Information Retrieval – Exercise 2 - Due Feb 23
IST 441

This exercise is worth 6 points.

In this exercise, you will familiarize yourself with vector space models of documents and queries, similarity measures and ranking. To get partial credit in case of miscalculations, please give detail to your solutions.

Given the following documents and queries:

D1: You say goodbye, I say hello
D2: You say stop, I say go
D3: Hello, hello, you say goodbye
D4: I say yes, you say no

Q1: say goodbye
Q2: you hello

Specify the vocabulary of tokens/terms using full text indexing and no stemming (ignore capitalization and punctuation), and define an alphabetical token/term order. Construct the following:

1. (3pt) The document term matrices (document-term matrix contains rows corresponding to the documents and columns corresponding to the terms) based on
    a. Binary: only consider whether a term *t* appears in a document *D*. Repeated terms in one document are counted as *1* in binary matrices.

    b. Raw term frequency
    The raw term frequency *tf(t in D)* is defined as the frequency of a term *t* appeared in document *D*.

    c. Normalized Term frequency
    See for an example - http://en.wikipedia.org/wiki/TFIDF
    Term frequency for a term *t* in a document *D* can be normalized by the total number of terms N_D in the document.
    Normalized tf(t in D) = raw term frequency(t in D)$/N_D$. = *tf(t in D)*$/N_D$.

    d. tf-idf weights.
    The inverse document frequency *idf(t)* of term *t* can be defined using this expression: $[\ln\ (N/(n_j+1)) + 1]$), where *N* is the total number of documents in the index, n_j is the document frequency of term *t* (document frequency is the number of documents that term *t* appeared in). Thus, for term *t* in document *D*:
    *tf-idf (t)= raw term frequency(t) * idf(t) = tf(t in D)**$[\ln\ (N/(n_j+1)) + 1]$.

2. (1pt) The query term matrices for the same weights
   a. Binary
      Only consider whether a term *t* appears in a query *Q*.
   b. Raw term frequency
      The raw term frequency *tf(t in Q)* is defined as the frequency of a term *t* appeared in query *Q*.

   c. Normalized Term frequency
      Term frequency for a query can be normalized by the total number of terms N_Q in the query.
      Normalized tf(t in Q) = raw term frequency(t in Q)/N_Q.

   d. tf-idf weights. (The inverse document frequency *idf(t)* for term *t* is calculated in problem 1)
      *tf-idf (t)= raw term frequency(t) * idf(t) = tf(t in Q)*[ln (N/(*n_j*+1)) + 1]*
      Note: *N* is the number of total documents. n_j is the document frequency of term *t*. Thus, *idf(t)* is the same as calculated in problem 1.

3. (1pt) (1) Using results from 1 and 2, calculate with all of the above weights the document – query similarity coefficients for the following similarity measures:
   a. Inner product
      Inner product is calculated on two vectors, a query *Q*, and a document *D*.
   b. Cosine measure
      Cosine similarity is calculated on a query *Q* and a document *D*.

   (2) Calculate Lucene scores using the following equation:
      For a query *Q* and document *D*, (*t* represents terms)

$$score(Q,D) = coord(Q,D) \cdot queryNorm(Q) \cdot \sum_{t \text{ in } Q} (tf(t \text{ in } D) \cdot idf(t)^2 \cdot t.getBoost() \cdot norm(D))$$

where,

coord(Q,D) = overlap between Q and D / maximum overlap
Maximum overlap is the maximum possible length of overlap between Q and D

$queryNorm(Q) = 1/sum\ of\ square\ weight^{1/2}$
$sum\ of\ square\ weight = q.getBoost()^2 \cdot \sum_{t \text{ in } Q} (idf(t) \cdot t.getBoost())^2$
In this exercise, *t.getBoost() = 1, q.getBoost() = 1*
So, $sum\ of\ square\ weight = \sum_{t \text{ in } Q} (idf(t))^2$
thus, $queryNorm(Q) = 1/(\sum_{t \text{ in } Q} (idf(t))^2)^{1/2}$

$tf(t \text{ in } D) = frequency^{1/2}$ (the frequency is the raw term frequency defined in problem 1)

$idf(t) = ln\ (N/(n_j+1)) + 1$ (*N*: number of documents, n_j: document frequency of term *t*)

$norm(D) = 1/number\ of\ terms^{1/2}$ (This is the normalization by the total number of terms in a document. Number of terms is the total number of terms appeared in a document *D*.)

4. (1pt) Devise a ranking based on the results of 3. Compare and discuss these results. Which would you say give the best rankings and why?