Automated Reading Comprehension Clustering

Charles Hathaway, David Hedin

December 15, 2015

Abstract

Determining the level of readability of documents, especially books, has lots of application in the domain of education. It helps to quantify and group books which may be used at a particular reading level, therefore enhancing the classroom expierence for both instructors and teachers. In this paper, we ran an experiment with a variety of books obtained from Project Gutenberg [1] organized by the project maintainers into a 2 groups; books for children, and adult fiction. To further enhance the analysis of this project, we also used our features to try and cluster books into clusters defined by the FleschKincaid readability metrics [2].

1 Introduction

There are several systems currently used to classify books based on reading comprehension level, for numerous applications ranging from selecting books for classrooms, to measuring an individuals literacy skills for both medical (autism, dyslexia, etc.) and educational purposes. In this paper, we analyze the results of utilizing several existing features to classify works in addition to a variety of novel features we created. The primary goal is to cluster books in groups representing the original designation of books; adult fiction and children fiction.

Given an input of 2292 books (320 children, 2002 adult, with 32 books overlapping) we achieved an F-score of 87.5%; this is a significant improvement over the 62% baseline [1]

2 Previous Works

Although not extensively, we did evaluate and learn from a number of previous works. Most significantly, we borrowed features from work done by Feng at al. [3]. Feng focused on the concept of documents being difficult to read due to

[1] There was some in-class discussion which suggested our baseline would be (total number of adult books)/(total number of books), which would put the baseline at around 86%. After testing this experimentally, and reasoning things out, we concluded the true baseline would be 62% as our algorithm had no idea what the sizes of the clusters were, and a truly random distribution would but half in each cluster, with one cluster having a higher chance of being correct than the other

"items fall[ing] out of memory before they can be semantically encoded". With this in mind, most of the novel metrics they designed focus on the number of entities mentioned in a document. Their work was motived to help organize documents by readability for adults with intellectual disabilities.

Our work is different than this in that we are using a much different data source (they were focused on news articles, where we are focused on fiction books). In addition, instead of focusing on adults with intellectual disabilities directly, we are instead more interested in examining books in relation to each other, with the assumption that easier to read books would be easier for both adults with ID, and developmentally delayed children.

3 Methodology and system design

The system is written in Python, and utilizes the Natural Language ToolKit (NLTK) [4]. The toolkit was trained using a portion of the Penn Treebank and the Conll2000 corpus. The potion used was provided by NLTK via it's distribution manager (nltk.download).

Once the corpus was downloaded and configured, we built the system in 3 stages. The first was retrieving the document to be analyzed; this was done ahead of time to verify the availability of the resources, and allow us to develop without worrying about going over any kind of bandwidth quota. However, now that development is mostly complete, this step can be skipped and documents can be downloaded directly from a Gutenberg mirror.

The next step is to process the data. This includes using the trained POS tagger, chunker, and NE tagger on all documents, running a variety of feature function on the documents, then recording the device. Our system writes data to a CSV file as it is processed, and uses a "Book" object to cache the results of each computation between features.

And lastly, we score the results of the system. This process involves reading in the CSV file and training our clustering algorithmn, then clustering our test data. Ultimately, we determine the accuracy of the clustering using a B^3 scoring system.

To get a better idea of how each feature contributes to the final score, we run the system multiple times with every combination of features.

Our initial experiment ran with 300-some-odd children books and over 2000 adult books. If we assume all books to be adult books, we can get a f-score of over 90%. To combat this problem, we opted to include a larger selection of children books.

3.1 System Usage

To setup your system, please follow the instruction in the README.md file provided in the repository.

Once configured, simply run "python src/main.py –help" to get a list of options and configuration settings. The simplest invocation of this command

requires you to specify 2 files; the list of adult books, and the list of children books. It will then print the clustering results with the default feature-set to standard out.

4 Linguistic Features

Based on previous work, a list of possible features was generated that could be used as a starting point for clustering the data. As well as this list, a new list of features was created with other features that may be useful for clustering.

Average number of words per sentence
Average number of syllables per word
Percentage of words with more than 3 syllables
Average number of noun phrases per sentence
Average number of common and proper nouns per sentence
Average number of verb phrases per sentence
Average number of adjectives per sentence
Average number of conjunctions per sentence
Average number of prepositional phrases per sentence
Total number of noun phrases in document
Total number of common and proper nouns in document
Total number of verb phrases in document
Total number of adjectives in document
Total number of conjunctions in document
Total number of prepositional phrases in document
Number of entity mentions in document
Number of unique entities in document
Average number of entity mentions per sentence
Average number of unique entities per sentence

Table 1: List of possible features from previous work [3]

From these two lists, a number of features were selected and grouped together to form 6 possible metrics for clustering the data. A correlation between nouns and initial results for clustering was the reason for deciding that features that used nouns should be in their own group. These clusters are documented in tables 3-8

5 Results

Originally, we ran all combinations of features to find the best combination. This was time consuming, and not very telling since there were a few features which clearly were superior to others. To help make more sense of the data,

Average word length in document
Total number of unique words in document
Ratio of unique words to total number of words in document
Ratio of proper nouns to common nouns in document
Length of document
Average number of proper nouns per sentence
Total number of proper nouns in document
Total number of passive sentences in document
Average number of prepositional phrases per sentence
Total number of prepositional phrases in document

Table 2: List of possible new features

Total number of noun phrases in document
Total number of proper nouns in document
Total number of common and proper nouns in document
Ratio of proper nouns to common nouns in document

Table 3: Document Wide Noun Features

Average number of noun phrases per sentence
Average number of proper nouns per sentence
Average number of common and proper nouns per sentence

Table 4: Sentence Wide Noun Features

Total number of verb phrases in document
Total number of adjectives in document
Total number of conjunctions in document
Total number of prepositional phrases in document

Table 5: Document Wide Non-Noun Features

Average number of verb phrases per sentence
Average number of adjectives per sentence
Average number of conjunctions per sentence
Average number of prepositional phrases per sentence

Table 6: Sentence Wide Non-Noun Features

Length of document
Average word length in document
Total number of unique words in document
Ratio of unique words to total number of words in document

Table 7: Document Wide Statistics Features

Average number of words per sentence
Average number of syllables per word

Table 8: Sentence Wide Statistics Features

we groups the features (as described above) and ran all combinations of those groups. This gave us much more manageable output, which is recorded in the table below. The "combination of features" column gives the table ID of the feature applied.

f-score	recall	precision	combination of features
0.8218207375	0.7606606607	0.8936757446	4
0.8218207375	0.7606606607	0.8936757446	8
0.8218207375	0.7606606607	0.8936757446	4+6
0.8218207375	0.7606606607	0.8936757446	4+8
0.8218207375	0.7606606607	0.8936757446	6+8
0.8218207375	0.7606606607	0.8936757446	4+6+8
0.7409439859	0.6349758162	0.8893661846	6
0.607092714	0.5114919282	0.7466449541	3
0.607092714	0.5114919282	0.7466449541	3+4
0.607092714	0.5114919282	0.7466449541	3+6
0.607092714	0.5114919282	0.7466449541	3+8
0.607092714	0.5114919282	0.7466449541	3+4+6
0.607092714	0.5114919282	0.7466449541	3+4+8
0.607092714	0.5114919282	0.7466449541	3+6+8
0.607092714	0.5114919282	0.7466449541	3+4+6+8
0.6021581601	0.5110111293	0.7328788591	3+4+5
0.6021581601	0.5110111293	0.7328788591	3+5+8
0.6011726778	0.5111912539	0.7295988678	4+5
0.6011726778	0.5111912539	0.7295988678	4+5+6
0.6011726778	0.5111912539	0.7295988678	4+5+8
0.6011726778	0.5111912539	0.7295988678	4+5+6+8
0.6005046017	0.5106208038	0.7287928993	3+5
0.6005046017	0.5106208038	0.7287928993	3+5+6
0.6005046017	0.5106208038	0.7287928993	3+4+5+6
0.6005046017	0.5106208038	0.7287928993	3+4+5+8
0.6005046017	0.5106208038	0.7287928993	3+5+6+8

0.6005046017	0.5106208038	0.7287928993	3+4+5+6+8
0.6000582208	0.511368158	0.7259677529	5
0.6000582208	0.511368158	0.7259677529	5+6
0.6000582208	0.511368158	0.7259677529	5+8
0.6000582208	0.511368158	0.7259677529	5+6+8
0.5980456604	0.5120879049	0.7186817211	4+7+8
0.5980456604	0.5120879049	0.7186817211	3+4+5+7
0.5980456604	0.5120879049	0.7186817211	3+5+6+7
0.5980456604	0.5120879049	0.7186817211	4+5+6+7
0.5980456604	0.5120879049	0.7186817211	5+6+7+8
0.5976007315	0.5124224422	0.7167424626	3+5+7
0.5976007315	0.5124224422	0.7167424626	5+6+7
0.5976007315	0.5124224422	0.7167424626	3+5+7+8
0.5976007315	0.5124224422	0.7167424626	3+4+5+6+7
0.5976007315	0.5124224422	0.7167424626	3+4+5+7+8
0.5976007315	0.5124224422	0.7167424626	3+5+6+7+8
0.5976007315	0.5124224422	0.7167424626	3+4+5+6+7+8
0.5968403761	0.512223071	0.714946885	7
0.5968403761	0.512223071	0.714946885	3+7
0.5968403761	0.512223071	0.714946885	4+7
0.5968403761	0.512223071	0.714946885	5+7
0.5968403761	0.512223071	0.714946885	6+7
0.5968403761	0.512223071	0.714946885	7+8
0.5968403761	0.512223071	0.714946885	3+4+7
0.5968403761	0.512223071	0.714946885	3+6+7
0.5968403761	0.512223071	0.714946885	3+7+8
0.5968403761	0.512223071	0.714946885	4+5+7
0.5968403761	0.512223071	0.714946885	4+6+7
0.5968403761	0.512223071	0.714946885	5+7+8
0.5968403761	0.512223071	0.714946885	6+7+8
0.5968403761	0.512223071	0.714946885	3+4+6+7
0.5968403761	0.512223071	0.714946885	3+4+7+8
0.5968403761	0.512223071	0.714946885	3+6+7+8
0.5968403761	0.512223071	0.714946885	4+5+7+8
0.5968403761	0.512223071	0.714946885	4+6+7+8
0.5968403761	0.512223071	0.714946885	3+4+6+7+8
0.5968403761	0.512223071	0.714946885	4+5+6+7+8

Table 9: Results of clustering

These results were less than stellar. We calculated the baseline to be 0.822113773588, which is slightly better than our results. However, when we compare the

f-score	recall	precision	# in cluster 1	# in cluster 2	combination of features
0.821	0.893	0.760	2	3330	flesch_kincaid_score
0.821	0.893	0.760	3330	2	flesch_kincaid_grade

0.821	0.893	0.760	2	3330	flesch_kincaid_score+flesch_kincaid_grade

Table 10: Using Kincaid as a Metric

6 Conclusion

Our best results are barely the same as the baseline; this can tell us a few interesting things. Either our features are totally useless (which seems unlikely, since the Kincaid readability tests have been used for years), or our corpus is not divided well or is difficult to assign features to. Given that Project Gutenberg only houses books who's copyright has expired, this leads to the interesting proposition that language has shifted so much our POS tag failed to achieve good results; however, we can't test this (at least not easily). We have no annotated corpus' to compare our results with, and therefore no way to verify the accuracy of the POS tagger.

As for the possibility that the gold standard we are using, bookshelves as defined by the Project Gutenberg curators, being not properly categorized raises another point of failure. In figures 1 and 2, we have 2 sample books from the Project Gutenberg collection. When presented during our proposal, it was difficult for the class to distinguish them with ease. This suggests that the categorization of these books may not be obvious or consistent.

References

[1] (Dec. 12, 2015). Project gutenberg, [Online]. Available: `http://www.gutenberg.org/` (visited on 12/12/2015).

[2] J. P. Kincaid, R. P. Fishburne Jr, R. L. Rogers, and B. S. Chissom, "Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel," DTIC Document, Tech. Rep., 1975.

[3] L. Feng, N. Elhadad, and M. Huenerfauth, "Cognitively motivated features for readability assessment," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2009, pp. 229–237.

[4] E. Loper and S. Bird, "Nltk: The natural language toolkit," in *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, Association for Computational Linguistics, 2002, pp. 63–70.

The train from 'Frisco was very late. It should have arrived at Hugson's Siding at midnight, but it was already five o'clock and the gray dawn was breaking in the east when the little train slowly rumbled up to the open shed that served for the station-house. As it came to a stop the conductor called out in a loud voice: "Hugson's Siding!" At once a little girl rose from her seat and walked to the door of the car, carrying a wicker suit-case in one hand and a round bird-cage covered up with newspapers in the other, while a parasol was tucked under her arm. The conductor helped her off the car and then the engineer started his train again, so that it puffed and groaned and moved slowly away up the track. The reason he was so late was because all through the night there were times when the solid earth shook and trembled under him, and the engineer was afraid that at any moment the rails might spread apart and an accident happen to his passengers. So he moved the cars slowly and with caution.

Figure 1: Dorothy and the Wizard in Oz, L. Frank Baum

Steve Tolman had done a wrong thing and he knew it. While his father, mother, and sister Doris had been absent in New York for a week-end visit and Havens, the chauffeur, was ill at the hospital, the boy had taken the big six-cylinder car from the garage without anybody's permission and carried a crowd of his friends to Torrington to a football game. And that was not the worst of it, either. At the foot of the long hill leading into the village the mighty leviathan so unceremoniously borrowed had come to a halt, refusing to move another inch, and Stephen now sat helplessly in it, awaiting theaid his comrades had promised to send back from the town.

Figure 2: Steve and the Steam Engine, Sara Ware Bassett