COS 435, Spring 2017 - Problem Set 1, Part 1

Due 11:59 pm Wednesday February 15 by DropBox submission
Due at 5:00 pm, Wednesday, Feb. 15, 2016 if submitting handwritten work on paper.

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. For each problem, list the students with whom you discussed general methods of solving the problem.

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 10am Thursday (2/16/15).
- Penalized 25% of the earned score if submitted by 4:30 pm Friday (2/17/15).
- Penalized 50% if submitted later than 4:30 pm Friday (2/17/15).

Submission

Submit your solutions as a PDF file using the Computer Science Department DropBox submission system for COS435 at
https://dropbox.cs.princeton.edu/COS435_S2017/HW1Part1 Name your file HW1Part1.pdf. If you have not used this facility before, consult the instructions at
https://csguide.cs.princeton.edu/academic/csdropbox - student
Note that you are automatically enrolled in CS DropBox using the registrar's COS435 enrollment list.

You may hand write your solutions as long as they are legible. In this case, you may either scan your writing to produce a PDF file for submission through DropBox or turn in your document by 5:00 PM Wed. Feb. 15 by placing it in TA Mayank Mahajan's CS Department mailbox on the second floor of the CS building. You may also hand in documents during class on Wed. Feb 15.

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
Does your method of Part a capture both 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 query keywords, but different query keywords, for a multi-keyword query? Explain.

Part e
Suppose we try to use this method on full-text documents and a large lexicon of terms. In this case we take every term appearing in the document except "stop words", and make it a keyword, putting the keywords in order of the frequency of appearance of the word in the document – most frequent first. (A "stop word" is a connecting word that gives little or no information, such as "the", "a", "of", "and".) Does your scoring method work in this case? Do you think it is practical? Explain.