DD2475 Information Retrieval

Written Exam, December 13, 2010, 14.00 – 19.00

The exam consists of two parts. Part 1 consists of 5 questions with a total maximum score of 20. Part 2 consists of 5 questions with a total maximum score of 30. The questions can be answered in English or Swedish. The limits for different grades are:
A: 42-50
B: 37-41 AND (≥10 on Part 1)
C: 33-36 AND ((≥10 on Part 1) AND (≥15 on Part 2))
D: 29-32 AND ((≥10 on Part 1) AND (≥15 on Part 2))
E: 25-28 AND ((≥10 on Part 1) AND (≥15 on Part 2))
Fx: ≥25 AND ((<10 on Part 1) OR (<15 on Part 2))
F: <25
In the event of grade F, the written exam has to be taken again. In the event of grade Fx, you will be allowed to orally complete parts of the exam to reach grade E.

Part 1: No aids allowed. Hand in your solutions to Part 1 in an assigned envelope before producing the aids to Part 2.

Part 2: All aids allowed EXCEPT I) other humans, II) phones, computers, reading pads.

Part 1: Theory (20 credits)

*We expect the answers to **all 5** questions to cover up to two (2) hand-written A4.*

1.1 (4 credits)

Term-document incidence matrices are most often not stored as 2D-arrays, but as another structure. Why?

1.2 (4 credits)

What is the difference between stemming and lemmatization? What are the advantages and disadvantages with each method?

1.3 (4 credits)

What is the difference between Boolean retrieval and ranked retrieval? Why are most publicly available search engines using ranked retrieval nowadays?

1.4 (4 credits)

Below is a piece of a positional index, where entries are in the form: docID1:<position1, position2, ...>; docID2:<position1, position2, ...>; etc.

zbigniew: 0:<1024,6555>; 1:<2,232>; 2:<0,233>; 3:<5,105,268,1323,3214>; 4:<234,8526,13283,17500,32698>
kazimierz: 1:<3>; 5:<32699>
brzeszynski: 1:<4,812>; 2:<234>; 4:<8525>; 6:<78, 378, 4798, 16788, 19234>; 7:<2, 207, 654, 891, 2024, 11026, 22704>
politial: 1:<7>; 4:<452,986>; 5:<25, 512, 765, 2187, 7453, 13818>; 6:<6526>; 7:<2064,11027>; 8:<4362,19254>
advisor: 1:<8,20560>, 7:<490,11030>; 8:<44,786,4360>

Suppose the query "zbigniew NEAR brzeszynski" returns the documents 1,2 and 4. Furthermore, suppose the query "political NEAR advisor" returns the documents 1 and 8. Describe a probable interpretation of the NEAR operator.

1.5 (4 credits)

The following 20 ranked results have been returned as a response to a query. Since we are experts in the area, we find results 2, 6, 8, 9, 10, 12, 15, 16, 17, 18, and 20 relevant.

We know that there are in total 20 relevant documents in the collection.

(a) Draw the precision-recall curve for the first 20 results shown above.

(b) What is *precision-at-10* for this list of results?

Part 2: Problems (30 credits)

*We expect the answers to **each** question to cover up to one (1) hand-written A4.*

2.1 (6 credits)

Consider the query "Michael Jackson", and the two following documents:

1) The Wikipedia article about Michael Jackson, which begins:

"Michael Joseph Jackson (August 29, 1958 – June 25, 2009) was an American recording artist, dancer, singer-songwriter, and philanthropist. Referred to as the King of Pop, Jackson is recognized as the most successful entertainer of all time by Guinness World Records. His contribution to music, dance and fashion, along with a much-publicized personal life, made him a global figure in popular culture for over four decades. The eighth child of the Jackson family, he debuted on the professional music scene along with his brothers as a member of The Jackson 5 in the mid-1960s, and began his solo career in 1971..."

2) The document whose entire contents is:

"Michael Jackson"

If the documents are represented by their tf-idf weights, which document would receive the higher cosine score with the query? In the light of this example, discuss the use of tf-idf as a measure for ranking search results. Which improvements would you suggest?

2.2 (6 credits)

How could an IR system combine use of a positional index and use of stop words? What is the potential problem, and how could it be handled?

2.3 (6 credits)

Consider an index with vector representation of documents. Consider the case of a query term that is not in the set of *M* indexed terms; thus, our standard construction of the query vector results in $V(q)$ not being in the vector space created from the collection. How would one adapt the vector space representation to handle this case?

2.4 (6 credits)

The Rocchio algorithm for relevance feedback suggests updating the query according to:

$$q_m = \alpha q_o + \beta \frac{1}{|D_r|} \sum_{d_i \in D_r} d_i - \gamma \frac{1}{|D_{nr}|} \sum_{d_j \in D_{nr}} d_j$$

As a response to the initial query q_o, the system returns the following top-10 list of ranked documents:

docA, *docB*, *docC*, *docD*, *docE*, *docF*, *docG*, *docH*, *docI*, *docJ*

The user then tags the returned documents with a ”+” if it is relevant or with a ”–” if it is not relevant. This is how the user labeled the initial ranked top-10 list:

– *docA*
+ *docB*
+ *docC*
– *docD*
+ *docE*
+ *docF*
– *docG*
– *docH*
+ *docI*
– *docJ*

A new query q_m is formulated according to the Rocchio formula, and used to retrieve a new list. What are the approximative values of α, β and γ in the three following cases? Motivate your answers.

(a) After one iteration of the Rocchio algorithm, the updated top-10 list looks like this:

docA, *docB*, *docC*, *docD*, *docE*, *docF*, *docG*, *docH*, *docI*, *docJ*

(b) After one iteration of the Rocchio algorithm, the updated top-10 list looks like this:

docB, *docC*, *docE*, *docF*, *docI*, *docA*, *docD*, *docG*, *docH*, *docJ*

(c) After one iteration of the Rocchio algorithm, the updated top-10 list looks like this:

docB, *docC*, *docE*, *docF*, *docI*, *docK*, *docL*, *docM*, *docN*, *docO*

2.5 (6 credits)

You are an investigating reporter, and someone has given you a memory stick with an index of a set of secret documents (but not the documents themselves).

(a) Assuming that the index is **not** encrypted (i.e., that you can access the data structure), what information can you obtain about the original documents from the index?

(b) Assume that the index is encrypted, but there is a query interface in which you can pose queries and get docIDs of matching documents. In this case, what can learn about the index and the documents?