

Big Data and Data Mining

Ranking

Flavio Bertini

flavio.bertini@unipr.it



Boolean model limits

- Thus far, our queries have all been boolean
 - Documents **either match or don't**
- Good for **expert users** with precise understanding of their needs and knowledge of the collection
 - Also good for applications: applications can easily make use of 1000s of results for further processing
- Not good for the majority of users
 - Most users incapable of writing Boolean queries (or they are, but they think it's too much work)
 - Most users don't want to wade through 1000s of results
 - This is particularly true of web search!

Boolean model limits: example

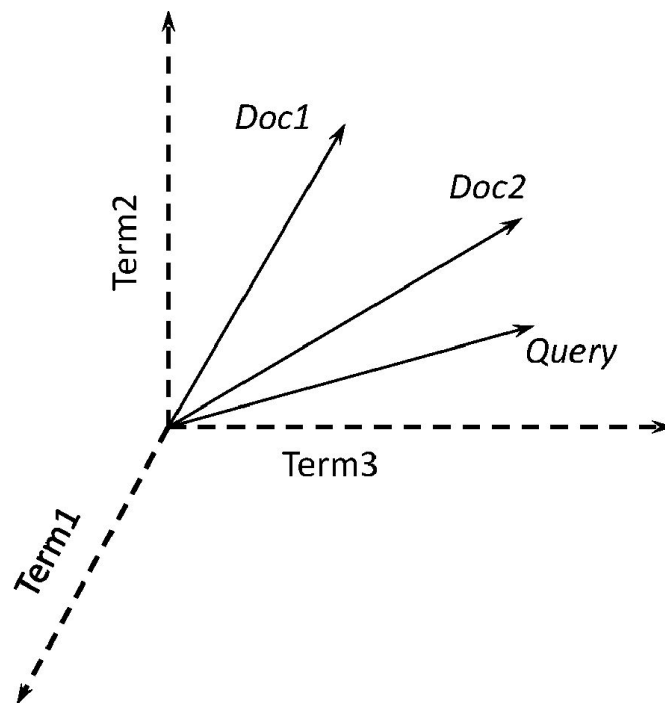
- Boolean queries often result in either **too few** (=0) or **too many** (1000s) results
 - For example, suppose a user is having troubles with his network card:
 - Query 1: “*standard user dlink 650*” → 200,000 hits
 - Query 2: “*standard user dlink 650 no card found*”: 0 hits
- It takes a lot of skill to come up with a query that produces a manageable number of hits
 - **AND** gives too few
 - **OR** gives too many

Ranked retrieval

- Rather than a set of documents satisfying a query expression, in **ranked retrieval**, the system returns an ordering over the (**top**) documents in the collection for a query
- When a system produces a ranked result set, large result sets are not an issue
 - We just show the **top k** (≈ 10) results
 - We don't overwhelm the user
 - Premise: the ranking model works
- Most known ranking model is Vector Space Model
 - Salton, G., Wong, A., & Yang, C. S. (1975). [A vector space model for automatic indexing](#). Communications of the ACM

Vector Space Model (VSM)

- Like in the bag-of-words model, documents are **represented by a vector** of “term weights”



- Collection represented by a matrix of **term weights**
 - d_{ij} is the weight of Doc_i for $Term_j$

$$D_i = (d_{i1}, d_{i2}, \dots, d_{it}) \quad Q = (q_1, q_2, \dots, q_t)$$

	$Term_1$	$Term_2$...	$Term_t$
Doc_1	d_{11}	d_{12}	...	d_{1t}
Doc_2	d_{21}	d_{22}	...	d_{2t}
\vdots	\vdots			
Doc_n	d_{n1}	d_{n2}	...	d_{nt}

VSM: vector representation

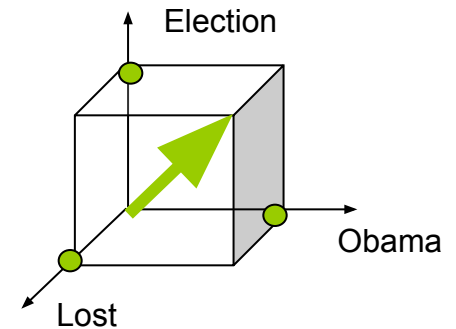
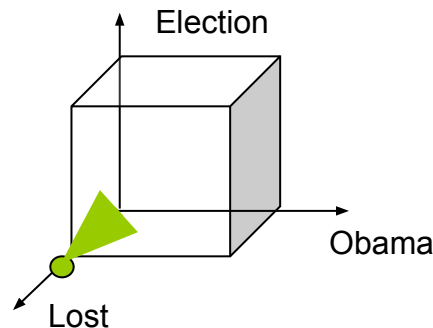
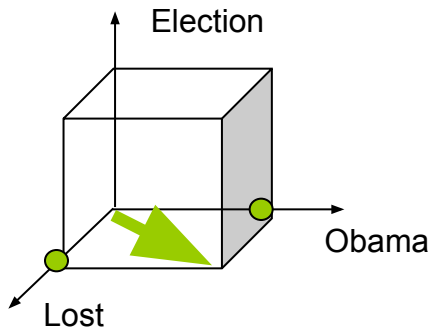
- An example with 3 terms (3-dimensional vector space) and 3 documents

D1 In last year election, Romney **lost election** against **Obama**

D2 I **lost** my faith in Clinton's ability to win

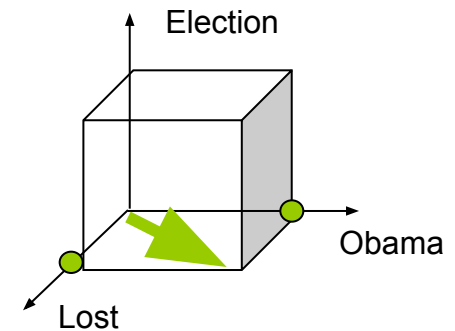
D3 **Obama** has **lost** control on Senate

We consider only the terms in figure



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



Queries as Vectors

- **Key idea 1:** Do the same for queries representing 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 \approx inverse of distance
- **Reminder:** We do this because we want to get away from the in-or-out Boolean model
- Instead: **rank more relevant documents** higher than less relevant documents

Recap

- In the boolean model, a document is either relevant or non-relevant
- Representing document and queries as vectors, we can compute the **similarity between the vector** of the document and the vector of the query:
 - This is an estimation of the relevance of the document with respect to the query
- We need to define the similarity between vectors but...
 - Before going on we must introduce the **term frequency**

TF: Term Frequencies

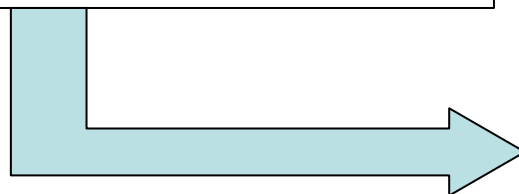
- So far, vectors were binary, representing the **presence or absence** of terms in a document
- We would like to assign a **weight** for each term in a vector space:
 - The term frequency $tf_{t,d}$ of term t in document d is defined as the number of times that t occurs in d
 - We want to use term frequency (**tf**) when computing query-document match scores



Example: TF as term weight

Three dimensional pictures are useful, but can be misleading for high-dimensional space (more than 3 terms considered):

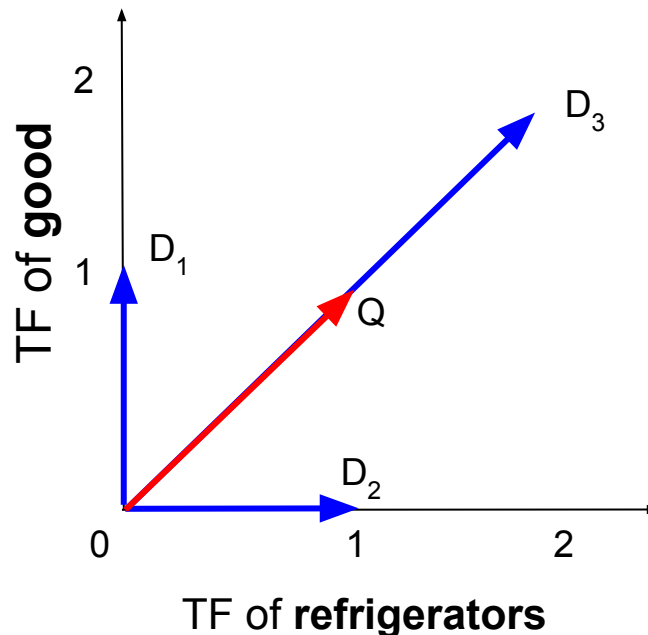
- D_1 Tropical Freshwater Aquarium Fish.
 D_2 Tropical Fish, Aquarium Care, Tank Setup.
 D_3 Keeping Tropical Fish and Goldfish in Aquariums, and Fish Bowls.
 D_4 The Tropical Tank Homepage - Tropical Fish and Aquariums.



Terms	Documents			
	D_1	D_2	D_3	D_4
Aquarium	1	1	1	1
Bowl	0	0	1	0
Care	0	1	0	0
Fish	1	1	2	1
Freshwater	1	0	0	0
Goldfish	0	0	1	0
Homepage	0	0	0	1
Keep	0	0	1	0
Setup	0	1	0	0
Tank	0	1	0	1
Tropical	1	1	1	2

Vector Space Distance

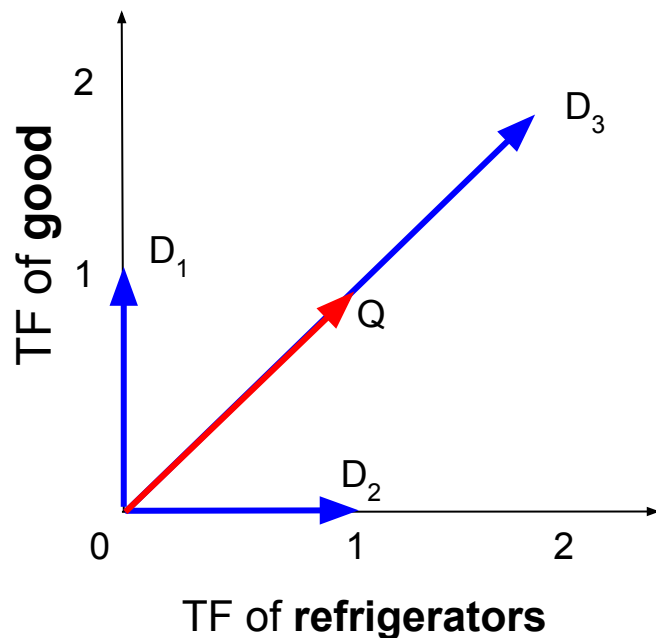
- How to formalize the proximity between two vectors when we have term frequencies?
- Example query Q: “Good refrigerators”
 - D_1 : “Good morning to all of you.”
 - D_2 : “Don’t put pizza in refrigerators”
 - D_3 : “Good Refrigerator Review: top five good refrigerators.”



Note: we are showing only the two interesting dimensions, “good” and “refrigerators”. There is a dimension for each indexed term (e.g., for pizza, morning, review...)

First cut: Euclidean Distance?

- We could consider the straight line distance between the endpoints, *i.e.*, the euclidean distance
 - Distance between Q and D_1 is 1
 - Distance between Q and D_2 is 1
 - Distance between Q and D_3 is $\sqrt{2} \approx 1.414$
- D_3 is the least similar, this doesn't seem right!



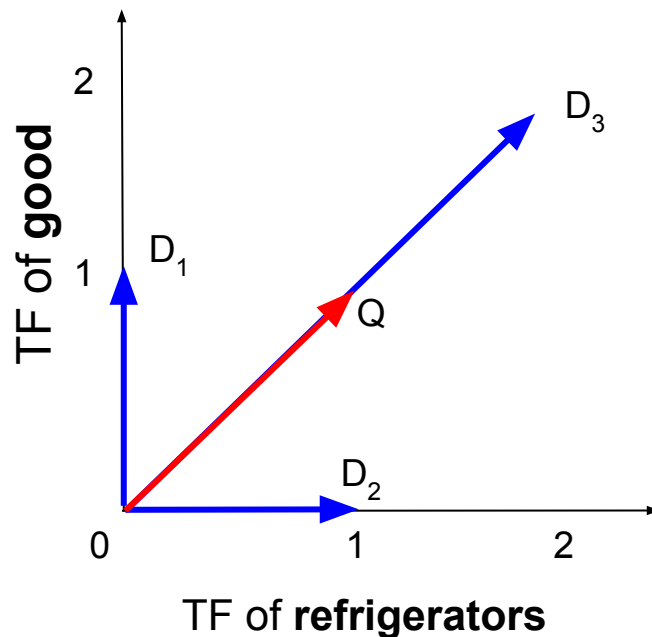
- Remember Q: “Good refrigerators”
 - D_1 : “Good morning to all of you.”
 - D_2 : “Don’t put pizza in refrigerators”
 - D_3 : “Good refrigerators review: top five good refrigerators.”

Angle instead of distance

- Thought experiment: take a document d and append it to itself many times. Call this longer document d'
- “Semantically” d and d' have the same content
- The Euclidean distance between the two documents can be quite large
- The angle between the two documents is 0, corresponding to maximal similarity
- **Key idea:** Rank documents according to angle with query

Example: Using angles

- We now consider the angle between vectors
 - Angle between Q and D_1 is 45°
 - Angle between Q and D_2 is 45°
 - Angle between Q and D_3 is 0°
- **D_3 is the most similar**, this is what we need!

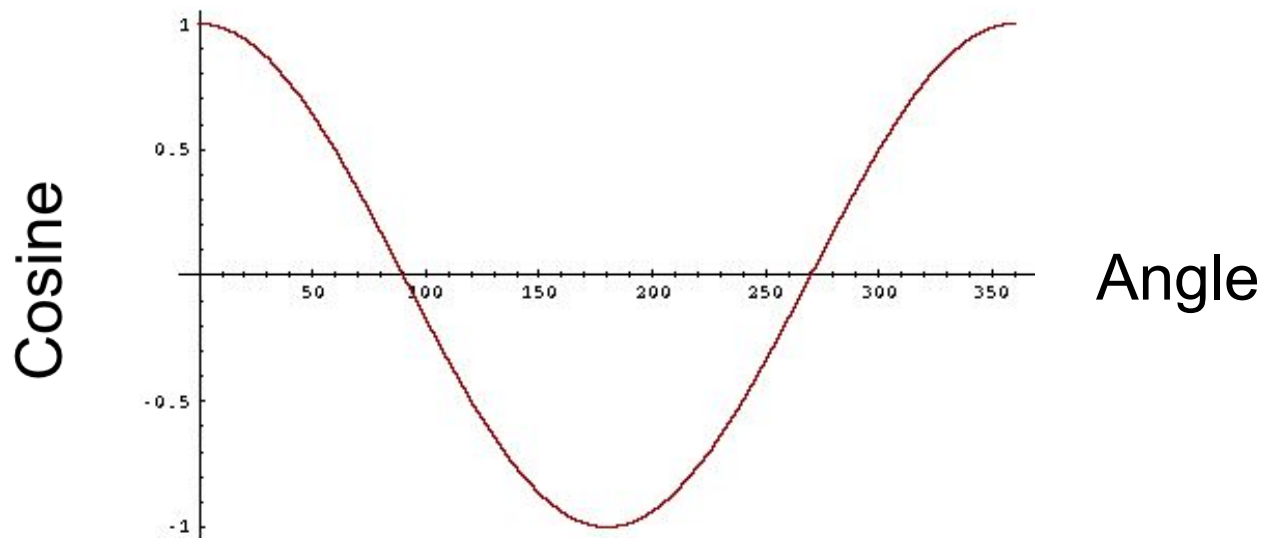


- Remember Q: “Good refrigerators”
 - D_1 : “Good morning to all of you.”
 - D_2 : “Don’t put pizza in refrigerators”
 - D_3 : “Good refrigerators review: top five good refrigerators.”



From Angles to Cosines

- The following two notions are equivalent.
 - Rank documents in increasing order of the angle between query and document
 - Rank documents in decreasing order of $\cos(\text{query}, \text{document})$
- Cosine is a monotonically **decreasing** function for the interval $[0^\circ, 180^\circ]$, meaning that it's inversely proportional of the angle



Cosine similarity

$$\cos(\vec{q}, \vec{d}) = \frac{\overbrace{\vec{q} \cdot \vec{d}}^{\text{Dot product}}}{\underbrace{|\vec{q}|}_{\text{Unit vectors}} \underbrace{|\vec{d}|}_{\text{Unit vectors}}} = \frac{\vec{q}}{|\vec{q}|} \cdot \frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2} \sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

- q_i is the tf weight of term i in the query
- d_i is the tf weight of term i in the document

$\cos(\vec{q}, \vec{d})$ is the cosine similarity of \vec{q} and \vec{d} ... or, equivalently, the cosine of the angle between \vec{q} and \vec{d} .

Example: Formula

	Election	Lost	Obama		Election	Lost	Obama
D1 [1	1	1]	D2 [0	1	0]

	Election	Lost	Obama		Election	Lost	Obama
D3 [0	1	1]	Q [0	0	1]

$$Sim(Q, D1) = \frac{\sum_{i=1}^3 Q_i D1_i}{\sqrt{\sum_{i=1}^3 Q_i^2} \sqrt{\sum_{i=1}^3 D1_i^2}}$$

$$Sim(Q, D2) = \frac{\sum_{i=1}^3 Q_i D2_i}{\sqrt{\sum_{i=1}^3 Q_i^2} \sqrt{\sum_{i=1}^3 D2_i^2}}$$

$$Sim(Q, D3) = \frac{\sum_{i=1}^3 Q_i D3_i}{\sqrt{\sum_{i=1}^3 Q_i^2} \sqrt{\sum_{i=1}^3 D3_i^2}}$$

Example: Instantiation

	Election	Lost	Obama		Election	Lost	Obama
D1 [1	1	1]	D2 [0	1	0]

	Election	Lost	Obama		Election	Lost	Obama
D3 [0	1	1]	Q [0	0	1]

$$\text{Sim}(Q, D1) = \frac{0*1 + 0*1 + 1*1}{\sqrt{1} * \sqrt{3}} = 0.577$$

$$\text{Sim}(Q, D2) = \frac{0*0 + 0*1 + 1*0}{\sqrt{1} * \sqrt{1}} = 0.000$$

$$\text{Sim}(Q, D3) = \frac{0*0 + 0*1 + 1*1}{\sqrt{1} * \sqrt{2}} = 0.707$$

Terms rarity

- Common terms are less informative than rare terms
- Consider a query like “*good refrigerators*”
 - With TF (term frequency) we consider the words frequency in each document
 - However, *good* is a very common word, that can be found in most documents, while *refrigerators* is less common and thus more informative for relevance
 - We **should weight more the rare words** than the common words
- We will use **inverse document frequency (idf)** to capture this, which in turn is based on document frequency (**df**)

DF: Document Frequency

- df_t is the document frequency of t : the number of documents that contain t
 - Note: despite its name is *document* frequency, it involves the **whole collection of documents!**
 - It measures **how common a term is in the whole collection of documents**
- df_t is an **inverse** measure of the informativeness of t
 - The **more common** is a word, the **less informative** will be for relevance (extreme case: stop words)
 - We use the **df** to formulate the **inverse** concept, in order to express the **rarity** of a word

IDF: Inverse Document Frequency

- We define the **inverse document frequency** (idf) of term t by

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

- N is the number of documents in the collection
- We use **log** (N/df_t) instead of N/df_t to smooth the effect of idf, for example:
 - a common word like “*home*” can have a frequency **1 million bigger** than a rare word such as “*fenestration*”
 - we apply the logarithm to flatten this **exponential** phenomenon of linguistics (Zipf’s law), otherwise very common words would have an idf of zero

Effect of IDF on Ranking

- Does idf have an effect on ranking for one-term queries, like
 - 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**
- IDF is never used alone, instead is used together with the TF for a better estimation of document relevance

TF-IDF 1/2

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

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

- 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**

TF-IDF 2/2

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

- The tf-idf weight of a term:
 - Increases with the number of occurrences within a document (**tf**)
 - Increases with the rarity of the term in the collection (**idf**)

How Search Engines Work



The user
inputs a query

Google

How search engines
how search engines work
how search engines work google
how search engines rank
how search engines see my site
how search engines work for dummies
how search engines operate
how search engines work pdf
how search engines work video
how search engines see your site
how search engines make money

Google Search I'm Feeling Lucky

The search engine search
the query terms in its very
large **index**



About 83,900,000 results (0.36 seconds)

Millions of results
found are **ranked**
accordingly to a
notion of **relevance**
and are shown as a
sorted list

Automated programs (a.k.a.
spiders or **crawlers**) scan all
pages in the web to build an
index



How search engines work

Web Videos Images News Shopping More Search tools

About 83,900,000 results (0.36 seconds)

You can find pages by following links from other pages but usually it is easier to **search** for things using a **search** engine. These are programs that **search** an index of the world wide web for keywords and display the results in order.

BBC Bitesize - How do search engines work?
www.bbc.co.uk/guides/ztbjq8f British Broadcasting Corporation

How Search Engines Work - The Beginners Guide to SEO ...
moz.com/beginners-guide-to-seo/how-search-engines-operate Moz

Search engines have two major functions: crawling and building an index, and providing search users with a ranked list of the websites they've determined are ...

How Internet Search Engines Work - HowStuffWorks
computer.howstuffworks.com/internet/basics/search-engine.htm

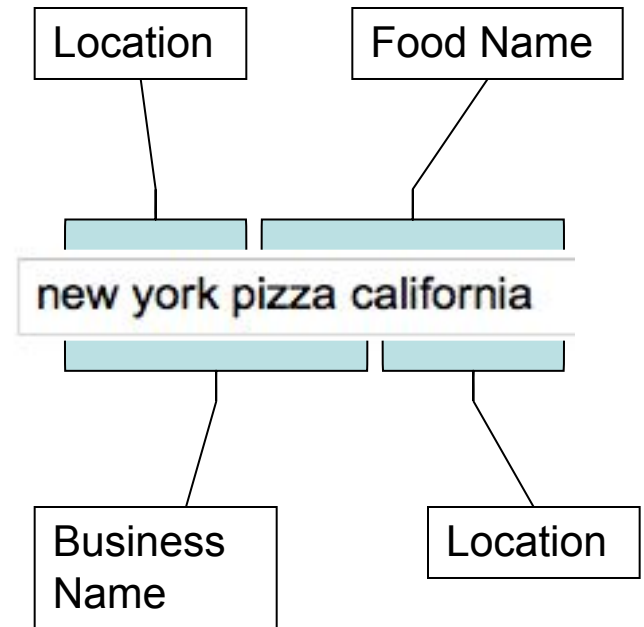
Internet search engines do your research for you. Learn how internet search engines like Google work, how internet search engines build an index and what ...

Images for How search engines work Report images

What is Relevant?

- How can a search engine know what the user is looking for?
 - **Information need:** *a desire to locate and obtain information to satisfy a conscious or unconscious need* [Wikipedia]
 - **User query:** the set of consecutive terms formulated by a user to express his information need
 - **Query intent:** the task, goal or intent of a user, expressed by a query. The same query formulated by different users may be the result of different intents

Query intent can be ambiguous:



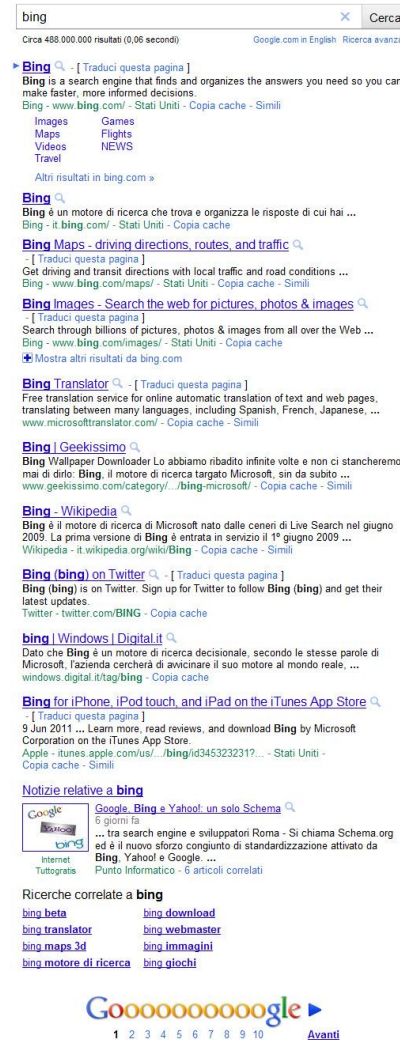


Ranking: Google PageRank

- A relevance notion based solely on term frequencies is not enough to rank billions of documents
- Complex measures are applied to evaluate the quality, reliability and authority of web pages:
 - Measures based **on topological network properties** (such as Google PageRank)
 - Measures based **on different field boosting** (Title, subtitle, body have different weight)
 - Measures based **on semantics and time-space properties** (freshness of a page)
- PageRank set up a **rich-get-richer loop**, whereby few sites dominate the top ranks

Search Engine Results Page

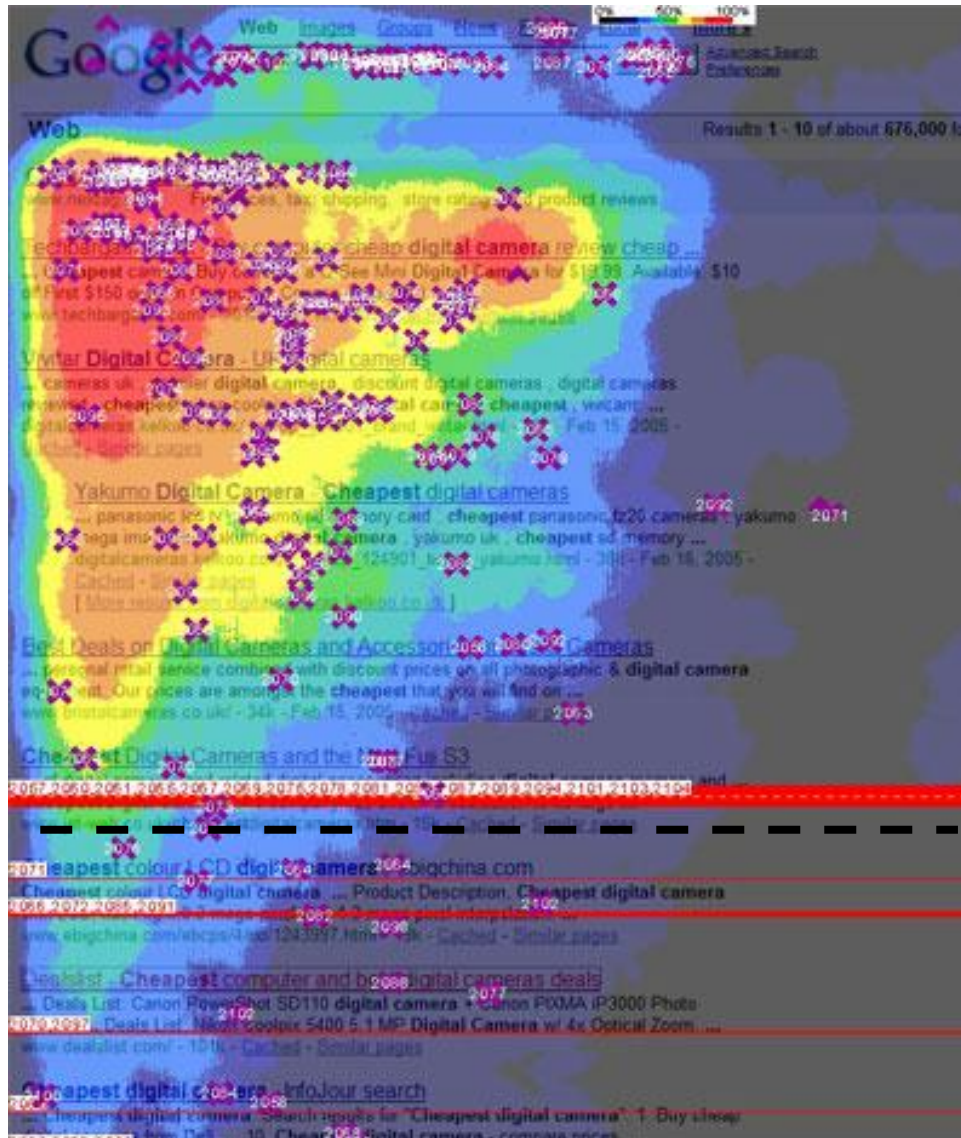
- A Search Engine Results Page (SERP) shows by default **10 results**
- Of these results, on average, the first **5 are visible** without scrolling down
- We will take in consideration the **user behavior on the first SERP** of search engines (the first 10 results)

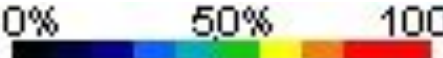



Visible
Area

Scroll
Area

Eye-tracking in 2005: Golden Triangle



- Color  represents percentage of time spent looking the area
- Purple X represents a mouse click on the page
- Dotted line - - - represents where the page breaks on the computer screen (Visible area/Scroll area)
- Red lines  indicate how far down the page scrolled before leaving the page

Results in 1st position are clicked more than 10 times results on 6th position

Eye-tracking in 2014: F-shape Scan

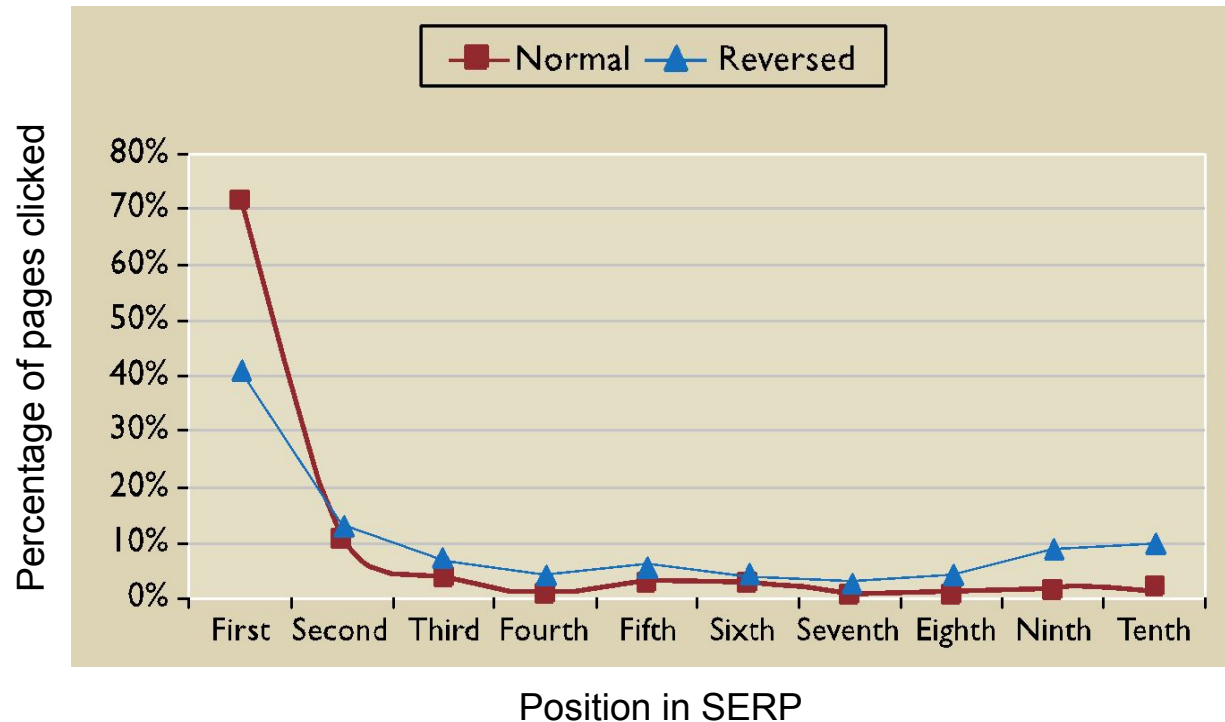


- In 2005, all results were links to a website. Now, we have a variety of results, and the results page layout varies from search to search.
- The first (vertical) glance allows to determine what categories Google has decided to show.
- Then, in a second (horizontal) glance, we go back and deep to the most relevant chunks of results.

The time spent scanning each result is halved: in 2005 was 2.5 seconds, in 2014 is 1.2 seconds.

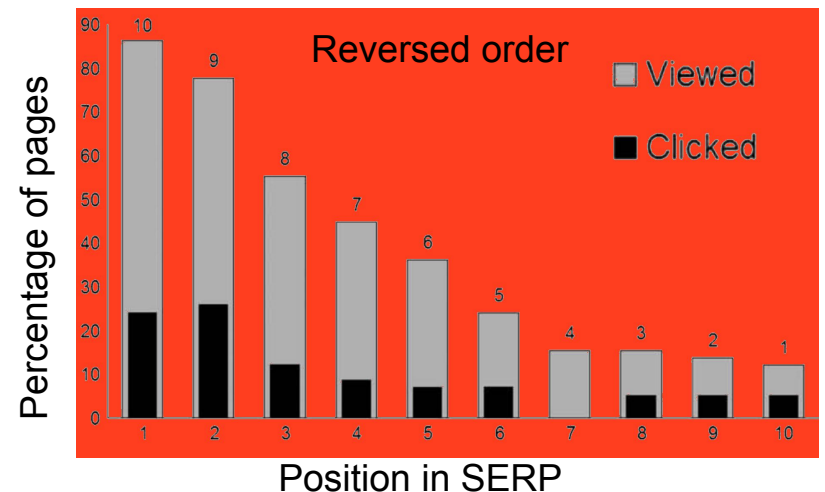
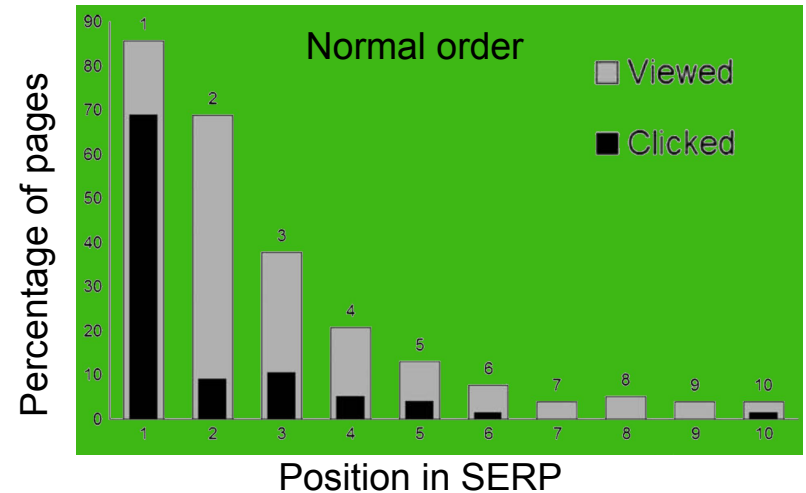
Result Positioning Bias

- When searching for information, the average searchers do not judge systematically all the results, instead they **simply click on top results**
- Studies have been conducted comparing the users' responses when they received results lists in **normal ordering versus a reversed order**



Click and Eye-tracking Bias

- View percentage and mouse clicks are compared in normal vs. reversed order of the first 10 results
- Even though results show some user awareness, **excessive trust in ranking** algorithms may negatively affect smaller positioned websites



Position Impact on Business

Web Images Videos Maps News Shopping Gmail more ▾

Google shopping

About 958,000,000 results (0.21 seconds)

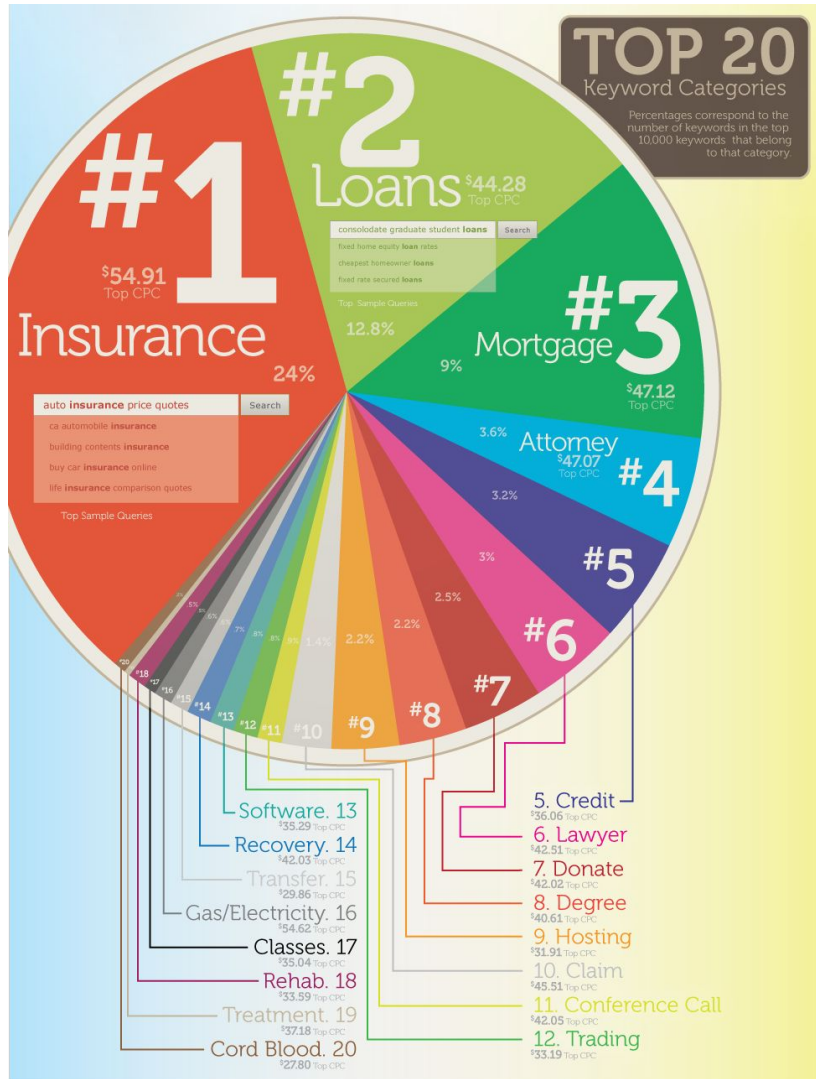
2006 AOL LEAKED DATA

	% OF CLICKS
Shopping.com - Online shopping. Price comparison, product reviews ... Shopping.com. Shop and find the lowest prices on products from online merchants. Price comparison shopping, product reviews, and store ratings - all the ... Electronics - Clothing - Shoes - Digital Cameras www.shopping.com/ - Cached - Similar	42.13%
Overstock.com: Online Shopping - Bedding, Furniture, Electronics ... Purchase name brands at clearance prices at Overstock.com! Buy discounted name brand Bedding, Furniture, Jewelry, Watches, Electronics, Clothing, Books, ... ✦ Show stock quote for OOSTK Furniture - Electronics - Clothing & Shoes - Jewelry www.overstock.com/ - Cached - Similar	11.90%
PriceGrabber.com - Comparison Shopping Beyond Compare Get the lowest prices on computers, electronics, furniture, cameras and more. Compare products and find the best deal with PriceGrabber.com. www.pricegrabber.com/ - Cached - Similar	8.50%
Welcome to Easton Town Center in Columbus, Ohio Site offers directions, store directory, dining and calendar of events. eastontowncenter.com/ - Cached - Similar	6.06%
Columbus Ohio Shopping - Find Stores and Malls in Columbus Ohio Directory of Places to Shop in Columbus Ohio including Local Malls and Stores. columbus.retailguide.com/ - Cached - Similar	4.92%
Shopping Columbus OH Ohio + Outlet Mall - City Guide Shopping Columbus OH Ohio, Outlet Malls, Jewelry, Furniture and Discount Retailers. www.discoverourtown.com/ Ohio - Cached - Similar	4.05%
Shopping - Google YouTube Calendar Photos Documents Reader Sites Groups - even more » - My Shopping List Search settings Sign in, Google Product Search ... www.google.com/prdhp?hl=en&tab=wf - Cached - Similar	3.41%
Yahoo! Shopping - Online Shopping with great products, prices and ... Yahoo! Shopping is the best place to read user reviews, explore great products and buy online. shopping.yahoo.com/ - Cached - Similar	3.01%
Amazon.com: Shopping: Sadie Frost, Jude Law, Sean Pertwee, Fraser ... Amazon.com: Shopping: Sadie Frost, Jude Law, Sean Pertwee, Fraser James, Sean Bean, Marianne Faithfull, Jonathan Pryce, Daniel Newman, Lee Whitlock, ... www.amazon.com/ » ... » Action & Adventure » Crime - Cached - Similar	2.85%
QVC Official Site - Online Shopping for Jewelry, Beauty ... Online Shopping, QVC live for Beauty, Jewelry, Electronics, Fashion, Handbags, Kitchen, Home, Food, Sports, Wellness, Crafts, Leisure, Toys, Todays Special ... www.qvc.com/ - Cached - Similar	2.99%

- Results in 1st position are clicked **more than 10 times** results on 6th position
- Suppose company A and company B are respectively on the 1st and 6th position for a valuable business keyword
- B should buy 10 times the clicks that A gets to obtain the same traffic!
- How much does a click cost?**

[[BMW given Google 'death penalty'. 2006](#)]

Click Economic Value



- In web advertising, the cost of a single click, also known as Cost-Per-Click (CPC), is **1\$ on average**. However...
 - On Google some keywords can be really expensive ([Google most expensive keywords 2011](#)):
 - **Insurance: 54.91\$**
 - Mortgage: 47.12\$
 - Attorney: 47.07
 - Claim: 45.51\$
 - Loans: 44.28\$
 - Lawyer: 42.51\$
 - Bing is even more expensive ([Bing most expensive keywords 2015](#)):
 - **Lawyers: 109.21\$**
 - Attorney: 101.77\$
 - Structured settlements: 78.39\$

Retrieval Models

- **Classic models**

- Boolean retrieval
- Vector Space model

- **Probabilistic Models**

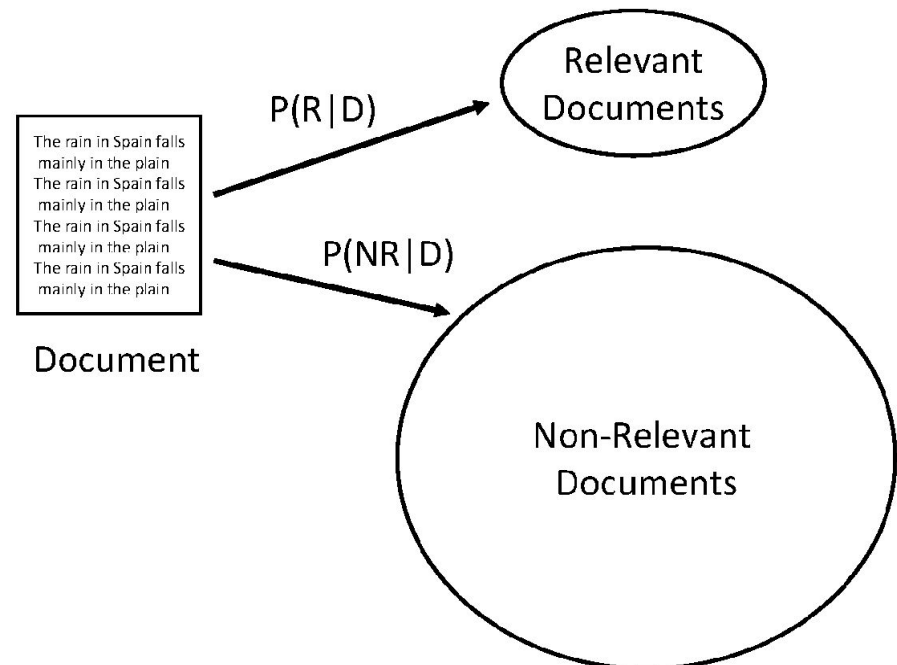
- BM25
- Language models

- Progress in retrieval models has corresponded with improvements in effectiveness

Probabilistic Models

- The Probability Ranking Principle in IR, [Robertson \(1977\)](#)
 - The ranking is given by a **probability of relevance**
 - Probability is estimated as accurately as possible using the available data → best effectiveness

“[...] If the **ranking of the documents is in order of decreasing probability of relevance** (where the probabilities are estimated as accurately as possible from the available data) **the overall effectiveness will be the best** that is obtainable on the basis of those data.”

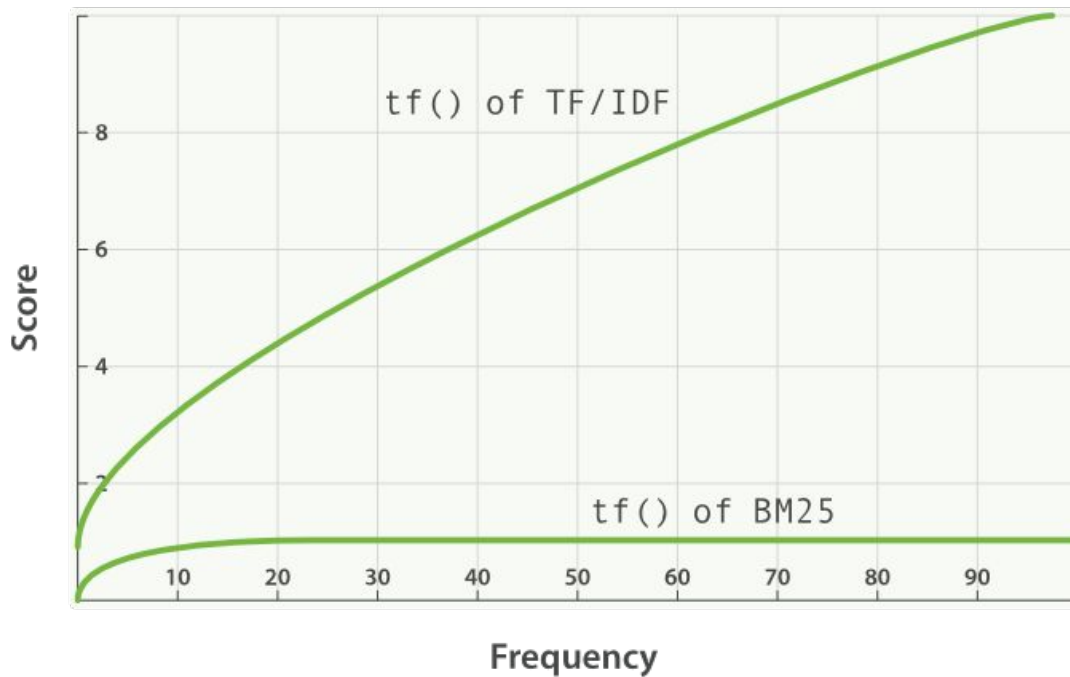


Okapi BM25

- The BM25 weighting scheme, often called *Okapi weighting*, after the system in which it was first implemented, was developed as a way of building a probabilistic model sensitive to
 - **term frequency**
 - **term rareness** in corpus (similarly to IDF)
 - **document length**
- BM25 became a popular and effective ranking algorithm while relying on probabilistic foundations
- The BM25 term weighting formulas have been used quite widely and quite successfully across a range of collections and search tasks

Okapi BM25: TF saturation

- In TF-IDF, a document similarity score could reach very high values if the term frequency of the query term is very high
- In Okapi BM25, the similarity score is always comprised between 0 and 1
 - At a certain point, highest values of term frequency do not affect the score



Okapi BM25: parameters

- Okapi BM25 has two parameters:
 - **k1**: this parameter controls **how quickly an increase in term frequency results in term-frequency saturation** (i.e., the slope of “tf() of BM25” curve)
 - The default value is 1.2
 - Lower values result in quicker saturation
 - Higher values in slower saturation
 - **b**: this parameter controls **how much effect document length normalization should have**
 - The default value is 0.75
 - A value of 0.0 disables normalization completely
 - A value of 1.0 normalizes fully

Language Model

- A statistical language model assigns a probability to a sequence of m words $P(w_1, \dots, w_m)$ by means of a probability distribution
- A separate language model is associated with each document in a collection. Documents are ranked based on the probability of the query Q in the document's language model $P(Q \mid M_d)$.
- Unigram language model
 - probability distribution over the words in a language
 - generation of text consists of pulling words out of a “bucket” according to the probability distribution and replacing them
- N-gram language model
 - some applications use bigram and trigram language models where probabilities depend on previous words

Efficiency Aspects of Ranking

- Computing the relevance score using non boolean models can be quite computationally-intensive:
 - Document is retrieved (has a relevance score) even if not all the query terms are found in it
 - Complex relevance formulas requires additional computations to compute score
- Strategies to control the efficiency:
 - The way we access documents affects computation time (one document at a time or one term at a time?)
 - Most of the time we are not interested in the least relevant documents



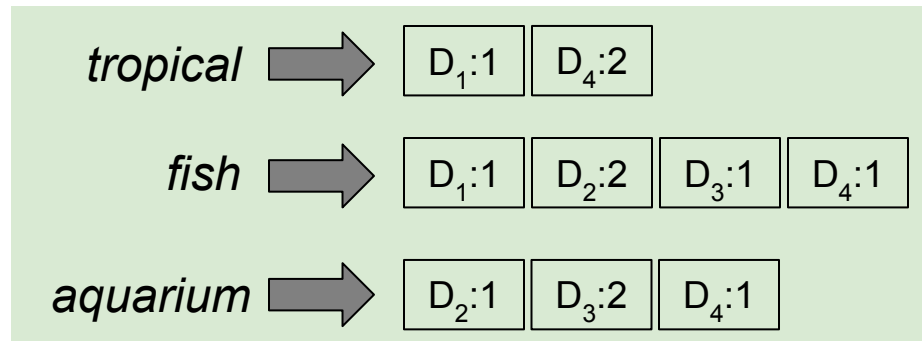
TAAT vs DAAT Techniques

- **TAAT** : “Term At A Time”
 - Scores for all docs computed concurrently, one query term at a time
- **DAAT** : “Document At A Time”
 - Total score for each doc (including all query terms) computed, before proceeding to the next
- Each has implications for how the retrieval **index** is structured and stored

TAAT: Example

Query: *tropical fish aquarium*

Inverted index:



Term frequency for each document:

$$D_1 = 1 + 1 = 2$$

$$D_2 = 2 + 1 = 3$$

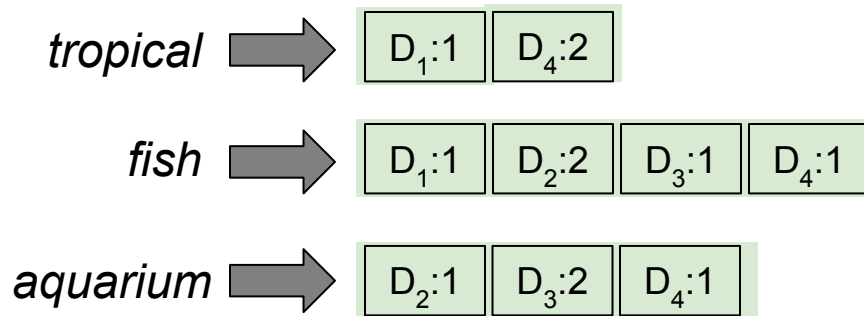
$$D_3 = 1 + 2 = 3$$

$$D_4 = 2 + 1 + 1 = 4$$

DAAT: Example

Query: *tropical fish aquarium*

Inverted index:



Term frequency for each document:

$$D_1 = 1 + 1 = 2$$

$$D_2 = 2 + 1 = 3$$

$$D_3 = 1 + 2 = 3$$

$$D_4 = 2 + 1 + 1 = 4$$

Efficient Scoring

- A large fraction of the CPU work for each search query is devoted just to **compute the score** of relevance
 - Generally, we have a tight budget on latency (e.g., 250ms) to answer the query, otherwise the user will not be happy
 - CPU provisioning doesn't permit exhaustively scoring every document for every query
- Basic idea: avoid scoring docs that won't make it into the top K

Safe Ranking

- The terminology “safe ranking” is used for methods that guarantee that the K docs returned are the **K absolute highest scoring** documents
 - (Not necessarily just under cosine similarity)
- Is it ok to be non-safe?
 - Yes, the results would be approximated but good enough to satisfy the user
- The goal is to run less computations while giving good results



Non-safe Ranking

- Non-safe ranking may be okay
 - Ranking function is only a proxy for user happiness
 - Documents close to top K may be just fine
- Non-safe: **Index “elimination”**
 - Only consider **high-idf (rare) query terms**, discard the common ones
 - Only consider **documents containing many query terms**, discard the ones with only few query terms
- Non-safe: **Champion lists**
 - For each term, the scores of top relevant documents are pre-computed (before any query is issued)

References

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze
Introduction to Information Retrieval
Cambridge University Press. 2008

The book is also online for free:

- HTML edition (2009.04.07)
- PDF of the book for online viewing
(with nice hyperlink features,
2009.04.01)
- PDF of the book for printing
(2009.04.01)

