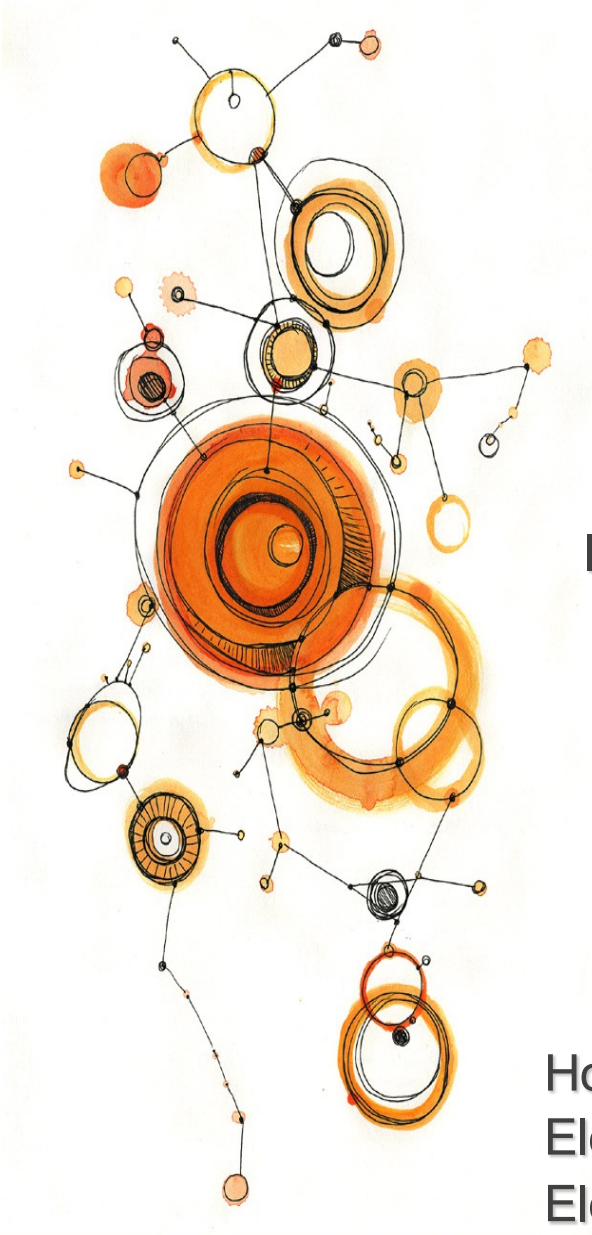


Modeling and Processing of Multimedia Data

International Second cycle degree programme (LM) in
Digital Humanities and Digital Knowledge (DHDK)
University of Bologna

Textual Information Retrieval Systems – Part II

Home page: http:// www-db.disi.unibo.it/courses/DMMMDB/
Electronic version: 1.02.TextualInformationRetrieval-II.pdf
Electronic version: 1.02.TextualInformationRetrieval-II-2p.pdf

Outline

- The Vector Space Model (VSM)
- Weighing techniques and ranking of the results
- Evaluation of IR systems: Precision and Recall metrics

Back to the limits of the Boolean model

1. Unexperienced users have difficulties in understanding what Boolean operators really mean
2. No notion of "partial matching"
3. No ranking of the documents
4. "Near-miss" and "Information overload" problems

Preliminaries on the Vector Space Model (VSM)

- The Boolean model just looks at binary (1/0) information:

  presence vs. absence of a term in a document

- Thus, for a conjunctive query all query terms must be present for a document to be considered relevant

- Let's take a different perspective: given a set of search terms, which translate our information need, we could argue that:
  1. The more such terms a doc contains, the better such doc is
  2. For a given search term, the more occurrences of such term in a doc, the better the doc is
- From 1. we derive that not all terms need to be present for a document to be relevant
- From 2. we derive the need to take into account term frequency information

Weighting terms: tf.idf

- Given a term t_i and a document doc_j, we can provide a measure of how well doc_j "fits" t_i, that is, which is the relevance of doc_j w.r.t. t_i
- Let $w_{i,j}$ denote the "weight" of t_i in doc_j
  - For the Boolean model it is $w_{i,j} \in \{0,1\}$
- In the tf.idf weighting schema, $w_{i,j}$ depends on two factors:
  - The first is the number of occurrences of t_i in doc_j, $freq_{i,j}$, and is called the **term frequency** of t_i in doc_j, also denoted $\mathbf{tf_{i,j}}$
    - One could also take "normalized" frequency values, i.e., $\mathbf{tf_{i,j} = freq_{i,j}/max_l\{freq_{l,j}\}}$
  - The second is a factor that is related to the "discriminating power" of t_i in the collection, and is negatively correlated with the number of docs, N_i, in which t_i appears (this is Ndocs in the inverted file figures).
    This is called the **inverse document frequency** of t_i, denoted $\mathbf{idf_i}$, and can be computed as $\mathbf{idf_i = log(N/ N_i)}$, where N is the number of docs in the collection
    - A term present in each document has $idf_i = 0$,
- The tf.idf scheme computes $w_{i,j}$ as $\boxed{\mathbf{w_{i,j} = tf_{i,j} * idf_i}}$
- We still have $w_{i,j} = 0$ if t_i is not present in doc_j

Documents as vectors: the VSM

- If we view each term as independent of (orthogonal to) the others, our docs are vectors in the $\Re^V$ space, with values along the i-th coordinate given by $w_{i,j}$
- This is called the **Vector Space Model (VSM)**, for which a measure of similarity among vectors can be easily defined…

A vector in $\Re^7$

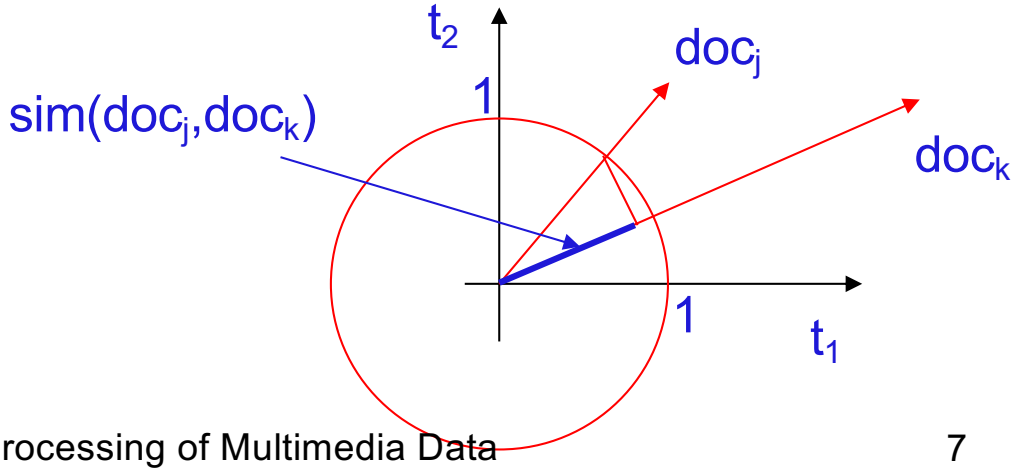
	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	13.1	11.4	0.0	0.0	0.0	0.1
Brutus	3.0	8.3	0.0	1.0	0.0	0.0
Caesar	2.3	2.3	0.0	0.5	0.3	0.3
Calpurnia	0.0	11.2	0.0	0.0	0.0	0.0
Cleopatra	17.7	0.0	0.0	0.0	0.0	0.0
mercy	0.5	0.0	0.7	0.9	0.9	0.3
worser	1.2	0.0	0.6	0.6	0.6	0.0

Cosine similarity

- The VSM model proposes to evaluate the similarity of two documents by measuring the correlation of the corresponding vectors
- A common way to measure correlation is computing the cosine of the angle between the two vectors:

$$\text{sim}(\text{doc}_j, \text{doc}_k) = \cos(\Theta_{j,k}) = \frac{\text{doc}_j \bullet \text{doc}_k}{\|\text{doc}_j\| \times \|\text{doc}_k\|} = \frac{\sum_{i=1}^{V} w_{i,j} \times w_{i,k}}{\sqrt{\sum_{i=1}^{V} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{V} w_{i,k}^2}}$$

- Normalization is to avoid that documents' length becomes the dominant factor



Ranking the documents (1)

- In order to rank the documents it is sufficient to compute their similarities w.r.t the **query vector q**
  - Thus, the query is a vector as well! (we don't have Boolean operators anymore)
- If query terms are not weighted (alternatively, one could take into account their idf values) we have $q = (w_{1,q},\ldots,w_{V,q})$, $w_{i,q} \in \{0,1\}$
- Let QT be the index set of the query terms, that is: $QT = \{i: w_{i,q} = 1\}$
- Then we can write

$$sim(doc_j, q) = \frac{\sum_{i \in QT} w_{i,j}}{\sqrt{\sum_{i=1}^{V} w_{i,j}^2} \times \sqrt{|QT|}} \propto \frac{\sum_{i \in QT} w_{i,j}}{\sqrt{\sum_{i=1}^{V} w_{i,j}^2}}$$

- Thus, to compute the similarity of a document w.r.t. to a query, we just need
  1) to accumulate its weights over the query terms, and
  2) to normalize by the length of the document vector

Ranking the documents (2)

- If documents' vectors have been normalized, i.e., $\left\|\text{doc}_j\right\| = \sqrt{\sum_{i=1}^{V} w_{i,j}^2} = 1$

  then we are left with: $\text{sim}(\text{doc}_j, q) \propto \sum_{i \in QT} w_{i,j}$

- Ok, now that we know how to compute similarities, we can rank documents and just return only the k best documents
- How?

Computing the k best documents

- We do not present all the details, even because each system has its own undisclosed “tricks”
- The idea is:
  1. Take the posting lists of the query terms (remind, here we have frequency of terms, and in the vocabulary the inverse document frequencies)
  2. Merge (take the union of) such lists
  3. Sort by decreasing similarity values
  4. Return the first k documents

VSM: an example

- Assume normalized vectors

q = computer science

2. Merge the lists

Term	N docs
computer	5
science	3

Doc #	wi,j
3	0.17
5	0.2
8	0.25
10	0.03
13	0.08

Doc #	wi,j
2	0.31
5	0.12
8	0.05

Doc #	sim
2	0.31
3	0.17
5	0.32
8	0.3
10	0.03
13	0.08

Doc #	sim
5	0.32
2	0.31
8	0.3
3	0.17
13	0.08
10	0.03

1. Take the posting lists of the query terms

3. Sort by decreasing similarity values

Retrieval effectiveness

- How can we evaluate the "quality of results" of a system?
- There are two "standard" measures:

Precision (Prec): this is the fraction of retrieved docs that are relevant

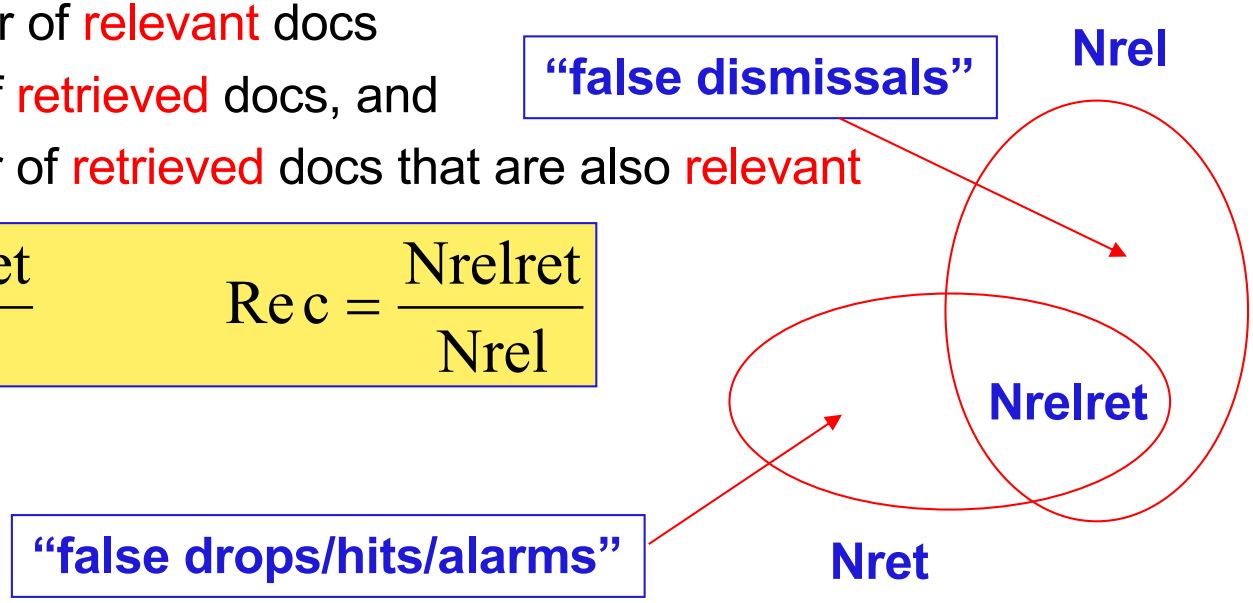
- In probabilistic terms, it is Prec = Prob{doc is relevant|doc is retrieved}

Recall (Rec): this is the fraction of relevant docs that are retrieved

- In probabilistic terms, it is Rec = Prob{doc is retrieved|doc is relevant}

- For a given query, let
  - Nrel be the number of relevant docs
  - Nret the number of retrieved docs, and
  - Nrelret the number of retrieved docs that are also relevant
- It is:

$$Prec = \frac{Nrelret}{Nret} \qquad Rec = \frac{Nrelret}{Nrel}$$



How to measure?

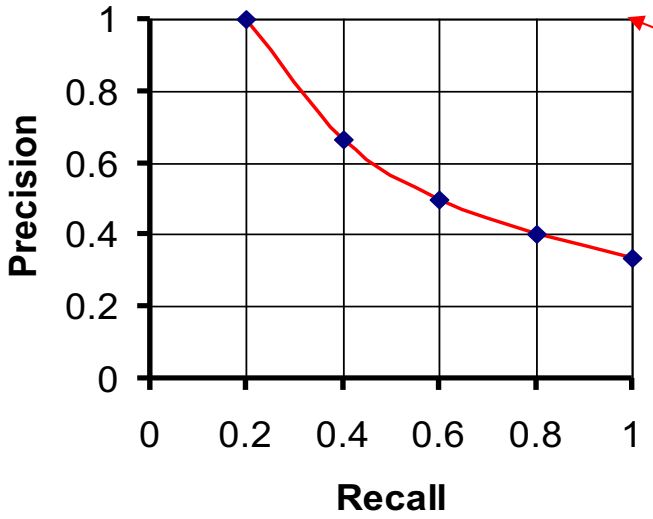
- To measure the performance of a system one resorts to so-called "test collections", which come with a set of queries, for each of which the set of relevant documents is known
  - In practice, relevance is assessed by a set of experts
- The reference test collections for information retrieval systems are the ones from TREC:
  - TREC stands for "Text Retrieval Conference": it is an annual conference dedicated to experiments over large text collections (N > 1 million, several GBytes of text)
- The collections come from different sources (e.g.: Wall Street Journal, Federal Register, US Patents, LA Times)
- As a final remark:
  if you don't have a test collection, you can still evaluate precision of results, but you cannot evaluate recall!

Precision-recall curve

- For a given query, it is customary to measure the precision at several levels of recall and then to plot (Prec,Rec) values, thus leading to a so-called Precision-Recall curve

Doc#	20	37	2	19	26	87	11	5	4	54	12	36	81	42	27
Relevant?	√		√			√				√					√

Assume there are 5 relevant documents for our query



(1,1) is the "ideal" system

- The retrieval performance of a system can then be evaluated by averaging precision values over a set of sample queries

Considerations on IR

- The vector space model is undoubtedly the most widely used text retrieval model
- It is commonly used by TREC participants, as well as by most Web search engines
- Its basic concepts are also adopted by systems other than IR ones, such as:
  - Multimedia systems
  - Information Filtering systems
  - Recommendation systems

- With respect to the Boolean model, VSM usually achieves a better quality of the result for simple queries (a few search terms)
- The Boolean model, on the other hand, is still preferred by experienced users who know how to precisely formulate their queries
- Other models (e.g., based on probability) exist, but they are not as popular as VSM

Free exercise 1.C

- Starting from Examples proposed in Exercises 1.A/1.B, and by focusing on the **textual documents** only
- Draw
  - the **term-document matrix** and
  - the **inverted index**

  representations for Boolean and VS retrieval models
- Identify some relevant **queries**
- Plot the two separate graphs of
  - **Average Precision vs. Nret documents** and
  - **Average Recall vs. Nret documents**

Free exercise 1.C: students to do

- Starting from the solution you proposed in Exercises 1.A/1.B, enrich your .ppt document with the description of your MM applications by adding
  - **visual examples** (i.e., **images**) of involved **relevant textual information**
  - provide a sketch of the **VS model-based inverted index** you built for solving your IR queries