Informatics 1 CG: Practical 6
Vector Space Models

Peter Bell

February 19, 2015

1 Introduction

In this practical you will use Matlab to construct vector space models for semantic processing, as described in your lectures. You will build models using data from two recent BBC news articles.

2 Getting started

The files needed for the practical have been packaged into a single file called `docs_material.tar.gz`. Download this file to your working directory, and unpackage its contents using the command `tar -zxvf docs_material.tar.gz` at a shell prompt.

The creates several text files, described below (where i={1,2}):

`doc_i.txt`	The text of the original news article.
`doc_keywords_i.txt`	The text of the article, after uninteresting words such as *and, but, to* have been removed. We call the remaining words "keywords".

To make it easier to process the data in Matlab, each unique keyword has been assigned an ID number, and there are three further files per article:

`keyword_mapping_i.txt`	A list showing the mapping of ID number to keywords.
`keyword_list_i.txt`	A list containing just the unique keywords, ordered by their ID number.
`doc_numbers_i.txt`	The article again, with uninteresting words removed, and all keywords replaced by numbers

Have a look at the contents of the files so that you understand the processing that has been done. The two most useful files for the exercise have been converted into Matlab variables: they are contained in `lab2_data.mat`. Open Matlab and load this file using the command `load lab2_data.mat`. This creates the following variables:

- `doc_numbers_1`, `doc_numbers_2` – each article in number form, as a Matlab column vector
- `keyword_list_1`, `keyword_list_2` – unique word lists for each article, imported as a Matlab *cell array.* (These are similar to vectors but do not have to contain numbers.) Elements are accessed using curly brackets instead: for example, `keyword_list_1{j}` for the jth unique keyword from article 1.

3 Constructing the Vector Spaces

You will now use the data to construct co-occurrence matrices for the most commonly-occurring words in each article. For each article, separately, you should:

1. Find the four most commonly-occurring words. To help you do this, write a Matlab function to count how many times each word appears. (If two words appear equally often, choose the one with the lowest ID number).

   [*Harder (optional)*: extend your function to *automatically* obtain the four words.]

2. For each of the four words, construct a *co-occurrence vector*, described in your lectures using a symmetric $(-5, +5)$ context – you should write another Matlab function to do this. The vector should be in column form, with the jth row corresponding to the word with ID number j. You can combine these vectors into a co-occurrence *matrix* for convenience.

3. For each of the four words, using your co-occurrence matrix, state the most frequently co-occurring words.

4 Measuring word similarity

Your vector space models can be used to measure how similar pairs of words.

4. For each article, calculate the *cosine distance* between the four co-occurrence vectors (once for each pair). A large cosine distance indicates that words are closely related. For each article, which of the pairs of words are most closely related, according to this measure?
5. Why would the Euclidean distance measure be less appropriate in this case for determining similar words?