Modern Information Retrieval

Probabilistic Information Retrieval

Hamid Beigy

Sharif university of technology

April 4, 2025

Table of contents

Introduction

Introduction

1. An information need may be expressed using different keywords (synonymy) such as aircraft vs airplane.
2. The same word can have different meanings (polysemy).
3. Vocabulary of searcher may not match that of the documents.
4. Solutions: refining queries manually or expanding queries automatically
5. Relevance feedback and query expansion aim to overcome the problem of synonymy.

Relevance Feedback

Basic idea of relevance feedback

1. The user issues a (short, simple) query.
2. The search engine returns a set of documents.
3. User marks some docs as relevant, some as non-relevant.
4. Search engine computes a new representation of the information need (should be better than the initial query).
5. Search engine runs new query and returns new results.
6. New results have (hopefully) better recall.

Types of query expansion

1. Manual thesaurus (maintained by editors, e.g., PubMed)
2. Automatically derived thesaurus (e.g., based on co-occurrence statistics)
3. Query-equivalence based on query log mining (common on the web as in the “palm” example)

Probabilistic Approach to IR

Probabilistic Approach to Retrieval

1. Given a user information need (query) and a collection of documents (document), a system must determine how well the documents satisfy the query
2. An IR system has an uncertain understanding of the user query, and makes an uncertain guess of whether a document satisfies the query
3. Probability theory provides a principled foundation for such reasoning under uncertainty
4. Probabilistic models exploit this foundation to estimate **how likely it is that a document is relevant to a query**.

Probabilistic vs. vector space model

1. Vector space models rank documents according to similarity to query.
2. The notion of similarity does not translate directly into an assessment of is the document a good document to give to the user or not?
3. The most similar document can be highly relevant or completely nonrelevant.
4. Probability theory is arguably a cleaner formalization of what we really want an IR system to do: give relevant documents to the user.

Probability Theory

1. For events A and B
   - Joint probability $P(A \cap B)$ of both events occurring
   - Conditional probability $P(A|B)$ of event A occurring given that event B has occurred
2. Chain rule gives fundamental relationship between joint and conditional probabilities:

$$P(AB) = P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

3. Similarly for the complement of an event $P(\overline{A})$:

$$P(\overline{A}B) = P(B|\overline{A})P(\overline{A})$$

4. Partition rule: if B can be divided into an exhaustive set of disjoint subcases, then $P(B)$ is the sum of the probabilities of the subcases. A special case of this rule gives:

$$P(B) = P(AB) + P(\overline{A}B)$$

1. Bayes' Rule for inverting conditional probabilities:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \left[\frac{P(B|A)}{\sum_{X\in\{A,\overline{A}\}} P(B|X)P(X)}\right] P(A)$$

   Can be thought of as a way of updating probabilities:

   - Start off with prior probability $P(A)$ (initial estimate of how likely event A is in the absence of any other information)
   - Derive a posterior probability $P(A|B)$ after having seen the evidence B, based on the likelihood of B occurring in the two cases that A does or does not hold

2. Odds of an event provide a kind of multiplier for how probabilities change:

$$\text{Odds:} \qquad O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1 - P(A)}$$

Probability Ranking Principle

The Document Ranking Problem

1. Ranked retrieval setup: given a collection of documents,
   - the user issues a query, and
   - an ordered list of documents is returned
2. Assume binary notion of relevance: $R_{d,q}$ is a random variable, such that
   - $R_{d,q} = 1$ if document d is relevant w.r.t query q
   - $R_{d,q} = 0$ otherwise
3. Probabilistic ranking orders documents decreasingly by their estimated probability of relevance w.r.t. query: $P(R = 1|d, q)$
4. Assume that **the relevance of each document is independent of the relevance of other documents**.
5. Probability ranking principle (PRP) states:
   - If the retrieved documents (w.r.t a query) are ranked decreasingly on their probability of relevance,
   - then the effectiveness of the system will be the best that is obtainable

1. Traditionally used with the PRP
2. Assumptions:
   - Binary: documents and queries represented as binary term incidence vectors
     - Document d represented by vector $\vec{x} = (x_1, \ldots, x_M) \in \{0,1\}^{|V|}$, where $x_t = 1$ if term t occurs in d and $x_t = 0$ otherwise
     - Different documents may have the same vector representation

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	1	1	0	0	0	1	
BRUTUS	1	1	0	1	0	0	
CAESAR	1	1	0	1	1	1	
CALPURNIA	0	1	0	0	0	0	
CLEOPATRA	1	0	0	0	0	0	
MERCY	1	0	1	1	1	1	
WORSER	1	0	1	1	1	0	
...							

   - Independence: no association between terms (not true, but practically works - naive assumption of Naive Bayes models)

1. To make a probabilistic retrieval strategy precise, need to estimate how terms in documents contribute to relevance
2. Find measurable statistics (term frequency, document frequency, document length) that affect judgments about document relevance
3. Combine these statistics to estimate the probability $P(R|d,q)$ of document relevance
4. **How exactly we can do this?**
5. $P(R|d,q)$ is modeled using term incidence vectors as $P(R|\vec{x},\vec{q})$

$$P(R=1|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=1,\vec{q})P(R=1|\vec{q})}{P(\vec{x}|\vec{q})}$$
$$P(R=0|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=0,\vec{q})P(R=0|\vec{q})}{P(\vec{x}|\vec{q})}$$

   - $P(\vec{x}|R=1,\vec{q})$: probability that if a relevant document is retrieved, then that document's representation is $\vec{x}$
   - $P(\vec{x}|R=0,\vec{q})$: probability that if a nonrelevant document is retrieved, then that document's representation is $\vec{x}$

6. Use statistics about the document collection to estimate these probabilities

1. $P(R|d,q)$ is modeled using term incidence vectors as $P(R|\vec{x},\vec{q})$

$$P(R=1|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=1,\vec{q})P(R=1|\vec{q})}{P(\vec{x}|\vec{q})}$$
$$P(R=0|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=0,\vec{q})P(R=0|\vec{q})}{P(\vec{x}|\vec{q})}$$

- $P(R=1|\vec{q})$: prior probability of retrieving a relevant document for a query $\vec{q}$.
- $P(R=0|\vec{q})$: prior probability of retrieving a nonrelevant document for a query $\vec{q}$.
- Estimate $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$ from percentage of relevant documents in the collection
- Since a document is either relevant or nonrelevant to a query, we must have that:

$$P(R=1|\vec{x},\vec{q}) + P(R=0|\vec{x},\vec{q}) = 1$$

Deriving a Ranking Function for Query Terms

1. Given a query q, ranking documents by $P(R = 1|d, q)$ is modeled under BIM as ranking them by $P(R = 1|\vec{x}, \vec{q})$
2. Rank documents by their odds of relevance (gives same ranking)

$$O(R|\vec{x}, \vec{q}) = \frac{P(R = 1|\vec{x}, \vec{q})}{P(R = 0|\vec{x}, \vec{q})} = \frac{\frac{P(R=1|\vec{q})P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|\vec{q})}}{\frac{P(R=0|\vec{q})P(\vec{x}|R=0,\vec{q})}{P(\vec{x}|\vec{q})}}$$
$$= \frac{P(R = 1|\vec{q})}{P(R = 0|\vec{q})} \cdot \frac{P(\vec{x}|R = 1, \vec{q})}{P(\vec{x}|R = 0, \vec{q})}$$

3. $\frac{P(R=1|\vec{q})}{P(R=0|\vec{q})}$ is a constant for a given query - **can be ignored**.

1. We make the Naive Bayes conditional independence assumption that **the presence or absence of a word in a document is independent of the presence or absence of any other word (given the query):**

$$\frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})} = \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$$

2. So we obtain

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$$

3. Since each x_t is either 0 or 1, we can separate the terms:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \times \prod_{t:x_t=1} \frac{P(x_t=1|R=1,\vec{q})}{P(x_t=1|R=0,\vec{q})} \times \prod_{t:x_t=0} \frac{P(x_t=0|R=1,\vec{q})}{P(x_t=0|R=0,\vec{q})}$$

Deriving a Ranking Function for Query Terms

1. Define
   - $p_t = P(x_t = 1|R = 1, \vec{q})$: Probability of a term appearing in relevant document
   - $u_t = P(x_t = 1|R = 0, \vec{q})$: Probability of a term appearing in a nonrelevant document.
2. Can be displayed as contingency table:

	document	relevant ($R = 1$)	nonrelevant ($R = 0$)
Term present	$x_t = 1$	p_t	u_t
Term absent	$x_t = 0$	$1 - p_t$	$1 - u_t$

3. **Assumption**: **terms not occurring in the query are equally likely to occur in relevant and nonrelevant documents**: If $q_t = 0$, then $p_t = u_t$
4. We only need to consider terms in the products that appear in the query:

$$O(R|\vec{x}, \vec{q}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=0,q_t=1} \frac{1-p_t}{1-u_t}$$

5. The left product is over query terms found in the document.
6. The right product is over query terms not found in the document.

1. Including the query terms found in the document into the right product, but simultaneously dividing by them in the left product, gives:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=q_t=1} \frac{1-u_t}{1-p_t} \cdot \frac{1-p_t}{1-u_t} \cdot \prod_{\substack{t:x_t=0 \\ q_t=1}} \frac{1-p_t}{1-u_t}$$

$$= O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} \cdot \prod_{t:q_t=1} \frac{1-p_t}{1-u_t}$$

2. The left product is still over query terms found in the document, but the right product is now over all query terms, hence constant for a particular query and can be ignored.
3. The only quantity that needs to be estimated to rank documents w.r.t a query is the left product
4. Hence the Retrieval Status Value (RSV) in this model:

$$RSV_d = \log \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} = \sum_{t:x_t=q_t=1} \log \frac{p_t(1-u_t)}{u_t(1-p_t)}$$

Deriving a Ranking Function for Query Terms

1. Equivalent: rank documents using the log odds ratios for the terms in the query c_t (functions as a term weight):

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} - \log \frac{u_t}{1-u_t}$$

2. The odds ratio is the ratio of two odds:
   - the odds of the term appearing if the document is relevant ($p_t/(1-p_t)$), and
   - the odds of the term appearing if the document is nonrelevant ($u_t/(1-u_t)$).
3. $c_t = 0$: term has equal odds of appearing in relevant and nonrelevant docs.
4. c_t positive: higher odds to appear in relevant documents.
5. c_t negative: higher odds to appear in nonrelevant documents.
6. So BIM and vector space model are identical on an operational level except that the term weights are different.
7. In particular: we can use the same data structures (inverted index etc) for the two models.

1. For each $t \in q$, estimate c_t in collection using a contingency table of counts of documents in the collection, where df_t is the number of documents that contain term t:

	documents	relevant	nonrelevant	Total
Term present	$x_t = 1$	s	$\mathrm{df}_t - s$	df_t
Term absent	$x_t = 0$	$S - s$	$(N - \mathrm{df}_t) - (S - s)$	$N - \mathrm{df}_t$
	Total	S	$N - S$	N

$$p_t = s/S$$
$$u_t = (\mathrm{df}_t - s)/(N - S)$$
$$c_t = K(N, \mathrm{df}_t, S, s) = \log \frac{s/(S-s)}{(\mathrm{df}_t - s)/((N - \mathrm{df}_t) - (S - s))}$$

2. **Avoiding zeros:**
   - If any of the counts is a zero, then the term weight is not well-defined.
   - Maximum likelihood estimates do not work for rare events.
   - To avoid zeros: add 0.5 to each count: use $S - s + 0.5$ in formula for $S - s$.

Simplifying assumption

1. Assuming that relevant documents are a very small percentage of the collection, approximate statistics for nonrelevant documents by statistics from the whole collection
2. Hence, u_t (the probability of term occurrence in nonrelevant documents for a query) is df_t/N and
$$\log[(1-u_t)/u_t] = \log[(N-\mathrm{df}_t)/\mathrm{df}_t] \approx \log N/\mathrm{df}_t$$
3. This should look familiar: idf.
4. The above approximation cannot easily be extended to relevant documents. (why? check it.)

Appraisal and Extensions

History and summary of assumptions

1. Among the oldest formal models in IR
   - Maron & Kuhns, 1960: Since an IR system cannot predict with certainty which document is relevant, we should deal with probabilities
2. Assumptions for getting reasonable approximations of the needed probabilities (in the BIM):
   - Boolean representation of documents/queries/relevance
   - Term independence
   - Out-of-query terms do not affect retrieval
   - Document relevance values are independent

How different are vector space and BIM?

- They are not that different.
- In either case you build an information retrieval scheme in the exact same way.
- For probabilistic IR, at the end, you score queries not by cosine similarity and tf-idf in a vector space, but by a slightly different formula motivated by probability theory.
- How to add term frequency and length normalization to the probabilistic model.

Okapi BM25: Overview

1. Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.
2. BIM was originally designed for short catalog records of fairly consistent length, and it works reasonably in these contexts
3. For modern full-text search collections, a model should pay attention to term frequency and document length
4. BestMatch25 (a.k.a BM25 or Okapi) is sensitive to these quantities
5. BM25 is one of the most widely used and robust retrieval models.
6. The simplest score for document d is just idf weighting of the query terms present in the document:

$$RSV_d = \sum_{t \in q} \log \frac{N}{\mathrm{df}_t}$$

Okapi BM25 basic weighting

1. Improve idf term [log N/df] by factoring in term frequency and document length.

$$RSV_d = \sum_{t \in q} \log \left[\frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1 + 1)\mathrm{tf}_{td}}{k_1((1 - b) + b \times (L_d / L_{\mathrm{ave}})) + \mathrm{tf}_{td}}$$

2. tf_{td}: term frequency in document d
3. L_d (L_{ave}): length of document d (average document length in the whole collection)
4. k_1: tuning parameter controlling the document term frequency scaling
5. b: tuning parameter controlling the scaling by document length

Okapi BM25 weighting for long queries

1. For long queries, use similar weighting for query terms

$$RSV_d = \sum_{t \in q} \left[\log \frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1+1)\mathrm{tf}_{td}}{k_1((1-b)+b \times (L_d/L_{\mathrm{ave}}))+\mathrm{tf}_{td}} \cdot \frac{(k_3+1)\mathrm{tf}_{tq}}{k_3+\mathrm{tf}_{tq}}$$

2. tf_{tq}: term frequency in the query q
3. k_3: tuning parameter controlling term frequency scaling of the query
4. No length normalization of queries (because retrieval is being done with respect to a single fixed query)
5. We tune parameters to optimize performance on a **development collection**.
6. Experiments have shown reasonable values are: $k_1, k_3 \in 1.2, 2]$ and $b = 0.75$.

Conclusions: Which ranking model should I use?

1. If you want something basic and simple: use vector space with tf-idf weighting.
2. If you want to use a state-of-the-art ranking model with excellent performance: use language models or BM25 with tuned parameters

References

Reading

1. Chapter 11 of Information Retrieval Book[1].
2. Section 7.2 of Search Engines - Information Retrieval in Practice Book[2].

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

[2] W. Bruce Croft, Donald Metzler, and Trevor Strohman (2009). *Search Engines - Information Retrieval in Practice*. Pearson Education.

References


Croft, W. Bruce, Donald Metzler, and Trevor Strohman (2009). *Search Engines - Information Retrieval in Practice*. Pearson Education.


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

Questions?