DD2475 Information Retrieval

Answer Suggestions to Written Exam, December 13, 2010, 14.00 – 19.00

These are only suggestions to answers, indicating solutions that would give full credit. However, other correct and exhaustive solutions will also receive the same credit.

Part 1: Theory (20 credits)

1.1 (4 credits)

If we represent the term-document matrix as a 2D-array, then the vast majority of array elements would be zero; the matrix is sparse. Thus, it is more economical to represent the matrix using linked lists as we did in the lab course. For realistic document collections, such a linked-list representation would fit into memory whereas the corresponding 2D array would not.

1.2 (4 credits)

Stemming is the process of chopping off the ends of words, leaving the stem; e.g., ”organiz|e”, ”organiz|ing”, ”organiz|ation” ⇒ ”organiz”. Lemmatization is the process of extracting the base form of words with the use of a vocabulary and morphological analysis; e.g., ”am”, ”are”, ”is”, ”was” ⇒ ”be”.

Stemming is faster and less complicated than lemmatization. On the other hand, stemming does not work for irregularly inflected words like "are" and "is". There is also a risk that stemming might lower precision; for instance, "stocking" is rewritten into "stock" which has a completely different meaning. Lemmatization is of greater benefit in languages with complex morphology (like Finnish), and languages with lots of compound words (like Swedish and German).

1.3 (4 credits)

In Boolean retrieval, documents are classified either relevant or non-relevant to a query, whereas in ranked retrieval, documents are ranked by their similarity to the query, according to some measure.

A non-specialist user might not be able to express their search need in terms of a Boolean expression. Furthermore, it is difficult to pose a Boolean query that returns a reasonable number of hits – they are most often too few or far to many. On the other hand, if the results are ranked according to similarity to the query, limiting the number of hits is not as critical, as the user might look at the most similar (=most relevant) hits first.

1.4 (4 credits)

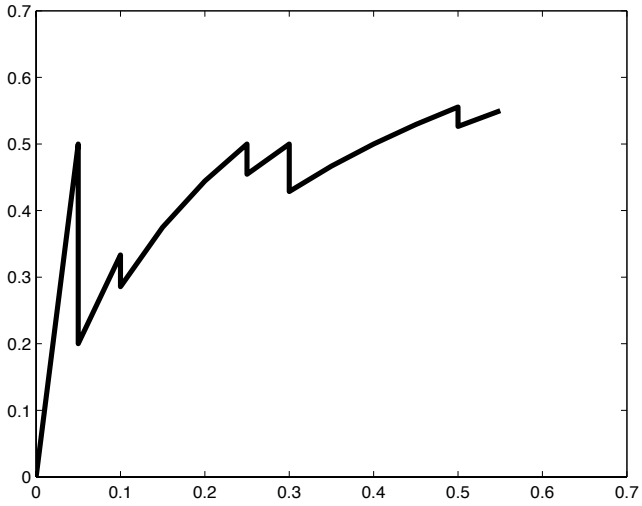
If we study the positions of the query terms in the returned documents, we can make the following table:

doc 1: 4-2 = 2 ("brzezynski" occurs two positions after "zbigniew")
doc 2: 234-233 = 1
doc 4: 8525-8526 = -1
doc 1: 8-7 = 1 ("advisor" occurs one position after "political")
doc 8: 4360-4362 = -2

On the other hand, document 7 is not returned, even though "advisor" occurs three positions after "political". A probable interpretation is that NEAR means "no more than two words before or after".

1.5 (4 credits)

(a)



(b) The *precision-at-10*, the precision among the 10 highest ranked documents, is 0.5, since there are 5 relevant documents among those 10.

Part 2: Problems (30 credits)

2.1 (6 credits)

Since the query is identical to the second document, the second document would receive the maximal cosine score of 1, whereas the Wikipedia article would receive a lower score. However, to a human it is apparent that the Wikipedia article is more relevant to the query than Document 2, implying that tf-idf is a quite inaccurate measure of relevance.

Suggested improvements could include checking whether a document contains full phrases, whether the search terms occur in headlines, early in document, in boldface, italics or capital letters, etc. If the text has been down loaded from the web it is also possible to use PageRank or some similar relevance measure.

2.2 (6 credits)

Only terms that are not stop words are stored in the index. The positions of a non stop word term in a document can be stored in the corresponding postings list along with the docID, In this way, the document can be recreated, but with empty spaces on original stop word locations.

There is a potential problem when posing a phrase query involving stop words. Comparing this query with the positional index will give back all occurrences of this phrase, but also of phrases where the stop words are replaced by any other word. For example, the phrase query ”angels fear to thread” (where ”to” is a stop word) would return documents containing ”angels fear * thread” where ”*” signifies any stop word – but also documents containing, e.g., ”angels fear woolen thread”, which is not relevant to the query. The problem could be solved by linearly searching through all returned documents to see if they contain the original phrase string.

2.3 (6 credits)

The algorithm can simply ignore the new term, and make a vector space comparison with the part of the query containing known terms. To realize this, one needs to study the way in which query vectors q and document vectors d are compared – with the cosine score. In the case of no extra query terms, the cosine score = length-normalized dot product:

$$\mathrm{CosineScore}(q,d) = \frac{q_1 d_1 + \ldots + q_n d_n}{\sqrt{q_1^2 + \ldots + q_n^2}\sqrt{d_1^2 + \ldots + d_n^2}}$$

Note that the query length is unimportant for the ranking of documents, since this is the same for all documents.

In this case, the query $q = [q_1, \ldots, q_n, q_{n+1}]$ contains an extra term which does not exist in the document $d = [d_1, \ldots, d_n]$. One could imagine adding an extra dimension to the document vector, $d_{n-1} = 0$. The cosine score is then:

$$\text{CosineScore}(q,d) = \frac{q_1 d_1 + \ldots + q_n d_n + q_{n+1} 0}{\sqrt{q_1^2 + \ldots + q_n^2 + q_{n+1}^2}\sqrt{d_1^2 + \ldots + d_n^2 + 0^2}}$$

Apart from a scale factor due to different query length, this is equal to simply ignoring the extra query term q_{n+1}: which would give the original cosine score above.

2.4 (6 credits)

(a) The ranking according to q_m is exactly the same as the original ranking according to q_o, which implies that $\alpha > 0$, while $\beta = 0$ and $\gamma = 0$.

(b) The documents marked as relevant have been moved up to the top of the list, which indicates that $\beta > 0$, and also that $\beta >> \alpha$. However, the documents marked as non-relevant are still rated higher than than number 11, 12, etc. in the original list, which indicates that $\gamma = 0$.

(c) The documents marked as relevant have been moved up to the top of the list, which indicates that $\beta > 0$, and also that $\beta >> \alpha$. Furthermore, all the documents marked as non-relevant have fallen out of the top-10 list, which indicates that $\gamma > 0$, and also that $\gamma >> \alpha$.

2.5 (6 credits)

(a) If the index is positional, and includes all words (i.e. stop words are not removed), it is possible to completely recover the documents. (If stop words have been removed, there will be gaps in the stop word positions). If the index is not positional, it is possible to retrieve the set of words occurring in the document (possibly with term frequencies).

(b) If the language is known, it is possible to extract the set of words from each document by systematically posing one-word queries of all the words in the language (taken from a dictionary) + a list of names, places, etc. One can test if the index is positional by testing queries like "the the the", which are unlikely to appear as phrases in any document, and see if such queries produce any results. If the index **is** positional one can try to submit phrase queries about interesting topics. However, it will not be possible to extract the exact contents of any documents by systematically all possible phrases – they are just too many.