Information Retrieval

Weike Pan

The slides are **adapted from those provided by Prof. Hinrich Schütze** at University of Munich (http://www.cis.lmu.de/~hs/teach/14s/ir/).

Chapter 6 Scoring, term weighting & the vector space model

- 6.1 Parametric and zone indexes
- 6.2 Term frequency and weighting
- 6.3 The vector space model for scoring
- 6.4 Variant tf-idf functions
- 6.5 References and further reading

Outline

- 6.1 Parametric and zone indexes
- 6.2 Term frequency and weighting
- 6.3 The vector space model for scoring
- 6.4 Variant tf-idf functions
- 6.5 References and further reading

6.1 Parametric and zone indexes

- Thus far, our queries have been Boolean. Documents either match or don’t.

- Good for expert users with precise understanding of their needs and of the collection.
- Good for applications: Applications can easily consume 1000s of results.

- Not good for the majority of users. Most users are not capable of writing Boolean queries ... or they are, but they think it’s too much work. Most users don’t want to wade through 1000s of results. This is particularly true of web search.

6.1 Parametric and zone indexes

Problem with Boolean search: Feast or famine

- Boolean queries often result in either too few (=0) or too many (1000s) results.

- In Boolean retrieval, it takes a lot of skill to come up with a query that produces a manageable number of hits.

6.1 Parametric and zone indexes

Feast or famine: No problem in ranked retrieval

- With ranking, large result sets are not an issue.
- Just show the top 10 results.
- Doesn’t overwhelm the user.

- Premise (前提): the ranking algorithm works, i.e., more relevant results are ranked higher than less relevant results

6.1 Parametric and zone indexes

Scoring as the basis of ranked retrieval

- How can we accomplish a relevance ranking of the documents with respect to a query?

- Assign a **score** to each query-document pair, say in [0,1]. This score measures how well the document and the query "**match**".

- **Sort** the documents according to the scores.

6.1 Parametric and zone indexes

Query-document matching scores

- How do we compute the score of a query-document pair?
  - If no query term occurs in the document: score should be 0.
  - The more **frequent** a query term in the document, the higher the score. (Notes: term frequency)
  - The **more** query terms occur in the document, the higher the score. (Notes: number of terms)

6.1 Parametric and zone indexes

Bag of words model

- We do not consider the **order** of words in a document.
- *John is quicker than Mary* and *Mary is quicker than John* are represented exactly the same way.
- In a sense, this is a step back: The positional index (see chapter 2) is able to distinguish these two documents.

Outline

- 6.1 Parametric and zone indexes
- 6.2 Term frequency and weighting
- 6.3 The vector space model for scoring
- 6.4 Variant tf-idf functions
- 6.5 References and further reading

6.2 Term frequency and weighting

TF (Term Frequency)

- Term frequency: number of occurrences of a term t in a document d
- Raw term frequency is not what we want because: A document with tf = 10 occurrences of the term is more relevant than a document with tf = 1 occurrence of the term. But not 10 times more relevant, i.e., the relevance does not increase proportionally with the term frequency.

- We define the TF weight of term t in document d as follows

$$w_{t,d} = \begin{cases} 1 + \log_{10} \text{tf}_{t,d} & \text{if } \text{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

6.2 Term frequency and weighting

IDF (Inverse Document Frequency)

- Rare terms in the collection (instead of in a document as that in TF) are more informative than frequent terms.
  - high weights for rare terms
  - low (positive) weights for frequent terms

- We define the IDF weight of term t in the collection as follows

$$\mathrm{idf}_t = \log_{10} \frac{N}{\mathrm{df}_t}$$

- The IDF weight is a measure of the informativeness of the term in the collection. Note that the document frequency denotes the number of documents in the collection that the term occurs in.

6.2 Term frequency and weighting

TF-IDF

- The TF-IDF weight of a term t in a document d is the product of the corresponding TF weight and IDF weight:

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$$

- Increases with the number of occurrences within a document (TF).
- Increases with the rarity of the term in the collection (IDF).
- The TF-IDF weight is the best known weighting scheme in information retrieval.
- We can also see that the IDF weight affects the ranking of documents for queries with at least two terms. It has little effect on document ranking for one-term queries.

Outline

- 6.1 Parametric and zone indexes
- 6.2 Term frequency and weighting
- 6.3 The vector space model for scoring
- 6.4 Variant tf-idf functions
- 6.5 References and further reading

6.3 The vector space model for scoring

Ranked retrieval in the vector space model

- Step 1. Represent the query as a weighted TF-IDF vector
- Step 2. Represent each document as a weighted TF-IDF vector
- Step 3. Compute the cosine similarity between the query vector and each document vector

$$\cos(\vec{q},\vec{d}) = \textsc{sim}(\vec{q},\vec{d}) = \frac{\vec{q}\cdot\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

- Step 4. Rank the documents with respect to the query
- Step 5. Return the top K (e.g., K = 10) documents to the user

Summary

- 6.1 Parametric and zone indexes
- 6.2 Term frequency and weighting
- 6.3 The vector space model for scoring
- 6.4 Variant tf-idf functions
- 6.5 References and further reading