COMP30027 Project2 Report

Anonymous

1 Introduction

Nowadays, social media becomes one of the most popular approaches for people to share and acquire information. Tons of messages are posted online with potential hidden information related to location, events around and so on. Automatic geolocation is one of the common analytic methods that predicts possible posted location of texts and is particularly useful in fields like disaster detecting [1] and stock market monitoring [2].

In this paper, we compare several different feature engineering strategies and classifiers on the effectiveness of text-based tweets geolocation prediction.

2 Related Work

Text Approach uses text information to predict location. Some of them combining traditional machine learning classifiers with feature extraction strategies, for instance, extracting local entities [4] from text using clustering algorithms like DBSCAN, selecting words from probabilities prospective [5], or using word embedding [6, 7, 8, 9]. Other common classifiers include multilayer model [3] and Neural Network based model [10, 11].

3 Text Data

The text data was directly collected from Twitter using the Twitter API, used only for research purpose. We chose the same number of tweets from 4 cities in Australia (Melbourne, Sydney, Perth and Brisbane). The text dataset consists of 248,828 tweets and is being randomly divided into train (103,364), development (37,316) and test (108,148) set, where only train and development sets have correct location label that could be used for evaluation.

4 Experiment Approach

4.1 Data Preprocess

The main purpose of preprocess was to filter out obvious infrequent/geolocation-irrelevant words in text, such as URL, emojis, special characters (like @ and #), numbers, punctuation symbols and stop words. The rest words were tokenized and transformed to lowercase, and all being preserved irrespective their length since Roller [12] mentioned short words contribution on geolocation prediction, more exploration about word length was done in feature engineering.

We also tried other preprocess approaches like word stemming and check spell, but eventually didnt use them since the problem of over/under stemming and time consuming.

After preprocessing, the size of unique words of train, development and test datasets are reduced to 72819, 40808 and 74319 separately.

4.2 Feature Engineering

Although preprocess eliminated some unhelpful words, the datasets still contained lots of noises words, like nan, like. Also, there were plenty of words frequently appear in development data but not in the training due to the characteristic of tweets, that people are likely to misspell and concatenate words together, like dandedong (188 times), perthfest (38 times). Therefore, we implemented and tested few feature engineering strategies to better address these problems.

The first one was provided from COMP30027, top words related to each city based on the Mutual Information and Chi-Square were selected.

Like MI, Konstantinos [5] used probability base approach Word Locality Heuristic (WLH) to determine words that are more useful in prediction, we also use this as our second strategy. After WLH or MI, noise words could be filtered out and unseen words were simply ignored.

Lastly, we combined TF-IDF text feature extraction with other feature engineering methods. To eliminate noises words, we simply assumed the train dataset is general enough to obtain the probability of a class given a word ($P(c|w)$), words with maximum $P(c|w)$ higher than threshold t were kept, also words with length smaller than l or total count is less than c were eliminated. As for unseen word, one was replacing it with most similar words in training set calculated by Word2Vec. Another one was breaking all words down into substring of length s, since most unseen words could be found in training set if we just slightly alter/insert/delete few characters or concatenate words together. Finally, we consider all tweets from the same city as one document and merge all tweets together to better use TF-IDF.

4.3 Classifier

We considered majority-class and stratified Dummy classifier as baseline. According to previous work, we ran several comparative experiment on classifiers that are easy to interpret, such as linear Support Vector Machine [5], Bernoulli Nave Bayes, Multinomial Nave Bayes [3, 5, 8] and Logistic Regression [5], and the combination (voting and stacking ensemble) of them, where voting ensemble integrate MNB, SVM and Logistic Regression and use hard voting, stacking ensemble put MNB and BNB as lower layer and use Logistic Regression as higher layer.

5 Experiment Results

5.1 Evaluation Metrics

In order to measure the effectiveness of different approaches, we used execution time, accuracy (ACC), macro-averaging precision, recall and fscore since the classes were distributed equally, also included predict label distribution, cities with most correct/incorrect predictions, and the most common misclassify error, which are all obtained from confusion matrix.

5.2 Results

Table 1 and 2 summarize part of the results, the highest value of each column is represented with bold. Although we try various combination of hyper-parameters of features engineering, due to page limitation we only list those representative enough. The best classifier is determined by their ACC and Time together, that is the classifier the has the highest accuracy with as low execution time as possible.

5.3 Interpretation and Error Analysis

As presented in table 5.1 and 5.2, MNB and Voting Ensemble classifier performed more accurately that increase accuracy to around 35.5%, so the model interpretation and error analysis were mainly focus on these three classifiers.

Probability-based model like MNB, BNB performed better when unrelated words were filtered out, since the appearances of irrelevant words largely decreased the posteriors probabilities and resulted in a low prediction score. After leaving out these words, NB predicted an instance simply based on how many informative words showed up and how pervasive were these words. For instance, 84.19% of melbourne were from Melbourne and it was a strong indicator of Melbourne. Also, the smoothing being chosen here was Laplace smoothing with alpha 0.01, means that we considered new values less important than the existing values since most of them are just interferences, like pleaseee.

As for classifiers with parameter assigned to each attributes, like SVM, Logistic Regression, the impact of unrelated words was being reduced by lower parameter, so that even without filtering, they could still perform automatic feature selection and produce nearly equal distributed results combing with MNB. LinearSVM created hyper plane dividing informative words (attributes with high-value parameter) into groups and didnt care about the distribution of irrelevant words (words with low-value parameter), texts with informative words were more likely to fall into same side of hyper plane and thus being classified as same label. Here we adopted penalty C as 0.95 to ignore special instances, like Melbourne tweets contains sydney, and generate more generalized model. Similarly, logistic regression would assign high-value parameter to label-correlated words and the appearance of these words will produce more confident prediction score that closer to 1 (if 1 is the label associated). We used lbfgs solver with l2 penalty that was suitable for multi-class problems and handling multinomial loss, also changed the inverse of regularization strength to 0.95 to allow few mistakes.

However, irrespective the feature engineering methods, the main error that all classifiers encountered was the prediction distribution was highly bias towards one label due to the lack of information, as some tweets were just lyric or advertisement that were inherently unpre-

Classifier	Best Classifier	ACC	Precision	Recall	Fscore	Time
Baseline	Dummy(maj)	0.25	0.0625	0.25	0.1	0.126616
Baseline	Dummy(strat)	0.250509	0.252651	0.252653	0.252651	0.374720
MI (top10)	MNB	0.294914	**0.662155**	0.294914	0.194591	0.754906
MI (top50)	MNB	0.301345	0.533528	0.301345	0.216524	1.011281
MI (top100)	MNB	0.307830	0.491527	0.307830	0.236774	1.516725
WLH (top 10%)	MNB	0.258575	0.571608	0.258575	0.119207	0.245079
WLH (top 30%)	Voting Ensemble	0.299710	0.652912	0.299710	0.202381	20.24341
WLH (top 50%)	MNB	0.329108	0.573644	0.329108	0.263804	0.217443
TF-IDF 1	Stack	0.343873	0.346076	0.343873	0.342224	2.298197
TF-IDF 2	MNB	0.339452	0.384120	0.339452	0.317352	0.111144
TF-IDF 3	Voting Ensemble	0.353012	0.354618	0.353012	**0.352226**	58.89453
TF-IDF 4	MNB	**0.356120**	0.371116	**0.356120**	0.350733	0.153969

1 TF-IDF t=0 + l=0 + c=0

2 TF-IDF t=0.5 + s=6 + l=3 + c=10 + merge tweets from same city + replace unseen words

3 TF-IDF 3: t=0 + s=6 + l=0 + c=0

4 TF-IDF 4: t=0.5 + s=6 + l=3 + c=0 + merge tweets from same city

Table 1: accuracy related evaluation

Classifier	Best Classifier	Distribution	Most correct	Least correct	Common error
Dummy(majority)	Dummy(maj)	100% Bris	Bris	Other	Other Bris
Dummy(stratified)	Dummy(strat)	25.55% Perth	Melb	Syd	Bris-Perth
MI (top10)	MNB	93.34% Bris	Bris	Syd	Syd-Bris
MI (top50)	MNB	89.49% Bris	Bris	Syd	Syd-Bris
MI (top100)	MNB	98.55% Bris	Bris	Perth	Melb-Bris
WLH (top 10%)	MNB	85.56% Bris	Bris	Syd	Syd-Bris
WLH (top 30%)	Voting Ensem	93.22% Syd	Syd	Melb	Melb-Syd
WLH (top 50%)	MNB	86.04% Bris	Bris	Syd	Syd-Bris
TF-IDF 1	Stack	32.99% Syd	Syd	Bris	Bris-Syd
TF-IDF 2	MNB	60.45% Bris	Bris	Syd	Syd-Bris
TF-IDF 3	Voting Ensem	28.55% Bris	Bris	Perth	Syd-Bris
TF-IDF 4	MNB	43.32% Bris	Bris	Perth	Perth-Bris

Table 2: prediction distribution related evaluation

dictable. 15 20% instances were empty after filtering (even more when using MI or WLH), and some others contained only unseen words, therefore, classifiers couldnt do anything more than guessing majority. And if we didnt eliminate those words, common words with different bias towards specific city, like nan prefer Perth in train but Brisbane in development, good prefer Brisbane in train but Melbourne in development, would also lead to higher change of misclassifying.

Another unavoidable error was the wrong indicator, like melbourne in Sydney tweets, NB would always predict towards Melbourne based on the posterior probabilities it learned, LinearSVM and Logistic Regression would also give Melbourne higher output due to high-value associated parameter, unless there were more other strong indicators of other cities in this tweet.

6 Conclusions

In this paper, we examined the performance of several feature engineering methods and classifiers that commonly being used in text geolocation tasks, using tweets data collected from 4 cities.

Firstly, we experimented words selection over MI and WLH, these methods manually selected informative words and use one-hot encoding to represent them as features, resulted in around 29% - 33% accuracy using MNB and hard voting ensemble classifier combing MNB, LinearSVM and Logistic Regression.

Secondly, we combined TF-IDF features with some other approaches together, like breaking words into substrings, selecting words based on P(c—w), length and overall count, merging all tweets from same city into one document, replacing new words with most similar existing words calculated by Word2Vec. The accuracy then was increased to around 35.5% with the same simple classifiers.

Finally, we summarized and interpreted the behavior of three classifiers above according to their hyperparameters, parameters and math logic behind. Also, we analyzed the potential reason of errors, one could be tweets were inherently unpredictable that only contained irrelevant words or new words that didnt appear in training. The other one is the appearance of strong indicator word of wrong city, like sydney in Melbourne tweet, would unavoidably made both probability-based and parameter-assignment-based models tend to predict incorrectly.

References

[1]: Ashktorab, Zahra, Christopher Brown, Manojit Nandi, and Aron Culotta. ”Tweedr: Mining twitter to inform disaster response.” In ISCRAM. 2014.

[2]: Mittal, Anshul, and Arpit Goel. ”Stock prediction using twitter sentiment analysis.” Standford University, CS229 (2011 http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentiment-Analysis.pdf) 15 (2012).

[3]: Tang, Haina, Xiangpeng Zhao, and Yongmao Ren. ”A multilayer recognition model for twitter user geolocation.” Wireless Networks (2019): 1-6.

[4]: Cheng, Zhiyuan, James Caverlee, and Kyumin Lee. ”You are where you tweet: a content-based approach to geo-locating twitter users.” In Proceedings of the 19th ACM international conference on Information and knowledge management, pp. 759-768. ACM, 2010.

[5]: Pappas, Konstantinos, Mahmoud Azab, and Rada Mihalcea. ”A Comparative Analysis of Content-based Geolocation in Blogs and Tweets.” arXiv preprint arXiv:1811.07497 (2018).

[6]: Lim, Kwan Hui, Shanika Karunasekera, Aaron Harwood, and Yasmeen George. ”Geotagging tweets to landmarks using convolutional neural networks with text and posting time.” In Proceedings of the 24th International Conference on Intelligent User Interfaces Companion (IUI19). 2019.

[7]: Ajao, Oluwaseun, Deepayan Bhowmik, and Shahrzad Zargari. ”Content-aware tweet location inference using quadtree spatial partitioning and jaccard-cosine word embedding.” In 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 1116-1123. IEEE, 2018.

[8]: Do, Tien Huu, Duc Minh Nguyen, Evaggelia Tsiligianni, Bruno Cornelis, and Nikos Deligiannis. ”Twitter user geolocation using deep multiview learning.” In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6304-6308. IEEE, 2018.

[9]: Thomas, Philippe, and Leonhard Hennig. ”Twitter geolocation prediction using neural networks.” In International Conference of the German Society for Computational Linguistics and Language Technology, pp. 248-255. Springer, Cham, 2017.

[10]: Rahimi, Afshin, Trevor Cohn, and Timothy Baldwin. ”A neural model for user geolocation and lexical dialectology.” arXiv preprint arXiv:1704.04008 (2017).

[11]: Rahimi, Afshin, Trevor Cohn, and Tim Baldwin. ”Semi-supervised user geolocation via graph convolutional networks.” arXiv preprint arXiv:1804.08049 (2018).

[12]: Roller, Stephen, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. ”Supervised text-based geolocation using language models on an adaptive grid.” In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1500-1510. Association for Computational Linguistics, 2012.