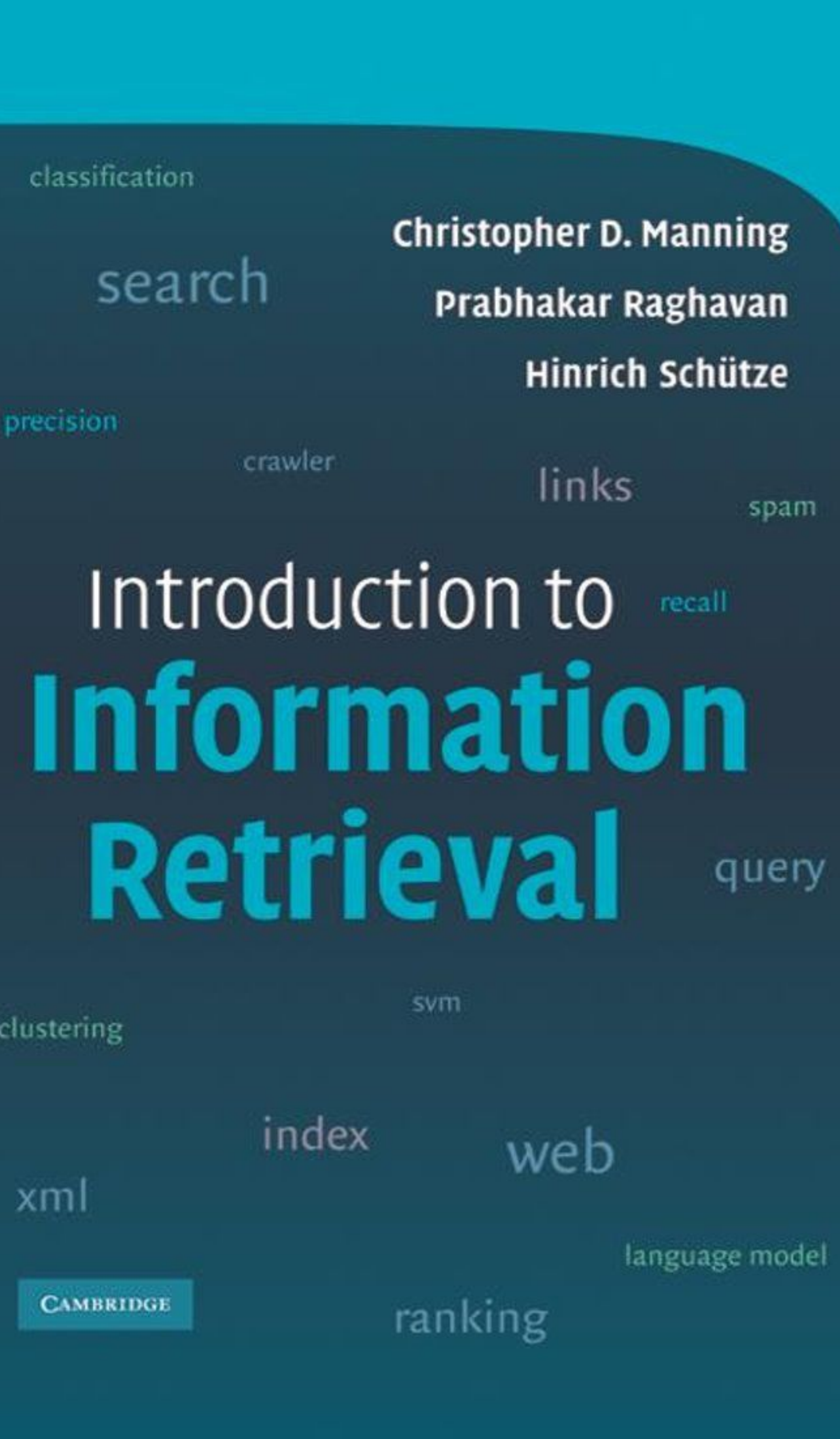
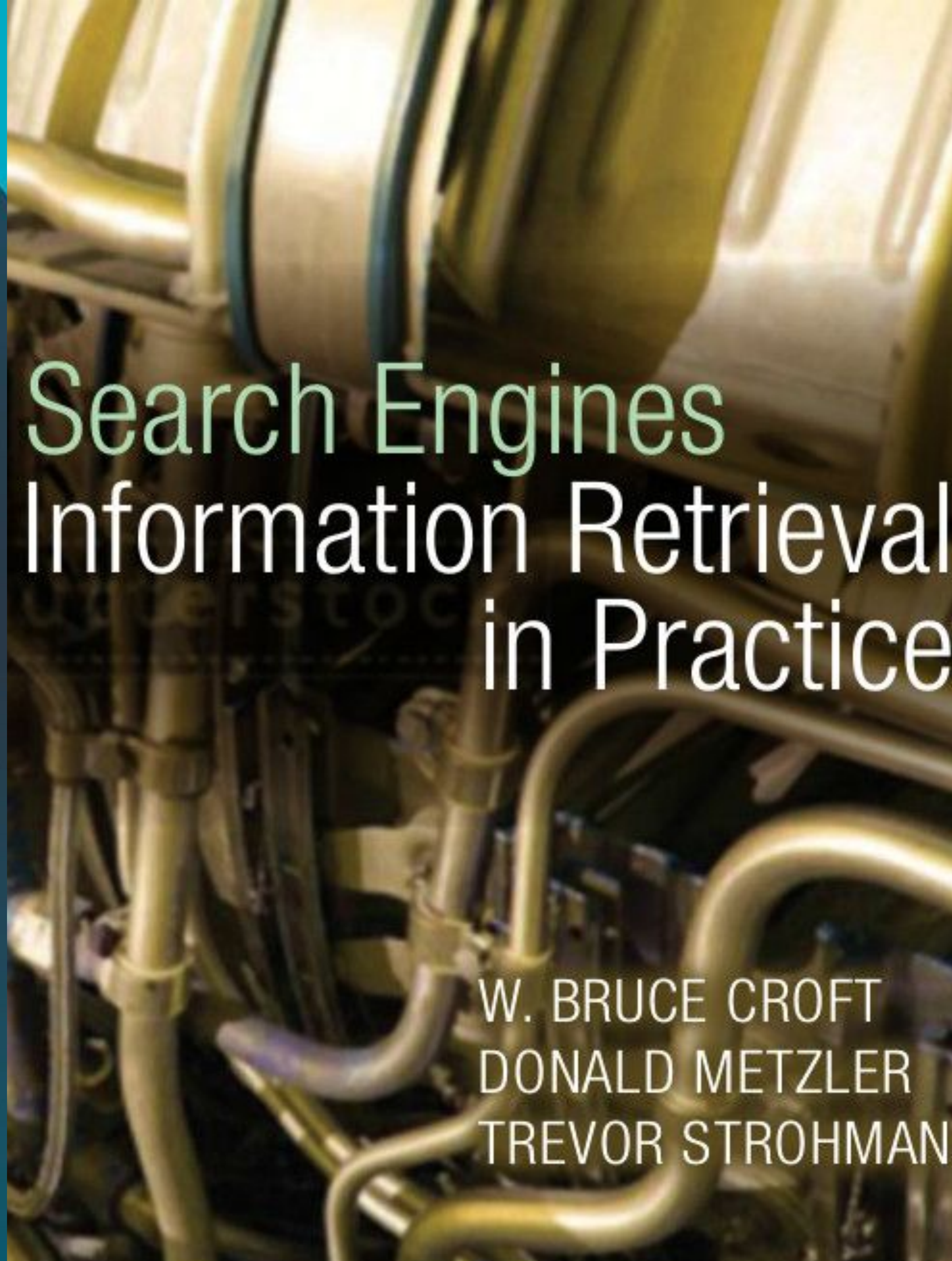
IN4325
Retrieval Models

Claudia Hauff (WIS, TU Delft)

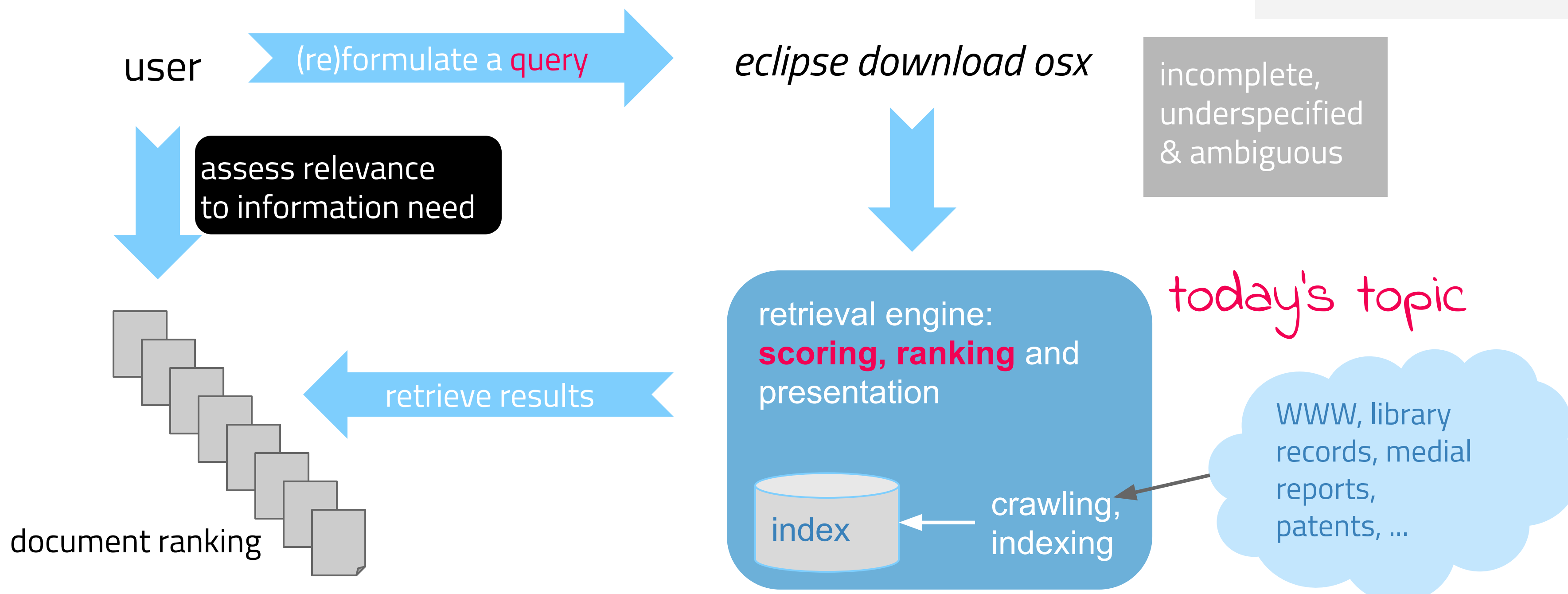

classification
Christopher D. Manning
search
Prabhakar Raghavan
Hinrich Schütze
precision
crawler
links
spam
Introduction to
recall
Information
Retrieval
query
svm
clustering
index
web
xml
language model
CAMBRIDGE
ranking


Search Engines
Information Retrieval
in Practice
W. BRUCE CROFT
DONALD METZLER
TREVOR STROHMAN

The big picture

The essence of IR

Information need: *Looks like I need Eclipse for this job. Where can I download the latest beta version for macOS Sierra?*



Information need
Topic the user wants to know more about

Query
Translation of need into an input for the search engine

Relevance
A document is relevant if it (partially) provides answers to the information need

Coming up with retrieval models

- Goal: formalize human decision making on relevance
- Problem: relevance is a complex concept (topical vs. user relevance)
- Approach:
  - Propose **theories** about relevance
  - Encode theories in **mathematical retrieval models**
  - **Evaluate** the models by comparing them to human relevance decisions (qrels)
  - **Failure analysis**
  - Propose improved retrieval models

repeat

- Good retrieval models produce outputs that **correlate** well with human decisions on relevance

High-level view of retrieval models

1) Given a query, calculate the score of each document in the collection S(Q,D), which is a measure of a **document's match to the query**

2) **Rank** the documents with respect to Q

3) Present the **top-k** ranked documents to the user

Questions:

- How are documents scored?
- What are the assumptions behind the scoring functions?

A succession of retrieval models

Five decades of research in retrieval models

Deep neural nets

Boolean retrieval

Vector space model

Probabi-listic models

Learning to rank

1940s/50s

today

Boolean model

president AND lincoln AND (biography OR life OR birthplace OR gettysburg) AND NOT (automobile OR car)

Boolean model

Also known as *"exact-match retrieval"*.

Documents are retrieved if they match the query specification.

Does not lead to a document ranking but a document **set**.

Assumption: all matching docs have the same relevance.

`grep` is a form of boolean retrieval.

Queries include Boolean logic (`AND, OR, NOT`) and regular expressions (e.g. wildcards).

Advantages:

- **Predictable** outcome
- Easy to **explain**
- Easy to implement
- Boolean query can be specified for any document feature (document text and metadata)

Disadvantages:

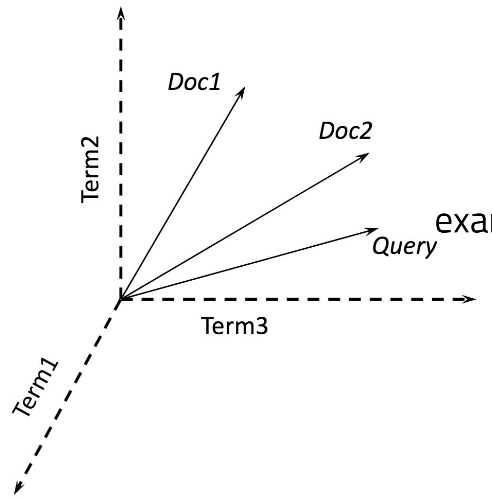
- Effective only for highly skilled users (simple queries do not work)
- No inherent order in the results

Vector space model

A vector space model for automatic indexing. A classic paper by G. Salton et al. from 1975.

Vector space model

S(Q,D)



example 3D space

Documents and **queries** are **represented** as t-dimensional vectors with t being the size of the vocabulary (number of unique words/stems/phrases, can be in the *millions*)

$$D_i = (\boxed{d_{i1}}, d_{i2}, \ldots, d_{it})$$

weight of the $1st$ term

A document collection with n documents can be represented as a matrix of term weights:

document row

$$\begin{array}{lllll} & Term_1 & Term_2 & \ldots & Term_t \\ Doc_1 & d_{11} & d_{12} & \ldots & d_{1t} \\ Doc_2 & d_{21} & d_{22} & \ldots & d_{2t} \\ \vdots & \vdots & & & \\ Doc_n & d_{n1} & d_{n2} & \ldots & d_{nt} \end{array}$$

Given a query (its vector representation), all documents in the corpus (their vector representations) are ranked according to their **similarity** to the query.

i.e. most successful in practice

Most commonly used similarity function: **cosine correlation**. It measures the cosine of the angle between the query and document vectors.

$$Cosine(D_i, Q) = \frac{\sum_{j=1}^{t} d_{ij} \cdot q_j}{\sqrt{\sum_{j=1}^{t} {d_{ij}}^2 \cdot \sum_{j=1}^{t} {q_j}^2}}$$

Vector space model

Inverse collection frequency is also sometimes used

Documents and **queries** are **represented** as t-dimensional vectors with t being the size of the vocabulary (number of unique words/stems/phrases, can be in the *millions*)

$$D_i = (\underbrace{d_{i1}}_{\text{weight of the } 1st \text{ term}}, d_{i2}, \ldots, d_{it})$$

Most common term weighting scheme: **tf.idf** (or variants thereof).

Term frequency weight:

$$\underbrace{tf_{ik}}_{\text{term frequency weight}} = \frac{\overbrace{f_{ik}}^{\text{frequency of term } k \text{ in document } Di}}{\underbrace{\sum_{j=1}^{t} f_{ij}}_{\text{document length}}}$$

Using the log(f) leads to better results (reduced impact of frequent terms)

Inverse document frequency:
Reflects the importance of the term in the corpus.
Two extremes:

1. Term that appears a lot in few documents (i.e. a *discriminating* term). Useful for retrieval.
2. Term that appears a few times in many documents (a stopword). Not useful.

$$\underbrace{idf_k}_{\text{inverse document frequency}} = \log \frac{\overbrace{N}^{\text{Collection size (num. documents)}}}{\underbrace{n_k}_{\text{Num. documents in which term } k \text{ occurs}}}$$

Term weighting in the vector space model:

$$d_{ik} = \frac{(\log(f_{ik}) + 1) \cdot \log(N/n_k)}{\sqrt{\sum_{k=1}^{t} [(\log(f_{ik}) + 1.0) \cdot \log(N/n_k)]^2}}$$

Vector space model

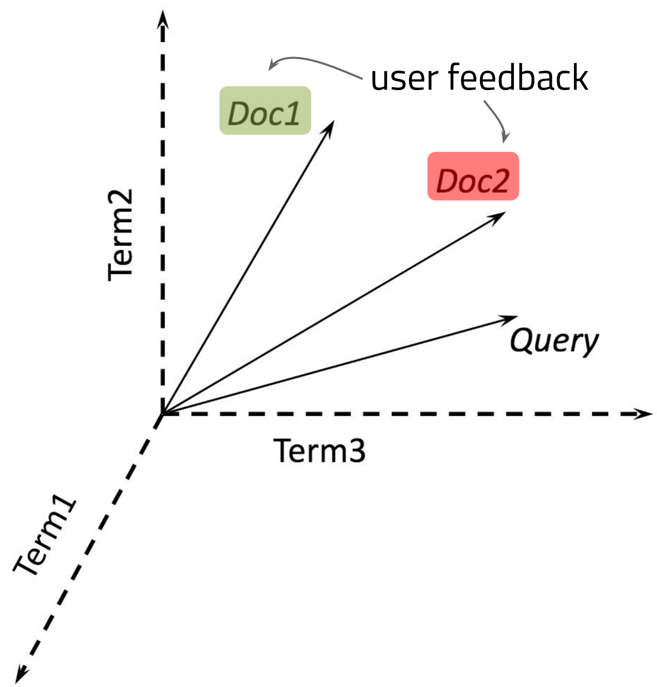
Document and collection frequencies

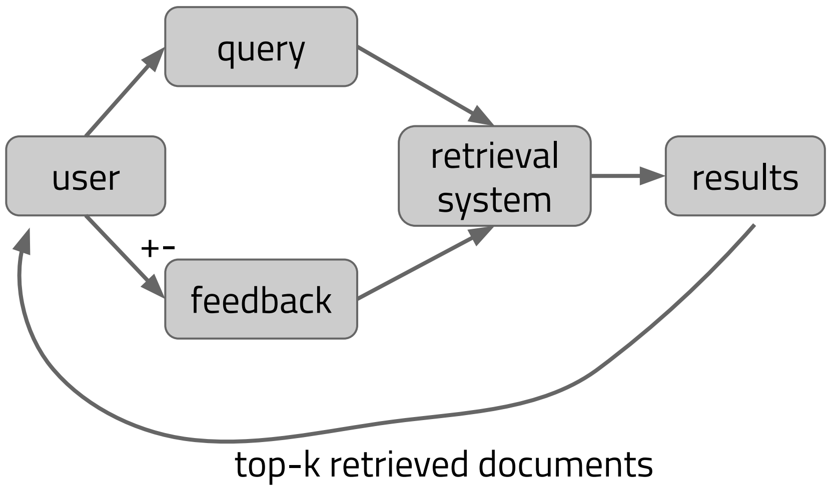
~7 million Wikipedia articles

	df	*cf*	*df/cf*	*idf*
netherlands	63,214	157,659	0.40	2.08
the	3,585,710	91,024,521	0.04	0.33
physics	123,068	248,338	0.50	1.79
actor	147,473	477,476	0.31	1.71
chess	14,690	83,641	0.17	2.72
indeed	55,735	80,597	0.69	2.14

Vector space model

How do we go about incorporating **relevance feedback**?



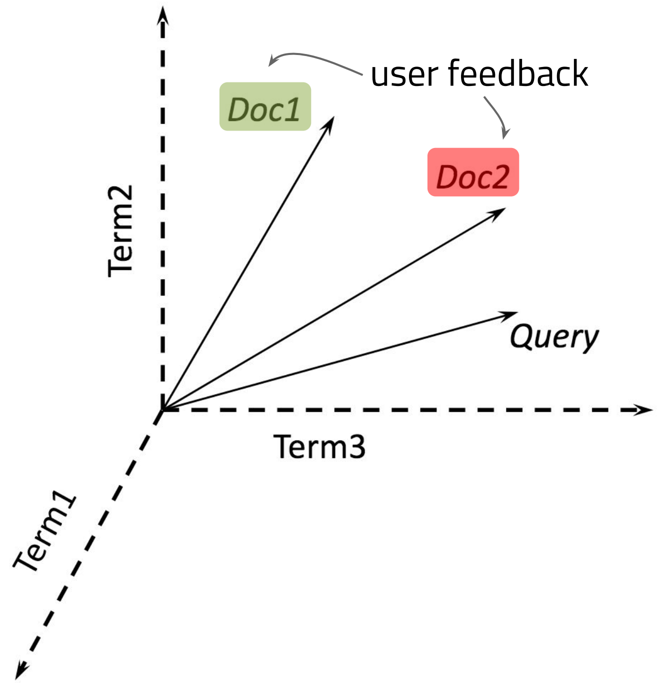


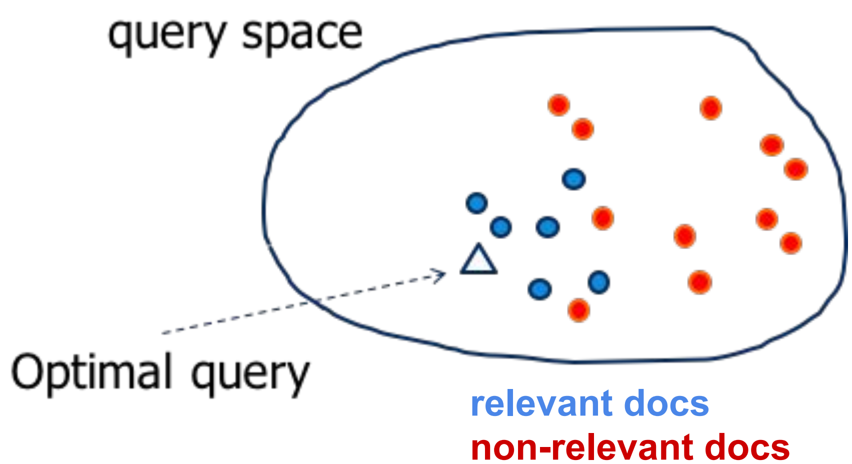
Relevance feedback
employ user feedback to improve the retrieval outcome; requires 2+ rounds of retrieval

Pseudo-relevance feedback
consider the top-k ranked documents to be relevant

Vector space model

How do we go about incorporating **relevance feedback**?



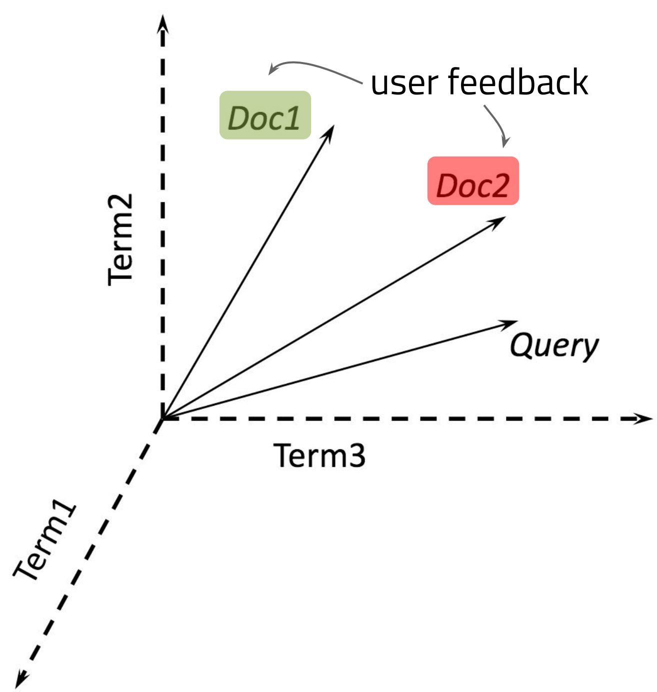


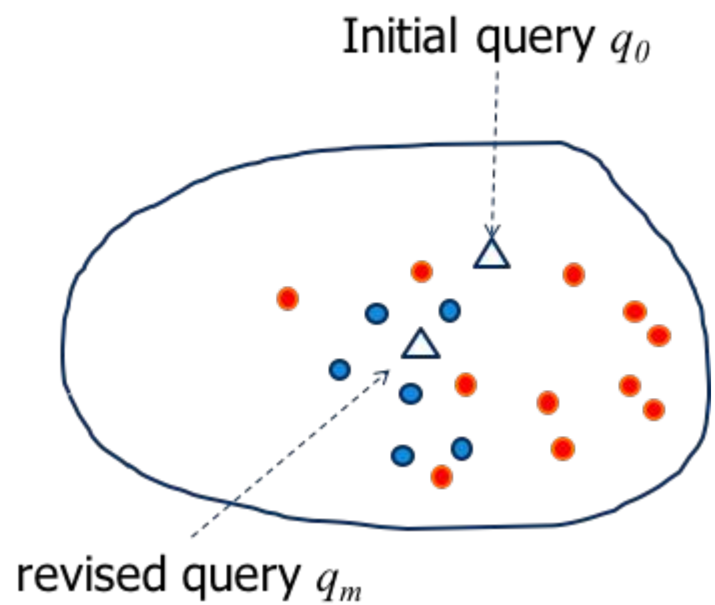
Relevance feedback employ user feedback to improve the retrieval outcome; requires 2+ rounds of retrieval

Cluster hypothesis relevant documents are more similar to each other than they are to non-relevant documents

Vector space model

How do we go about incorporating **relevance feedback**?





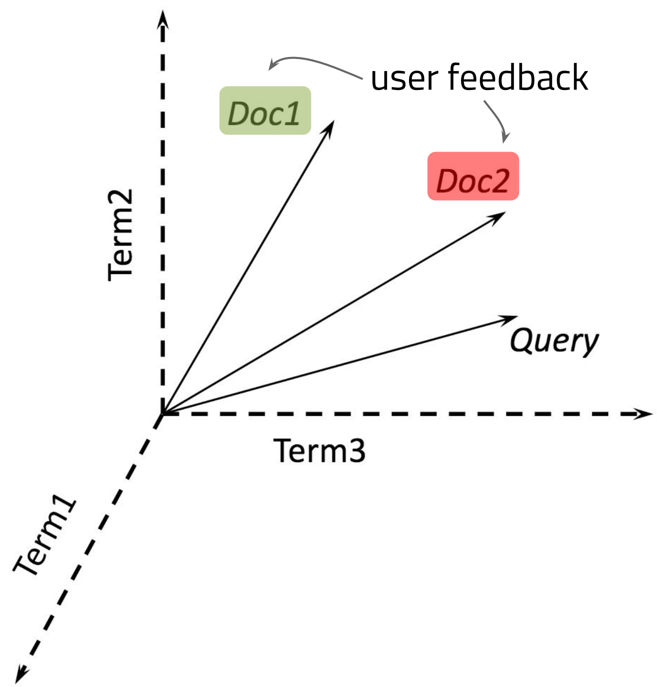
Relevance feedback
employ user feedback to improve the retrieval outcome; requires 2+ rounds of retrieval

Cluster hypothesis
relevant documents are more similar to each other than they are to non-relevant documents

Vector space model

Relevance feedback
employ user feedback to improve the retrieval outcome; requires 2+ rounds of retrieval

How do we go about incorporating **rel. feedback**?



Rocchio's algorithm:
Idea: the *optimal query* should maximize the difference between the average vector representing the relevant documents and the average vector representing the non-relevant documents

Operationalization: modify the weights in query vector Q to produce a new query Q' according to:

weight of jth term in document i

$$q'_j = \alpha . q_j + \beta . \frac{1}{|Rel|} \sum_{D_i \in Rel} d_{ij} - \gamma . \frac{1}{|Nonrel|} \sum_{D_i \in Nonrel} d_{ij}$$

initial weight of query term j

set of identified (non-)relevant docs.

simple approximation: all documents in the corpus

Free parameters: alpha, beta, gamma

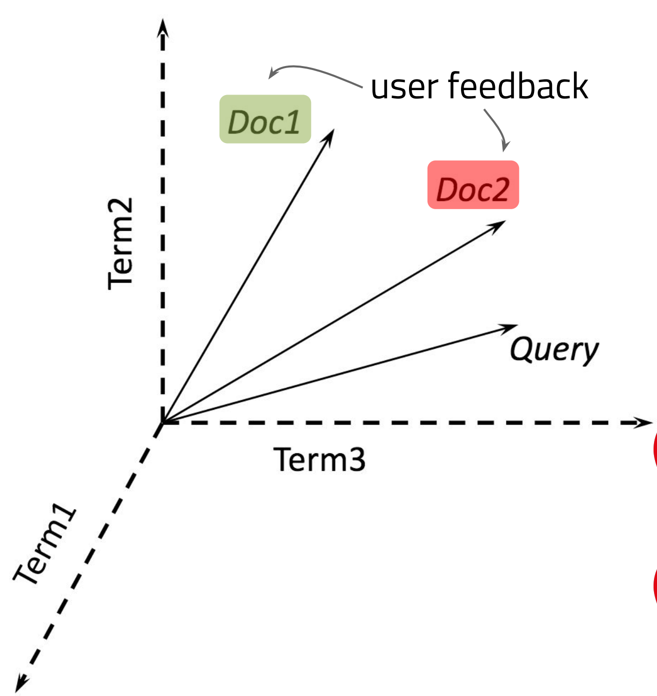
Negative query weights are dropped.

Result: *expanded query* with additional query terms (usually limited to top 50). Followed by a retrieval round.

Vector space model

Relevance feedback employ user feedback to improve the retrieval outcome; requires 2+ rounds of retrieval

How do we go about incorporating **rel. feedback**?



Rocchio's algorithm:
Idea: the *optimal query* should maximize the difference between the average vector representing the relevant documents and the average vector representing the non-relevant documents

Operationalization: modify the weights in query vector Q to produce a new query Q' according to:

$$q'_j = \alpha . q_j + \beta . \frac{1}{|Rel|} \sum_{D_i \in Rel} d_{ij} - \gamma . \frac{1}{|Nonrel|} \sum_{D_i \in Nonrel} d_{ij}$$

Should the weights change over time (more and more retrieval rounds)?

What are good weights for "Find Similar Pages"?

In what cases is Rocchio's algorithm unlikely to work?

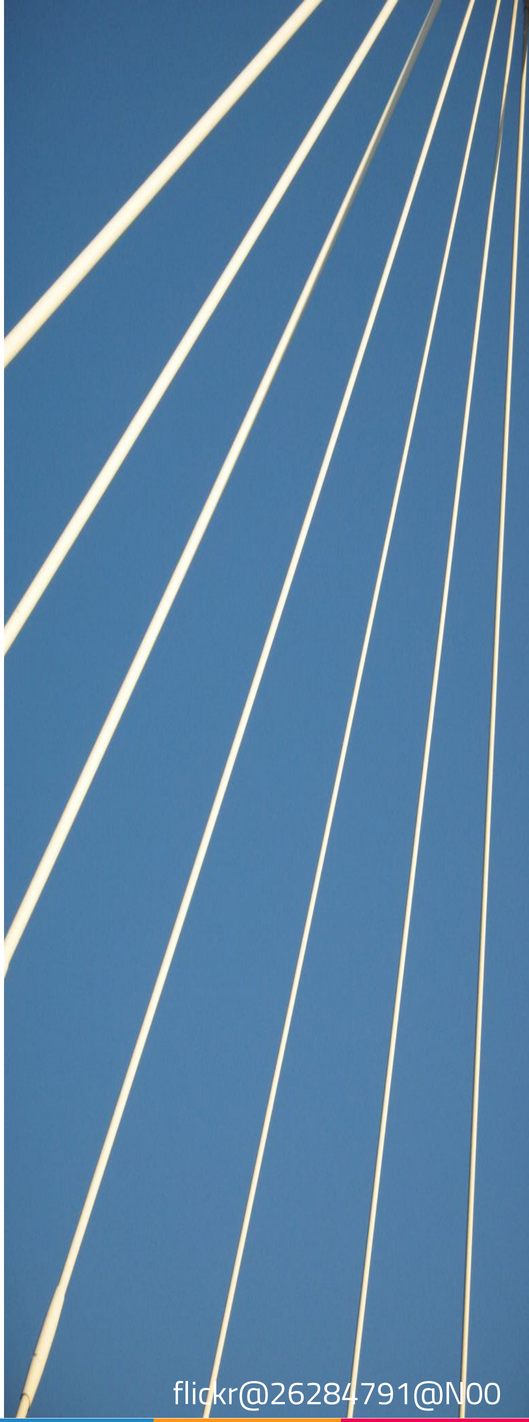
Vector space model

Advantages:

- Easy to implement: three components (tf, idf, doclen)
- Intuitive to understand
- Easy to employ different term weighting schemes, relevance feedback
- Decades of empirical work (developed in the 60s/70s); we know what works and what does not

Issues:

- Assumption: similarity of query and document vectors is **correlated with relevance**
- Assumption: vectors provide a good query and document representation
- In its pure form, it models only topical relevance (though features related to user relevance can be incorporated into the model)



Probabilistic models

Different classes of models fall under this category

Probabilistic models

A whole lot of empiricism, **where is the theory in all of this**?

Probabilistic models

A whole lot of empiricism, **where is the theory in all of this**?

Ideally: **make assumptions explicit** and show theoretically that a ranking algorithm based on the retrieval model will achieve better effectiveness.

Probabilistic retrieval models have a strong foundation for modeling the **uncertainty** that is part of the IR process.

Standard baseline approaches today are probabilistic.

Stephen Robertson

Probabilistic Ranking Principle

Assumption: a document's relevance is independent of other documents.

We still do not know how to estimate the probability of relevance.

Which proved to be great for IR researchers; many different ways to estimate these probabilities were proposed.

"*If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of* ***decreasing probability of relevance*** *to the user who submitted the request, 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.*

http://www.staff.city.ac.uk/~sb317/papers/ProbabilityRankingPrinciple.pdf

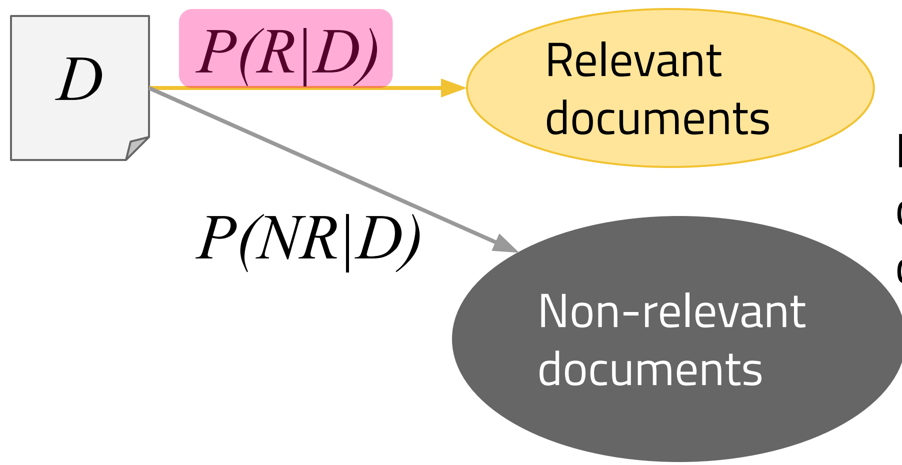
Probabilistic models:

Binary Independence Model

Binary Independence Model

conditional probability of relevance given doc. representation D

Bayes Decision Rule: Given a **new** document D, it should be classified as relevant if $P(R|D)>P(NR|D)$



Binary relevance: two document sets per query

How can we compute $P(R|D)$ and $P(NR|D)$?

We can’t easily. Lets compute $P(D|R)$ and $P(D|NR)$ instead.

Binary Independence Model

set of relevant documents

Lets compute $P(\boxed{D}|\boxed{R})$ and $P(D|NR)$ instead.

new document

Useful, as $P(D|R)$ and $P(R|D)$ are related (Bayes' rule):

$$P(R|D) = \frac{P(D|R)\boxed{P(R)}}{\boxed{P(D)}}$$

A priori probability of relevance: how likely is any document relevant?

Normalizing constant

We then rank documents by their **likelihood ratio**:

$$\frac{P(D|R)}{P(D|NR)}$$

Binary Independence Model

How do we compute $P(D|R)$ and $P(D|NR)$?

Model assumptions:

- Documents are represented as vector of **binary** features (1 if a term is present in the document and 0 otherwise)
- **Term independence** (same for R and NR):

$$P(D|R) = \prod_{i=1}^{t} P(d_i|R)$$

not realistic, but simplifies the maths

Likelihood ratio:

probability that term i occurs in a document from the relevant set

$$\frac{P(D|R)}{P(D|NR)} = \prod_{i:d_i=1} \frac{p_i}{s_i} \cdot \prod_{i:d_i=0} \frac{1-p_i}{1-s_i}$$

product over the terms with a value of 1 in the document representation

probability that term i occurs in a document from the non-relevant set

Binary Independence Model

$$\frac{P(D|R)}{P(D|NR)} = \prod_{i:d_i=1} \frac{p_i}{s_i} \cdot \prod_{i:d_i=0} \frac{1-p_i}{1-s_i}$$

$$= \prod_{i:d_i=1} \frac{p_i}{s_i} \cdot \overbrace{\left(\prod_{i:d_i=1} \frac{1-s_i}{1-p_i} \cdot \prod_{i:d_i=1} \frac{1-p_i}{1-s_i} \right)} \cdot \prod_{i:d_i=0} \frac{1-p_i}{1-s_i}$$

$$= \prod_{i:d_i=1} \frac{p_i(1-s_i)}{s_i(1-p_i)} \cdot \prod_{i} \frac{1-p_i}{1-s_i}$$

product over all terms, and thus the same for all documents -> ignore

$$\propto \sum_{i:d_i=1} \log \frac{p_i(1-s_i)}{s_i(1-p_i)}$$

logarithm avoids accuracy issues of multiplying a lot of small numbers

BIM scoring function

Binary Independence Model

$$\sum_{i:d_i=1} \log \frac{p_i(1-s_i)}{s_i(1-p_i)}$$

BIM scoring function (sometimes also called “Retrieval Status Value” or RSV)

Note: the query terms do not explicitly appear in the function. However, the query provides us with **clues** on the **relevant set** (which we have to estimate).

Assumptions:

- Terms not appearing in the query have the same probability of occurrence in the relevant and non-relevant documents (p=s); BIM sum only over terms appearing in the query and the document.
- s can be estimated by the term occurrence in the whole collection (#relevant docs << #non-relevant docs)
- p is constant, e.g. 0.5, if we know nothing about our relevant set

Binary Independence Model

$$\sum_{i:d_i=1} \log \frac{p_i(1-s_i)}{s_i(1-p_i)}$$

BIM scoring function

BIM formula using our assumptions:

Number of documents in the corpus

$$\log \frac{0.5(1-\frac{n_i}{N})}{\frac{n_i}{N}(1-0.5)} = \log \frac{N - n_i}{n_i}$$

Number of documents containing term i

Looks like a variant of *idf* (derived in a principled manner), *tf* components are missing as we assumed binary "features".

Binary Independence Model

Relevance feedback employ user feedback to improve the retrieval outcome; requires 2+ rounds of retrieval

$$\underbrace{\sum_{i:d_i=1} \log \frac{p_i(1-s_i)}{s_i(1-p_i)}}_{\text{BIM scoring function}}$$

In practice, relevance feedback is mostly absent. IDF-like weighting is not very effective.
Is this all just an academic exercise?

No! BIM is the basis of **BM25** ("best match" version 25), one of the most popular baselines today.

Relevance feedback provides us with additional information; we arrive at better estimates: $s_i = (n_i - r_i + 0.5)/(N - R + 1.0)$ $\quad p_i = (r_i + 0.5)/(R + 1)$

smoothing to avoid log(0)

	Relevant	Non-relevant	Total
$d_i = 1$	r_i	$n_i - r_i$	n_i
$d_i = 0$	$R - r_i$	$N - n_i - R + r_i$	$N - n_i$
Total	R	$N - R$	N

Contingency table of term occurrences for a given query

Number of relevant documents containing term i

Number of documents containing term i

Number of relevant documents for the given query

Number of documents in the collection

Probabilistic models:

BM25

An improvement over BIM

BM25 ranking algorithm

BM25 extends the BIM model by including document weights and query term weights

BM25 is grounded in probabilistic arguments and experimental validation (TREC, CLEF, NTCIR, etc.). Most common scoring variant:

constants

frequency of term i in the query

R=r=0 in the absence of relevance information

$$\sum_{i \in Q} \log \frac{(r_i + 0.5)/(R - r_i + 0.5)}{(n_i - r_i + 0.5)/(N - n_i - R + r_i + 0.5)} \cdot \frac{(k_1 + 1) f_i}{K + f_i} \cdot \frac{(k_2 + 1) qf_i}{k_2 + qf_i}$$

sum over all query terms

frequency of term i in the document

$$K = k_1((1 - b) + b \cdot \frac{dl}{avdl})$$

document length

average document length in the corpus

BM25 scoring example

Setting:

- Corpus with 500K documents
- Query Q: "president lincoln"
- $df(president)=40K$, $df(lincoln)=300$
- Length of document D: 90% of the average document length
- Constants: $k1=1.2$, $b=0.75$, $k2=100$

$tf(president)$	$tf(lincoln)$	$BM25(Q,D)$
15	25	20.66
15	1	12.74
15	0	5.00
1	25	18.20
0	25	15.66

BM25 hyperparameters

Determines impact of doc. term frequency (at 0, only term *presence* is considered).
Common value: 1.2

$$\sum_{i \in Q} \log \frac{(r_i + 0.5)/(R - r_i + 0.5)}{(n_i - r_i + 0.5)/(N - n_i - R + r_i + 0.5)} \cdot \frac{(k_1 + 1) f_i}{K + f_i} \cdot \frac{(k_2 + 1) q f_i}{k_2 + q f_i}$$

Determines impact of query term frequency.
Common range 0-1000

$$K = k_1((1 - b) + b \cdot \frac{dl}{avdl})$$

Normalizes the tf component by document length.

At 0, no length normalization. At 1, full normalization.
Common value: 0.75.

BM25: popular baseline

- BM25 is an effective and **robust** ranking algorithm
- BM25's parameter values should be **tuned**(!)

- Topical relevance
- Explicit assumption: binary notion of relevance
- Term frequencies were added to the model (*BIM to BM25*) to improve retrieval effectiveness

flickr@visualanthology

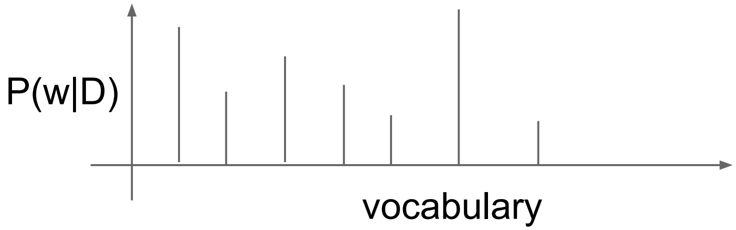
Probabilistic models:

Language Modeling for IR

Term frequencies become part of the model …

Language models

- **Unigram language model**: probability distribution over the words (the *vocabulary*) in a language (the *collection* or *document*)
- In IR, unigram LMs represent the *topical content*



- A LM representation of a document can be used to *generate* new text by sampling terms from the distribution (the text won't have a syntactic structure, but that's fine)

Language models

$$P(D|Q) = \frac{P(Q|D) \times P(D)}{P(Q)}$$

- Idea: rank the documents by the **likelihood** of the query according to the document's language model

- If we throw all terms into a "bag" and randomly draw terms, for which document is the probability greater of drawing the query term *CSKA*?

Pontus Wernbloom's last-minute volley earned a draw to keep CSKA Moscow in the hunt in their last-16 Champions League match against Real Madrid.
Real had looked set to return to Spain with a lead at the tie's halfway stage, but paid for their wastefulness.
Cristiano Ronaldo fired under Sergei Chepchugov to open the scoring but fluffed a fine 84th-minute chance.
CSKA had rarely threatened but Wernbloom crashed home with virtually the game's final kick to snatch a draw.

Greece has avoided a nightmare scenario by agreeing to a 130bn euros (£110bn; $170bn) bailout deal, Finance Minister Evangelos Venizelos has said.
He said the deal was probably the most important in Greece's post-war history. The cabinet was meeting to discuss how to pass the reforms stipulated by international lenders, which include huge spending cuts and beefed-up monitoring by eurozone officials. Trade unions have called strikes and protests for Wednesday.

Language models

$$P(D \mid Q) = \frac{P(Q \mid D) \times P(D)}{P(Q)}$$

Query Q is “generated” by a **probabilistic model** based on document D

Pontus Wernbloom's last-minute volley earned a draw to keep CSKA Moscow in the hunt in their last-16 Champions League match against Real Madrid.
Real had looked set to return to Spain with a lead at the tie's halfway stage, but paid for their wastefulness.
Cristiano Ronaldo fired under Sergei Chepchugov to open the scoring but fluffed a fine 84th-minute chance.
CSKA had rarely threatened but Wernbloom crashed home with virtually the game's final kick to snatch a draw.

to	0.0617
the	0.0493
a	0.0493
s	0.0370
but	0.0370
in	
0.0246	
their	0.0246
with	0.0246
wernbloom	0.0246
real	0.0246
had	0.0246
draw	0.0246
cska	0.0246
…	
moscow	0.0123

Maximum likelihood estimate

Language models

$$P(D \mid Q) = \frac{P(Q \mid D) \times P(D)}{P(Q)}$$

$$P(D \mid Q) \propto P(Q \mid D) \times \boxed{P(D)}$$

Uniform: every document has the same probability of being relevant

Assumptions:

- Term independence (makes the problem more tractable)
- Multinomial language model

$$P(Q \mid D) = \prod_i P(q_i \mid D)$$

Now: problem reduced to estimating $P(q_i|D)$

Term frequency based.
No explicit IDF component in LM!

Language models

Smoothing

$$P(Q \mid D) = \prod_i P(q_i \mid D)$$

$$Q = \{CSKA, Moscow, Greece\}$$

$$P(Q \mid D) = P(CSKA \mid D) P(Moscow \mid D) P(Greece \mid D)$$

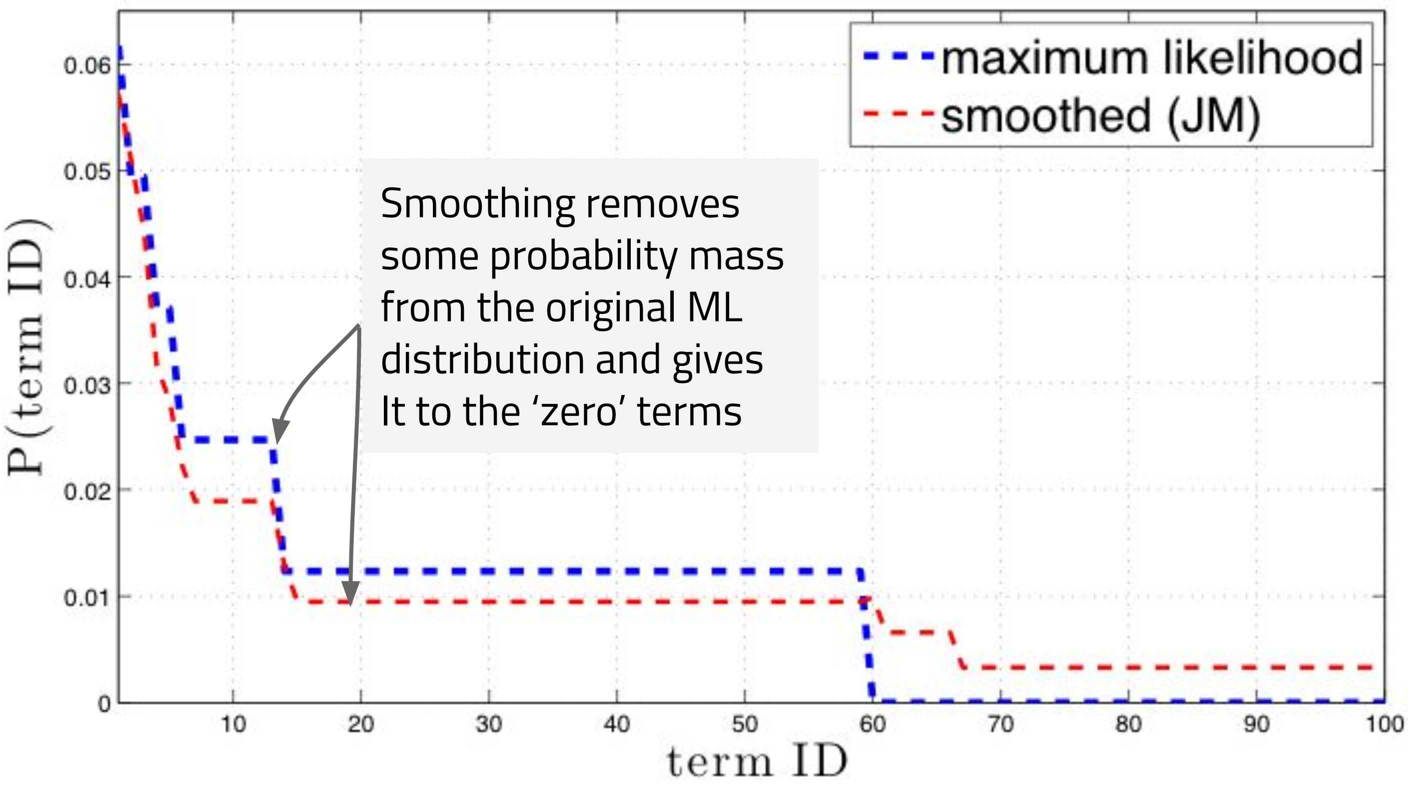
$$P(Q \mid D) = 0.0246 \times 0.0123 \times 0$$

$$P(Q \mid D) = 0$$

Smoothing methods 'smooth' the document's language model (maximum likelihood prob. distribution) to avoid terms with zero probability.

Language models

Smoothing



Language models

Smoothing

General idea: discount probabilities of **seen words**, assign extra probability mass to **unseen words** with a fallback model (the *collection language model*)

$$P(w \mid D) = \begin{cases} P_{smoothed}(w \mid D) & \textit{if word } w \textit{ is seen} \\ \alpha_d P(w \mid \mathbb{C}) & \textit{otherwise} \end{cases}$$

Jelineck-Mercer (JM) smoothing: linear interpolation (amount of smoothing controlled) between ML and collection LM

$$P_\lambda(w \mid D) = (1-\lambda) P_{ml}(w \mid D) + \lambda P(w \mid \mathbb{C}), \quad \lambda \in (0,1)$$

Language models

Smoothing

General idea: discount probabilities of **seen words**, assign extra probability mass to **unseen words** with a fallback model (the *collection language model*)

$$P(w \mid D) = \begin{cases} P_{smoothed}(w \mid D) & \textit{if word } w \textit{ is seen} \\ \alpha_d P(w \mid \mathbb{C}) & \textit{otherwise} \end{cases}$$

Term-dependent Jelineck-Mercer smoothing: different terms are smoothed to different degrees

$$P_{\lambda_w}(w \mid D) = (1 - \lambda_w) P_{ml}(w \mid D) + \lambda_w P(w \mid \mathbb{C})$$

Language models

Smoothing

General idea: discount probabilities of **seen words**, assign extra probability mass to **unseen words** with a fallback model (the *collection language model*)

$$P(w \mid D) = \begin{cases} P_{smoothed}(w \mid D) & \textit{if word } w \textit{ is seen} \\ \alpha_d P(w \mid \mathbb{C}) & \textit{otherwise} \end{cases}$$

Dirichlet smoothing: longer documents receive less smoothing

"count" of term w in D

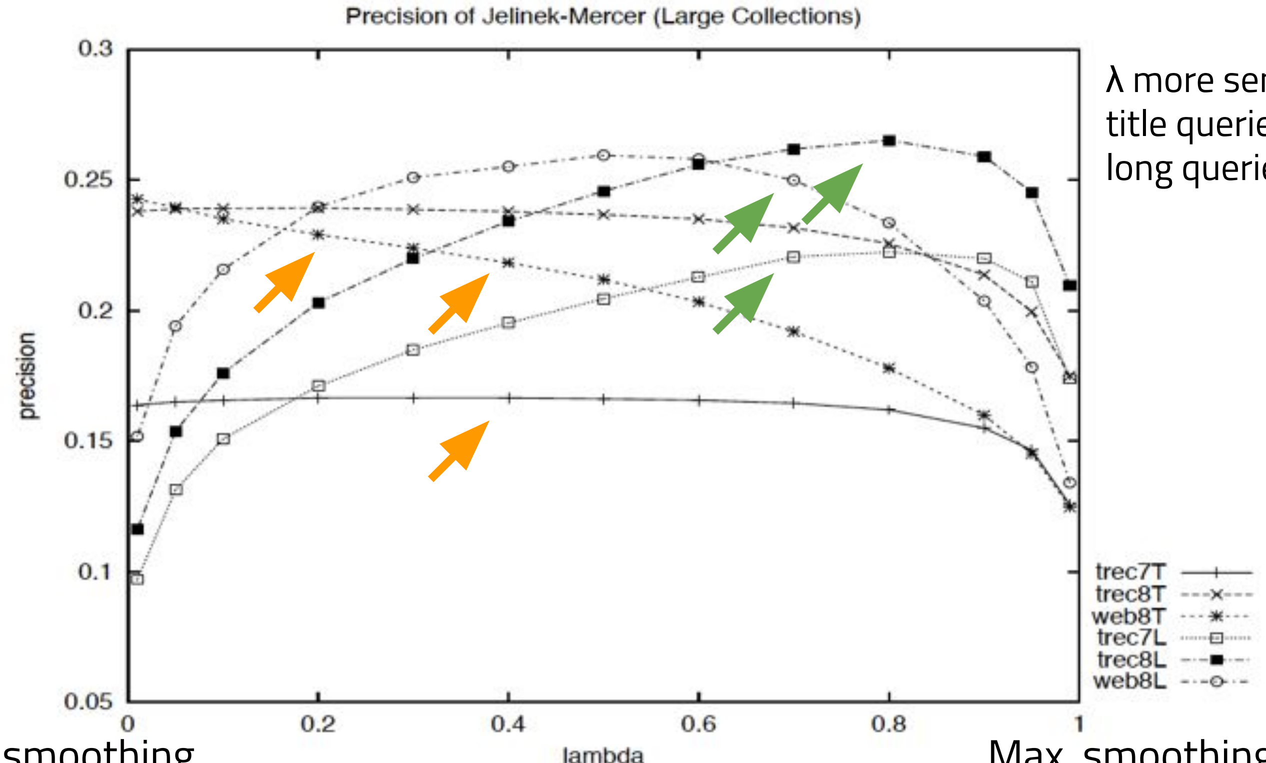
$$P_\mu(w \mid D) = \frac{c(w;D) + \mu P(w \mid \mathbb{C})}{\sum_w c(w;D) + \mu}, \ \textit{usually } \mu > 100$$

Language models

Smoothing: an empirical study
Jelineck-Mercer smoothing

<top>
<num> Number: 503
<title> Vikings in Scotland?
<desc> Description: What hard evidence proves that the Vikings visited or lived in Scotland?
<narr> Narrative: A document that merely states that the Vikings visited or lived in Scotland is not relevant. A relevant document must mention the source of the information, such as relics, sagas, runes or other records from those times.
</top>

5 TREC corpora with 50 topics per corpus



λ more sensitive for long queries;
title queries: good λ=0.1
long queries: good λ=0.7

T=title

L=title+desc+narr

Min. smoothing

Max. smoothing

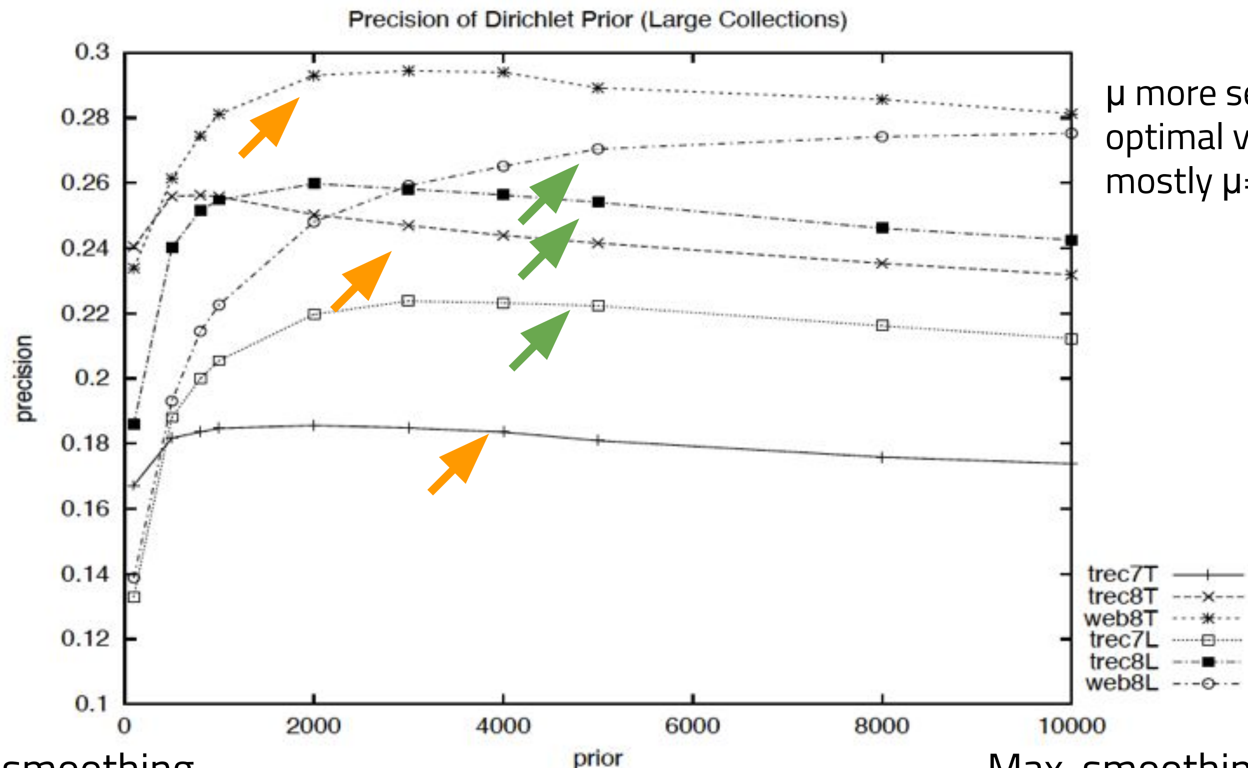
https://dl.acm.org/citation.cfm?id=384019

Language models

Smoothing: an empirical study
Dirichlet smoothing

<top>
<num> Number: 503
<title> Vikings in Scotland?
<desc> Description: What hard evidence proves that the Vikings visited or lived in Scotland?
<narr> Narrative: A document that merely states that the Vikings visited or lived in Scotland is not relevant. A relevant document must mention the source of the information, such as relics, sagas, runes or other records from those times.
</top>

5 TREC corpora with 50 topics per corpus



μ more sensitive for long queries; optimal value similar for T/L queries; mostly μ=2000

T=title

L=title+desc+narr

Min. smoothing

Max. smoothing

https://dl.acm.org/citation.cfm?id=384019

Language models

What about other sources of evidence?

On the web (and elsewhere), several sources of information to estimate content models:

- E.g. the content of the web page + the anchor texts of all hyperlinks pointing to the document
- Potentially very different representations of the same document

$$P(D \mid Q) \propto P(D) \prod_{i=1}^{n} \Big((1-\lambda-\mu) P(q_i \mid C) + \lambda P_{content}(q_i \mid D) + \mu P_{anchor}(q_i \mid D) \Big)$$

Language models

The document prior P(D)

So far: P(D) is assumed to be **uniform**

- Each document is equally likely to be drawn for a query

What can influence the probability of a document being relevant to an **unseen query**?

- Document length
- Document quality (PageRank, HITS, etc.)
- Document source (Wikipedia pages receive a high prior)
- Recency
- Language
- ...

Language models

The document prior P(D)

Another TREC task: **Entry page search**

- Find an entry page (homepage) of an organisation

Ad hoc retrieval systems purely based on content perform poorly

Priors (or other model components) can be

- Estimated from training data
- Defined based on some general modelling assumptions

https://dl.acm.org/citation.cfm?id=564383

Language models

The document prior P(D)

Another TREC task: **Entry page search**

- Find an entry page (homepage) of an organisation

Exploratory analysis: does the document length influence P(D)?

Somewhat, short documents are more likely to be entry pages than than longer ones.

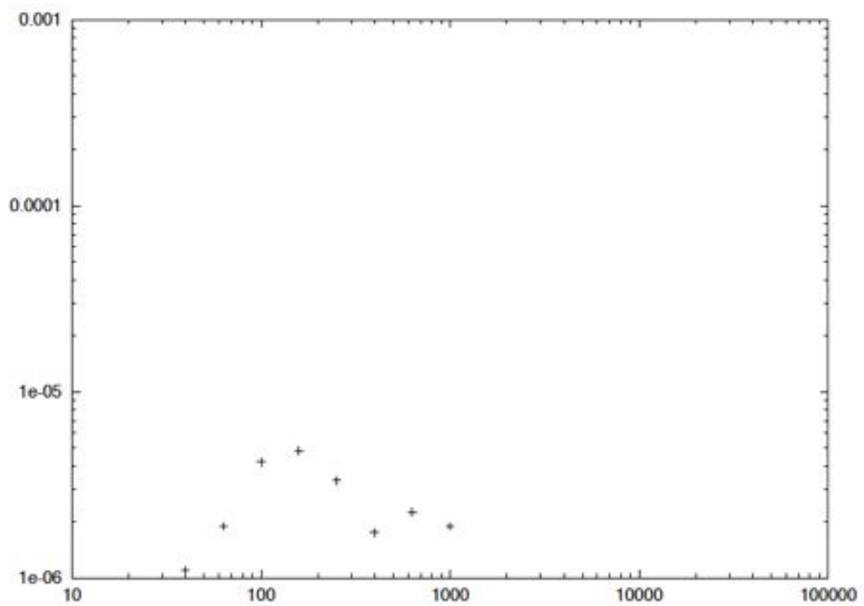


Figure 2: Prior probability of relevance given document length on the Entry Page task ($P(entry\ page|doclen)$)

Language models

The document prior P(D)

Another TREC task: **Entry page search**

- Find an entry page (homepage) of an organisation

Exploratory analysis:
does the number
of links pointing to a
document influence P(D)?

YES!

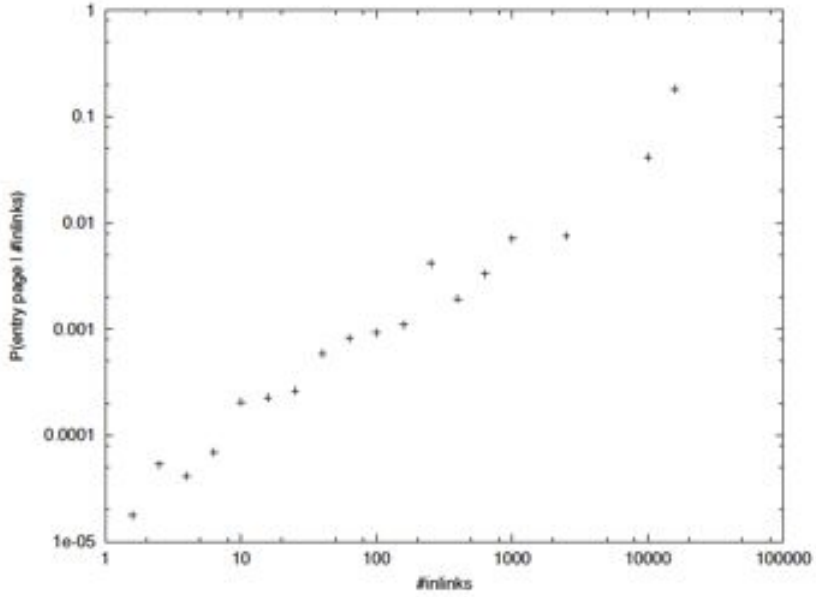


Figure 3: Prior probability of relevance given number of in-links on the Entry Page task ($P(entry\ page|inlinkCount)$)

Language models

The document prior P(D)

Another TREC task: **Entry page search**

- Find an entry page (homepage) of an organisation

Exploratory analysis: does the URL form influence P(D)? **YES!**

URL types:

Root: http://www.sigir.org
Subroot: http://www.sigir.org/sigirlist
Path: http://www.sigir.org/sigirlist/issues/
File: http://www.sigir.org/resources.html

URL type	Entry page	WT10g
root	79 (73.1%)	12,258 (0.7%)
subroot	15 (13.9%)	37,959 (2.2%)
path	8 (7.4%)	83,734 (4.9%)
file	6 (5.6%)	1,557,719 (92.1%)

Language models

The document prior P(D)

Another TREC task: **Entry page search**

- Find an entry page (homepage) of an organisation

Ranking	Content (λ=0.1) MRR	Anchors (λ=0.1) MRR
$P(Q\|D)$	0.3375	0.4188
$P(Q\|D)P_{doclen}(D)$	0.2634	0.5600
$P(Q\|D)P_{URL}(D)$	**<u>0.7705</u>**	0.6301
$P(Q\|D)P_{inlink}(D)$	0.4974	0.5365

https://dl.acm.org/citation.cfm?id=564383

Axiomatic approach to IR on two slides

Axiomatic framework

- Can we **analytically** predict whether a retrieval model will work well?
- Idea: **formalize retrieval constraints** (axioms) and explore retrieval models that fulfil the constraints
- Constraints a *reasonable* retrieval function should satisfy:

Constraints	Intuitions
TFC1	to favor a document with more occurrence of a query term
TFC2	to favor document matching more distinct query terms
TFC2	to make sure that the change in the score caused by increasing TF from 1 to 2 is larger than that caused by increasing TF from 100 to 101.
TDC	to regulate the impact of TF and IDF
LNC1	to penalize a long document(assuming equal TF)
LNC2, TF-LNC	to avoid over-penalizing a long document
TF-LNC	to regulate the interaction of TF and document length

- Allows us to constraint parameter space, investigate new functions

https://dl.acm.org/citation.cfm?id=1009004

Axiomatic framework in the neural age

- Neural nets cannot be analyzed analytically (millions of parameters)

- Idea: create **diagnostic datasets**, each designed to fulfil a specific axiom; explore how well neural models are able to rank documents within these datasets

- Findings (so far):
  - Most neural nets are not doing too well with semantic axioms
  - BERT “ignores” these axioms completely but outperforms traditional models by a wide margin (1)
  - Thus … we are missing vital axioms

(1) https://chauff.github.io/documents/publications/ECIR2020-Camara.pdf

Lecture Summary

- Most well-known retrieval models: boolean, vector space and probabilistic models
  - **BM25** and **Language Modeling** are popular baselines today
  - Models are rooted in theory and **validated empirically** (often leading to “adaptations” of the theory)
  - No machine learning until early 2000s (manual tuning over decades instead)

- Difficulty of machine learnt methods: data, data and more data

That's it!

Next week:

Paper review due (March 20).

Core IR proposal due (March 20).

Slack: in4325.slack.com

Email: in4325-ewi@tudelft.nl