nlp-communicating-exercises

March 26, 2019

1 22. Natural Language Processing

22.1 This exercise explores the quality of the n-gram model of language. Find or create a monolingual corpus of 100,000 words or more. Segment it into words, and compute the frequency of each word. How many distinct words are there? Also count frequencies of bigrams (two consecutive words) and trigrams (three consecutive words). Now use those frequencies to generate language: from the unigram, bigram, and trigram models, in turn, generate a 100-word text by making random choices according to the frequency counts. Compare the three generated texts with actual language. Finally, calculate the perplexity of each model.

22.2 Write a program to do **segmentation** of words without spaces. Given a string, such as the URL "thelongestlistofthelongeststuffatthelongestdomainnameatlonglast.com," return a list of component words: ["the," "longest," "list," ...]. This task is useful for parsing URLs, for spelling correction when words runtogether, and for languages such as Chinese that do not have spaces between words. It can be solved with a unigram or bigram word model and a dynamic programming algorithm similar to the Viterbi algorithm.

22.3 *Zipf's law* of word distribution states the following: Take a large corpus of text, count the frequency of every word in the corpus, and then rank these frequencies in decreasing order. Let f_I be the Ith largest frequency in this list; that is, f_1 is the frequency of the most common word (usually "the"), f_2 is the frequency of the second most common word, and so on. Zipf's law states that f_I is approximately equal to α / I for some constant α. The law tends to be highly accurate except for very small and very large values of I.

22.4 Choose a corpus of at least 20,000 words of online text, and verify Zipf's law experimentally. Define an error measure and find the value of α where Zipf's law best matches your experimental data. Create a log–log graph plotting f_I vs. I and α / I vs. I. (On a log–log graph, the function α / I is a straight line.) In carrying out the experiment, be sure to eliminate any formatting tokens (e.g., HTML tags) and normalize upper and lower case.

22.5 (Adapted from @Jurafsky+Martin:2000.) In this exercise you will develop a classifier for authorship: given a text, the classifier predicts which of two candidate authors wrote the text. Obtain samples of text from two different authors. Separate them into training and test sets. Now train a language model on the training set. You can choose what features to use; n-grams of words or letters are the easiest, but you can add additional features that you think may help. Then compute the probability of the text under each language model and chose the most probable model. Assess the accuracy of this technique. How does accuracy change as you alter the set of features? This subfield of linguistics is called **stylometry**; its successes include the identification of the author of the disputed *Federalist Papers* @Mosteller+Wallace:1964 and some disputed works of Shakespeare @Hope:1994. @Khmelev+Tweedie:2001 produce good results with a simple letter bigram model.

22.6 This exercise concerns the classification of spam email. Create a corpus of spam email and one of non-spam mail. Examine each corpus and decide what features appear to be useful for classification: unigram words? bigrams? message length, sender, time of arrival? Then train a classification algorithm (decision tree, naive Bayes, SVM, logistic regression, or some other algorithm of your choosing) on a training set and report its accuracy on a test set.

22.7 Create a test set of ten queries, and pose them to three major Web search engines. Evaluate each one for precision at 1, 3, and 10 documents. Can you explain the differences between engines?

22.8 Try to ascertain which of the search engines from the previous exercise are using case folding, stemming, synonyms, and spelling correction.

22.9 Estimate how much storage space is necessary for the index to a 100 billion-page corpus of Web pages. Show the assumptions you made.

22.10 Write a regular expression or a short program to extract company names. Test it on a corpus of business news articles. Report your recall and precision.

22.11 Consider the problem of trying to evaluate the quality of an IR system that returns a ranked list of answers (like most Web search engines). The appropriate measure of quality depends on the presumed model of what the searcher is trying to achieve, and what strategy she employs. For each of the following models, propose a corresponding numeric measure.

1. The searcher will look at the first twenty answers returned, with the objective of getting as much relevant information as possible.

2. The searcher needs only one relevant document, and will go down the list until she finds the first one.

3. The searcher has a fairly narrow query and is able to examine all the answers retrieved. She wants to be sure that she has seen everything in the document collection that is relevant to her query. (E.g., a lawyer wants to be sure that she has found *all* relevant precedents, and is willing to spend considerable resources on that.)

4. The searcher needs just one document relevant to the query, and can afford to pay a research assistant for an hour's work looking through the results. The assistant can look through 100 retrieved documents in an hour. The assistant will charge the searcher for the full hour regardless of whether he finds it immediately or at the end of the hour.

5. The searcher will look through all the answers. Examining a document has cost \$$A$; finding a relevant document has value \$$B$; failing to find a relevant document has cost \$$C$ for each relevant document not found.

6. The searcher wants to collect as many relevant documents as possible, but needs steady encouragement. She looks through the documents in order. If the documents she has looked at so far are mostly good, she will continue; otherwise, she will stop.