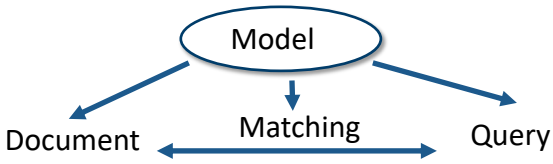
Recall: IR Model



There are many IR models

- The relevance decision mechanism can be either ***strict*** or ***flexible*** (e.g. set retrieval vs. ranked retrieval)
- The representation of the data can have a ***varying degree of abstraction*** (e.g. unigrams vs. concepts)

Progress in retrieval models has corresponded with improvements in effectiveness

3

Recall: Relevance

We have seen that relevance is a complex and subjective concept

- Many factors to consider
- People often disagree when making relevance judgments

Retrieval models make various assumptions about relevance to simplify the problem

- e.g., topical relevance (query and document are about the same topic)
  vs.
  user relevance (taking into account the usefulness of document from a user's perspective such as novelty, freshness, language, etc.)
- e.g., binary vs. multi-valued relevance

4

Outline

Classical Models (in brief):

- Boolean, Vector Space

Probabilistic Models

- BM25, Language Models, Divergence From Randomness
- Semi-structured models: Fields-based Models
- Proximity Models

5

Boolean, Vector Space

CLASSICAL MODELS

6

Recall: Boolean Retrieval

Recall the example of using an inverted index:

- Query for 'time' AND 'dark'

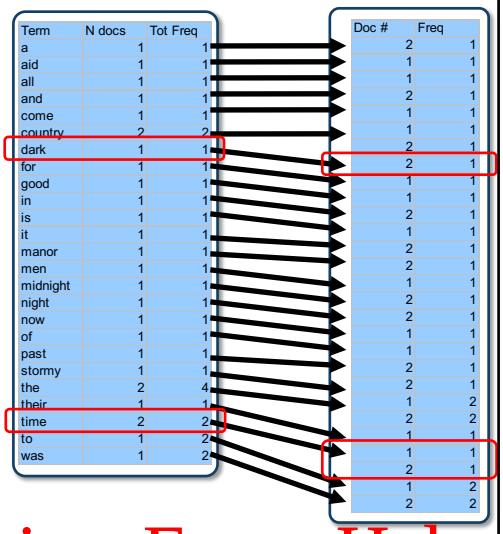
There are 2 docs with "time" in dictionary

- **IDs 1** and **2** from posting file

There is 1 doc with "dark" in dictionary

- **ID 2** from posting file

Therefore, only doc ID 2 satisfied the query.



Term	N docs	Tot Freq
a	1	1
aid	1	1
all	1	1
and	1	1
come	1	1
country	2	2
dark	1	1
for	1	1
good	1	1
in	1	1
is	1	1
it	1	1
manor	1	1
men	1	1
midnight	1	1
night	1	1
now	1	1
of	1	1
past	1	1
stormy	1	1
the	2	4
their	1	1
time	2	2
to	1	2
was	1	2

Doc #	Freq
2	1
1	1
1	1
2	1
1	1
1	1
2	1
2	1
1	1
1	1
2	1
1	1
2	1
2	1
1	1
2	1
2	1
1	1
1	1
2	1
2	1
1	2
2	2
1	1
1	1
2	1
1	2
2	2

7

Standard Boolean Model

Assumptions

- A document is represented as a ***set*** of keywords (i.e. model of documents)
- Queries are ***boolean*** expressions of keywords, connected by AND, OR, and NOT (i.e. model of queries)

Relevance

- A document is judged to be relevant if the index terms in the document ***satisfy*** the logical expression in the query (i.e. relevance decision mechanism)

$$R(D, Q) = D \rightarrow Q$$

F.W. Lancaster; E.G Fayen (1973). Information Retrieval On-Line, Melville Publishing Co., Los Angeles, California

G. Salton, E.A. Fox, and H. Wu. (1983). Extended Boolean information retrieval. Communications of the ACM 26(11), 1022-1036.

8

Relevance Decision Mechanism

Based on a Closed World Assumption (CWA)

Example

- Three index terms: ***A, B, C***

Query = "A ∧ B ∧ ¬ C"

- ***Retrieved*** : d_1
- ***Not retrieved*** : d_2, d_3

9

Advantages and Disadvantage of the Boolean Model

Advantages

- Simple queries are easy to understand
- Exact and simple to program
- The whole panoply of Boolean Algebra available

Disadvantages

- Effectiveness depends entirely on the user
- Complex query syntax is often misunderstood (if understood at all)
- Problems of Null output or *Information Overload*
- Output is not ordered in any useful fashion (set retrieval)

10

Classical Models #2

Vector Space Model

- Documents are represented by a term vector
- Queries are similarly represented by a term vector

Similarity metric is ad-hoc

- Cosine
- Dice
- Coordination matching

Which one to use?

What is the ideal one?

How do you find?

Ad-hoc weightings (i.e., not theoretically based)

No guidance on when to stop ranking

Not necessarily optimal ranking

What are the drawbacks of the Vector space retrieval model?

G. Salton, A. Wong, C. S. Yang. (1975). A vector space model for automatic indexing. Communications of the ACM, 18 (11), 613-620.

BM25

PROBABILISTIC MODELS AND BEST MATCH

Why use probabilities ?

IR deals with Uncertain Information

Explain why IR is an Uncertain process?

- **Document Representation**: How good is this representation
- **Information need and Query Representation**: How representative is the query viz information need
- **Matching Process**: How relevant is the result to the query? Is the ranking optimal?
- **Results presentation**: What is the impact of the presentation of the results?

Uncertainty is everywhere!

Probability theory seems to be the most natural way to quantify uncertainty

13

Probability Ranking Principle

W.S. Cooper argued:

p(R|D)

query

relevance

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

... where the probabilities are estimated as accurately a possible on the basis of whatever data made available to the system for this purpose ...

... then the **overall effectiveness** of the system to its users **will be the best** that is obtainable on the basis of that data.

Ranking Criteria: Probability of relevance of the document with respect to the query

(Robertson 1977)

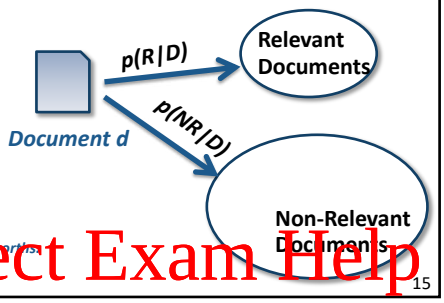
14

Another way to view ranking…

IR as a *classification problem*

- Given a document D, we want to decide whether this is relevant or not
- Classify as relevant or non-relevant and retrieve if it is relevant
- If we can calculate the probability of relevance (p(R|D)) or non-relevance (p(NR|D)) then we can use the set with the highest probability

If p(*R*|*D*) > p(*NR*|*D*) then *D* is relevant, Otherwise *D* is not relevant



C. J. van Rijsbergen: (1979): Information Retrieval (2nd edition): Butterworths

15

Bayes Classifier

Bayes Decision Rule

- A document *D* is relevant if $P(R|D) > P(NR|D)$

Estimating probabilities

- Use **Bayes Rule**

$$p(R \mid D) = \frac{p(D \mid R)p(R)}{p(D)}$$

$$p(NR \mid D) = \frac{p(D \mid NR)p(NR)}{p(D)}$$

- Classify a document as relevant if

$$\frac{p(D/R)}{p(D/NR)} > \frac{p(NR)}{p(R)}$$

–lhs is *likelihood ratio*

p(*D*|*R*), *p*(*D*|*NR*) - probability that if a relevant (non-relevant) document is retrieved, it is *D*.

16

Estimating Probabilities

How do we compute all those probabilities?

- Cannot compute exact probabilities – we have to use estimates

Binary Independence Retrieval (BIR)

- Document represented by a vector of binary features indicating term occurrence (or non-occurrence)

Independence Assumption

- Terms occur in documents independently $p(a \cap b) = p(a).p(b)$

Queries

- Binary vectors of terms
- Classify a document as relevant if **p(D|R) p(R) > p(D|NR) p(NR)** (after using **Bayes Rule**)

$$\frac{p(D/R)}{p(D/NR)} > \frac{p(NR)}{p(R)}$$

h: Likelihood ratio — *Cut Off Point*

17

Binary Independence Model

Assume independence of terms

- doc represented by a vector of binary features indicating term occurrence (or non-occurrence)
- p_i is probability that term i occurs (i.e., has the value 1) in a document from the relevant set; s_i is probability of occurrence of i in the non-relevant set

After a bit of math (see next slide)

$$\frac{p(D \mid R)}{p(D \mid NR)} = \prod_{i:d_i=1} \frac{p_i(1-s_i)}{s_i(1-p_i)} \cdot \prod_i \frac{1-p_i}{1-s_i}$$

Retrieval Status Value (RSV) — *Constant for each query*

Avoid multiplying lots of small numbers (scoring function):

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

S.E. Robertson, K. Sparck Jones (1976). Relevance weighting of search terms. Journal of the American Society for Information Science, 27: 129–146

18

Binary Independence Model

See Textbook

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

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

Added & cancel each other

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

$p_i = p(d_i = 1/R)$
$1 - p_i = p(d_i = 0/R)$
$s_i = p(d_i = 1/NR)$
$1 - s_i = p(d_i = 0/NR)$

$\prod_{i:d_i=1}$ means that it is a product over the terms that have the value 1 in the document

19

Ranking Documents

How do we score documents in practice?

- Query provides information about relevant documents
- If we assume p_i constant (0.5), s_i is approximated by entire collection, then we get an **idf-like weight**

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

Relevance Score for a Document:

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

c_i

Score Computation

Documents	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

$$c_i \approx K(N, n_i, R, r_i) = \log \frac{r_i/(R-r_i)}{(n_i-r_i)/(N-n_i-R+r_i)}$$

Add 0.5 to every expression

0

Looking back

In the absence of any information about the relevant documents,

- The term weight derived from the binary independence model is very similar to idf weight
- There is no tf component
- Because in BIR we consider only a binary representation

So we started with a binary representation; That is throwing away all the frequency information;

Performance Issues

- How good is this model?
  - Not very good: if tf is ignored, most effectiveness measures could drop by up to 50%
  - Let's bring back tf!

Used Bayes rule and came up with a ranking where in the worst case, it is equivalent to an **idf**-based ranking!

21

BM25

Widely used ranking approach based on binary independence model

- Adds document and query term weights

$$R(D,Q) = \sum_{i \in Q} \log \frac{(r_i + 0.5)/(R - r_i + 0.5)}{(n_i - r_i + 0.5)/(N - n_i - R + r_i + 0.5)} \bullet \frac{(k_1 + 1)tf_i}{K + tf_i} \bullet \frac{(k_2 + 1)qf_i}{k_2 + qf_i}$$

- k_1, k_2 and b are parameters whose values are set empirically
- $K = k_1((1 - b) + b \cdot \frac{dl}{avdl})$, *dl* is document length
- Typical TREC value for k_1 is 1.2, k_2 varies from 0 to 1000, b = 0.75

Still one of the best performing ranking models

K. Sparck Jones, S. Walker, S.E. Robertson (2000). A probabilistic model of information retrieval: development and comparative experiments. IPM, 36(6), 779-808

S.E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford (1994). Okapi at TREC-3. TREC 1994.

22

Building language models, Smoothing

LANGUAGE MODELLING

23

Language Model

Unigram language model

- Probability distribution over the words in a language

Model M		
	0.2	girl
	0.1	cat
	0.35	the
	0.25	boy
	0.1	meet

girl	cat
0.2	0.1

girl meet

$P(s \mid M) = 0.2 \times 0.1 = 0.02$

Models the *probability* of generating strings in the language

A *topic* in a document or query can be represented as a language model

i.e., words that tend to occur often when discussing a topic will have high probabilities in the corresponding language model

24

Unigram and higher-order models

P (● ● ● ●)

= P (●) P (● | ●) P (● | ● ●) P (● | ● ● ●)

Unigram Language Models

P (●) P (●) P (●) P (●)

Easy. Effective!

Bigram (generally, *n*-gram) Language Models

P (●) P (● | ●) P (● | ●) P (● | ●)

25

Why Use Language Models?

Use language models to model the process of query generation:

- User thinks of some relevant documents
- Pick some keywords to use as a query

Metrological office warned of high winds, rain across west of Scotland. There will be local floods, traffic chaos. The bad weather will continue for the foreseeable future.

Forecast, Glasgow, weather

Note: some words are not in the document

How do different retrieval models handle missing query terms?

26

Generative Probabilistic Models

What is the probability of producing the query from a document? P(Q|D)

- Referred to as the ***query-likelihood***

Assumptions:

- The probability of a document being relevant is strongly correlated with the probability of a query given a document
  - **P(D|R) is correlated with p(Q|D)**
- User has a reasonable idea of the terms that are likely to appear in the "ideal" document
- User's query terms can distinguish the "ideal" document from the rest of the corpus

The query is generated as a representative of the "ideal" document

- System's task is to estimate for each of the documents in the collection, which is most likely to be the "ideal" document

J.M. Ponte and W.B. Croft (1998). "A Language Modeling Approach to Information Retrieval." SIGIR 98: 275-281.
W.B. Croft, J. Lafferty (2003). Language modeling for information retrieval. Kluwer Academic Publishers.

Language Models

Basic Idea:

- Let's assume we point blindly, one at a time, at 3 words in a document
- What is the probability that I, by accident, pointed at the words "Weather", "Glasgow" and "Forecast"?
- Compute the probability, and use it to rank the documents.

Query-Likelihood Model

- **Rank documents by the probability that the query could be generated by the document model**
- Given a query, start with P(D|Q)
- Using **Bayes' Rule**:

$$p(D|Q) = \frac{p(D)p(Q|D)}{p(Q)} \underset{rank}{=} p(D)p(Q|D)$$

Standard LM Approach

Assume that query terms are drawn identically and independently from a document

$$p(Q \mid D) = \prod_{q_i \in Q} p(q_i \mid D)$$

Maximum Likelihood Estimate of $p(q_i \mid D)$

- Simply use the number of times the query term q_i occurs in the document divided by the total number of term occurrences.

$$p(q_i \mid D) = \frac{f_{q_i,D}}{|D|}$$

Problem: Zero Probability Problem

- If any of the query words are missing from the document, the score will be zero

29

Zero Probability Problem

May not wish to assign a probability of zero to a document that is missing one or more of the query terms

- Missing 1 out of 4 query words same as missing 3 out of 4

Document texts are a *sample* from the topic's model

- Missing words should not have zero probability of occurring

***Smoothing* is a technique for estimating probabilities for missing (or unseen) words**

- Lower (or discount) the probability estimates for words that are seen in the document text
- Assign that "left-over" probability to the estimates for the words that are not seen in the text

30

The Need for Smoothing

Zero probabilities spell disaster

- We need to *smooth* probabilities
  - **Discount** nonzero probabilities
  - Give some probability mass to **unseen** terms

Smoothing

- E.g. adding 1, ½ or ε to counts, Dirichlet priors, discounting, and interpolation
- A simple idea that works well in practice is to use a mixture between the document and the collection distribution (**Jelinek-Mercer**)

$$p(q_i \mid D) = (1-\lambda)p(q_i \mid D) + \lambda p(q_i \mid C)$$

A non-occurring term is possible, but no more likely than would be expected by chance in the collection

C. Zhai, J. Lafferty. (2004) A study of smoothing methods for language models applied to information retrieval. ACM TOIS (22:2), 179-214

31

Jelinek-Mercer Smoothing

Avoiding the estimation problem and overcoming data sparsity

- Lower (discount) the probability estimates for words that are seen in a document text,
  - Assign the leftover probability to estimates for the words that are not seen in the text
  - Unseen words can be estimated from the collection

$$p(q_i \mid C) = \frac{c_{q_i}}{|C|}$$

How many times the query term occurred in the collection divided by the total number of word occurrences in the collection

Justify the use of the Jelinek-Mercer smoothing method

$$p(q_i / D) = (1-\lambda)\frac{f_{q_i,D}}{|D|} + \lambda\frac{c_{q_i}}{|C|} \qquad p(Q \mid D) = \prod_{q_i \in Q} p(q_i / D)$$

32

Applying Log Scale

To avoid computational issues with low probability values

$$log\big(p(Q \mid D)\big) = \sum log(p(q_i \mid D))$$

$$log\big(p(Q \mid D)\big) = \sum_{q_i \in Q} \log\left((1-\lambda)\frac{f_{q_i,D}}{|D|} + \lambda\frac{c_{q_i}}{|C|}\right)$$

In general, the following works well:
λ= 0.1 for short queries
λ= 0.7 for long queries

Small values of λ (close to zero) produce less smoothing - the query tends to act like a Boolean AND

As λ approaches 1, the query acts more like a Boolean OR= Coordination level match

33

Dirichlet Smoothing

Dirichlet Smoothing

- A very effective smoothing method in IR especially for short queries
- Choose lambda (λ) values with respect to document length
- Small values of μ produces less smoothing

$$\lambda = \frac{\mu}{|D| + \mu}$$

$$log\big(p(Q/D)\big) = \sum_{q_i \in Q} \log\left(\frac{f_{q_i,D} + \mu\left(\frac{c_{q_i}}{|C|}\right)}{|D| + \mu}\right)$$

Typical values of μ vary from 1000 to 2000

34

LM vs. Probabilistic Model

The main difference is whether "Relevance" figures explicitly in the model or not

- LM approach does not attempt to model relevance
- Although, some later extensions do try to put relevance back into the model (e.g. see [Lavrenko'01])

LM is an intuitive and generally effective ranking model, that is also computationally tractable

However: its performance might vary across collections, and might be seen as less robust than, say, BM25.

V. Lavrenko, B. Croft. (2001). Relevance based language models. In SIGIR'01

Divergence From Randomness

INFORMATION THEORETIC MODELS

Divergence From Randomness (DFR)

DFR

- Information-theoretic framework for generating IR document and term weighting models, based on a simple notion:

"The more the divergence of the within-document term-frequency from its frequency within the collection, the more the information carried by the word t in the document d."

Hence:

- Assuming that the occurrence of a term is random in the whole collection,
- The various different weighting models from the DFR framework measure the divergence of the actual term distribution from that obtained under a random process

G. Amati, (2003). Probabilistic Models for Information Retrieval based on Divergence from Randomness. PhD thesis, Univ. of Glasgow.
G. Amati, C.J. van Rijsbergen, (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM TOIS 20(4), 357 – 389

Components of DFR

Three Components

$$w_{d,q} = \sum_{t \in q \cap d} (-\log P_1) \cdot (1 - P_2)$$

- **Randomness model** computes P1
  - Measures the divergence of the tf from what would be random for this collection (Hence the –log())
  - e.g. calculated using Binomial distribution or Poisson
- **Risk/Information Gain Model** computes P2
  - The score of a document shouldn't gain linearly from repeated occurrences
  - Information gain models, e.g. Laplace after-effect model controls this
- **Document Length Normalisation** – acts upon tf directly
  - The magnitude of tf in a document also depends on the document length
  - However, we don't want long documents being retrieved too easily
  - Hence, we normalise tf wrt document length, e.g. Normalisation 2

Components of DFR

Let's explain each component in more detail:

PL2: Poisson randomness model, ***L***aplace after-effect model and Normalisation ***2***

39

Measuring Randomness

We want to measure how much information the tf occurrences of a term t within a document *d* has

- Compared to its expected occurrences in the corpus D:

$$P(tf \mid d, D)$$

- This is like trying to pull *tf* aces out of a suite of cards in *l(d)* trials, where there are *TF* aces in total

We can compute this using the binomial distribution:

$$P(tf \mid d, D) = \begin{pmatrix} TF \\ tf \end{pmatrix} p^{tf} (1-p)^{TF-tf}$$

- where *p* is the prior of the document, with *p* = 1/N

40

Binomial =~ Poisson

Approximations

- Note the factorials in the Binomial: $\binom{n}{k} = \frac{n!}{k!(n-k)!}$
  - n! = 1 x 2 x 3 x … x n
  - Factorials are expensive (and inaccurate) to compute
- The Poisson distribution is an approximation of the binomial distribution
  - Which can be cheaply computed using a Stirling series or the Lanzcos approximation

PL2

- An approximation of the **P**oisson distribution using Stirling series with the two other DFR components (explained next):
  - Information Gain: **L**aplace
  - Document frequency normalisation: Normalisation **2**

41

Information Gain

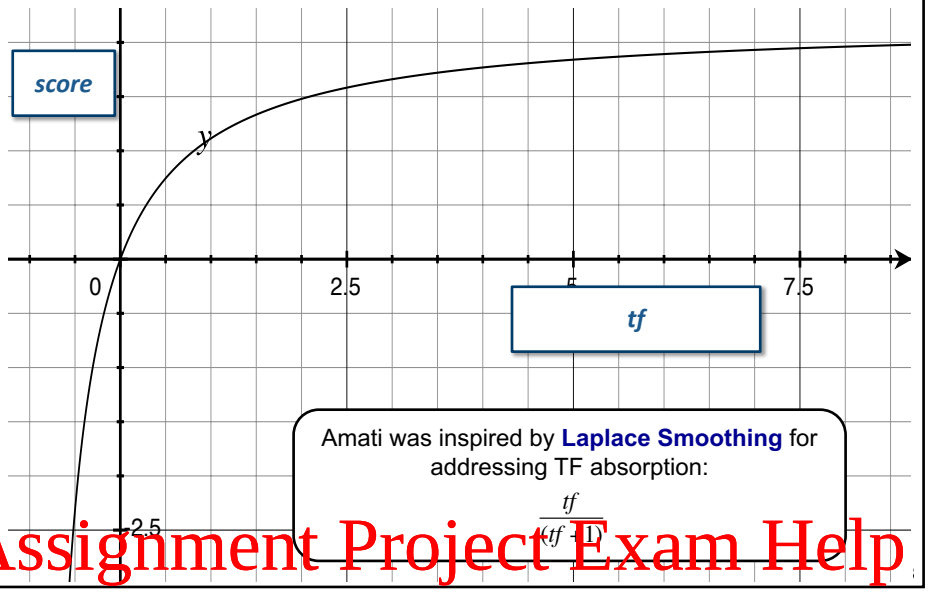
Not all term occurrences are equal

- Informative words are usually rare in the collection
  - This follows from Zipf's law
- However, when they do occur, their frequency is very high, indicating the importance of these terms in the respective documents
- Hence, the **gain** in probability of observing one more occurrences of *t* after observing *tf* should decrease

  $P(tf|d,D) - P(tf-1|d,D) > P(tf+1|d,D) - P(tf|d,D)$
  - In this way, increases in term frequency should be **absorbed**

42

Term Frequency Absorption



Document Length normalisation

Problem

- Basic models treat documents as if they are of the same length
  - However, longer documents are more likely to be retrieved by chance
  - The magnitude of the *tf* in a document also depends on the **document length**

Solution

- We can replace *tf* with ***tfn***, a **normalised term frequency**, with respect to a standard length:
  - Normalisation 2:

$$\mathbf{tfn} = \mathrm{tf} \cdot \log\left(1 + \frac{c \cdot \mathbf{avg\_length}}{\mathrm{l(d)}}\right)$$

44

PL2 & Summary

Randomness Model

- Approximation of ***Poisson*** distribution using Stirling series, P

After-Effect:

- ***Laplace***, L

Document frequency normalisation:

- Normalisation *2*

$$PL2: \frac{1}{tfn+1}(tfn.\log_2\frac{tfn}{\lambda}+(\lambda+\frac{1}{12.tfn}-tfn).\log_2 e+ 0.5.\log_2(2\pi.tfn))$$

$$\lambda = \frac{TF}{N}$$

$$tfn = tf \cdot \log_2(1+c\cdot\frac{\text{average length}}{length(d)})$$

DFR provides an information-theoretic framework for term weighting models

- Can also be applied to query expansion, *fields*, *proximity*

45

DFR Models

It is not a single model but a framework to generate models

- All models show an excellent performance, esp. where the term distribution follows the distribution of the parametric model assumed (e.g. PL2 assumes Poisson)

DFR contains many parameter-free models

- We do not need to learn or estimate parameters
- Effective even on document collections where the term distribution ***imperfectly*** follows the assumed parametric distribution
- E.g. DPH (See Web Track 2009-2012) is a "one-size-fits-all" weighting model: one distribution to fit all sorts of term distribution.

The retrieval functions are efficient to implement similar to BM25

46

Comparison of Models Effectiveness

GOV2, TREC 2006 Terabyte track	MAP
TF.IDF (with doc length normalisation)	0.2665
BM25	0.3015
Dirichlet Language Modelling	0.2675
PL2	0.3045
DPH	0.2920

- All weighting models exhibit similar performances for this dataset
- The DFR DPH parameter-free model is effective

47

BM25F, PL2F

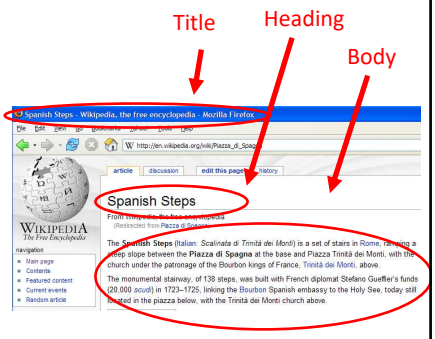
FIELDS-BASED MODELS

48

Retrieval with Document Fields

Document Structure

- Normal weighting models consider the document without any structure
- Instead, fields provide means to incorporate document *structure* in retrieval
- A document is indexed as a bag-of-words *per field*



Possible Fields

- Metadata from form-like fixed fields
- Subject, title, from/to headers in emails
- Post, comments in blogs
- HTML tags and the *anchor text* of incoming hyperlinks in Web documents

49

Fields to an Indexer

```
<DOC>
<DOCNO>doc1</DOCNO>
<URL>http://en.wikipedia.org/Piazza de Spagna</URL>
<HTML><HEAD>
<TITLE>Spani
</HEAD>
<BODY>
<h2>Spanish
Stairs in Ro
..
<ATEXT
src="http://en.wikipedia.org/Fontanna_della_Barcaccia">
Spanish Steps</ATEXT>
<ATEXT
src=http://en.wikipedia.org/Fontanna della Barcaccia>Pi
azza de Spagna</ATEXT>
</DOC>
```

URL

TITLE

set of

BODY

ATEXT

Different fields can bring different terms, of different usefulness.

How to weight such term occurrences from different fields?

50

Retrieval with Document Fields

How to combine the scores fro[illegible]lds?

Field importance

Term score (in field)

– Score Combination $score(d,Q) = \sum_{t \in Q} \sum_{f} w_f \cdot w(tf_f)$

Fields in the document

– Frequency combination $score(d,Q) = \sum_{t \in Q} w(\sum_{f} w_f \cdot tf_f)$

BM25F

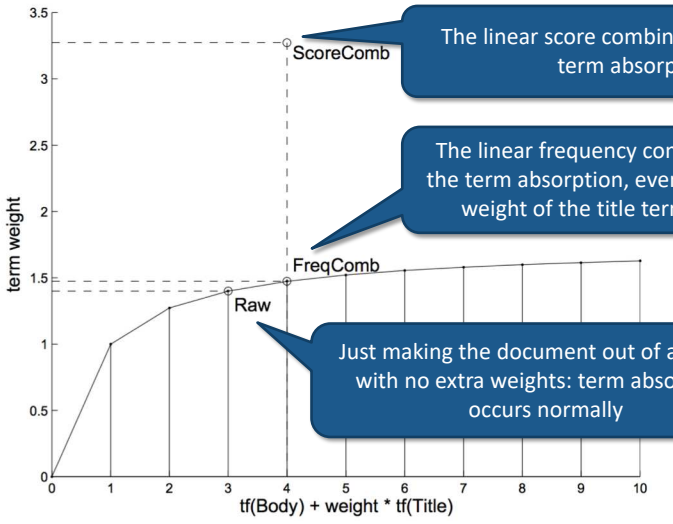
– Robertson et al. (2004) proposed the combination of field term frequencies instead of field relevance scores, which has advantages including:
  - Equal field weights degenerates to unstructured case
  - Collection statistics and document length are clearly defined for each field

– In contrast, for score combination, the non-linear nature of tf absorption functions (e.g. the Laplace effect in PL2) makes their combination non-trivial… more details in the next slide

S.E. Robertson, H. Zaragoza, M. Taylor. (2004). Simple BM25 extension to multiple weighted fields. CIKM'04.

TF absorption for combining fields

Figure 1: *tf* component of term weight



The linear score combination breaks the term absorption

The linear frequency combination retains the term absorption, even if we double the weight of the title term occurrences

Just making the document out of all fields, with no extra weights: term absorption occurs normally

Figure from [Robertson et al. CIKM 2004]

Term Distributions

The distribution of words in fields depends on the functionality of each field

- The title offers a short & concise description for documents
- The amount of anchor text depends on the document's in-degree
- Indeed, Hawking et al. (2004) showed that anchor text requires different length normalisation than the body of Web documents

Consider increasing the length of each field:

- Increasing URL length – not important, might be negative feature
- Increasing title length – not important
- Increasing body length – could make term occurrences less informative
- Increasing anchor text length – document has more inlinks!

S.E. Robertson, H. Zaragoza, M. Taylor. (2004). Simple BM25 extension to multiple weighted fields. CIKM'04.
C. Macdonald, V. Plachouras, B. He, C. Lioma and I Ounis. (2005). University of Glasgow at WebCLEF 2005: Experiments in per-field normalisation and language specific stemming. CLEF 2005.

Robertson et al. (2004) and Macdonald et al. (2005):

Both proposed ***per-field normalisation*** and ***weighting*** BM25F & PL2F

S.E. Robertson, H. Zaragoza, M. Taylor. (2004). Simple BM25 extension to multiple weighted fields. CIKM'04.
C. Macdonald, V. Plachouras, B. He, C. Lioma and I Ounis. (2005). University of Glasgow at WebCLEF 2005: Experiments in per-field normalisation and language specific stemming. CLEF 2005.

Per-field normalisation

Recall Frequency combination: $score(d,Q) = \sum_{t \in Q} w(\sum_f w_f \cdot tf_f)$

- Assume we can rewrite to show the normalised term frequency, tfn, used by the weighting model

$$score(d,Q) = \sum_{t \in Q} w(tfn = Norm(\sum_f w_f \cdot tf_f))$$

Per-field normalisation (e.g. BM25F, PL2F)

- Normalises the term frequency before their combination across fields:

$$score(d,Q) = \sum_{t \in Q} w(tfn = \sum_f w_f \cdot Norm(tf_f))$$

DFR and Document Fields (Macdonald et al., 2005)

Field Normalisation:

- Normalise and weight the term frequencies for each field independently, with respect to that field's length
- Replace Normalisation 2 with Normalisation 2F

$$tfn = \sum_f w_f \cdot tf_f \log\left(1 + c_f \frac{avglen_f}{l_f(d)}\right)$$

- c_f is a parameter related to each field f
- w_f is the weight of each field f
- $avglen_f$ is the average length of field f in the collection
- $l_f(d)$ is the length of field f in document d

PL2F: **P**oisson randomness model, **L**aplace after-effect model and Normalisation **2F**

Field-based Models

GOV, TREC 2004 Web track Topic Distillation	MAP
PL2	0.1246
PL2F	0.1391
BM25F	0.1497

- Field-based models enhance retrieval effectiveness, by allowing the importance of term occurrences in each field to be appropriately weighted
  - But **appropriate training** is required to ascertain field weights

Phrase Retrieval

PROXIMITY MODELS

Phrases

Some search engines support "phrase operators"

- Filter retrieved documents to only those containing the quoted phrase
- Inverted index is enriched with the positions of occurrences

But when they don't?

- Documents containing the query terms in close proximity may still be more relevant…

Q: Why are unigrams insufficient?

- Consider looking for documents on University of Glasgow?
- Then nothing would stop pages about Glasgow Caledonian University ranking higher!
- We don't want users to resort to "phase operators"

D. Metzler, W.B. Croft. (2005). A Markov random field model for term dependencies. SIGIR'05, 472-479.
X. Peng, D. Metzler, W.B. Croft, M. Smith. (2007). Incorporating Term Dependency in the DFR Framework. SIGIR'07.

Proximity

Proximity models boost documents where the query terms occur in close proximity

Basic Idea:

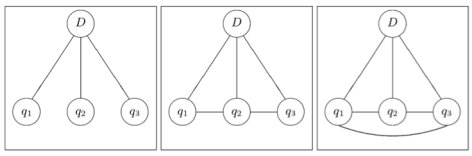
- Count occurrences of **pairs** of query terms in each document, weighting their occurrences appropriately

Metzler & Croft proposed a Markov Random Field model to formalise a proximity weighting model

D. Metzler, W.B. Croft. (2005). A Markov random field model for term dependencies. SIGIR'05, 472-479.

Dependencies

A Markov chain is used to characterise the dependencies between query terms:



Full independence Sequential dependence Full dependence

- **Full independence**: normal LM retrieval – each term is considered alone
- **Sequential dependence**: we consider adjacent pairs of terms
- **Full dependence**: we consider all pairs of query terms

The final document score considers weights for terms, pairs and the entire n-gram query:

$$score(d,Q) = \lambda_1 \sum_{t \in Q} w(t,d) + \lambda_2 \sum_{c \in Q_o} w(c,d) + \lambda_3 \sum_{c \in Q_u} w(c,d)$$

terms ordered pairs unordered pairs

61

Worked Example

Query

- white house rose garden

Score(d,Q)

- ***Full independence***
  - 0.8 * [w(white, d) + w(house, d) + w(rose, d) + w(garden, d)]
- ***Sequential dependence***
  - + 0.1 *[w('white house', d) + w('house rose', d) + w('rose garden', d)]
- ***Full dependence***
  - + 0.1 * [w('white rose', d) + w('white garden', d) + w('house garden', d)]
  - + 0.1 *w('white house rose garden', d)]

NB: w('white house', d) can be implemented as counting exact matches, or simply the number of windows in a document in which the *ordered* or *unordered* phrase occurs.

How to make these counts? We can use standard unigram posting lists with *position* information.

62

Example of Estimating N-gram Frequencies

- Consider w('white house', d)
- We can calculate "pf", the frequency of the pair p in a document by using the inverted index posting lists for 'white' and 'house':

white	0, <1,5>	5, <3>	6, <4>
house	0, <2,6>	3, <2,4,6>	6, <5>

'white house' *id=0 pf=2* *id=3 pf=0* *id=5 pf=0* *id=6 pf=1* **PF=3**

- In essence, we can simulate a posting list for 'white house' without indexing it as a bigram
- How can we calculate the 'IDF' of a phrase? (i.e. based on PF)
  - We must parse the combined posting lists BEFORE we can start scoring documents
  - This is expensive, particularly if the index cannot be held in memory

63

Proximity using DFR

DFR can be used to calculate proximity too.

- Recall the Binomial for term occurrences

$$\binom{TF}{tf} p^{tf}(1-p)^{TF-tf} \quad \Rightarrow \quad \binom{PF}{pf} p^{pf}(1-p)^{PF-pf}$$

- The randomness reference is the collection (e.g. TF)
  - However, recall that for proximity, PF is expensive to calculate
  - We do not have a posting list for a pair already telling us the size of the intersection, i.e. the global frequency of the pair
- For proximity, we examine only the document, using a binomial:

$$\binom{l(d)}{pf} p^{pf}(1-p)^{l(d)-pf}$$

  - This models how many occurrences of a pair within a document of the given length
  - Approximating the Binomial using Lanzcos functions, we call this pBiL2

J Peng, C Macdonald, B He, V Plachouras, & I Ounis. Incorporating term dependency in the DFR framework. In Proceedings of SIGIR 2007.
C Macdonald & I Ounis. Global Statistics in Proximity Weighting Models. In Proceedings of N-Gram Workshop at SIGIR 2010.

64

Results for Proximity

GOV2, TREC 2006 Terabyte track	MAP Global Stats	MAP No Global Stats
BM25	0.2743	
+MRF (SD. window size 2)	0.2964	0.2945
+MRF (FD, window size 2)	0.2763	0.2819
+DFR (SD. window size 2)	0.2998	0.2888
+DFR (FD, window size 2)	0.2769	0.2862

- Adhoc retrieval tasks on large corpora benefit from term dependence modelling

Summary

Retrieval models are one of the most important topics in IR

- An understanding of the underlying assumptions and concepts behind them is very rewarding
- We focussed on a class of probabilistic retrieval models called *generative models*, which use word-based and structure evidence

The complexity of modern IR applications is such that, increasingly, many other types of evidence are being used & combined

- Metadata, PageRank scores, Spam scores, and many other features!
- Increasingly, *discriminative models* are being used (e.g. **Learning to Rank**)
- The days of "the all-encompassing" IR model (ala BM25, LM, DFR) are probably behind us although they will always form the backbone of the deployed features