CS 544: Data Viz

Final Project

Adam King

December 6, 2017

1 Reason for Viz

The data for this visualization is historic word probability data for English. The two main measures of word probability are the word's frequency and the word's average contextual probability. Recent work in cognitive science and linguistics have shown that these types of measures correlate with people's identification and processing speed of the words, i.e. more frequent or probable words are identified and processed faster. These measures also have been shown to affect the overall number of sounds in the word (i.e. more frequent or predictable words tend to have fewer phonemes) and the exact stored representation of words (i.e. more frequent or more predictable words are pronounced with a shorter duration in milliseconds compared to what we might expect given the sounds in the word). These types of changes to words help drive linguistic change and evolution over time. Because of that, studying factors like word frequency or average word probability is useful for studying the historic evolution of language.

Interestingly, frequency and average contextual probability of words have been found to have *independent* effects on the shape of words. However, there has been little work done to tease apart any differences in these effects. Simply put, the question that this viz seeks to help answer is: do frequency and contextual probability pattern differently for words across time. Specifically, is one of these factors more *stable* than the other. To predict future steps in a languages evolution, we would need to make predictions based off of the current state of the language and its past. Knowing which variable, frequency or contextual probability, is more stable would let us decide the best factors for this predictive model. To capture this, it is helpful to have a visualization that shows the change over time for either feature.

This visualization is challenging because there are 3 dimensions to graph: frequency, average contextual probability and time. As well, there are a **lot** of words so we need someway to prevent the graph from being too crowded and make it impossible for the viewer to see anything interesting. Because of this, I opted to incorporate some elements of interactivity and filtering so a viewer could (theoretically) explore the entirety of the data while not being overwhelmed.

2 Data

The data for this project was a 200 year history for English provided by the Google n-grams corpus[1]. The Google n-grams offers n-gram counts for several languages from Google's collection of scanned books. For each n-gram, they provide the n-gram, a year and the frequency of that n-gram for

[1] http://storage.googleapis.com/books/ngrams/books/datasetsv2.html

books published in that year. Using this, we can chart the change in time for specific terms or n-grams.

As I mentioned before, I was interested in two different measures for a word's probability. The frequency is the number of times a word appears for a given year. The average probability is a little more complicated. Following the standard for many linguistic and cognitive science papers, I *estimate* the average contextual probability of a word by calculating the trigram average. This is definitely a simplification of the problem, but has been shown to work. To do this, I find each trigram for which a target work is the final word and calculate the probability of that word given the previous two. I then average the probabilities for each trigram context to determine, on average, how likely a word is given its context. This measure highly correlated with frequency (more frequent words tend to be more probable in all contexts), though slightly different. The name of the game here is to see *how* different the two are. To make the frequency factor more similar to the trigram probability, I converted it into a unigram probability which is the frequency of the term divided by the sum of the frequencies of all words for the year. To semi-normalize the data, I log-transformed the data.

All of this work was done in pre-processing, before any visualization. I did all the preprocessing in `python` and ended up with a large JSON file. To reduce the overall size of the data, I aggregated all years into decades[2] and limited the words for the project to the ≈15,000 most frequent words. The eventual JSON file had entries for each word, and for each word a value for unigram and trigram probability for each decade. Though Google n-grams go back to the 1600s, the data was very sparse before 1800 and so I opted to limit the graph to nothing before 1800.

3 Viz

On to the viz! The visualization is presented as a 2-d scatter plot where the x-axis is a measure for contextual probability and y-axis is a measure of frequency. Because word length is correlated with frequency and probability, the radii of the points are determined by the length of the word. This allows views to see right-before-their-eyes the fact that more frequent/probable words tend to be shorter.

Below the actual chart is a time line which allows the viewer to scroll throw time and see the change for the two factors over time. This lets the user move through different periods of time and see how either factor changes. The viz also shows the path for the next and previous 50 years so a viewer can see the path of a word after the interaction has stopped.

To avoid bombarding the viewer with every word, s/he can select a subset of the English lexicon using a regular expression match. The regular expression matching allows for a power means of subsetting the lexicon.

In terms of the actual factors, I have two conditions for each. The viewer can either see the actual values for the -log unigram or trigram probabilities plotted on the graph or see the ordinal ranks for either factor. The rank is simply the order of the word in a sorted list for either measure. This view is useful as it forces the lexicon to take advantage of the entire space for the chart; even though the data is non-parametric, the log transformation is likely to push a lot of mass towards the mean which can make the visualization *crowded*.

Enough talk, go look at the graph!

[2] This **still** yielded 20GB of data....