MANCHESTER 1824
The University of Manchester

COMP38210
Workshop 4: Improved indexing

John McNaught
&
Sandra Sampaio

MANCHESTER 1824
The University of Manchester

Overview

- Ranked retrieval
- Term frequency
- Zipf's law, Luhn
- Document frequency
- Inverse document frequency
- Collection frequency
- tf-idf
- Vector space

MANCHESTER 1824
The University of Manchester

Ranked retrieval

- Boolean model
  - Documents match or don't match
- Results are unranked
  - Maybe by most-recent-date order, though
- Web search → many results
  - Need some means of ranking
  - Typically aim to get top *n* (10?) ones correct
    - As user will likely not look much further

MANCHESTER 1824
The University of Manchester

Ranked retrieval

- Attempt more than matching query with documents
- Score each document to say *how well it matches* query
  - Assign real number score in range 0..1
- Then rank according to scores

MANCHESTER 1824
The University of Manchester

Points to consider

- Intuitive: if query term occurs many times in document, document more relevant?
- But: rare terms more informative than frequent terms
- Documents have different lengths
  - Will clearly affect counts

MANCHESTER 1824
The University of Manchester

(IIR book)

Term-document count matrices

Each document is a count vector in $\mathbb{N}^{|V|}$: a column below

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

MANCHESTER 1824
The University of Manchester

(IIR book)

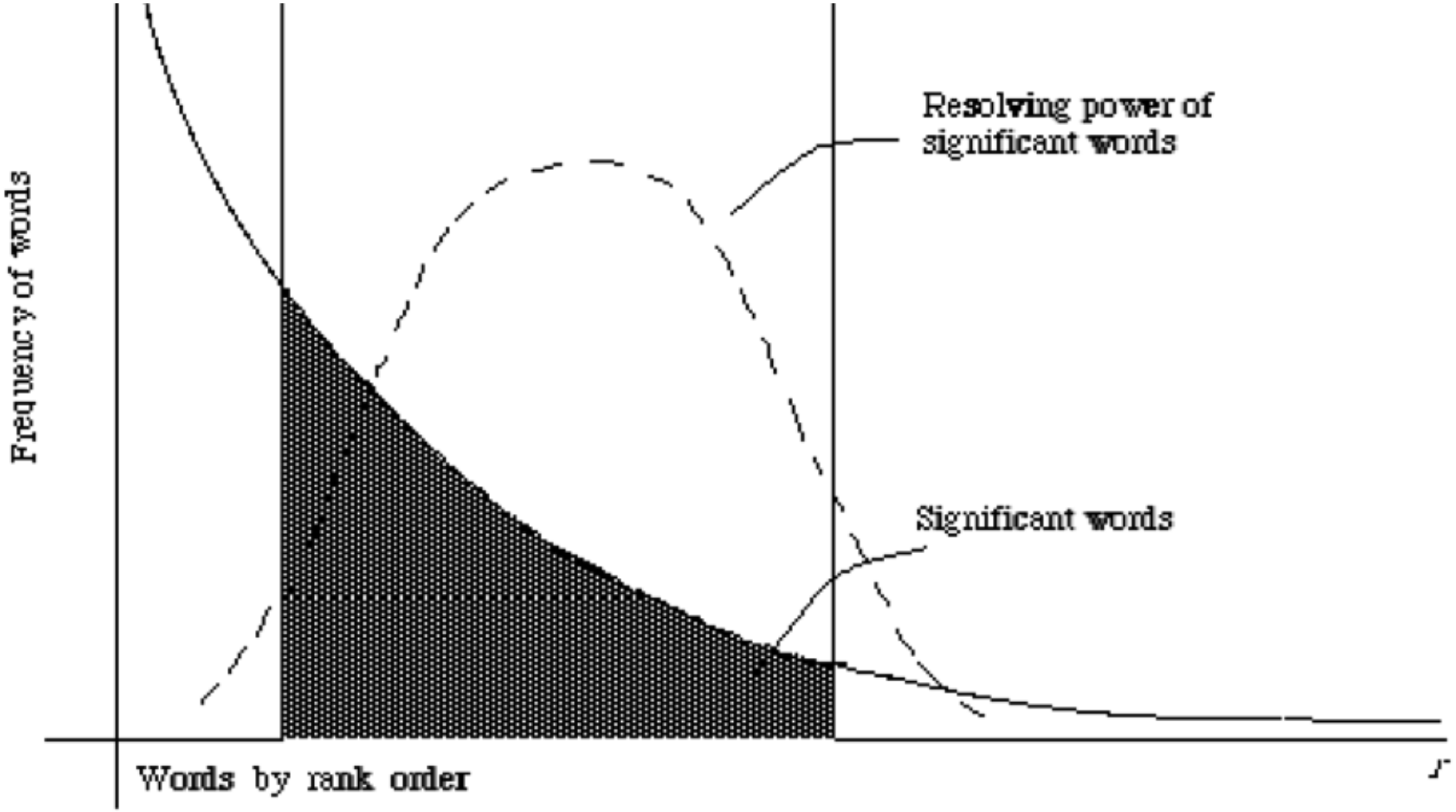
Term frequency (tf)

- Term frequency $tf_{t,d}$ of term *t* in document *d* is defined as the number of times that *t* occurs in *d*.
- Want to use tf when computing query-document match scores. But how?
- Raw term frequency is not what we want:
  - Document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term
  - *But not 10 times more relevant*
- Relevance does not increase proportionally with term frequency

NB: frequency = count in IR

Zipf and Luhn

http://www.dcs.gla.ac.uk/Keith/pdf/Chapter2.pdf



Zipf's Law: Frequency * rank ≃constant

(IIR book)

Document frequency (df)

- Rare terms more informative than frequent terms
  - Recall stop words
- Consider term in query that is rare in collection (e.g., *arachnocentric*)
- A document containing this term very likely to be relevant to query *arachnocentric*
- → Want a high weight for rare terms like *arachnocentric*.

MANCHESTER 1824
The University of Manchester

Sec. 6.2.1

(IIR book)

Document frequency, continued

- Frequent terms less informative than rare terms
- Consider query term that is frequent in collection (e.g., *high, increase, line*)
- A document containing such a term is more likely to be relevant than a document that doesn't
- But it's not a sure indicator of relevance
- → For frequent terms, want high positive weights for words like *high, increase, and line*
- But lower weights than for rare terms
- Use document frequency (df) to capture this

(IIR book)

idf weight

- df_t is the document frequency of *t*: the number of documents that contain *t*
  - df_t is an inverse measure of the informativeness of *t*
  - $df_t \leq N$
- We define the idf (inverse document frequency) of *t* by

$$idf_t = \log_{10}(N/df_t)$$

  - We use log (N/df_t) instead of N/df_t to "dampen" the effect of idf.

Will turn out the base of the log is immaterial.

MANCHESTER 1824
The University of Manchester

Smoothing effect of log

- 5 * log(1000/1) = 5 * 3.00 = 15.0
- 5 * log(1000/2) = 5 * 2.70 = 13.5
- 5 * log(1000/3) = 5 * 2.52 = 12.6
- 5 * log(1000/10) = 5 * 2.00 = 10.0
- 5 * log(1000/11) = 5 * 1.96 = 9.8
- 5 * log(1000/100) = 5 * 1.00 = 5.0
- Want to distinguish
  - Term occurs in 1 or 2 documents
  - Term occurs in many documents
  - Term occurs in great many documents
- Typically do not care if term occurs in (say) 60 or 61 documents: too fine grained for weighting purposes

(IIR book)

idf example, suppose N = 1 million

term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1,000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

$$\text{idf}_t = \log_{10}(N/\text{df}_t)$$

There is one idf value for each term t in a collection

(IIR book)

Effect of idf on ranking

- Does idf have an effect on ranking for one-term queries? E.g.
  - iPhone
- idf has no effect on ranking one term queries
  - idf affects the ranking of documents for queries with at least two terms
  - For the query capricious person, idf weighting makes occurrences of capricious count for much more in the final document ranking than occurrences of person

(IIR book)

Collection vs. Document frequency

- The collection frequency of *t* is the number of occurrences of *t* in the collection, counting multiple occurrences
- Example:

Word	Collection frequency	Document frequency
insurance	10440	3997
try	10422	8760

- Which word is a better search term (and should get a higher weight)?

MANCHESTER 1824
The University of Manchester

(IIR book)

tf-idf weighting

- The tf-idf weight of a term is the product of its tf weight and its idf weight

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

- Best known weighting scheme in information retrieval
  - Note: the "-" in tf-idf is a hyphen, not a minus sign!
  - Alternative names: tf.idf, tf x idf
- Increases with the number of occurrences within a document
- Increases with the rarity of the term in the collection

Final ranking of documents for a query

$$\text{Score}(q,d) = \sum_{t \in q \cap d} \text{tf.idf}_{t,d}$$

- Score of a document *d* is sum over all query terms of number of times each query term occurs in document *d*
- Refine: add up tf-idf weight of each query term found in *d*

MANCHESTER 1824
The University of Manchester

(IIR book)

Binary → count → weight matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Each document is now represented by a real-valued vector of tf-idf weights $\in \mathbb{R}^{|V|}$

(IIR book)

Documents as vectors

- So we have a |V|-dimensional vector space
- Terms are axes of the space
- Documents are points or vectors in this space
- Very high-dimensional: tens of millions of dimensions when you apply this to a web search engine
- These are very sparse vectors - most entries are zero

MANCHESTER 1824
The University of Manchester

Sec. 6.3

(IIR book)

Queries as vectors

- Key idea 1: Do the same for queries: represent them as vectors in the space
- Key idea 2: Rank documents according to their proximity to the query in this space
- proximity = similarity of vectors
- proximity ≈ inverse of distance
- Recall: We do this because we want to get away from the you're-either-in-or-out Boolean model.
- Instead: rank more relevant documents higher than less relevant documents

MANCHESTER 1824
The University of Manchester

Resources

- Chapter 6, section 6.2 of Manning et al., Introduction to Information Retrieval (online, free)
- Zipf's law illustrated
  - http://www.wordcount.org/main.php

    (an award-winning "artistic experiment")
- The idf page
  - http://staff.city.ac.uk/~sb317/idf.html