CIS530 Computational Linguistics
Final Project Report

Xiaozhuo Cheng, Boyi He

April 25, 2017

1 Introduction

The final project is to develop a system that predicts the relative difficulty of a text. The training set we used contains 461 excerpts labeled with scores out of 10, where 1 is the easiest and 10 is the most difficult.

Our group focused on extracting a variety of lexical and syntactic features from the corpus, including n-grams bag of words, vocabulary size, word length variation, sentence length variation, Type-Token ratio, Part-of-Speech (POS) tags, etc. The objective is to find an appropriate feature representation of text readability. Actually, any of word richness, syntactic complexity, and even specialized medical knowledge might cause reading difficulty, so we expected them to be quite predictive.

After several approaches, our best model is to combine POS tags, Named Entity Recognition (NER) labels and Brown cluster identities altogether as features, and to create a support vector machine to predict the readability score.

2 Method

Here are the resources and tools we used:

1. NLTK

   We used **word_tokenize**, **sent_tokenize**, and **FreqDist** to tokenize the data and get frequency distributions from tokenized excerpts.

2. Stanford CoreNLP

   We used CoreNLP to generate xml files for the data sets which will be used to extract features. Similar to the second homework, we represented the excerpts with Part of speech, Named entities, syntactic dependencies and syntactic production rules.

3. Brown Cluster file

   We used Brown Cluster file to generate Brown cluster features for a given text by substituting a word with its corresponding cluster id. The Brown Cluster files we used are:

   - brown-rcv1.clean.tokenized-CoNLL03.txt-c100-freq1.txt
   - brown-rcv1.clean.tokenized-CoNLL03.txt-c3200-freq1.txt
     https://github.com/c-amr/camr/tree/master/resources/wclusters-engiga

4. scikits.learn

   We used **Epsilon-Support Vector Regression** (**SVR**) from scikits.learn to train our model to predict the readability score.

5. Scipy

   We used **scipy.stats.spearmanr** to calculate the Spearman correlation between two rankings, in order to evaluate our predictions in validation step.

3 Final System

Our final system uses the combination of POS tags, Brown Cluster identities and NER features. Each of the features will map every token in an excerpt into a tag/cluster. We calculated the distribution over all the tags/clusters for each excerpt, which is a vector. The three vectors for each excerpt were concatenated into a single vector, which is our feature representation. We used the train set to train the model, and used SVR as our machine learning model.

4 Experiments

4.1 Local Preliminary Results

As claimed in previous part, we implemented features of various types, including classical features to measure readability, features related to language models such as n-gram bag-of-words and syntactic features.

To find which features are more predictive in this task, we divided the training set into two parts. We used the first part to train our model, and the second part to validate it using Spearman correlation. To avoid over-fitting, we did a k-fold cross validation. But after some experiments, we found that the performance of a model strongly relied on how we separated the training set. This might be caused by the small size of the training corpus or the big variance of the data. To minimize the effect of random partition on the variation results, we ran 2-fold cross-validation multiple times (typically 20 times) using the model, and took the average.

Here are the local preliminary results we got (with SVR):

- Among the classical features we implemented, vocabulary size has the highest correlation, which is around 0.228.

Features	Spearman Correlation
Vocabulary Size	0.228305740222
Fraction of Frequent Words (more than 6 times)	0.153923146718
Fraction of Rare Word (more than 3 times)	0.13116525617
Median Word Length	0.0547891496984
Mean Word Length	-0.0237772968644
Mean Sentence Length	0.0275755521293
Type-Token Ratio	0.156250021302

Table 1: Classical Features

- Among the Unigram and Bigram bag-of-words we implemented, top 4000 Unigram and top 4500 Bigram has the highest correlations, which are around 0.197 and 0.155. TF-IDF is also predictive relatively, whose correlation is around 0.199. However, all theses scores are lower compared to the correlation score got by vocabulary size simply.

Features	Spearman Correlation
Unigram (top 3000)	0.181302233801
Unigram (top 4000)	0.197170603032
Unigram (top 4500)	0.130161168733
Unigram (top 5000)	0.177991692817
Bigram (top 3000)	0.0895149373835
Bigram (top 4000)	0.103821785743
Bigram (top 4500)	0.154769792246
Bigram (top 5000)	0.0909119347687
TF-IDF	0.19881474157
Mutual Information	0.168683389154

Table 2: Weighting Schemes and N-gram Bag-of-Words

- Syntactic features are the most predictive ones, compare to the features mentioned above. Note the Brown Cluster Identities has the highest estimated score as a single feature, which is around 0.326.

Features	Spearman Correlation
POS Tags	0.255703582528
Universal POS Tags	0.201036072875
NER Labels	0.0394201744184
Dependency Relations	0.212124035841
Syntactic Productions	0.254497563442
Brown Cluster Identities	0.325705977545

Table 3: Syntactic Features

- Now we tried to combine the features. Based on the combinations with the highest correlation score, we generated our final predictor.

Features	Spearman Correlation
Unigram (top 4000) + Bigram (top 4500)	0.158804254091
POS Tags + Unigram (top 4000)	0.203226099794
POS Tags + Universal POS Tags	0.226266354806
POS Tags + Dependency Relations	0.243528304823
POS Tags + Syntactic Productions	0.181608394708
POS Tags + Brown Cluster Identities	0.339476492653
Syntactic Productions + Brown Cluster Identities	0.217129309489
POS Tags + Syntactic Productions + Brown Cluster Identities	0.215198142283
POS Tags + NER Labels + Brown Cluster Identities	0.341134698426

Table 4: Feature Combinations

4.2 Submitted Preliminary Results

Models	Spearman Correlation
POS Tags + Brown Cluster Identities, SVR	0.38232619055
POS Tags + Brown Cluster Identities (updated), SVR	0.386003567059
POS Tags + Brown Cluster Identities + NER Labels, SVR	0.391568018354

Table 5: Submitted Preliminary Results

- Result of the first submission is close to (a little higher than) our local estimation. So now we know the combination of POS tags and Brown Cluster Identities works well, and we are more confident about our local evaluations.
- To improve the performance of Brown Cluster Identities, in the second submission, we replaced the Brown Cluster file of 100 clusters to another Brown Cluster source file of 3200 clusters. But this does not make any obvious improvement.
- In the final submission, we also included NER features considering its practical meaning in this task, and its relevance to POS tags and Brown Cluster Identities.

5 Discussion and Analysis

1. Why Brown Cluster Identities works well in this task?

Brown Clusters partition a corpus of words into word clusters. The words that are grouped together have the same or similar types, so it is likely that some hard words, which are difficult to read, such as medical terminology are grouped together.

Also, for words that do not appear in the precomputed Brown clusters, we set them with a new cluster ID, 8888. For an excerpt with high difficulty, it may have high ratios for clusters that have plenty of hard words and a high ratio for the new cluster, which implies there are more rare words. This could be a reason for Brown Cluster to have the highest estimated score as a single feature.

2. Why combining POS tags and NER labels is helpful?

   From Table 3 we can see, Brown Clusters are quite predictive in this task. Based on our analysis, that means learning about word difficulty, word similarity or unknown words might help us measure text readability. Note that POS tags can somewhat indicate the similarity between words, and named entities often do not appear in the precomputed Brown clusters. So POS tags and NER labels are kind of 'similar' to Brown Cluster Identities.

   But on the other hand, POS tags and NER labels also provide addition information to this model. So combining them is helpful.

3. Why classical features does not work well?

   Since most classical features are quite simple. They cannot indicate word difficulty, word similarity or unknown words at all.

4. Why n-gram bag-of-words does not work well?

   Different from classical features, n-gram bag-of-words can indicate word difficulty, word similarity or unknown word using normalized frequency distribution to some degree. Also, this kind of features work very well for the previous homework to represent an author's writing style. But this time, why it doesn't work very well?

   The reason might be the size of the training corpus. This time, the training set contains only 461 excerpts, which is quite small. Due to the corpus size, difficult words are hard to appear in many different excerpts, so they can hardly be chosen in bag-of-words.

5. How can we trust our local validations? Why?

   As mentioned earlier, the local performance relies greatly on how to separate the training set into two parts. For example, although we got an average Spearman correlation score of 0.339 locally using POS tags and Brown Cluster Identities as features, the minimum score in 20 groups of 2-fold cross validation is often smaller than 0.2. This is probably caused by too much noise existed in validation set. Consider the equation to compute Spearman correlation coefficient, one sample whose ranking is far away from what the model predicted it to be might have a great negative effect on the validation result.

   But after the first submission, we found that the submission result is even higher the local validation score we got. That means, there are not so much noise in the test set, and we can be more confident with our model.

6. What about optional data set?

   We have tried with optional data set, but finally gave it up because we need to find a way to coordinate the optional set with the training set to continue.

   If we simply used the optional set to validate, it doesn't work. For example, we divided the training set into two halves, one to generate a model and the other to validate that model. We also applied the model on the optional data, and then measured the Spearman correlation between the model outputs and the optional scores provided (opposite). However, after experiments, we found that we cannot see a direct relationship between the two local validation results. So we cannot simply use the optional set to validate, since its corpus is quite different from corpus of the training set.

7. Additional ways for improvement?

   - To find the relationship between each single word and text reading difficulty. And then used 'word difficulty' to predict readability of the entire excerpt.

- Try to filter out noise data existed in training corpus.
- Find a way to coordinate the optional training set with the training set and then use the optional data to build our model.
- Do this task as a multi-class classifier rather than a regression. Now the objective is to find the most similar group (with score from 1 to 10) for each test excerpt.