Introduction to Information Retrieval

http://informationretrieval.org

IIR 11: Probabilistic Information Retrieval

Hinrich Schütze

Institute for Natural Language Processing, Universität Stuttgart

2011-05-23

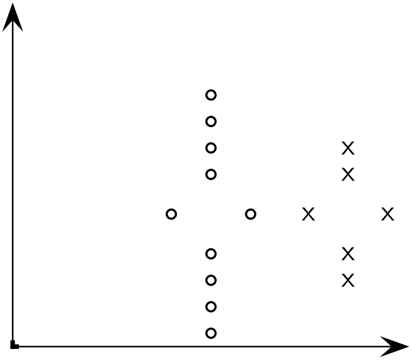
Overview

Outline

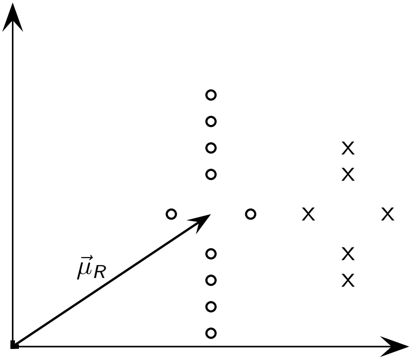
Relevance feedback: Basic idea

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

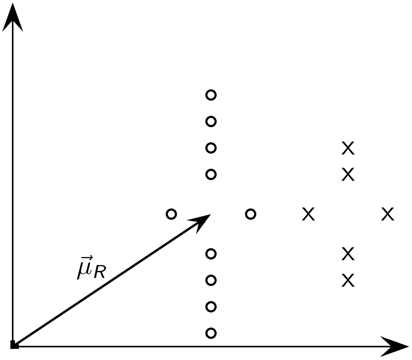
Rocchio illustrated



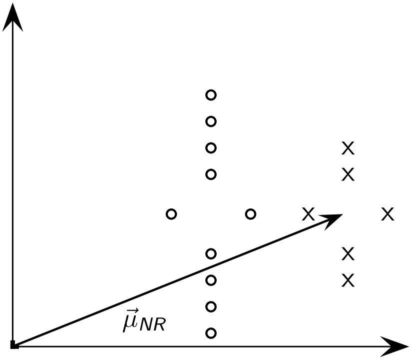
Rocchio illustrated



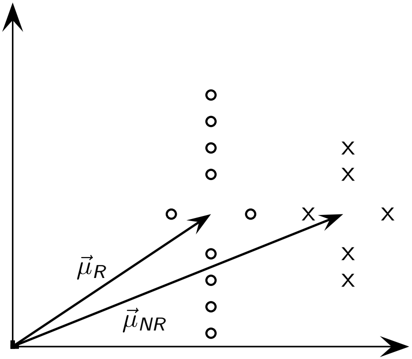
Rocchio illustrated



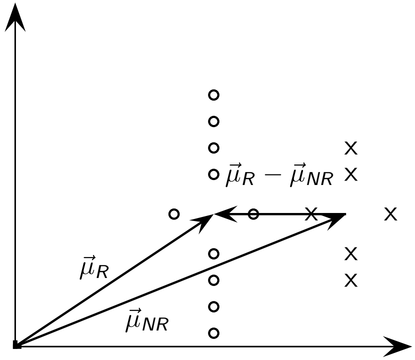
Rocchio illustrated



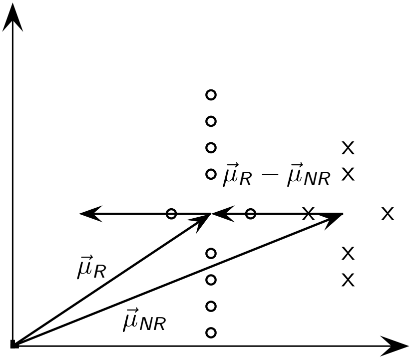
Rocchio illustrated



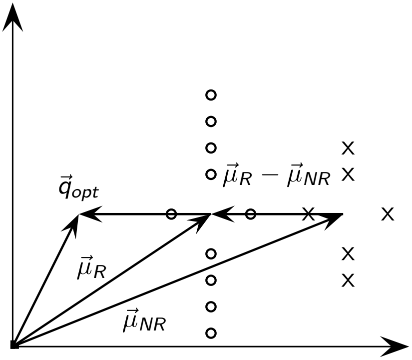
Rocchio illustrated



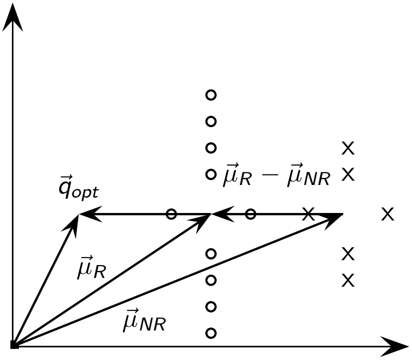
Rocchio illustrated



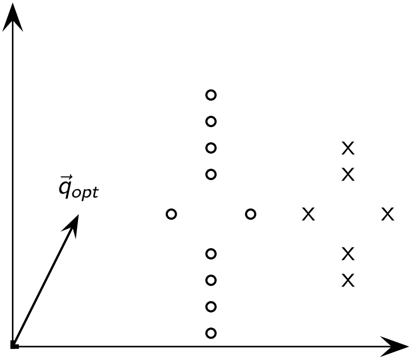
Rocchio illustrated



Rocchio illustrated



Rocchio illustrated



Types of query expansion

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

Query expansion at search engines

- Main source of query expansion at search engines: query logs
- Example 1: After issuing the query [herbs], users frequently search for [herbal remedies].
  - → "herbal remedies" is potential expansion of "herb".
- Example 2: Users searching for [flower pix] frequently click on the URL photobucket.com/flower. Users searching for [flower clipart] frequently click on the same URL.
  - → "flower clipart" and "flower pix" are potential expansions of each other.

Take-away today

- Probabilistically grounded approach to IR
- Probability Ranking Principle
- Models: BIM, BM25
- Assumptions these models make

Outline

Relevance feedback from last lecture

Relevance feedback from last lecture

- Previous lecture: in relevance feedback, the user marks documents as relevant/nonrelevant

Relevance feedback from last lecture

- Previous lecture: in relevance feedback, the user marks documents as relevant/nonrelevant
- Given some known relevant and nonrelevant documents, we compute weights for non-query terms that indicate how likely they will occur in relevant documents

Relevance feedback from last lecture

- Previous lecture: in relevance feedback, the user marks documents as relevant/nonrelevant
- Given some known relevant and nonrelevant documents, we compute weights for non-query terms that indicate how likely they will occur in relevant documents
- Today: develop a probabilistic approach for relevance feedback and also a general probabilistic model for IR

Probabilistic Approach to Retrieval

Probabilistic Approach to Retrieval

- Given a user information need (represented as a query) and a collection of documents (transformed into document representations), a system must determine how well the documents satisfy the query

Probabilistic Approach to Retrieval

- Given a user information need (represented as a query) and a collection of documents (transformed into document representations), a system must determine how well the documents satisfy the query
  - An IR system has an uncertain understanding of the user query, and makes an uncertain guess of whether a document satisfies the query

Probabilistic Approach to Retrieval

- Given a user information need (represented as a query) and a collection of documents (transformed into document representations), a system must determine how well the documents satisfy the query
  - An IR system has an uncertain understanding of the user query, and makes an uncertain guess of whether a document satisfies the query
- Probability theory provides a principled foundation for such reasoning under uncertainty

Probabilistic Approach to Retrieval

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

Probabilistic IR Models at a Glance

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model
  - Probability ranking principle

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model
  - Probability ranking principle
    - Binary Independence Model, BestMatch25 (Okapi)

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model
  - Probability ranking principle
    - Binary Independence Model, BestMatch25 (Okapi)
- Bayesian networks for text retrieval

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model
  - Probability ranking principle
    - Binary Independence Model, BestMatch25 (Okapi)
- Bayesian networks for text retrieval
- Language model approach to IR

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model
  - Probability ranking principle
    - Binary Independence Model, BestMatch25 (Okapi)
- Bayesian networks for text retrieval
- Language model approach to IR
  - Important recent work, will be covered in the next lecture

Probabilistic IR Models at a Glance

- Classical probabilistic retrieval model
  - Probability ranking principle
    - Binary Independence Model, BestMatch25 (Okapi)
- Bayesian networks for text retrieval
- Language model approach to IR
  - Important recent work, will be covered in the next lecture
- Probabilistic methods are one of the oldest but also one of the currently hottest topics in IR

Exercise: Probabilistic model vs. other models

Exercise: Probabilistic model vs. other models

- Boolean model

Exercise: Probabilistic model vs. other models

- Boolean model
  - Probabilistic models support ranking and thus are better than the simple Boolean model.

Exercise: Probabilistic model vs. other models

- Boolean model
  - Probabilistic models support ranking and thus are better than the simple Boolean model.
- Vector space model

Exercise: Probabilistic model vs. other models

- Boolean model
  - Probabilistic models support ranking and thus are better than the simple Boolean model.
- Vector space model
  - The vector space model is also a formally defined model that supports ranking.

Exercise: Probabilistic model vs. other models

- Boolean model
  - Probabilistic models support ranking and thus are better than the simple Boolean model.
- Vector space model
  - The vector space model is also a formally defined model that supports ranking.
  - Why would we want to look for an alternative to the vector space model?

Probabilistic vs. vector space model

Probabilistic vs. vector space model

- Vector space model: rank documents according to similarity to query.

Probabilistic vs. vector space model

- Vector space model: rank documents according to similarity to query.
- 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?"

Probabilistic vs. vector space model

- Vector space model: rank documents according to similarity to query.
- 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?"
- The most similar document can be highly relevant or completely nonrelevant.

Probabilistic vs. vector space model

- Vector space model: rank documents according to similarity to query.
- 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?"
- The most similar document can be highly relevant or completely nonrelevant.
- Probability theory is arguably a cleaner formalization of what we really want an IR system to do: give relevant documents to the user.

Outline

Basic Probability Theory

Basic Probability Theory

- For events A and B

Basic Probability Theory

- For events A and B
  - Joint probability $P(A \cap B)$ of both events occurring

Basic Probability Theory

- 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

Basic Probability Theory

- 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
- Chain rule gives fundamental relationship between joint and conditional probabilities:

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

Basic Probability Theory

- 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
- Chain rule gives fundamental relationship between joint and conditional probabilities:

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

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

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

Basic Probability Theory

- 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
- Chain rule gives fundamental relationship between joint and conditional probabilities:

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

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

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

- 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:

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

Basic Probability Theory

Basic Probability Theory

Bayes' Rule for inverting conditional probabilities:

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

Basic Probability Theory

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:

Basic Probability Theory

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)

Basic Probability Theory

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

Basic Probability Theory

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

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

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

Outline

The Document Ranking Problem

The Document Ranking Problem

- Ranked retrieval setup: given a collection of documents, the user issues a query, and an ordered list of documents is returned

The Document Ranking Problem

- Ranked retrieval setup: given a collection of documents, the user issues a query, and an ordered list of documents is returned
- Assume binary notion of relevance: $R_{d,q}$ is a random dichotomous variable, such that

The Document Ranking Problem

- Ranked retrieval setup: given a collection of documents, the user issues a query, and an ordered list of documents is returned
- Assume binary notion of relevance: $R_{d,q}$ is a random dichotomous variable, such that
  - $R_{d,q} = 1$ if document d is relevant w.r.t query q

The Document Ranking Problem

- Ranked retrieval setup: given a collection of documents, the user issues a query, and an ordered list of documents is returned
- Assume binary notion of relevance: $R_{d,q}$ is a random dichotomous variable, such that
  - $R_{d,q} = 1$ if document d is relevant w.r.t query q
  - $R_{d,q} = 0$ otherwise

The Document Ranking Problem

- Ranked retrieval setup: given a collection of documents, the user issues a query, and an ordered list of documents is returned
- Assume binary notion of relevance: $R_{d,q}$ is a random dichotomous variable, such that
  - $R_{d,q} = 1$ if document d is relevant w.r.t query q
  - $R_{d,q} = 0$ otherwise
- Probabilistic ranking orders documents decreasingly by their estimated probability of relevance w.r.t. query: $P(R = 1|d, q)$

The Document Ranking Problem

- Ranked retrieval setup: given a collection of documents, the user issues a query, and an ordered list of documents is returned
- Assume binary notion of relevance: $R_{d,q}$ is a random dichotomous variable, such that
  - $R_{d,q} = 1$ if document d is relevant w.r.t query q
  - $R_{d,q} = 0$ otherwise
- Probabilistic ranking orders documents decreasingly by their estimated probability of relevance w.r.t. query: $P(R = 1|d, q)$
- Assume that the relevance of each document is independent of the relevance of other documents

Probability Ranking Principle (PRP)

Probability Ranking Principle (PRP)

- PRP in brief

Probability Ranking Principle (PRP)

- PRP in brief
  - 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

Probability Ranking Principle (PRP)

- PRP in brief
  - 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
- PRP in full

Probability Ranking Principle (PRP)

- PRP in brief
  - 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
- PRP in full
  - If [the IR] system's response to each [query] is a ranking of the documents [...] in order of decreasing probability of relevance to the [query], where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose, the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data

Binary Independence Model (BIM)

Binary Independence Model (BIM)

- Traditionally used with the PRP

Assumptions:

Binary Independence Model (BIM)

- Traditionally used with the PRP

Assumptions:

- 'Binary' (equivalent to Boolean): documents and queries represented as binary term incidence vectors

Binary Independence Model (BIM)

- Traditionally used with the PRP

Assumptions:

- 'Binary' (equivalent to Boolean): documents and queries represented as binary term incidence vectors
  - E.g., document d represented by vector $\vec{x} = (x_1, \ldots, x_M)$, where $x_t = 1$ if term t occurs in d and $x_t = 0$ otherwise

Binary Independence Model (BIM)

- Traditionally used with the PRP

Assumptions:

- 'Binary' (equivalent to Boolean): documents and queries represented as binary term incidence vectors
  - E.g., document d represented by vector $\vec{x} = (x_1, \ldots, x_M)$, where $x_t = 1$ if term t occurs in d and $x_t = 0$ otherwise
  - Different documents may have the same vector representation

Binary Independence Model (BIM)

- Traditionally used with the PRP

Assumptions:

- 'Binary' (equivalent to Boolean): documents and queries represented as binary term incidence vectors
  - E.g., document d represented by vector $\vec{x} = (x_1, \ldots, x_M)$, where $x_t = 1$ if term t occurs in d and $x_t = 0$ otherwise
  - Different documents may have the same vector representation
- 'Independence': no association between terms (not true, but practically works - 'naive' assumption of Naive Bayes models)

Binary incidence matrix

	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	
...							

Each document is represented as a binary vector $\in \{0, 1\}^{|V|}$.

Binary Independence Model

Binary Independence Model

To make a probabilistic retrieval strategy precise, need to estimate how terms in documents contribute to relevance

- Find measurable statistics (term frequency, document frequency, document length) that affect judgments about document relevance

Binary Independence Model

To make a probabilistic retrieval strategy precise, need to estimate how terms in documents contribute to relevance

- Find measurable statistics (term frequency, document frequency, document length) that affect judgments about document relevance
- Combine these statistics to estimate the probability $P(R|d,q)$ of document relevance

Binary Independence Model

To make a probabilistic retrieval strategy precise, need to estimate how terms in documents contribute to relevance

- Find measurable statistics (term frequency, document frequency, document length) that affect judgments about document relevance
- Combine these statistics to estimate the probability $P(R|d,q)$ of document relevance
- Next: how exactly we can do this

Binary Independence Model

Binary Independence Model

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

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

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

Binary Independence Model

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

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

- $P(\vec{x}|R=1,\vec{q})$ and $P(\vec{x}|R=0,\vec{q})$: probability that if a relevant or nonrelevant document is retrieved, then that document's representation is $\vec{x}$
- Use statistics about the document collection to estimate these probabilities

Binary Independence Model

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

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

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

Binary Independence Model

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

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

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

- $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$: prior probability of retrieving a relevant or nonrelevant document for a query $\vec{q}$

Binary Independence Model

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

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

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

- $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$: prior probability of retrieving a relevant or 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

Binary Independence Model

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

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

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

- $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$: prior probability of retrieving a relevant or 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:

Binary Independence Model

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

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

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

- $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$: prior probability of retrieving a relevant or 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)

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 $P(R=1|\vec{x},\vec{q})$
- Easier: rank documents by their odds of relevance (gives same ranking)

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

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

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

Deriving a Ranking Function for Query Terms (2)

Deriving a Ranking Function for Query Terms (2)

It is at this point that 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):

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

So:

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

Exercise

Exercise

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

Exercise

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

Why is this wrong? Good example?

Exercise

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

Why is this wrong? Good example?

PRP assumes that the relevance of each document is independent of the relevance of other documents.

Exercise

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

Why is this wrong? Good example?

PRP assumes that the relevance of each document is independent of the relevance of other documents.

Why is this wrong? Good example?

Deriving a Ranking Function for Query Terms (3)

Deriving a Ranking Function for Query Terms (3)

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

Deriving a Ranking Function for Query Terms (3)

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

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

Deriving a Ranking Function for Query Terms (4)

Deriving a Ranking Function for Query Terms (4)

- Let $p_t = P(x_t = 1|R = 1, \vec{q})$ be the probability of a term appearing in relevant document

Deriving a Ranking Function for Query Terms (4)

- Let $p_t = P(x_t = 1|R = 1, \vec{q})$ be the probability of a term appearing in relevant document

Deriving a Ranking Function for Query Terms (4)

- Let $p_t = P(x_t = 1|R = 1, \vec{q})$ be the probability of a term appearing in relevant document
- Let $u_t = P(x_t = 1|R = 0, \vec{q})$ be the probability of a term appearing in a nonrelevant document

Deriving a Ranking Function for Query Terms (4)

- Let $p_t = P(x_t = 1|R = 1, \vec{q})$ be the probability of a term appearing in relevant document
- Let $u_t = P(x_t = 1|R = 0, \vec{q})$ be the probability of a term appearing in a nonrelevant document

Deriving a Ranking Function for Query Terms (4)

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

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

Deriving a Ranking Function for Query Terms

Deriving a Ranking Function for Query Terms

Additional simplifying assumption: terms not occurring in the query are equally likely to occur in relevant and nonrelevant documents

- If $q_t = 0$, then $p_t = u_t$

Now we need only to consider terms in the products that appear in the query:

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

Deriving a Ranking Function for Query Terms

Additional simplifying assumption: terms not occurring in the query are equally likely to occur in relevant and nonrelevant documents

- If $q_t = 0$, then $p_t = u_t$

Now we need only to consider terms in the products that appear in the query:

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

- The left product is over query terms found in the document and the right product is over query terms not found in the document

Deriving a Ranking Function for Query Terms

Deriving a Ranking Function for Query Terms

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

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

- 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.

Deriving a Ranking Function for Query Terms

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

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

- 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.
- → The only quantity that needs to be estimated to rank documents w.r.t a query is the left product

Deriving a Ranking Function for Query Terms

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

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

- 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.
- → The only quantity that needs to be estimated to rank documents w.r.t a query is the left product
- 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

Deriving a Ranking Function for Query Terms

Equivalent: rank documents using the log odds ratios for the terms in the query c_t:

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

- The odds ratio is the ratio of two odds: (i) the odds of the term appearing if the document is relevant ($p_t/(1-p_t)$), and (ii) the odds of the term appearing if the document is nonrelevant ($u_t/(1-u_t)$)

Deriving a Ranking Function for Query Terms

Equivalent: rank documents using the log odds ratios for the terms in the query c_t:

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

- The odds ratio is the ratio of two odds: (i) the odds of the term appearing if the document is relevant $(p_t/(1-p_t))$, and (ii) the odds of the term appearing if the document is nonrelevant $(u_t/(1-u_t))$
- $c_t = 0$: term has equal odds of appearing in relevant and nonrelevant docs

Deriving a Ranking Function for Query Terms

Equivalent: rank documents using the log odds ratios for the terms in the query c_t:

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

- The odds ratio is the ratio of two odds: (i) the odds of the term appearing if the document is relevant ($p_t/(1-p_t)$), and (ii) the odds of the term appearing if the document is nonrelevant ($u_t/(1-u_t)$)
- $c_t = 0$: term has equal odds of appearing in relevant and nonrelevant docs
- c_t positive: higher odds to appear in relevant documents

Deriving a Ranking Function for Query Terms

Equivalent: rank documents using the log odds ratios for the terms in the query c_t:

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

- The odds ratio is the ratio of two odds: (i) the odds of the term appearing if the document is relevant ($p_t/(1-p_t)$), and (ii) the odds of the term appearing if the document is nonrelevant ($u_t/(1-u_t)$)
- $c_t = 0$: term has equal odds of appearing in relevant and nonrelevant docs
- c_t positive: higher odds to appear in relevant documents
- c_t negative: higher odds to appear in nonrelevant documents

Term weight c_t in BIM

Term weight c_t in BIM

- $c_t = \log \frac{p_t}{(1-p_t)} - \log \frac{u_t}{1-u_t}$ functions as a term weight.

Term weight c_t in BIM

- $c_t = \log \frac{p_t}{(1-p_t)} - \log \frac{u_t}{1-u_t}$ functions as a term weight.
- Retrieval status value for document d: $RSV_d = \sum_{x_t=q_t=1} c_t$.

Term weight c_t in BIM

- $c_t = \log \frac{p_t}{(1-p_t)} - \log \frac{u_t}{1-u_t}$ functions as a term weight.
- Retrieval status value for document d: $RSV_d = \sum_{x_t=q_t=1} c_t$.
- So BIM and vector space model are identical on an operational level ...

Term weight c_t in BIM

- $c_t = \log \frac{p_t}{(1-p_t)} - \log \frac{u_t}{1-u_t}$ functions as a term weight.
- Retrieval status value for document d: $RSV_d = \sum_{x_t=q_t=1} c_t$.
- So BIM and vector space model are identical on an operational level . . .
- . . . except that the term weights are different.

Term weight c_t in BIM

- $c_t = \log \frac{p_t}{(1-p_t)} - \log \frac{u_t}{1-u_t}$ functions as a term weight.
- Retrieval status value for document d: $RSV_d = \sum_{x_t=q_t=1} c_t$.
- So BIM and vector space model are identical on an operational level . . .
- . . . except that the term weights are different.
- In particular: we can use the same data structures (inverted index etc) for the two models.

How to compute probability estimates

How to compute probability estimates

For each term t in a query, estimate c_t in the whole 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

$$p_t = s/S$$

$$u_t = (\mathrm{df}_t - s)/(N - S)$$

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

Avoiding zeros

Avoiding zeros

- If any of the counts is a zero, then the term weight is not well-defined.

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.

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 (expected likelihood estimation = ELE)

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 (expected likelihood estimation = ELE)
- For example, use $S - s + 0.5$ in formula for $S - s$

Exercise

Exercise

- Query: `Obama health plan`
- Doc1: `Obama rejects allegations about his own bad health`
- Doc2: `The plan is to visit Obama`
- Doc3: `Obama raises concerns with US health plan reforms`

Exercise

- Query: `Obama health plan`
- Doc1: `Obama rejects allegations about his own bad health`
- Doc2: `The plan is to visit Obama`
- Doc3: `Obama raises concerns with US health plan reforms`

Estimate the probability that the above documents are relevant to the query. Use a contingency table. These are the only three documents in the collection

Simplifying assumption

Simplifying assumption

- Assuming that relevant documents are a very small percentage of the collection, approximate statistics for nonrelevant documents by statistics from the whole collection

Simplifying assumption

- Assuming that relevant documents are a very small percentage of the collection, approximate statistics for nonrelevant documents by statistics from the whole collection
- Hence, u_t (the probability of term occurrence in nonrelevant documents for a query) is df_t/N and

$$\log[(1-u_t)/u_t] = \log[(N-\mathrm{df}_t)/\mathrm{df}_t] \approx \log N/\mathrm{df}_t$$

Simplifying assumption

- Assuming that relevant documents are a very small percentage of the collection, approximate statistics for nonrelevant documents by statistics from the whole collection
- Hence, u_t (the probability of term occurrence in nonrelevant documents for a query) is df_t/N and

$$\log[(1-u_t)/u_t] = \log[(N-\mathrm{df}_t)/\mathrm{df}_t] \approx \log N/\mathrm{df}_t$$

- This should look familiar to you ...

Simplifying assumption

- Assuming that relevant documents are a very small percentage of the collection, approximate statistics for nonrelevant documents by statistics from the whole collection
- Hence, u_t (the probability of term occurrence in nonrelevant documents for a query) is df_t/N and

$$\log[(1-u_t)/u_t] = \log[(N-\mathrm{df}_t)/\mathrm{df}_t] \approx \log N/\mathrm{df}_t$$

- This should look familiar to you ...
- The above approximation cannot easily be extended to relevant documents

Probability estimates in relevance feedback

Probability estimates in relevance feedback

- Statistics of relevant documents (p_t) in relevance feedback can be estimated using maximum likelihood estimation or ELE (add 0.5).

Probability estimates in relevance feedback

- Statistics of relevant documents (p_t) in relevance feedback can be estimated using maximum likelihood estimation or ELE (add 0.5).
  - Use the frequency of term occurrence in known relevant documents.

Probability estimates in relevance feedback

- Statistics of relevant documents (p_t) in relevance feedback can be estimated using maximum likelihood estimation or ELE (add 0.5).
  - Use the frequency of term occurrence in known relevant documents.
- This is the basis of probabilistic approaches to relevance feedback weighting in a feedback loop

Probability estimates in relevance feedback

- Statistics of relevant documents (p_t) in relevance feedback can be estimated using maximum likelihood estimation or ELE (add 0.5).
    - Use the frequency of term occurrence in known relevant documents.
- This is the basis of probabilistic approaches to relevance feedback weighting in a feedback loop
- The exercise we just did was a probabilistic relevance feedback exercise since we were assuming the availability of relevance judgments.

Probability estimates in adhoc retrieval

Probability estimates in adhoc retrieval

- Ad-hoc retrieval: no user-supplied relevance judgments available

Probability estimates in adhoc retrieval

- Ad-hoc retrieval: no user-supplied relevance judgments available
- In this case: assume that p_t is constant over all terms x_t in the query and that $p_t = 0.5$

Probability estimates in adhoc retrieval

- Ad-hoc retrieval: no user-supplied relevance judgments available
- In this case: assume that p_t is constant over all terms x_t in the query and that $p_t = 0.5$
- Each term is equally likely to occur in a relevant document, and so the p_t and $(1 - p_t)$ factors cancel out in the expression for RSV

Probability estimates in adhoc retrieval

- Ad-hoc retrieval: no user-supplied relevance judgments available
- In this case: assume that p_t is constant over all terms x_t in the query and that $p_t = 0.5$
- Each term is equally likely to occur in a relevant document, and so the p_t and $(1 - p_t)$ factors cancel out in the expression for RSV
- Weak estimate, but doesn't disagree violently with expectation that query terms appear in many but not all relevant documents

Probability estimates in adhoc retrieval

- Ad-hoc retrieval: no user-supplied relevance judgments available
- In this case: assume that p_t is constant over all terms x_t in the query and that $p_t = 0.5$
- Each term is equally likely to occur in a relevant document, and so the p_t and $(1 - p_t)$ factors cancel out in the expression for RSV
- Weak estimate, but doesn't disagree violently with expectation that query terms appear in many but not all relevant documents
- Combining this method with the earlier approximation for u_t, the document ranking is determined simply by which query terms occur in documents scaled by their idf weighting

Probability estimates in adhoc retrieval

- Ad-hoc retrieval: no user-supplied relevance judgments available
- In this case: assume that p_t is constant over all terms x_t in the query and that $p_t = 0.5$
- Each term is equally likely to occur in a relevant document, and so the p_t and $(1 - p_t)$ factors cancel out in the expression for RSV
- Weak estimate, but doesn't disagree violently with expectation that query terms appear in many but not all relevant documents
- Combining this method with the earlier approximation for u_t, the document ranking is determined simply by which query terms occur in documents scaled by their idf weighting
- For short documents (titles or abstracts) in one-pass retrieval situations, this estimate can be quite satisfactory

Outline

History and summary of assumptions

History and summary of assumptions

- Among the oldest formal models in IR

History and summary of assumptions

- 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

History and summary of assumptions

- 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
- Assumptions for getting reasonable approximations of the needed probabilities (in the BIM):

History and summary of assumptions

- 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
- Assumptions for getting reasonable approximations of the needed probabilities (in the BIM):
  - Boolean representation of documents/queries/relevance

History and summary of assumptions

- 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
- Assumptions for getting reasonable approximations of the needed probabilities (in the BIM):
  - Boolean representation of documents/queries/relevance
  - Term independence

History and summary of assumptions

- 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
- 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

History and summary of assumptions

- 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
- 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?

How different are vector space and BIM?

- They are not that different.

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.

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 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.
- Next: how to add term frequency and length normalization to the probabilistic model.

Okapi BM25: Overview

Okapi BM25: Overview

- Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.

Okapi BM25: Overview

- Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.
- BIM was originally designed for short catalog records of fairly consistent length, and it works reasonably in these contexts

Okapi BM25: Overview

- Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.
- BIM was originally designed for short catalog records of fairly consistent length, and it works reasonably in these contexts
- For modern full-text search collections, a model should pay attention to term frequency and document length

Okapi BM25: Overview

- Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.
- BIM was originally designed for short catalog records of fairly consistent length, and it works reasonably in these contexts
- For modern full-text search collections, a model should pay attention to term frequency and document length
- BestMatch25 (a.k.a BM25 or Okapi) is sensitive to these quantities

Okapi BM25: Overview

- Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.
- BIM was originally designed for short catalog records of fairly consistent length, and it works reasonably in these contexts
- For modern full-text search collections, a model should pay attention to term frequency and document length
- BestMatch25 (a.k.a BM25 or Okapi) is sensitive to these quantities
- BM25 is one of the most widely used and robust retrieval models

Okapi BM25: Starting point

Okapi BM25: Starting point

- The simplest score for document d is just idf weighting of the query terms present in the document:

Okapi BM25: Starting point

- The simplest score for document d is just idf weighting of the query terms present in the document:

Okapi BM25: Starting point

- 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

Okapi BM25 basic weighting

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

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

Okapi BM25 basic weighting

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

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

- tf_{td}: term frequency in document d

Okapi BM25 basic weighting

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

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

- tf_{td}: term frequency in document d
- L_d (L_{ave}): length of document d (average document length in the whole collection)

Okapi BM25 basic weighting

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

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

- tf_{td}: term frequency in document d
- L_d (L_{ave}): length of document d (average document length in the whole collection)
- k_1: tuning parameter controlling the document term frequency scaling

Okapi BM25 basic weighting

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

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

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

Exercise

Exercise

- Interpret BM25 weighting formula for $k_1 = 0$
- Interpret BM25 weighting formula for $k_1 = 1$ and $b = 0$
- Interpret BM25 weighting formula for $k_1 \mapsto \infty$ and $b = 0$
- Interpret BM25 weighting formula for $k_1 \mapsto \infty$ and $b = 1$

Okapi BM25 weighting for long queries

Okapi BM25 weighting for long queries

- For long queries, use similar weighting for query terms

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

Okapi BM25 weighting for long queries

- For long queries, use similar weighting for query terms

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

- tf_{tq}: term frequency in the query q

Okapi BM25 weighting for long queries

- For long queries, use similar weighting for query terms

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

- tf_{tq}: term frequency in the query q
- k_3: tuning parameter controlling term frequency scaling of the query

Okapi BM25 weighting for long queries

- For long queries, use similar weighting for query terms

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

- tf_{tq}: term frequency in the query q
- k_3: tuning parameter controlling term frequency scaling of the query
- No length normalization of queries (because retrieval is being done with respect to a single fixed query)

Okapi BM25 weighting for long queries

- For long queries, use similar weighting for query terms

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

- tf_{tq}: term frequency in the query q
- k_3: tuning parameter controlling term frequency scaling of the query
- No length normalization of queries (because retrieval is being done with respect to a single fixed query)
- The above tuning parameters should ideally be set to optimize performance on a development test collection. In the absence of such optimization, experiments have shown reasonable values are to set k_1 and k_3 to a value between 1.2 and 2 and $b = 0.75$

Which ranking model should I use?

Which ranking model should I use?

- I want something basic and simple → use vector space with tf-idf weighting.

Which ranking model should I use?

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

Which ranking model should I use?

- I want something basic and simple $\rightarrow$ use vector space with tf-idf weighting.
- I want to use a state-of-the-art ranking model with excellent performance $\rightarrow$ use language models or BM25 with tuned parameters
- In between: BM25 or language models with no or just one tuned parameter

Take-away today

- Probabilistically grounded approach to IR
- Probability Ranking Principle
- Models: BIM, BM25
- Assumptions these models make

Resources

- Chapter 11 of IIR
- Resources at `http://ifnlp.org/ir`