The Role of the Query

- Retrieval result hinges crucially on query
- Query construction requires / benefits from domain knowledge
- Query terms can be ambiguous
- + and – operators in the query can help to
  - disambiguate ambiguous terms
  - make query more precise
  - are not well-understood by common users
- In IR, queries are normally treated as short documents themselves
- + and – operators are not part of the document, but function as additional filters

Retrieval Models

- **Boolean Model**
  - term presence/absence ➔ relevant/irrelevant, no ranking
  - formalized query syntax, such as Boolean logic, NEAR-operators
  - IR-systems of the pre-web era, as used in libraries, law enforcement etc.
  - suitable for small collections, recall-oriented
- **Vector Space Model**
  - queries and docs are represented in a vector space
  - term weighting, scoring function ➔ graded relevance, ranking
  - free text queries
  - web IR systems, used as add-on in classic environments
- **Probabilistic Models**, e.g. Language Modeling
  - probabilistic weighting scheme, generative story
  - similar to VSM, but better theoretical footing
  - especially suited for long queries, more tolerant regarding missing terms

Retrieval models are differentiated by how they represent documents, how they represent queries, and by the relevance function.

Term Weights

- Terms that are more frequent in a document characterize the document better than terms that are rare
- **Term Frequency** ($tf_{t,d}$): The frequency of term t in document d
- $tf_{t,d}$ alone is unintuitive: Terms that are common in all documents in the collection are not informative
- In a document collection, rare terms are more informative than common ones (and vice versa, cf. stop words like "the")
- If a query contains a term that is rare in the collection, those documents that do contain the term are probably relevant.
- **Document Frequency** ($df_{t,}$): Number of documents with a term
- **Inverse Document Frequency** (idf_t):

  N: number of documents

$$idf_t = \log \frac{N}{df_t}$$

Term	Doc #	Freq
ambitious	2	1
be	2	1
brutus	1	1
brutus	2	1
capitol	1	1
caesar	1	1
caesar	2	2
did	1	1
enact	1	1
hath	2	1
I	1	2
i'	1	1
it	2	1
julius	1	1
killed	1	2
let	2	1
me	1	1
noble	2	1
so	2	1
the	1	1
the	2	1
told	2	1
you	2	1
was	1	1
was	2	1
with	2	1

Inverse Document Frequency

- There is one idf_t for each term t in the collection
- Example values (for $\log_{10}$):

term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1,000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

Tf-idf: a standard term weighting scheme in IR

- tf and idf can be combined into a single term weight: **Tf-idf**:

$$w_{t,d} = tf_{t,d} \times \log \frac{N}{df_t}$$

- Tf-idf characterizes the weight of a term in a document, in relation to the entire collection that the document is part of
- Best-known weighting scheme in IR
- Properties
  - Increases with the number of occurrences of a term in a document
  - Increases with the rarity of a term in the entire collection
  - handles stop words naturally
- A more elaborate weighting formula:

more query words

repetition of query words

$$sim(d,q) = \sum_{w \in Q} \frac{tf_D(w)}{tf_D(w) + K|d|} \times \log\left(\frac{N}{df(w)}\right)$$

penalize common words

penalize long docs

Vector Space Model

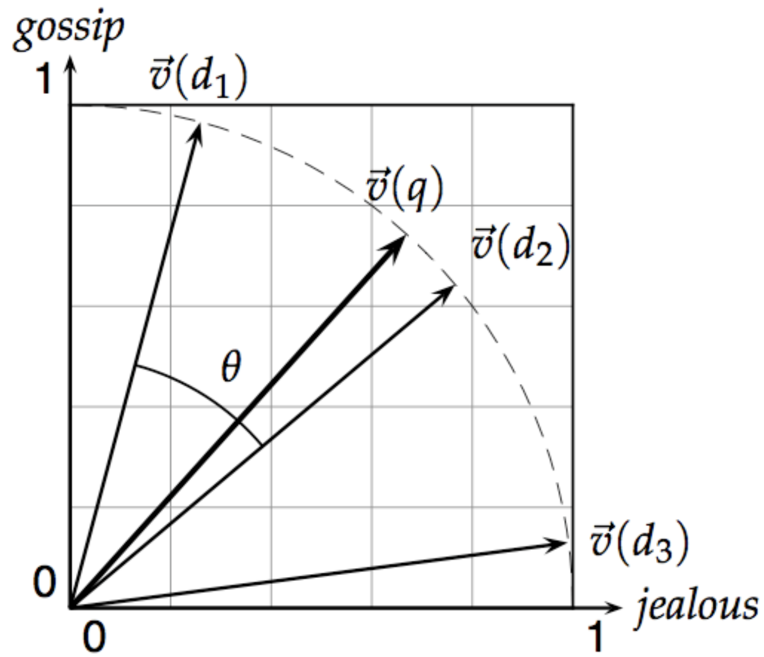
- Idea: Relevant documents are those that are most similar to the query
- Similarity is established by means of comparison, so how should we compare query and documents?
- Term overlap (on the basis of the index) does not capture document properties.
- Rather: Represent query and documents as vectors, and compare them using vector-similarity measures ➔ Vector Space Model
- Query and documents are modelled as points / vectors in n-dimensional space, with n = number of distinct terms in the collection
- In physical space, a point is characterized by its location in three dimensions (~ values for the attributes x, y, and z).
- In vector space, each term is an attribute, and its Tf-idf value is its value (other weighting schemes are possible)

Vector Space Example

- Relevance score: use cosine similarity of the angle between vectors.
- Scales to be used with arbitrary number of terms (i.e. dimensions)
- Returns numerical similarity for each doc-query pair

$$\mathrm{sim}(d_1, d_2) = \vec{v}(d_1) \cdot \vec{v}(d_2)$$

$$= \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|}$$



Cosine similarity illustrated. $\mathrm{sim}(d_1, d_2) = \cos\theta$.

(only two dimensions “gossip” and “jealous” shown)

Other sources for term weights

- Fields: For structured collections. E.g.: term in book title more important than in description
- Zones: For semi-structured collections, such as HTML documents. E.g. term in <h1>-Tag more important than in text; terms in certain frames irrelevant
- Static document weights: Some documents are more trusted than others, independent of query. Leads to tiered indices: separate indices for documents with high, medium and low ranks