Twitter Authorship Attribution with Classical Machine Learning Methods

Team128 Man FU(989145), Yu DONG(928922), Anqi ZHAO(1110621)

1 INTRODUCTION

Authorship attribution (AA) is the process of attempting to identify the likely authorship of a given document, given a collection of documents whose authorship is known.[7] It is a rising topic in Natural language processing and presents a few promising applications. In this project, we will be researching methods of AA on a data set of tweets to compare different algorithms' performances on modern communications.

2 FEATURES DESCRIPTION

Feature Preprocessing

Tweets contain lots of unique properties indicating the users' characteristics, which contributes to the prediction in our project. However, it's a great challenge to choose typical features for a higher accuracy.

Firstly, we did feature preprocessing like removing meaningless words like stop words, and stemming words for later efficient training. Also, tweets were tokenized and and punctuations in them were deleted from tweets. Moreover words are all in lower case after preprocessing.

After several attempts in the project, emoticons, punctuations, mentions("username"), retweets("RT"), hashtags("#word") were found to have played important roles in representing users' behaviours and characteristics.[5] In this case, we added them back to features by using regular expressions.

Besides, word frequency in tweets represents users' attitudes, interests, concerned topics. Therefore, feature engineering also applied into our feature selection.

Feature Engineering

CountVectorizer CountVectorizer provided by Scikit Learn library in Python was used in feature extraction at first. It assisted in converting text reviews into numeric matrices.[6] Based on parameters setting, the words with high frequency will be selected as features for the corresponding tweet to represent related author for later classification.

Word2vec Word2vec supported by gensim library was also applied for feature extraction. Word vectors was used in calculating similarities between words.[4] Thanks to word2vec, useful features can be selected with smaller feature dimension[1], which helps a lot for the big dataset and computation. Moreover, the combination with decomposition in sklearn relieves the pressure on computers in typical feature selecting.

TF-IDF Term frequency-inverse document frequency (TF-IDF) was used on the dataset so that we could focus on more characteristic vocabulary that users may write in their tweets. A high term frequency in a users tweet may be an indication of their unique style, while a high document frequency in all tweets indicate that this may be a too commonly used word and may not reflect individual users' writing styles.

3 MACHINE LEARNING METHODS

After prepossessing the data and selecting features, we tried out three machine learning methods to compare which would yield the better categorization scores.

Logistic Regression

Using a TF-IDF score on our data, we trained a logistic model to predict the if a sentence, or rather the group of words in that sentence, is likely to appear together in a person's tweet. It takes in the training data of key words mentioned in a users' tweet, and generates a curve to indicate the likelihood of the sentence being said by a user. When inputted with the test data, it will compare the tweet's keywords with this curve and return the user with the most likelihood of sending this tweet.

The time complexity of logistic regression is O((f+1)csE) for training and O((f+1)cs) for testing where f is the number of features, c is the number of classes, s is the number of samples and E is the number of epochs. Training and testing using this model is very slow when using relatively large data sets such as the current. However, with it's long training time, the categorization score could only go up to around 5%, which was not a significant score.

Support Vector Machine

The Linear support vector machine method finds a separable boundary that maximize the margin between classes.[2][3] Because labelling a tweet with specific user id is a multi-class problem, we used the default multi-class method with parameter of 'ovr' for classification. Compared with the parameter of 'crammer single', the 'ovr' method leads to better accuracy and this method is less expensive to compute. The loss function measures the discrepancy between predicted user id and true user id for a same tweet. In this project, we used default loss function whose name is 'squared hinge'. To avoid over-fitting with too many features, we use penalty function of L1 to get sparse solution.

K Nearest Neighbors

The k-nearest neighbours (KNN) method, as a type of lazy learning, is the easiest to train among the machine learning methods we tried as it only needs to store the training instances. When performing predictions, KNN takes in the test instance and compare it with all existing training instances to find the k most similar training instances, and returns the average value of the k instances. Thus, it was the method with the fastest training and slowest testing. In this project, the prediction we make is the user's number or rather their label, which is nominal rather than interval. Thus for k > 1, we returned the mode of the k most similar training instances instead, and in the case that mode does not exist, the 1NN result is returned.

4 MODELS SELECTION

After some preliminary tests, we identified the linear SVM model to produce the best results. Thus, we went into further research and testing to study how to raise the accuracy of the linear SVM method specifically to a greater extent. Initially, our features selection algorithm eliminated punctuation marks before feeding the data to the machine learning algorithm.This produced an accuracy score of 0.18. After further observing the data, we realized that users may also exhibit certain styles of using punctuation as well. Thus, we changed our feature selection algorithm to keep punctuation instead. Running the same algorithm with the punctuated data, the accuracy score rose to 0.20 which proved that users indeed had their own styles of using punctuation. We also tried changing the penalty parameters. Changing the penalty from the default L2 to L1 raised the accuracy by 0.20 to 0.24. The advantage of using L1 over L2 in linear SVM is that it prevents over-fitting.

5 CRITICAL ANALYSIS

Initially, before choosing features, we considered choosing 5000 users who send the most tweets. Compared with 9297 users in raw text, reducing the numbers of users can reduce the computation on training model. The assumption was that less labels with more frequency should improve the accuracy of prediction on the test data set.

We also tried using tokens rather than TF-IDF on this limited training set. Assumption here was that tokens allowed the model to detect distinct unique features for tweets and improves the precision of labelling.

Later we used linear SVM to train and predict based on word frequency by using CountVectorizer on total data set. The accuracy was 0.16771. So we added emoticons,punctuations, mentions, retweets, and hashtags in our features. With the help of them, the accuracy was 0.18399. So these are important features which cannot be ignored. Besides, after analysing the results, we realized that when using TF-IDF on full data was a better feature model as and the machine learning algorithm has access to all users' writing styles as TF-IDF actually produces a better result with larger data sets and is trained against all users that could possibly appear in the test set. Later, it turned out our assumption was right. The accuracy was raised to 0.24240. With the using of parameter "sublinear_tf = true" which uses (1+log(TF)) instead of direct term frequency(TF), the accuracy was improved to 0.24588.

When choosing models, we first tried KNN since it is a simple model and fast for training. We get the accuracy of 0.03, 0.05, 0.06 when setting the nearest neighbor of 5, 3 and 1 respectively. This implies the prediction has the highest accuracy when the model labels the tweet with most similar user. Then we deployed the SVM model on the data set. Since SVM find the perfect boundary between classes, it takes a long time for model training. But the prediction on test data can achieve 0.19, highest among all the methods we have tried. We also tried LR, but it does not have good performance for multi-class problem.

Feature Improvement	Accuracy
1	0.168
2	0.184
3	0.242
4	0.246

Model	Accuracy
5-NN	0.03
3-NN	0.05
1-NN	0.06
LR	0.12
SVM	0.19

6 CONCLUSION

After comparing the performance of different models, we find that SVM can achieve highest accuracy with the features we extracted through TF-IDF. In the future, we can focus on finding more unique features which can improve the prediction accuracy.

REFERENCES

[1] Long Ma and Yanqing Zhang. 2015. Using Word2Vec to process big text data. In *2015 IEEE International Conference on Big Data (Big Data)*. IEEE, 2895–2897.

[2] Ben Rubinstein. 2019. Lecture 10. Soft-Margin SVM,Lagrangian Duality. (2019).

[3] Ben Rubinstein. 2019. Lecture 9. support vector machines. (2019).

[4] Tianze Shi and Zhiyuan Liu. 2014. Linking GloVe with word2vec. *arXiv preprint arXiv:1411.5595* (2014).

[5] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. 2010. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 841–842.

[6] Abinash Tripathy, Ankit Agrawal, and Santanu Kumar Rath. 2016. Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications* 57 (2016), 117–126.

[7] Ying Zhao and Justin Zobel. 2007. Searching with style: Authorship attribution in classic literature. In *Proceedings of the thirtieth Australasian conference on Computer science-Volume 62*. Australian Computer Society, Inc., 59–68.