cs545, Fall 2012, Homework 2

Total points: 100
Due date: 5pm, Friday, 28th of September

Submission instructions TBD
Late policy: No late homeworks accepted.

You may discuss problems at a high level with classmates, but you must solve and write them up yourselves.

This homework will give you practice coding up simple analyses of text data. Your goal will be to analyze the Zipfian distributions of word and morpheme frequency across 11 languages: Bulgarian, Czech, English, Estonian, Farsi, Hungarian, Polish, Romanian, Slovene, Slovak, and Serbian. Corresponding to each language is a data file named `orwell-xy.txt` where 'xy' is a two-letter code for for the language name. The data consists of translations of Orwell's novel 1984, and the files use a tab-delimited format with four columns:

1. Token ID
2. Word token
3. Morphological analysis of token
4. Lemma (morphological root of token)

Notes: Please do no further lowercasing, filtering, or tokenization of the data. The files are encoded in UTF-8, so make sure to use the appropriate read and write methods.[1] For linear regression and computing means and standard deviations, I recommend using the Python numpy library,[2] and the Python matplotlib library for making plots.[3]

Question 1

In class we studied the morphological analysis of English using FSA's. Our first goal is to assess the morphological complexity of these 11 languages. For each language, find the lemma which has the greatest number of unique word types. List these lemmas along with the number of words (types, not tokens) that they take. Which language has the least morphological complexity by this measure?

Let's continue this analysis. Now for each language, compute the ratio of unique words types to unique lemma types. This ratio will tell us the average number of word-forms that each lemma can take, giving another measure of morphological complexity. List the languages along with their ratios. Are these results roughly consistent with the previous analysis?

[1] e.g. in Python: `codecs.read(..,'r','utf-8')`

[2] See here for linear regression:
`http://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.lstsq.html#numpy.linalg.lstsq`
The numpy functions `numpy.average` and `numpy.std` compute means and standard deviations of lists.

[3] The functions I used are:
`matplotlib.pyplot.plot`, `matplotlib.pyplot.clf`, and `matplotlib.pyplot.savefig`.

Question 2

In this question we will analyze the distribution over word frequencies found in our text and compare it to the Zipf power law. Recall from class that under the Zipf power law, the frequency of a word is inversely proportional to it *rank* (where the rank of the most frequent word is 1, that of the second most frequent word is 2, that of the third most frequent word is 3, etc):

$$freq(w) = Z \cdot \frac{1}{rank(w)^k}$$

In this equation, $freq(w) = \frac{count(w)}{\sum_{w'} count(w')}$, Z is a normalization constant, and k is a parameter. In the classical Zipfian distribution $k = 1$.

Write a program that computes the empirical rank and frequency of all the words in each language.

- For each language, create a plot showing the relationship between these two values (rank on the x-axis, frequency on the y-axis).
- Do the plots look similar? Do they appear to be power laws? If you've done this correctly, it should be very hard to see the graphs and compare them. It will appear that the frequency drops down to nearly zero very quickly in both cases.
- However, recall from lecture that what distinguishes a power law from an exponential function is that it has a "fat tail." Let's try zooming in on the tail to see this. Again, plot the empirical frequencies, but this time begin at $rank(w) = 300$. Do you see the fat tail now?
- An even better way to do this is to plot the log-log graphs (i.e., plot $\log(freq(w))$ on the y-axis and $\log(rank(w))$ on the x-axis). Do so for each language and show the results. If the frequencies obey a power law, the log-log plot will be linear, and the negative of the slope will give the value of k in the Zipf equation above. Do the plots look linear?

Question 3

Finally, let's estimate the exponent k for each language. To do so, run a least-squares linear regression on each language's log-log plot and report the negative of the resulting slopes. Compute and report the mean and standard deviation of the slopes over the 11 languages. How close are the estimated values of k to the classical Zipf law (in which $k = 1$)? Are the results similar across languages? How do the cross-language differences in k compare to the cross-language differences in morphological complexity that we observed in Question 1?

Now, let's see how much our estimates of k were skewed by differences in morphological complexity. To do so, recompute the log-log data but this time use lemma frequency instead of word frequency (no need to report all the new graphs). Re-estimate the values of k using linear regression, and report the new mean and standard deviation. Are the new values closer to the classical Zip law of $k = 1$? Do the languages now appear to have more similar frequency distributions?