Report layout of example report we saw

* + abstract

1. introduction
2. Related work
3. Overview

3.1 Project Build Process

4. Task Design

4.1 Project Build Process

5. Dataset

5.1 source of data

6. Feliss kappa (measurement)

6.1 Tweet annotation label

7. Labelling

8. Work quality of the worker

8.1 accuracy vs Hits

8.2 worker average time

8.3 worker average time

9 extras

10 conclusions

Acknowledgments

References

Abstract

Most of the world’s population is bi-language. Moreover, even though the dominant language in the internet (particularly in social media) is English, there are evidences that most of the dynamic content is created by non-native English speakers. That is why the problem of distinguishing between native and non-native speakers is drawing attention. Potential applications of this task are: Teaching English more efficiently, identifying target audience based on native language etc.  
In this work we took an approach that is content independent - we are modelling text by the function words which occur in it. We then use machine learning techniques to distinguish between native and non-native English speakers, yielding solid results.

Introduction

Native language Identification (NLI) is the task of determining an author's native language based on his writing of a different language. NLI is well studied (Koppel et al. 2005). This work is focused on the similar somewhat simplified task of Native language classification – based on a text written in English – decide if the writer is a native (English) speaker, or not. One popular approach for classification is Bag of Words (BOW). In BOW the text is represented by a feature vector of dimension N (N is the size of the vocabulary or a subset of the vocabulary- i.e. N most frequent words). Each dimension is the count of the corresponding word form the vocabulary occurring in the text. Usually stop words are excluded from the feature vector. This approach (as will be shown in the results section of this document) is very effective for this task of classification. However, this approach suffers from a main disadvantage which is content dependency. For instance, if a classification model is trained on specific domain corpus (e.g. Sports, Politics, traveling etc) this model will consider words from the specific domain with much higher importance compared to words out of the specific domain. This presents an issue in generalization across domain. Also there is a time relevance issue – domain dominant words can change significantly over time (this is particularly noticeable domains such as politics and sports – where things changes rapidly). This could lead to poor classification results that might force retraining of the model for the new domain or the new period.

In order to overcome the content dependency issue we took a content independent approach by observing the use of function words (which do not carry content) for native and non-native English speakers. Because function words are not domain nor period dependent - this approach is robust to the issues described above and yields firm results for binary classification.

We use 2 different data sets for native and non-native English speakers respectively. Native language data set was extracted from Reddit – a popular American discussion website. We had 6 different countries of origin for native English speakers (USA, UK, Ireland, New-Zealand, Canada and Australia) for each we used an equal portion in this work. Non-native English speakers dataset was taken from TOFEL (Test Of English As A Foreign Language) – a collection of assays written by non-native English speakers as a university entrance test. The country of origin of the writer was not specified. We used function words dictionary from the academic resources of Sequence Publishing as a base and extended it manually during our training process.  
We used several well-known classifiers (SVM, Decision Tree and Naive Base) for this task and compared the results of the function words approach vs. BOW approach.

Related work

The task of native language identification (NLI) has got a fair amount of attention (Koppel et al. 2005, Jarvis & Paquot 2015 and many more) - 2 Shared tasks were held 2013 and 2017 in which dozens teams were competing. In this work our focus was classification and not identification of native language – i.e. determine if a text was written by an English native speaker or not (binary classification).   
McNamara & Crossley (2011) focused on identifying shared lexical features of non-native English speakers and found that some of these shared features are style related.  
 The approach taken in this work - of observing the use of function words for classification - was used before for the similar task of identifying translations source language (Koppel & Ordan 2011). This approach loosens the dependency upon content and focuses on stylistic characteristics. As shown (Koppel & Ordan 2011) some function words are over-represented and under-represented according to the source language, creating a solid base for classification.

Dataset

We used the Reddit dataset released by Rabinovich et al. (2018) as our native English speaker corpus. Reddit is a popular online community consisting of thousands of forums in a wide range of topics. The dataset includes Reddit posts whose content is generated by users specifying their country as a flair (metadata attribute). Following Rabinovich et al. (2018), we view the country information as an accurate, (though not perfect). Native English speaker countries used in this work consist of: US, UK, Australia, New Zealand, Ireland and Canada. For our non-native data set we used TOEFL dataset. The TOEFL test is used internationally as a measure of academic English proficiency, among other purposes, to inform admissions decisions for students seeking to study at institutions of higher learning where English is the language of instruction. The dataset consists of essays written by authors of 10 different native languages on 8 topics sampled as evenly as possible.  
The native languages of the authors in the TOEFL dataset are: Arabic, Chinese, French, German, Hindi, Italian, Japanese, Korean, Spanish, Telugu and Turkish.

.

Methods

## preprocessing

As a first step we converted datasets to lower case.  
Since our methodology focuses on function words we needed sentences that are in sufficient length. We sampled sentence containing 45, 20, 15 tokens in them in 3 different settings. As TOEFL was the smaller dataset, we fixed the sample size of the larger dataset of Reddit accordingly.   
For the non-native English speakers, we had 11044 sentences for the 45 word sentences (across the entire dataset). For the native English speakers, we had 6 countries of origin to sample form. We randomly sampled an even portion of 1840 45-word sentences from each which summed up to 11040 sentences.

## Feature vectors

For the sake of content independent classification, we relied on function words – words that are in general not related to content and considered as a stylistic property.   
We downloaded a dictionary from the academic resources at [Sequence Publishing](http://www.sequencepublishing.com/1/academic.html)   
 Our dictionary consists of 311 function words. We examined the function word used in those sentences and searched a pattern in both classes (Native and Non-nativ). For each sample we created a 311-dimension feature vector, where the kth dimension in the vector is the number of occurrences of the kth function word from our dictionary in the given sample (sentence).   
To examine the effectiveness of this approach, we confronted it with the Bag of Words (BOW) approach, that is known to yield excellent results, and is strongly depend upon content. We took the 230 most frequent words from the 2 datasets, leaving function words and other special characters out and removing duplicates – summing up to 343. We constructed a 343-dimension feature vector- where the kth dimension in the vector is the number of occurrences of the kth word from the 343 most frequent words we collected -in the given sample (sentence).   
  
Classifiers

For both method we used 3 classifiers which we used from Sklearn python library.

## SVM

We use sklearn.svm as our main classifier. We used an RBF kernel, cost 1, and to improve run time we increased the cache size to 7000.

SVC (C=1.0, cache\_size=7000, class\_weight=None, coef0=0.0, decision\_function\_shape='ovr', degree=3, gamma='auto', kernel='rbf’, max\_iter=-1, probability=False, random\_state=None, shrinking=True, tol=0.001, verbose=False)

## Naïve Bayes

We used sklearn naive Bayes as the second classifier. It was set on the default parameters. MultinomialNB (alpha=1.0, class\_prior=None, fit\_prior=True)

## Decision Tree

We used ~. It was set on the default parameters. DecisionTreeClassifier (class\_weight=None, criterion='gini', max\_depth=None, max\_features=None, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, min\_samples\_leaf=1, min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0, presort=False, random\_state=0, splitter='best')

Experiments

## Word usage

## Common words

To understand the process of classifying the correct class, we examined the frequent words in both method. Our intuition was that Native speakers in both methods will use more diverse words while the Non-natives with less richer language will use more frequently the common words.

Top 10 function words used by both classes

|  |  |  |  |
| --- | --- | --- | --- |
| Native (11040 samples) | | Non-native (11044 samples) | |
| Function word(s) | Count | Function word(s) | Count |
| The | 29792 | The | 30795 |
| To | 16770 | To | 23452 |
| And | 15245 | And | 17640 |
| Of | 14777 | Of | 14846 |
| A | 12432 | In | 14372 |
| In | 10226 | A | 13726 |
| That | 9192 | That | 12590 |
| It | 7575 | Is | 10852 |
| Is | 7381 | I | 7780 |
| I | 6888 | It | 7280 |
| total | 130278 | total | 153333 |

If we look on the top 10 function words used by each class as a group, they are the same on both classes. An interesting thing that can be deducted from this table is that Non-natives have used the top 10 function words 17.69% more then the Natives, meaning (assuming the word count for each class is close) that they use less un common function words which support our line of thought. To dig deeper, we wanted to see the variance of each class. We calculated the average use of function words for each class (Native=909.5, Non-native=1089). Then we summed the power of 2 of the subtraction of each word count from the average. At the end we divided by the number of words. If our intuition is right, the Non-natives will have a much bigger variance. The results support our claim. For Non-natives 9,702,525 and for Natives 7,066,042.

אני מציע למחוק את הטבלה הבאה כי יש בה יותר מידי function words מה שמראה שהמילון שלנו לא הכי טוב

Top 10 words used by both classes

|  |  |  |  |
| --- | --- | --- | --- |
| Native (11040 samples) | | Non-native (11044 samples) | |
| Word | Count | Word | Count |
| 's | 4726 | People | 7239 |
| People | 2341 | Life | 3015 |
| Has | 1614 | Time | 2513 |
| Just | 1609 | Young | 2380 |
| Eu | 1378 | Think | 2262 |
| Were | 1245 | New | 2224 |
| Think | 1181 | Cars | 2089 |
| ‘re | 1129 | Things | 2013 |
| Even | 1081 | Better | 1888 |
| Get | 1055 | knowledge | 1866 |
| total | 17359 | total | 27489 |

Despite the words are different, again the Non-native repeated their top 10 words more then the Natives.

## Biggest difference usage in words

For each word in our feature vector, we compared the difference between classes. We retrieved the top 10 words with are farthest in usage

|  |  |  |  |
| --- | --- | --- | --- |
| Function words | | | |
| word | Count Native | Count Non-native | Difference |
| to | 16770 | 23452 | 6682 |
| In | 10226 | 14372 | 4146 |
| Is | 7381 | 10852 | 3471 |
| That | 9192 | 12590 | 3398 |
| Will | 1474 | 4674 | 3200 |
| More | 1747 | 4470 | 2723 |
| n’t | 3856 | 1435 | 2421 |
| Are | 3788 | 6191 | 2403 |
| And | 15245 | 17640 | 2395 |
| Can | 1552 | 3915 | 2363 |

A quick look shows that the 10 most different function words are all used more by Non-native speakers.

|  |  |  |  |
| --- | --- | --- | --- |
| BOW | | | |
| word | Count Native | Count Non-native | Difference |
| people | 2341 | 7239 | 4898 |
| ‘s | 4726 | 1255 | 3471 |
| Life | 255 | 3015 | 2760 |
| Young | 65 | 2380 | 2315 |
| Cars | 36 | 2089 | 2053 |
| Knowledge | 55 | 1866 | 1811 |
| New | 475 | 2224 | 1749 |
| Time | 841 | 2513 | 1672 |
| Facts | 49 | 1578 | 1529 |
| Things | 496 | 2013 | 1517 |

And again, the pattern is in the numbers. The Non-native over using the top 10 most different words except for ‘s which maybe should have been in our function words dictionary.

## Parameters

To get good result, we first need to consider couple of key parameters, sentence length and size of the BOW feature vector. Other than that, after doing some cross validation we came up that a 80-20% split of the train-test data is good.

## Sentence length

The first parameter is the number of tokens desired in each sentence we take from our raw data into our data set. If we take a sentence with 3 words, we will have a problem classifying. So, we inspected 3 different sizes: 10, 25, 45 (this size are not words in the sentence but tokens. For example, ‘.’ could be a token). Since the data size with tokens length 45 is ~11,000 for class non-native we set the data size for all experiment to be 22000 sample. Also we used BOW dictionary size 343 (top 230 words from each class – 343 is after removing duplicates)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Function words | | |
| Length | 10 | 25 | 45 |
| SVM | 71.51% | 76.52% | 82.18% |
| Decision tree | 62.23% | 66.53% | 69.66% |
| Naive Bayes | 72.1% | 75.66% | 79.76% |
|  | BOW | | |
| Length | 10 | 25 | 45 |
| SVM | 83.9% | 89.47% | 93.38% |
| Decision tree | 81.16% | 83.72% | 87.52% |
| Naive Bayes | 86.43% | 90.46% | 93.45% |

For function words classification method, it was best to select 45 tokens length.

## Size of BOW feature vector

To show the true loss of using function words method, we need to maximize the performance of the BOW method. So, we examined the sizes: 230, 500, 1000, 2000, 3000 (number of words from each class – the combination is smaller or equal to two times the size)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Top X words from class | Feature vector size after removing duplicates | SVM | Decision Tree | NB |
| 230 | 343 | 93.39 | 87.53 | 93.46 |
| 500 | 728 | 93.43 | 88.66 | 94.7 |
| 1000 | 1445 | 92.87 | 88.54 | 96.4 |
| 2000 | 2848 | 90.9 | 88.86 | 97.17 |
| 3000 | 4275 | 89.13 | 89.09 | 97.44 |

We can see SVM performance is decreasing while the feature vector size is increasing. Decision tree remains around the same accuracy. Naive Bayes accuracy is increasing as the feature vector size is increasing. Since SVM is our main classifier, we selected 230 words from each class.

## Data size

We tried to minimize the size of the train data to see what’s the optimal size and maybe change our data size and\or our 80-20% spilt.

Since we used on most of our experiments test size 4417 samples, i recommend to leave only the second table of the 2000 test size. Also the 2000 table is more monotonic. Also Maybe remove some rows – too much data

We fixed the test data size on 1000 samples and changed the train data size from 2500 to the 21084 (all the data minus the 1000 for the test set):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Function words | | | BOW | | |
|  | SVM | Decision tree | Naive Bayes | SVM | Decision tree | Naive Bayes |
| 2500 | 78.2 | 66.5 | 79.1 | 87.5 | 86.4 | 92.6 |
| 4500 | 79.5 | 67.6 | 80 | 89.7 | 87.8 | 93.2 |
| 6500 | 79.6 | 68.7 | 80 | 90.8 | 86.5 | 93 |
| 8500 | 80.1 | 65.2 | 80 | 91.4 | 88 | 92.8 |
| 10500 | 80.5 | 65.9 | 80.4 | 91.5 | 86.7 | 93 |
| 12500 | 80.9 | 67.6 | 80.8 | 92.1 | 86.2 | 92.9 |
| 14500 | 81 | 67 | 80.5 | 92.4 | 86.2 | 93 |
| 16500 | 81.1 | 66 | 80.9 | 92.5 | 86.3 | 93.2 |
| 18500 | 81.5 | 68.6 | 80.8 | 92.5 | 87.3 | 93.3 |
| 20500 | 81.7 | 68.8 | 80.8 | 92.5 | 87.6 | 93.2 |
| 21084 | 81.5 | 68.8 | 80.8 | 92.5 | 87.6 | 93.2 |

We can see that SVM accuracy is almost increasing monotonically in regard to the growth the train size.

We fixed the test data size on 2000 samples and changed the train data size from 2500 to the 20084 (all the data minus the 2000 for the test set):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Function words | | | BOW | | |
|  | SVM | Decision tree | Naive Bayes | SVM | Decision tree | Naive Bayes |
| 2500 | 78.9 | 66.4 | 79 | 88 | 85.8 | 92.5 |
| 4500 | 79.9 | 67.3 | 79.5 | 90 | 86.8 | 93.1 |
| 6500 | 80.1 | 68.2 | 79.5 | 91.1 | 85.9 | 92.7 |
| 8500 | 80.7 | 66.7 | 79.7 | 91.7 | 86.9 | 92.7 |
| 10500 | 80.9 | 67.6 | 80.2 | 91.7 | 86.7 | 93 |
| 12500 | 81.1 | 68.5 | 80.6 | 92.3 | 86.6 | 92.9 |
| 14500 | 81.5 | 67.4 | 80.3 | 92.5 | 86.4 | 93 |
| 16500 | 81.7 | 67.7 | 80.4 | 92.6 | 86.2 | 93 |
| 18500 | 81.9 | 69.3 | 80.5 | 92.9 | 86.8 | 93.1 |
| 20084 | 82 | 69.3 | 80.5 | 93 | 86.8 | 93.1 |

As for SVM, the accuracy is rising monotonically as the train data size grow on both methods. No point withholding train data.

As for Decision Tree, the accuracy wasn’t monotonic but did reach a maximum accuracy when the train data size was maximal.

As for Naive Bayes, on function words it reached to the best accuracy when train data size was 12500 and on BOW on 4500, 18500 and 20084.

## Full run

After setting the parameters, 45 for tokens length and 343 as for BOW feature vector size, with 80-20% split we measured the precision, recall and accuracy for each classifier on both methods. Instead of over crowding the table, we will show f score for each class which is the harmonic average of the precision and recall summed

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Function words | | | |
|  | Non-native f score | Native f score | weighted f score | accuracy |
| SVM | 0.819 | 0.823 | 0.821 | 82.18% |
| Decision tree | 0.695 | 0.697 | 0.696 | 69.66% |
| Naive Bayes | 0.799 | 0.795 | 0.797 | 79.76% |

We achieved best accuracy of 82.18% using SVM.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Bow | | | |
|  | Non-native f score | Native f score | weighted f score | accuracy |
| SVM | 0.931 | 0.935 | 0.933 | 93.38% |
| Decision tree | 0.876 | 0.874 | 0.875 | 87.52% |
| Naive Bayes | 0.934 | 0.934 | 0.934 | 93.45% |

We achieved best accuracy of 93.45% using NB. SVM came close with 93.38%. observing f score of each class shows that the classifiers on both methods act almost the same for each class.

## Different Native English types

For our Native English we sampled authors from 6 different English spoken countries: US, UK, Australia, New-Zealand, Ireland and Canada. As most of NLP tools and knowledge of English is focused on US English, we were interested in testing the influence of this mixture of "Englishes" on the results. For that we compared the original setting of 6 types of native English to solely US English as our native data set.

As explained in the above, the TOEFL sample came from unknown origin of country. Although, the Reddit came from 6 different Native English-speaking countries. We decided to check this classification problem using only American English. The intuition we had is that the results will improve. We used tokens length 45 and BOW feature vector size 343.

|  |  |  |
| --- | --- | --- |
|  | Function words | BOW |
| SVM | 82.5 | 93.7 |
| Decision tree | 70.2 | 88.72 |
| Naive Bayes | 80.08 | 93.88 |

Comparison to the results from tables at 5.4 (the difference is that at 5.4 the native data came from 6 different countries and at 5.5 it came only from US):

On both methods: all 3 classifiers performed better when the native data came only from the US.

Results

On BOW we achieved accuracy up to 97% using Naive Bayes (see 5.2.2). On function words we achieved up to 82.5% using SVM (see 5.5). Despite the big difference, the us

Conclusions

We presented a method using function words that is robust to domain changes and time changes. It ignores the content in the sentence and search patterns in usage of its function words. Despite the big difference of accuracy on both method, the usage of function words is better. The run time is smaller because there is no need to go over all the data to create a dictionary and the size of the function words is significantly smaller (311 fixed size vs 3000 where we achieved accuracy of 97%). Also, it is likely that the same function words model we trained using this data set will work on different data sets while the BOW model is likely to fail upon switching domains. In addition, function words usage doesn’t change much over the year while usage of words changes often due to slang.

Future work

We plan to do the same classification problem using RNN (recursive neural network). Our goal will be not only to separate natives and non-natives, but also to indicate the origin country, a multi class problem.