Simple, Proven Approaches to Text Retrieval

S. E. Robertson & K.Sparck Jones

Presenters:
Gündüz Vehbi Demirci
Emir Gülümser

Outline

- Terms and Matching
- Weights
- Iterative Searching
- Longer Queries
- Elaborations

Terms and Matching

- Stop words are eliminated (Economy reasons)
- Terms are stems (Roots)
- Request as an unstructured list of terms
- If it is unweighted output could be ranked by number of matching terms
- If it is, ranked output is sum of weights

Weights

- *Collection Frequency*
  - CFW(i) = logN – logn
  - n = number of documents term t(i) occurs
  - N = number of documents in collection
- *Term Frequency*
  - TF(i,j) = number of occurrences of term t(i) in document d(j)

Weights

- *Document Length*
  - DL(j) = total of term occurrences in document d(j)
  - NDL(j) = DL(j) / (Avg. DL for all docs)
- *Combinig the Evidence*
  - For one term t(i) and one document d(j)
  - CW(i,j) = [CWF(i) * TF (i,j) * (K1+1)] / [K1 * ((1-b) + (b * (NDL(j)))) + TF(i,j)]

Iterative Searching

- *Relevance Weights*

  for term t(i)

  r = number of known relevant documents term t(i) occur in
  R = number of known relevant document for a request

  $$RW(i) = \log \left[\left((r+0.5)\,(N-n-R+r+0.5) \right) / \left((n\,r+0.5)(R-r+0.5) \right) \right]$$

Query Expansion

- All terms taken from relevant documents are ranked according to their Offer Weight

$$OW(i) = r * RW(i)$$

Iterative combination

- The relevance weight may be substituted for the collection frequency weight in the combined weight, formula to give
- Combined Iterative Weight:

$$CIW(i,j) = [\, RW(i) * TF(i,j) * (K1+1) \,] \;/\; [\, K1 * ((1-b) + (b * (NDL(j)))) + TF(i,j) \,]$$

Longer queries

- If you have requests longer than a few words or a sentence, i.e. ones in which query term stems may occur with different frequencies QF(i), then for each query term–document match
- Compute the Query Adjusted Combined Weight: QACW (I) = QF(i) * CW(i,j)
- Query Adjusted Combined Iterative Weight: QACIW (I) = QF(i) * CIW(i,j)

Elaborations

- It may be sensible, for some files, to index explicitly on complex or compound terms where a suitable lexicon is available:
  - Assist in the construction of the inverted file, which will also supply the necessary counting data for weighting.
- Discovering, by inspection, what multi-word strings there are in a file:
  - Quite different, and very expensive enterprise.
- It may also be possible to require conjoint matching as a preliminary to ranking output, though it may not be practicable to go beyond document co-presence and require text proximity.
- But it is always important in using multi-word terms to recognize the need to allow for the value of matches on individual components of a compound as well as on the compound as a whole.
- These elaborations are tricky to manage, and are not recommended for beginners.

Thank you for listening!