support Vector Classifier and report the outcomes in the results section.

4.2 Predicting city names for tweets

Our approach for city prediction leverages the estimated coordinates and their respective clusters, obtained in the previous step, as features and utilizes simple heuristics and additional classification models. In an alternative experimental setting, we also consider the tf-idf feature space as we did in subsection **4.1**. We try the following methods:

1. **Cluster-mode** - Consider the clusters obtained during coordinates estimation and take the most frequently occurring city across all training points in that cluster to be the city value for each test set point labelled with that cluster during the classification in subsection **4.1**.

2. With the coordinates as features, perform a multi-class classification. For this approach, we experiment with Logistic Regression, Naive Bayes, Support Vector Classifier and k-Nearest Neighbours with number of neighbours as 3, 10, 50 and 100.

3. With the tf-idf values of tokens in the training dataset's tweets as features, perform a multi-class classification. Like with cluster prediction, here too we experiment with Logistic Regression, Naive Bayes and Support Vector Classifier.

5 Results

We use the evaluation script provided on the task's website and discuss the results obtained for each subproblem below.

5.1 Coordinates estimation

For the subproblem of coordinates estimation, the task defines mean and median error distance as the evaluation criteria. The results for each experimental setting are listed in table 1.

The lowest value for both median and mean error distance is observed upon clustering with 10 clusters and using the Logistic Regression classifier trained on word vector based features. For the three classification models, the best overall performance is observed for Logistic Regression followed by Gaussian Naive Bayes and Support Vector Classifier. Use of word embeddings based features reduces the error values for Logistic Regression but increases the same for the Naive Bayes classifier. Due to already poor performance of the Support Vector Classifier, we chose to not further experiment with it for this subproblem. Changing the number of clusters parameter influences each of the classification models' performance differently. For Logistic Regression, the performance worsens considerably on increasing the number of clusters. The Naive Bayes Classifier's performance exhibits different trends on increasing the number of clusters for the two feature spaces. For Support Vector Classifier, the error increases on increasing the number of clusters from 10 to 100 and slightly decreases on further increasing the number of clusters to 500.

The median error distance value observed is better than several submissions on the task's official website where the worst and best reported values are 5848.35 and 16.13 respectively. The mean error distance, on the other hand, is only moderately better than the worst reported value of 6175.27. For this subproblem, our results are better than five of the official submissions. Considering that we were able to run our approaches on only a fraction of the training dataset, we regard our results as satisfactory.

5.2 City name prediction

Our results for the subproblem of city name prediction were a lot less promising and are listed in table 2.

model	**num clusters**	**features**	**median error distance**	**mean error distance**
LR	10	*tf-idf*	2556.50	5123.39
LR	100	*tf-idf*	4510.56	5665.62
LR	500	*tf-idf*	6211.03	6856.52
LR	**10**	**word-vec**	**2540.31**	**5022.17**
LR	100	word-vec	3882.87	5414.24
LR	500	word-vec	5176.85	6088.25
GNB	10	*tf-idf*	5621.24	6527.85
GNB	100	*tf-idf*	3808.87	5545.30
GNB	500	*tf-idf*	3939.71	5547.64
GNB	10	word-vec	8274.67	8311.35
GNB	100	word-vec	7776.05	7405.35
GNB	500	word-vec	6529.79	6716.52
SVC	10	*tf-idf*	7142.75	7780.72
SVC	100	*tf-idf*	11765.07	10353.57
SVC	500	*tf-idf*	11730.65	10338.68

Table 1: Results for coordinates estimation. LR, GNB and SVC stand for Logistic Regression, Gaussian Naive Bayes and Support Vector Classifier respectively.

classifier model	**feature model**	**class accuracy**
cluster-mode	coordinates	0.0719
LR	coordinates	0.0359
GNB	coordinates	0.0013
SVC	coordinates	0.0012
LR	*tf-idf*	**0.1052**
GNB	*tf-idf*	0.0925
SVC	*tf-idf*	0.0342

Table 2: Results for city name prediction. Cluster-mode refers to the first approach discussed in section **4.2**.

The best classification accuracy of 7.1% for cluster-mode is observed when using the coordinates values obtained through Logistic Regression classification on word vector based features with 10 clusters. The results obtained across all experimental settings involving the k-Nearest Neighbour model were poor and hence are not included in the table above. The best overall accuracy of 10.5% is observed using the Logistic Regression classifier trained on the tf-idf feature space.

The best classification accuracy obtained is better than that of six results reported on the official website.

We believe the large number of classes and highly skewed distribution of data points across them(as illustrated in figure 1) to be a major factor in the poor classification accuracy observed. Overall, the estimated coordinates values obtained in the previous subsection appear to be an unreliable feature space for city name prediction.

6 Conclusion

6.1 Contribution

In this report, we implemented a clustering and classification based approach to geolocation prediction, as proposed by Rahimi et al. 2016 for their tool pigeo, for the Twitter Geolocation Prediction Shared Task. We further extended their technique to an alternate word embeddings based feature space. We also proposed simple heuristic based and traditional classification based approaches for city prediction. We experimented with the proposed techniques in various settings involving different classification models and clustering parameter values.

6.2 Knowledge Gained

During the course of this miniproject, we took a sizeable leap from skimming through research papers to critically reviewing them, implementing the proposed ideas by ourselves and coming up with simple novel modifications to them. We feel much more comfortable with our proficiency in handling decently large datasets and running simple experiments on remote servers. Our knowledge of elementary data analysis and machine learning concepts has improved with this project, especially with python libraries like scikit-learn, pandas and numpy. We gained significant insights into the pipeline of a standard NLP project from data cleaning, analysis, feature engineering to building models and evaluating them. Through this project we were able to reinforce our concepts from the course and gain better understanding.

6.3 Future Work

In an alternate experimental setting based on Miura et al. 2016, we experiment with a pipeline involving stemming, tf-idf vectorization and Stochastic Gradient Descent Classification. Our initial experiments over the entire training data (with 100 clusters) yielded an accuracy of 12.04%, and 1271 and 4004 median and mean error distances respectively. Running this pipeline with higher clusters and additional features such as word embeddings may provide higher classification accuracy.

A combination of oversampling and undersampling methods such as SMOTE could help address the skewed distribution across city labels and the resulting biased classification. This would result in a much larger feature space due to the addition of synthetic samples. Methods such as Incremental/Online-Learning could handle such large data and yield better results. Additional approaches leveraging the extensive metadata that can be sourced from the Twitter API could potentially provide considerable performance improvements.

References

M. Dredze, M.J. Paul, S. Bergsma, and H. Tran. 2013. Carmen: A twitter geolocation system with applications to public health. *AAAI Workshop - Technical Report*, pages 20–24.

fastText. fasttext. https://fasttext.cc/.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Geolocation prediction in social media data by finding location indicative words. pages 1045–1062.

Bo Han, Afshin Rahimi, Leon Derczynski, and Timothy Baldwin. 2016. Twitter geolocation prediction shared task of the 2016 workshop on noisy user-generated text. pages 213–217.

Chieh-Yang Huang, Hanghang Tong, Jingrui He, and Ross Maciejewski. 2019. Location prediction for tweets. *Frontiers in Big Data*, 2:5.

Incremental/Online-Learning. Incremental/online learning. https://scikit-learn.org/0.15/modules/scaling_strategies.html.

Yasuhide Miura, Motoki Taniguchi, Tomoki Taniguchi, and Tomoko Ohkuma. 2016. A simple scalable neural networks based model for geolocation prediction in twitter. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 235–239, Osaka, Japan. The COLING 2016 Organizing Committee.

Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2016. pigeo: A python geotagging tool. pages 127–132.

Tweepy. Tweepy's website. https://www.tweepy.org/.

Twitter. Api rate limits. https://developer.twitter.com/en/docs/twitter-api/rate-limits.

Wikipedia. Word embedding. https://en.wikipedia.org/wiki/Word_embedding.