Modern Information Retrieval

Scores in a complete search system

Hamid Beigy

Sharif university of technology

March 3, 2025





- 1. Introduction
- 2. Improving scoring and ranking
- 3. A complete search engine
- 4. References

Introduction



1. We define term frequency weight of term t in document d as

$$tf_{t,d} = \sum_{x \in d} f_t(x)$$
 where $f_t(x) = \begin{cases} 1 & \text{if } x = t \\ 0 & \text{otherwise} \end{cases}$

2. The log frequency weight of term t in d is defined as follows

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

3. We define the idf weight of term t as follows:

$$\mathsf{idf}_t = \mathsf{log}_{10} \frac{N}{\mathsf{df}_t}$$

4. We define the tf-idf weight of term t as product of its tf and idf weights.

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

Hamid Beigy (Sharif university of technology)



1. Cosine similarity between query q and document d is defined as

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

- 2. q_i is the tf-idf weight of term *i* in the query.
- 3. d_i is the tf-idf weight of term *i* in the document.
- 4. $|\vec{q}|$ and $|\vec{d}|$ are the lengths of \vec{q} and \vec{d} .
- 5. $\vec{q}/|\vec{q}|$ and $\vec{d}/|\vec{d}|$ are length-1 vectors (= normalized).
- 6. Computing the cosine similarity is time-consuming task.





Mean: 3.07 Median/Mode: 2.00

Google

Looking versus clicking

٢

- $1. \ {\rm Users} \ {\rm view} \ {\rm results} \ {\rm two} \ {\rm more} \ {\rm often}/ \ {\rm thoroughly}.$
- 2. Users click most frequently on result one.



- 1. The first rank has average click rate of 31.7%.
- 2. Only 0.78% of Google searchers clicked from the second page.





Importance of ranking

- 1. Viewing abstracts: Users are a lot more likely to read the abstracts of the top-ranked pages (1, 2, 3, 4) than the abstracts of the lower ranked pages (7, 8, 9, 10).
- 2. Clicking: Distribution is even more skewed for clicking
- 3. In 1 out of 2 cases, users click on the top-ranked page.
- 4. Even if the top-ranked page is not relevant, 30% of users will click on it.
 - Getting the ranking right is very important.
 - Getting the top-ranked page right is most important





Improving scoring and ranking



- 1. The scoring algorithm can be time consuming
- 2. Using heuristics can help saving time
- 3. Exact top-score vs approximative top-score retrieval

We can lower the cost of scoring by searching for K documents that are likely to be among the top-scores

- 4. General optimization scheme:
 - 4.1 find a set of documents A such that K < |A| << N, and whose is likely to contain many documents close to the top-scores
 - 4.2 return the K top-scoring document included in A



- 1. While processing the query, only consider terms whose idf_t exceeds a predefined threshold Thus we avoid traversing the posting lists of low idf_t terms, lists which are generally long
- 2. Only consider documents where all query terms appear



- 1. We know which documents are the most relevant for a given term
- 2. For each term t, we pre-compute the list of the r most relevant (with respect to w(t, d)) documents in the collection
- 3. Given a query q, we compute

$$A = \bigcup_{t \in q} r(t)$$

r can depends on the document frequency of the term.



- 1. Only consider documents which are considered as high-quality documents
- 2. Given a measure of quality g(d), the posting lists are ordered by decreasing value of g(d)
- 3. Can be combined with champion lists, *i.e.* build the list of r most relevant documents wrt g(d)
- 4. Quality can be computed from the logs of users' queries



- 1. Some sublists of the posting lists are of no interest
- 2. To reduce the time complexity:
 - query terms are processed by decreasing *idf*_t
 - postings are sorted by decreasing term frequency $tf_{t,d}$
 - Once idf_t gets low, we can consider only few postings
 - Once *tf*_{t,d} gets smaller than a predefined threshold, the remaining postings in the list are skipped



- 1. The document vectors are gathered by proximity
- 2. We pick \sqrt{N} documents randomly \Rightarrow leaders
- 3. For each non-leader, we compute its nearest leader \Rightarrow followers
- 4. At query time, we only compute similarities between the query and the leaders
- 5. The set A is the closest document cluster
- 6. The document clustering should reflect the distribution of the vector space







- 1. This technique can be seen as a generalization of champion lists
- 2. Instead of considering one champion list, we manage layers of champion lists, ordered in increasing size:

index 1	/ most relevant documents		
index 2	next m most relevant documents		
index 3	next <i>n</i> most relevant documents		

Indexed defined according to thresholds

Dictionary	$v(\vec{d}_1)$	$\vec{v(d_2)}$	$\vec{v(d_3)}$
affection	0.996	0.993	0.847
jealous	0.087	0.120	0.466
gossip	0.017	0	0.254







- 1. Priority is given to documents containing many query terms in a close window
- 2. Needs to pre-compute *n*-grams
- 3. And to define a proximity weighting that depends on the window size *n* (either by hand or using learning algorithms)



- 1. Index elimination
- 2. Champion lists
- 3. Static quality score
- 4. Impact ordering
- 5. Cluster pruning
- 6. Tiered indexes
- 7. Query-term proximity

A complete search engine



- 1. Many techniques to retrieve documents (using logical operators, proximity operators, or scoring functions)
- 2. Adapted technique can be selected dynamically, by parsing the query
- 3. First process the query as a phrase query, if fewer than K results, then translate the query into phrase queries on bi-grams, if there are still too few results, finally process each term independently (real free text query)





References



1. Chapters 7 of Information Retrieval Book¹

¹Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press. Hamid Beigy (Sharif university of technology)



Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze (2008). Introduction to Information Retrieval. New York, NY, USA: Cambridge University Press.

Questions?