The Reddit Ngram Viewer

Randal S. Olson[1,*]

[1]University of Pennsylvania, Philadelphia, PA, 19104, USA
[*]E-mail: rso@randalolson.com

Introduction

When the Google Ngram Viewer [1] was released to the public in late 2010, it took the online world by storm. It suddenly become possible to analyze phrase usage in over two centuries of English literature and observe long-term trends in how the English language was used. In this paper, my aim is to introduce a similar tool with a different text corpus: The Reddit Ngram Viewer.

For the uninitiated, Reddit.com is one of the most popular link aggregation web sites on the web. Dubbed "the front page of the Internet," Reddit serves over 200,000,000 unique visitors (called "Redditors") every month [2], who share, vote on, and discuss links with each other in distinct communities called "subreddits." Links that receive large numbers of upvotes (i.e., positive votes) rise to the top of Reddit's link sharing page, whereas links that receive large numbers of downvotes (i.e., negative votes) eventually fade away into obscurity.

Reddit's comments present a unique text corpus to analyze because they are quite different from published literature: Redditors range from the pre-teens to the retired, hold a wide variety of professions, and hail from all over the world. More importantly, comments on Reddit are more conversational in nature than published literature, which allows us to analyze what the average Internet citizen is discussing on a day-to-day basis.

In the following section, I describe the methods I used to create the data set underlying the Reddit Ngram Viewer for those seeking to perform research on Reddit's comment corpus. I have purposely excluded any trends or interpretations from this paper and instead leave said interpretations to future research.

Methods

The Reddit Ngram Viewer data set was built on top of the full Reddit comment corpus compiled by Jason Baumgartner [3]. The only modifications to the raw data prior to parsing them into n-grams were:

- Sorting the comments by their time of posting in ascending order, since the comments for some months (e.g., December 2008) are not completely sorted by the date.
- Removing all comments that had been deleted, as indicated by a comment with the body text of "[deleted]". Once comments are deleted on Reddit, they are no longer available in the archive.
- Removing all URLs from the comments.

With the data adequately preprocessed, the process of generating n-grams from the data set proceeded as follows. The "body" field was used as the representative text for each comment.

1. Group all of the comments into day-by-day bins based on their time of posting in UTC ("created_utc" in the data set).
2. Replace the following special characters with unique text identifiers (e.g., "aaaaaaaaaa"): ', -, ;, #, and &. This allows the special characters to be retained in the n-grams.

3. Using scikit-learn's CountVectorizer module [4], compute all 1-, 2-, and 3-grams from the comments in each day bin.

4. For all of the computed n-grams, replace the unique text identifiers with their corresponding special character.

This process resulted in millions of n-gram/n-gram count pairs, which I saved to an output file for future processing. However, due to limited storage space, the following n-grams were pruned from the data set:

- Any n-gram with more than 40 characters. Based on visual inspection of a subset of the data set, the large majority of n-grams with more than 40 characters were gibberish.
- Any n-grams that begin with a disallowed special character (i.e., any special character other than ', -, ;, #, and &). Some special characters were maintained by the CountVectorizer module, but they were not useful for this analysis.
- Any 1-ngram with a daily proportion < 0.000001 and any 2- or 3-gram with a proportion < 0.00001. "Daily proportion" here means the proportion of that ngram's type for the day, e.g., if "hahaha" appeared 100 times on January 4, 2015 and there were 20,000 1-grams on January 4, 2015, then the daily proportion of "hahaha" for that day would be 0.005. Generally, this means that any n-gram that wasn't mentioned at least 50 times in a day was excluded from this analysis.

Once the n-grams were filtered to a more reasonable subset of about 625,000 n-grams, I sorted the n-grams into separate csv files filed in directories based on the first 5 characters of the n-gram. e.g., "data are" was filed into "d/a/t/a/_/data_are.csv". Note that spaces in the n-grams were replaced with underscores in the file names. Similarly, all allowed special characters were replaced with dashes (-) in the file names. In each n-gram's file, the following fields are stored for every day in the date range:

- Date in the format MM/DD/YYYY
- Daily proportion, as defined above

Although far from ideal for storage purposes, this file structure allows for fast look-up of specific n-grams without the need for an advanced database structure to handle the millions of n-gram/n-gram count pairs.

Acknowledgments

Thank you to Ritchie King and the FiveThirtyEight data team for supporting the development of the Reddit Ngram Viewer. I gratefully acknowledge the support of the Michigan State University High Performance Computing Center and the Institute for Cyber Enabled Research (iCER), who provided the computational resources to process the full Reddit comment corpus.

References

[1] Google. Google Ngram Viewer, September 2015. `https://books.google.com/ngrams`.

[2] Reddit. About Reddit, September 2015. `https://www.reddit.com/about`.

[3] Jason Baumgartner. Complete Public Reddit Comments Corpus, September 2015. `https://archive.org/details/2015_reddit_comments_corpus`.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.