6. *Word embeddings.*

The large number of English words can make language-based applications daunting. To cope with this, it is helpful to have a *clustering* or *embedding* of these words, so that words with similar meanings are clustered together, or have embeddings that are close to one another.

But how can we get at the meanings of words? John Firth (1957) put it thus:

> *You shall know a word by the company it keeps.*

That is, words that tend to appear in similar contexts are likely to be related. In this mini-project , you will investigate this idea by coming up with an embedding of words that is based on co-occurrence statistics.

The description here assumes you are using Python with NLTK.

- First, download the Brown corpus (using `nltk.corpus`). This is a collection of text samples from a wide range of sources, with a total of over a million words. Calling `brown.words()` returns this text in one long list, which is useful.
- Remove stopwords and punctuation, make everything lowercase, and count how often each word occurs. Use this to come up with two lists:
  - A *vocabulary* V, consisting of a few thousand (e.g., 5000) of the most commonly-occurring words.
  - A shorter list C of at most 1000 of the most commonly-occurring words, which we shall call *context words.*
- For each word $w \in V$, and each occurrence of it in the text stream, look at the surrounding window of four words (two before, two after):

$$w_1 \;\; w_2 \;\; w \;\; w_3 \;\; w_4.$$

  Keep count of how often context words from C appear in these positions around word w. That is, for $w \in V, c \in C$, define

$$n(w,c) = \# \text{ of times } c \text{ occurs in a window around } w.$$

  Using these counts, construct the probability distribution $\Pr(c|w)$ of context words around w (for each $w \in V$), as well as the overall distribution $\Pr(c)$ of context words. These are distributions over C.
- Represent each vocabulary item w by a $|C|$-dimensional vector $\Phi(w)$, whose c'th coordinate is:

$$\Phi_c(w) = \max\left(0,\ \log \frac{\Pr(c|w)}{\Pr(c)}\right).$$

  This is known as the (positive) *pointwise mutual information*, and has been quite successful in work on word embeddings.

Now, use this matrix to come up with a 100-dimensional representation of words.

(a) *Give a description of your 100-dimensional embedding.* The description should be concise and clear, and should make it obvious exactly what steps you took to obtain your word embeddings. Below, we will denote these as $\Psi(w) \in \mathbb{R}^{100}$, for $w \in V$. Also clarify exactly how you selected the vocabulary V and the context words C.

(b) *Investigate your embedding using nearest neighbor.* Pick a collection of 25 words $w \in V$. For each w, return its nearest neighbor $w' \neq w$ in V. A popular distance measure to use for this is *cosine distance*:

$$1 - \frac{\Psi(w) \cdot \Psi(w')}{\|\Psi(w)\|\|\Psi(w')\|}.$$

Do the results make any sense?

(c) *Investigate your embedding by clustering the vocabulary.* Using the vectorial representation $\Psi(\cdot)$, cluster the words in V into 100 groups. Clearly specify what algorithm and distance function you use for this, and the reasons for your choices. Look over the resulting 100 clusters. Do any of them seem even moderately coherent? Pick out a few of the best clusters and list the words in them.

Note: The Brown corpus is very small. Current work on word embeddings uses data sets that are several orders of magnitude larger.