COS 435, Spring 2014 - Problem Set 1

Due at 1:30PM, Wednesday, Feb. 19, 2014

Collaboration and Reference Policy

You may discuss the general methods of solving the problems with other students in the class. However, each student must work out the details and write up his or her own solution to each problem independently.

Some problems have been used in previous offerings of COS 435. You are NOT allowed to use any solutions posted for previous offerings of COS 435 or any solutions produced by anyone else for the assigned problems. You may use other reference materials; you must give citations to all reference materials that you use.

Lateness Policy

A late penalty will be applied, unless there are extraordinary circumstances and/or prior arrangements:

- Penalized 10% of the earned score if submitted by 11:59 pm Wed. (2/19/14).
- Penalized 25% of the earned score if submitted by 4:30 pm Friday (2/21/14).
- Penalized 50% if submitted later than 4:30 pm Friday (2/21/14).

Problem 1:

Consider a collection of 25,000 documents. Each document is assigned a non-empty *ordered* list of at least 1 and at most 10 keywords by the authors of the document. The keywords come from a fixed list of 100 keywords given to the authors by the publisher. The order of the keywords in the list of keywords for a document reflects the order of importance of the keywords in the document: the first keyword is the most important keyword, although it could be tied in importance with the second keyword, etc. (Imagine the ACM giving authors of all technical papers a list of 100 keywords related to computer science topics and asking each author to choose and order by importance at least 1 and at most 10 keywords that best describe the topic of the paper.)

A query on the collection of documents is an arbitrarily long list of keywords from among the 100 keywords. Any scoring of documents with respect to a query should have the following properties:

- For a single keyword query (e.g. “security”), if the keyword appears at position j in the list of keywords for document A and appears at position k > j in the list of

keywords for document B, then A scores as a better match than B. (For example, if A has keyword list "fraud, security, privacy" and B has keyword list "privacy, copyright, security, encryption", then A scores higher than B as a match for the query "security" because "security" is in position 2 in A's list and in position 3 in B's list.) Note that the order from the beginning of the list of keywords for a document is used; the size of the list of keywords for a document is not considered in this scoring.

- For a query containing more than one keyword, if the list of keywords for document A contains more of the query keywords than the list of keywords for document B, then A scores as a better match than B.

Note that the list of keywords for a ***query*** is ***not*** ordered by importance.

Part a

The (arguably) simplest scoring of a document with respect to a query is to count the number of query terms that appear in the document. This would work well with the constrained specification of this problem (documents represented by keywords and queries restricted to keywords) if it were not for the importance of the order of keywords assigned to a document. One solution is to assign a weight for each (document, keyword) pair - the weight is positive if the keyword is in the list of keywords assigned to the document and 0 otherwise. For a single document, the values of the positive weights depend on the positions of the corresponding keywords on the ordered list of keywords for the document. Then, hopefully, calculating a score for a document by adding up the weights for all the keywords that appear in the query and ranking documents according to their scores would satisfy the two properties given above.

Propose a method of assigning weights to (document, keyword) pairs so that the scoring described in the previous paragraph satisfies the two properties above. It is important to note the (document, keyword) weights are assigned without any knowledge of specific queries, i.e the same weights must be used for all queries.

Part b

Show the weights assigned to the following 3 documents under your method of Part a. For each of the two following queries, give the score for each document:

document A has keyword list "agents, interfaces"
document B has keyword list "network, protocol, interfaces, instrumentation"
document C has keyword list "interfaces, proxy, agents"

query 1: interfaces
query 2: interfaces, agents

Part c

How well does your method of Part a capture the scoring properties given at the beginning of this problem description? Justify your answer. Be specific.

Part d
Does your method of scoring distinguish between different documents that contain the same number of the query keywords, but different query keywords, for a multi-keyword query? Explain.

Part e
Does your method scale to full-text documents and a large lexicon of terms? That is, if we consider the sequence of all terms in each document, and queries can use any terms in the lexicon, is the scoring method effective? Explain. Do you think it is practical?

Problem 2 (2012 exam problem):

PageRank is usually applied to the graph of a collection of documents without regard to the content of the documents. In this problem we will change that. For each pair of documents, there is a pre-computed real-valued similarity measure ranging between 0 and 1 (e.g. cosine similarity in the vector model). Let sim(i,j) denote this similarity between the i^{th} and j^{th} documents. We use this value to modify the PageRank calculation, which we call **pr-s** for "PageRank with similarity". For the graph of a collection of **n** documents, the new PageRank equation is:

$$\textbf{pr-s}_{\textbf{new}}\textbf{(k)} = \alpha/n + (1-\alpha)\sum_{\textbf{i with edge from i to k}} \textbf{((1+sim(i,k)) * pr-s(i))}$$

Part a: Even if the underlying graph has no sinks, PageRank with similarity is not guaranteed to converge. Why not? Be specific.

Part b: Modify the definition of PageRank with similarity so that the idea of using similarity in this way is retained but the resulting calculation does converge assuming no sinks. Give the new equation and an argument (not a proof) that it converges.

Part c: Does PageRank with similarity enhance or detract from the goals of the original PageRank? Justify your answer.