Wednesday Schedule

8:30-9:00	Continental Breakfast
9:00-10:30	Intro to Text Retrieval (Paul McNamee)
10:30-10:50	Break
10:50-12:10	Learning to Rank (Kevin Duh)
12:10-1:00	Lunch Break
2:00-3:00	Learning to Rank Lab (Kevin Duh)
3:00-3:30	Break
3:30-5:00	Learning to Rank Lab (Kevin Duh)

human language technology
center of excellence

Information Retrieval from Soup to Nuts

Paul McNamee

19 June 2018

JOHNS HOPKINS
UNIVERSITY

Hello

maoi

tom yam soup

Paul McNamee
JHU APL & HLTCOE
mcnamee@jhu.edu

- **Research Interests**
  - IR, IE, NER, MT, entity linking > other text stuff
- **Other interests**
  - flatwater paddler
  - python, lisp > java > perl > c++
- **Love spending summer evenings doing HLT**
  - TREC (14x), TAC (7x), CLEF (10x), FIRE (2x), NTCIR (2x), CoNLL (3x), numerous CLSP & COE workshops

over

What is Information Retrieval?

Google is college aged - Founded ~ 20 years ago

What is Information Retrieval?

- **Field concerned with the organization, storage, and retrieval of information**
  - **Especially text**
  - **Also retrieval of semi-structured data (XML), video images, speech, music, etc...**

- **Requires algorithms and data structures**
  - **For manipulating natural language**
  - **To efficiently store and process data**

I never waste memory on things that can easily be stored and retrieved from elsewhere – A. Einstein

over

Text is Unstructured

- **Vis-à-vis RDBS**
  - **Compare**
    - **SELECT SALARY FROM EMPTBL WHERE BASEPAY > $100,000**
    - **“Find salary surveys for CS/IT professionals in the Washington DC area”**
  - **SQL semantics are clearly specified**
    - **A single omission results in a completely incorrect response to a query**
    - **Language is less well-defined; missing one relevant document might not be catastrophic**

over

Small Differences in Language Matter

- Find salary surveys for computer scientists in:
  - Seattle, Washington
  - Washington, DC

- List Fortune 500 CEOs who are not male

over

Variability and Ambiguity

- **Language usually provides no canonical way to reference people and things**
  - **President Carter, Pres. Carter, Jimmy Carter; the 39th president, Rosalynn Carter's husband**
- **Ambiguity (*polysemy*) pervasive**
  - **jaguar, bank, see, hornet, red, aa,**
- **Distinctions vary in granularity**
  - **cool (popular) vs. cool (low in temperature)**
  - **list (to recite items in a list) vs. (to include in a list)**

over

Word Choices

- **Text documents have a limited vocabulary with discrete occurrences; words have many *synonyms***
  - **query: ‘fast automobiles’**
    - **should also match ‘fast cars’**

- **Inflectional variation (*morphology*)**
  - **query about ‘juggling’**
    - **should match jugglers, juggler, jongleur**

over

Summary: Three Classic Problems

- **Polysemy**
  - Words can have multiple meanings
  - lead: (chem element, to be in charge of)

- **Synonymy**
  - The same concept can be expressed using different words

- **Morphology**
  - Many word forms are related
    - juggle, juggling, juggled, jugglers
    - actor, actress
    - go, going, went

Pre-history of IR

- 300 BCE Ptolemy I founds Great Library at Alexandria which grows to include 700,000+ volumes (scrolls)
- 1230s St. Anthony of Padova creates concordance for Latin Vulgate
- 1247 Cardinal Hugo employs 500 monks to build a concordance
- 1470s Johannes Gutenberg builds printing press
- 1714 Henry Mills conceives of the typewriter
- 1872 21-year old Melvil Dewey invents a classification code
- 1890 Dr. James Strong (and students) create an 'exhaustive' concordance
- 1900 John Ambrose invents the vacuum tube

over

Entry from Strong's Concordance

Ob 18 flame, and the house of Esau for s', 7179
Na 1:10 they shall be devoured as s' "
Mal 4: 1 all that do wickedly, shall be s': "
1Co 3:12 precious stones, wood, hay, s'; *2562*

stubborn
De 21:18 man have a s' and rebellious son, 5637
20 This our son is s' and rebellious, "
J'g 2:19 doings, nor from their s' way. 7186
Ps 78: 8 a s' and rebellious generation; 5637
Pr 7:11 (She is loud and s'; her feet abide "

stubbornness
De 9:27 look not unto the s' of this people, 7190
1Sa 15:23 and s' is as iniquity and idolatry. 6484

stuck
1Sa 26: 7 his spear s' in the ground at his 4600
Ps 119:31 I have s' unto thy testimonies: *1692
Ac 27:41 the forepart s' fast, and remained *2043*

studs
Ca 1:11 borders of gold with s' of silver. 5351

studieth
Pr 15:28 of the righteous s' to answer: 1897
24: 2 For their heart s' destruction, and "

study See also STUDIETH.
Ec 12:12 much s' is a weariness of the flesh. 3854
1Th 4:11 that ye s' to be quiet, and to do *5389*
2Ti 2:15 S' to shew thyself approved unto **4704*

11 The words of the wise are like goads, their collected sayings like firmly embedded nails[p]—given by one Shepherd. 12 Be warned, my son, of anything in addition to them.

Of making many books there is no end, and much study wearies the body.[q]

13 Now all has been heard;
here is the conclusion of the matter:
Fear God[r] and keep his commandments,[s]
for this is the whole duty of man.[t]
14 For God will bring every deed into judgment,[u]
including every hidden thing,[v]
whether it is good or evil.

3853. לְהָבִים **Lehâbîym**, *leh-haw-beem'*; plur. of 3851; *flames*; *Lehabim*, a son of Mizrain, and his descend.:—Lehabim.

3854. לַהַג **lahag**, *lah'-hag*; from an unused root mean. to *be eager*; intense mental *application*:—study.

3855. לַהַד **Lahad**, *lah'-had*; from an unused root mean. to *glow* [comp. 3851] or else to *be earnest* [comp. 3854]; *Lahad*, an Isr.:—Lahad.

over

Advent of Computer Science

- 1941 Harvard Mark I computer (Howard Aiken and Thomas J. Watson Sr.)
- 1945 Vannever Bush conceives of MEMEX device ("As we may think" in Atlantic Monthly)
- 1948 Claude Shannon's work in information theory, coins term 'bit'
- 1962 First Comp Sci. degree program offered by Purdue U.
- 1963 ASCII standard developed
- 1972 Tomlinson sends first email message
- 1975 Microsoft founded by Gates and Allen
- 1977 Apple II personal computer
- 1981 IBM PC
- 1984 Apple Macintosh with windowing interface
- 1984 1,000 Internet hosts

Birth of the Web

- 1989 Tim Berners-Lee invents World-Wide-Web
- 1992 1,000,000 Internet hosts, but only 50 web sites
- 1994 Two Stanford graduate students found Yahoo, a manually build on-line directory
- 1995 AltaVista indexes 15 million web pages
- 1996 Two other Stanford graduate students collaborate on Google
- 1997 Lawrence and Giles paper characterizing Web
- 1999 Excite search engine sold for $6.7 billion; around same time automotive division of Volvo sold for $6.3 billion.
- 2000 1 billion web pages on public web; 10 million web sites, 93 million or so Internet hosts
- 2002 Google claims 3 billion page index
- 2004 Google IPO
- 2004 Microsoft unleashes Web search engine
- 2006 Google's stock value exceeds $150 billion (> Coke, IBM, AT&T)
- 2009 Microsoft rebrands Web search as Bing

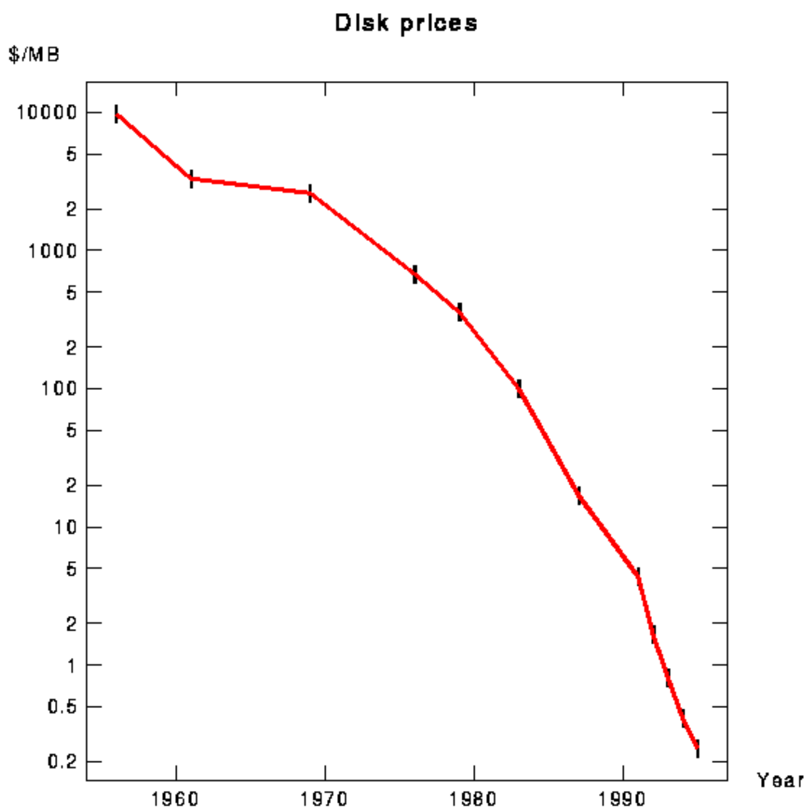
over

20 to 30 years ago

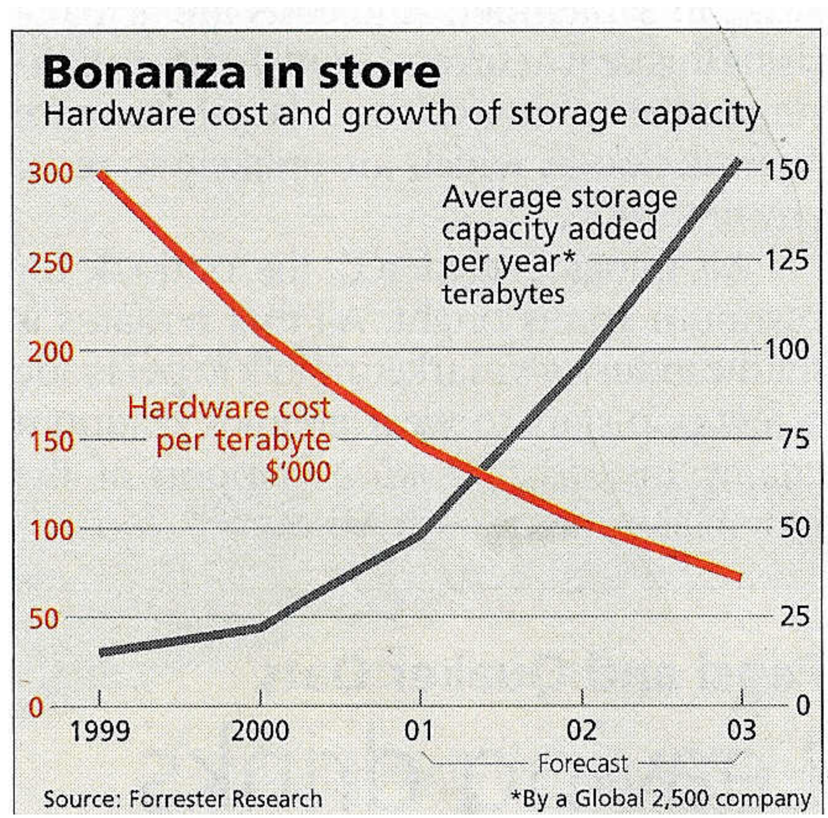
- Wikipedia: In 1982, a [BLANK] could store much more data than a personal computer hard drive
- How big was a laptop hard disk in 1998?
- How much would 1 TB of storage cost in 1998?

Why Has IR Thrived?

- **Dropping prices for external storage is one of the greatest factors**



From www.lesk.com



Sample Task

Suppose I offer you $1 million if you can correctly identify a street in Ohio where a CPK is located next to a Saks 5th Ave. You have 30 seconds. Can you do it?

The Feynman Problem-Solving Algorithm: (1) write down the problem; (2) think very hard; (3) write down the answer. – Murray Gellmann

index

Key Data Structure: Inverted Files

- **Inverted files are a data structure that stores for each term, a list of documents containing that term**
- **Commonly include the number of times that term occurs; possibly even the word-order**
  - **Large binary files, only 15-20% the size of the indexed text**

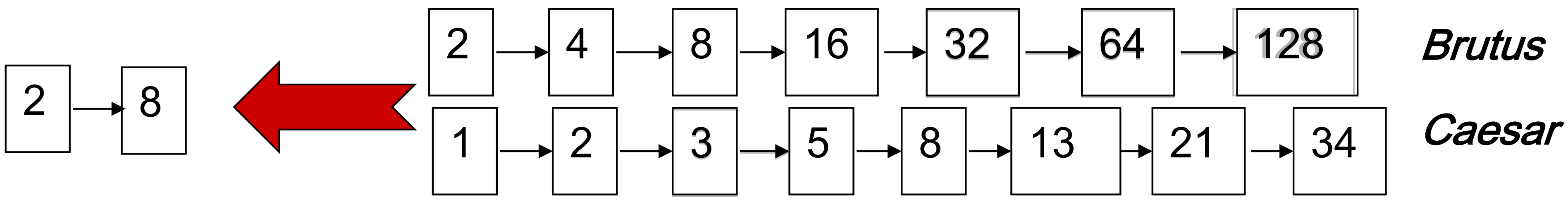
lists called “postings lists”

	doc	cnt	doc	cnt	doc	cnt	doc	cnt
cpk	1	2	6	1	87	1	92	1
saks	1	8	17	2	45	1		
starbucks	5	1	6	1	87	3	101	3

Term starbucks occurs in 4 documents. It occurs 3 times in document 87.

The merge (Boolean AND)

- **Walk through the two postings simultaneously, in time linear in the total number of postings entries**



If the list lengths are *x* and *y*, the merge takes O(*x*+*y*) operations.

<u>Crucial</u>: postings sorted by docID.

How Inverted Files are Created

- **Documents are parsed to extract words and these are saved with the Document ID.**

Doc 1

Now is the time for all good men to come to the aid of their country.

Doc 2

It was a dark and stormy night in the country manor. The time was past midnight

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

How Inverted Files are Created

- **After all documents have been parsed the temporary inverted file is sorted**
- **‘Sort-based’ inversion**
  - **See Managing Gigabytes Section 5.2**
  - **(Or Zobel/Moffat paper)**

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

➔

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

How Inverted Files are Created

- **Multiple term entries for a single document are merged and frequency information added**

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

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

index

How Inverted Files are Created

- **The file is commonly split into a Dictionary and a Postings (or Inverted) File**

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

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

Building Inverted Files

- Doc 1: Socrates is a man
- Doc 2: All men are mortal
- Doc 3: Socrates is mortal, mortal

Records in inverted file are pairs (docid and count)

Dictionary

Term	ID	DF	#Occur	Pointer
socrates	0	2	2	
is	1	2	2	
a	2	1	1	
man	3	1	1	
all	4	1	1	
men	5	1	1	
are	6	1	1	
mortal	7	2	3	

Inverted File

Doc	times	Doc	times
1	1	3	1
1	1	3	1
1	1		
1	1		
2	1		
2	1		
2	1		
2	1	3	2

Data structures usually rely on termids vs. strings

index

Summary: Inverted Files

- Permit fast search for individual terms
  - Up-front cost to make searches fast
- Associated with each term is a list of document IDs (and optionally, frequency and/or positional information)
- These lists can be used to solve Boolean queries:
  - country: d1, d2
  - manor: d2
  - country and manor: d2

Complexity of Index Construction

- **Time**
  - **Linear in the length of the text**
  - **Assumption: vocabulary fits in memory**
  - **Easily parallelizable (Map/Reduce)**
- **Space**
  - **20-30% of input text is typical (for a position-less index)**
  - **Clever compression techniques ~10-15%**

Alternatives?

- **Grep**
  - **Never index, just do linear scan**

- **Suffix trees, Suffix arrays**
  - **Requires 4x space of input text**
  - **Most useful if text and SA fit in memory**

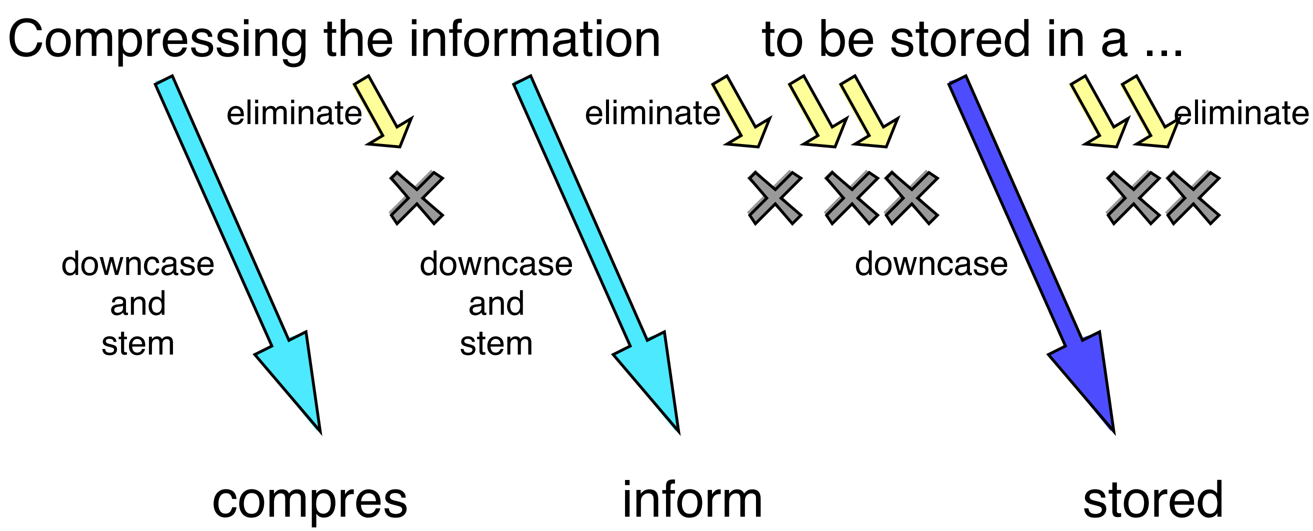
Summary: Boolean Search

- Pros
  - ➢ Good performance with well-constructed queries
    - – ~25% more accurate on human constructed queries than an automatic non-Boolean model
  - ➢ Representation is space-compact
  - ➢ Results are transparent
    - – Docs contain, or don't contain terms of interest

- Negatives
  - ➢ Ignores if a document contains query terms more than once
  - ➢ If a document contains other words besides the query terms, (is unfocused), there is no penalty
  - ➢ Document scores are 0/1 (specificity is low)
  - ➢ Long/Complex queries are hard to construct
    - – All words for concept 'weapon': knife or gun or hammer or sword or bow-and-arrow or rope or candlestick ...

Representing Documents: Tokenization

tok



Identical processing is done to both documents and queries

tok

Issues

- Word Segmentation
  - RateMyProfessor.com, 珠穆朗玛峰
- Punctuation, Hyphenation
  - sanjeev@grumpy-bear.jhu.edu
- Case
  - “us” vs. U.S.
- Numbers
  - Flight 93, Y2K, 1%, 3rd place, 1-800-CONTACTS, 3.14159
- Abbreviations
  - parked on Bureau Dr. Pepper and salt make peas taste...
  - JHU vs. Johns Hopkins
- Misspellings
- Diacritical marks
  - resume vs. résumé vs. resumé, schuetze vs. schütze

“I Can’t Believe It’s Not Butter” is a single proper noun.

Popular steps

- Stopword removal
  - remove / discard common words: the, a, an, of, with, ...
  - "to be or not to be"
- Simple normalization of word forms
  - 'stemming' or suffix removal
  - golfing, golfers, golfed transformed to "golf"
- Most systems do both
  - Neither is harmless
  - Both can be useful, but stemming is the more valuable

tok

Stopping

- **Motivation**
  - ➢ **Reduce size of inverted index**
    - – **With compression, this effect is minimal (4%)**
  - ➢ **High frequency words have low discrimination power**

- **Standard lists exist**
  - ➢ **Google: English stopword list**

Stemming

- **Motivation**
  - Treat morphological word variants identically
  - Also reduces the size of the lexicon
- **Example**
  - remove plural forms, map cat<u>s</u> to cat
  - juggle, juggling, juggler, juggles
    - probably shouldn't be confused with 'jug'
    - but, suffix removal won't find jongleur
  - physics & physician
- **The technique is conflationary**
  - Distinctions are lost
  - Can help and can sometimes hurt

tok

Simple "S" stemming

- IF a word ends in "ies", but not "eies" or "aies"
  - THEN "ies" → "y"

studies -> study

- IF a word ends in "es", but not "aes", "ees", or "oes"
  - THEN "es"→ "e"

leaves -> leave

Not leaf …

- IF a word ends in "s", but not "us" or "ss"
  - THEN "s" → NULL

books -> book

Harman, JASIS 1991

tok

Porter Stemmer

Uses a list of suffixes and applies transformation rules until no further rules can be applied

Multiple versions

Freely available: http://snowball.tartarus.org/

- **Too aggressive**
  - organization / organ
  - policy / police
  - execute / executive
  - army / arm
- **Too timid**
  - european / europe
  - cylinder / cylindrical
  - create / creation
  - search / searcher

Word-based Information Retrieval

- Most traditional information retrieval systems index documents according to the words in those documents.

- Word-based retrieval is language-specific (e.g., a retrieval system for English will not work as well for Arabic, Japanese, Korean, Turkish, and other languages).

- Word-based retrieval performs poorly when the documents to be retrieved are garbled or contain spelling mistakes (e.g., from OCR or speech transcription).

N-gram Tokenization

- Represent text as overlapping substrings
- Fixed length of *n* of 4 or 5 is effective in alphabetic languages
- For text of length *m*, there are *m-n+1* n-grams

	s	w	i	m	m	e	r	s	
_	s	w	i	m					
	s	w	i	m	m				
		w	i	m	m	e			
			i	m	m	e	r		
				m	m	e	r	s	
					m	e	r	s	_

- Advantages: simple, address morphology, surrogate for short phrases, robust against spelling & diacritical errors, language-independent
- Disadvantages: conflation (e.g., simmer, polymers), n-grams can incur both speed and disk usage penalties

Against: Damashek (1995)

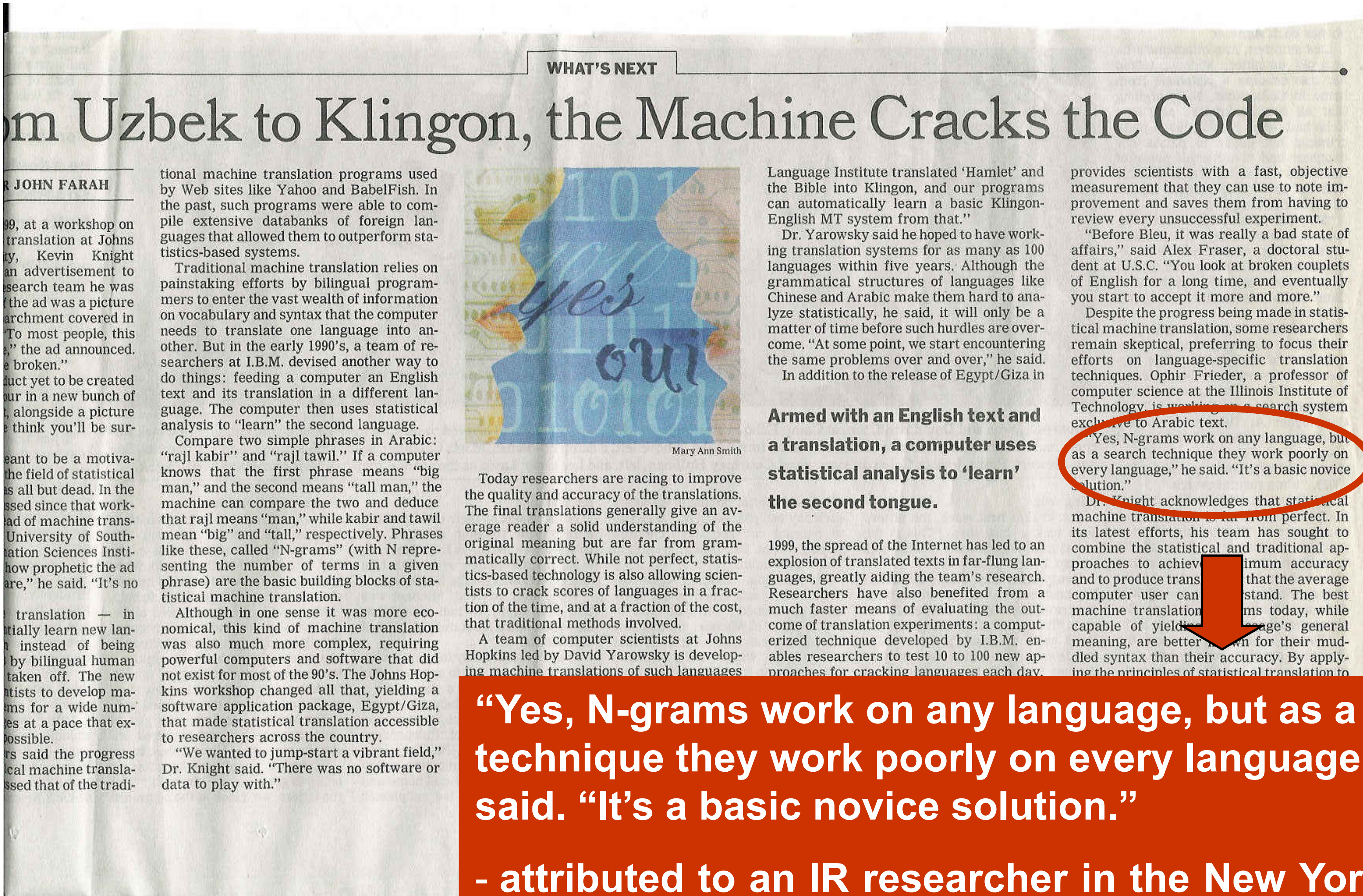
- Marc Damashek and colleagues developed an IR system (ACQUAINTANCE) based on n-grams
  - *‘Gauging Similarity with n-Grams: Language Independent Categorization of Text’*, Science, vol. 267, 10 Feb 1995
  - Increased size of ‘n’, considered many languages
  - The article described system performance at TREC-3 as:
    - “on a par with some of the best existing retrieval systems.”

- The article elicited strong reaction
  - IR luminary Gerard Salton wrote a response
    - “decomposition of running texts into overlapping n-grams ... is too rough and ambiguous to be usable for most purposes.”
    - “for more demanding tasks, such as information retrieval, the n-gram analysis can lead to disaster”
    - “decomposition of text words such as HOWL into HOW and OWL raises the ambiguity of the text representation and lowers retrieval effectiveness”

Pro: Asian Languages (1999)

- *Information Processing and Management* 35(4) was devoted to IR in Asian Languages
  - Many Asian languages lack explicit word boundaries
- Korean
  - Lee et al., KRIST Collection (13K docs)
    - 2-grams outperform words, decompounding cited
- Chinese
  - Nie and Ren, TREC 5/6 Chinese Collection (165K docs)
    - 2-grams (0.4161 avg. prec.) comparable to words (0.4300)
    - Combination of both is best (0.4796)
- Japanese
  - Ogawa and Matsuda, BMIR-J2 (5K docs)
    - M-grams (unigrams and bigrams) comparable to words

Against: "A Basic Novice Solution"



WHAT'S NEXT

om Uzbek to Klingon, the Machine Cracks the Code

R JOHN FARAH

99, at a workshop on translation at Johns ty, Kevin Knight an advertisement to esearch team he was the ad was a picture archment covered in To most people, this ," the ad announced. e broken."
luct yet to be created our in a new bunch of , alongside a picture e think you'll be sur-

eant to be a motiva- the field of statistical s all but dead. In the ssed since that work- ad of machine trans- University of South- ation Sciences Insti- how prophetic the ad are," he said. "It's no

translation — in tially learn new lan- instead of being by bilingual human taken off. The new ntists to develop ma- ms for a wide num- es at a pace that ex- possible.
rs said the progress ical machine transla- ssed that of the tradi-

tional machine translation programs used by Web sites like Yahoo and BabelFish. In the past, such programs were able to compile extensive databanks of foreign languages that allowed them to outperform statistics-based systems.

Traditional machine translation relies on painstaking efforts by bilingual programmers to enter the vast wealth of information on vocabulary and syntax that the computer needs to translate one language into another. But in the early 1990's, a team of researchers at I.B.M. devised another way to do things: feeding a computer an English text and its translation in a different language. The computer then uses statistical analysis to "learn" the second language.

Compare two simple phrases in Arabic: "rajl kabir" and "rajl tawil." If a computer knows that the first phrase means "big man," and the second means "tall man," the machine can compare the two and deduce that rajl means "man," while kabir and tawil mean "big" and "tall," respectively. Phrases like these, called "N-grams" (with N representing the number of terms in a given phrase) are the basic building blocks of statistical machine translation.

Although in one sense it was more economical, this kind of machine translation was also much more complex, requiring powerful computers and software that did not exist for most of the 90's. The Johns Hopkins workshop changed all that, yielding a software application package, Egypt/Giza, that made statistical translation accessible to researchers across the country.

"We wanted to jump-start a vibrant field," Dr. Knight said. "There was no software or data to play with."

Mary Ann Smith

Today researchers are racing to improve the quality and accuracy of the translations. The final translations generally give an average reader a solid understanding of the original meaning but are far from grammatically correct. While not perfect, statistics-based technology is also allowing scientists to crack scores of languages in a fraction of the time, and at a fraction of the cost, that traditional methods involved.

A team of computer scientists at Johns Hopkins led by David Yarowsky is developing machine translations of such languages

Language Institute translated 'Hamlet' and the Bible into Klingon, and our programs can automatically learn a basic Klingon-English MT system from that."

Dr. Yarowsky said he hoped to have working translation systems for as many as 100 languages within five years. Although the grammatical structures of languages like Chinese and Arabic make them hard to analyze statistically, he said, it will only be a matter of time before such hurdles are overcome. "At some point, we start encountering the same problems over and over," he said.

In addition to the release of Egypt/Giza in

Armed with an English text and a translation, a computer uses statistical analysis to 'learn' the second tongue.

1999, the spread of the Internet has led to an explosion of translated texts in far-flung languages, greatly aiding the team's research. Researchers have also benefited from much faster means of evaluating the outcome of translation experiments: a computerized technique developed by I.B.M. enables researchers to test 10 to 100 new approaches for cracking languages each day.

provides scientists with a fast, objective measurement that they can use to note improvement and saves them from having to review every unsuccessful experiment.

"Before Bleu, it was really a bad state of affairs," said Alex Fraser, a doctoral student at U.S.C. "You look at broken couplets of English for a long time, and eventually you start to accept it more and more."

Despite the progress being made in statistical machine translation, some researchers remain skeptical, preferring to focus their efforts on language-specific translation techniques. Ophir Frieder, a professor of computer science at the Illinois Institute of Technology, is working on a search system exclusive to Arabic text.

"Yes, N-grams work on any language, but as a search technique they work poorly on every language," he said. "It's a basic novice solution."

Dr. Knight acknowledges that statistical machine translation is far from perfect. In its latest efforts, his team has sought to combine the statistical and traditional approaches to achieve imum accuracy and to produce trans that the average computer user can stand. The best machine translation ms today, while capable of yield age's general meaning, are better n for their muddled syntax than their accuracy. By applying the principles of statistical translation to

"Yes, N-grams work on any language, but as a search technique they work poorly on every language," he said. "It's a basic novice solution."

- attributed to an IR researcher in the New York Times on 31 July 2003

The Truth is Out There...

What should we conclude?

1. N-grams are not effective
2. N-grams are effective, but only in Asian Languages
3. Some IR Researchers do not like n-grams
4. Something else?

tok

Monolingual Tokenization

		words	stems	morf	4-stem	4-grams	5-grams
BG	Bulgarian	0.2164		0.2703	0.2822	**0.3105**	0.2820
CS	Czech	0.2270		0.3215	0.2567	**0.3294**	0.3223
DE	German	0.3303	0.3695	0.3994	0.3464	0.4098	**0.4201**
EN	English	0.4060	**0.4373**	0.4018	0.4176	0.3990	0.4152
ES	Spanish	0.4396	**0.4846**	0.4451	0.4485	0.4597	0.4609
FI	Finnish	0.3406	0.4296	0.4018	0.3995	0.4989	**0.5078**
FR	French	0.3638	**0.4019**	0.3680	0.3882	0.3844	0.3930
HU	Hungarian	0.1976		0.2921	0.2836	**0.3746**	0.3624
IT	Italian	0.3749	**0.4178**	0.3474	0.3741	0.3738	0.3997
NL	Dutch	0.3813	0.4003	0.4053	0.3836	0.4219	**0.4243**
PT	Portuguese	0.3162		0.3287	0.3418	0.3358	**0.3524**
RU	Russian	0.2671		0.3307	0.2875	**0.3406**	0.3330
SV	Swedish	0.3387	0.3756	0.3738	0.3638	0.4236	**0.4271**
Average		0.3230		0.3605	0.3518	0.3894	**0.3923**
% change				11.6%	8.9%	20.5%	**21.4%**
Avg-8		0.3719	0.4146	0.3928	0.3902	0.4214	**0.4310**
% change			11.5%	5.6%	4.9%	13.3%	**15.9%**

tok

5 Non-European Languages

		words	stems	morf	4-stem	4-grams	5-grams
AR	Arabic	0.2054		0.2216	0.2373	**0.2731**	0.2356
BN	Bengali	0.2630		0.2933	0.2886	**0.3247**	0.3173
FA	Farsi	0.3406		0.3559	0.3629	**0.3986**	0.3821
HI	Hindi	0.2429		0.2477	0.2484	**0.3305**	0.3271
MR	Marathi	0.2572		0.3310	0.2939	**0.4114**	0.3739
Average-18		0.3072		0.3409	0.3336	**0.3778**	0.3742
% change				11.0%	8.6%	**23.0%**	21.8%

tok

Bilingual: English to X

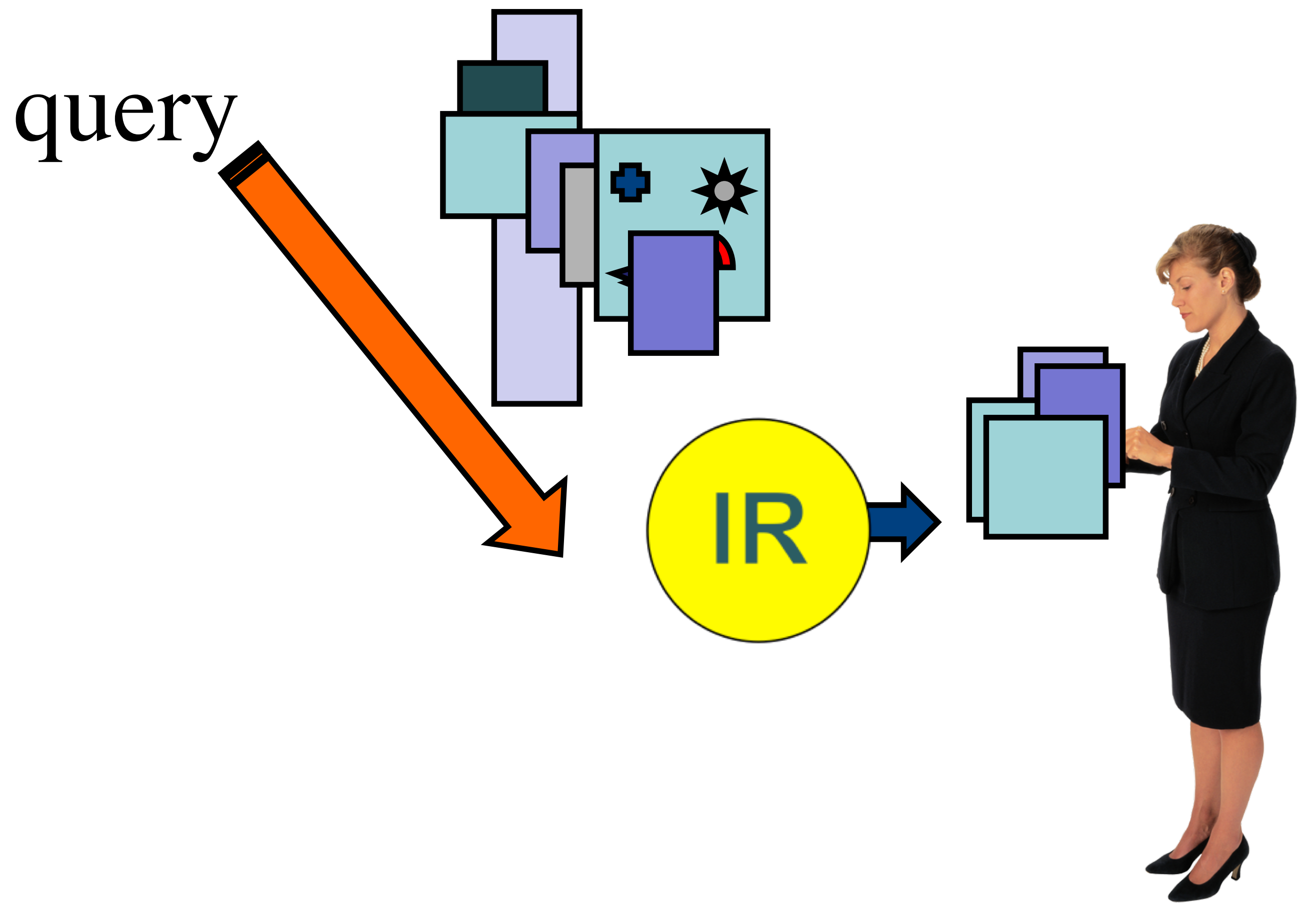
		Acquis Corpus			Europarl Corpus		
		words	stems	5-grams	words	stems	5-grams
BG	Bulgarian	0.0591	x	**0.0898**	x	x	x
CS	Czech	0.1107	x	**0.2479**	x	x	x
DE	German	0.1802	0.2097	**0.2952**	0.2427	0.2646	**0.3519**
ES	Spanish	0.2583	0.3072	**0.3661**	0.3509	0.3721	**0.4294**
FI	Finnish	0.1286	0.1755	**0.3552**	0.2135	0.2488	**0.3744**
FR	French	0.2508	0.2733	**0.3013**	0.2942	0.3233	**0.3523**
HU	Hungarian	0.1087	x	**0.2224**	x	x	x
IT	Italian	0.2365	0.2656	**0.2920**	0.2913	0.3132	**0.3395**
NL	Dutch	0.2474	0.2249	**0.3060**	0.2974	0.2897	**0.3603**
PT	Portuguese	0.2009	x	**0.2544**	0.2365	x	**0.2931**
SV	Swedish	0.2111	0.2270	**0.3016**	0.2447	0.2534	**0.3203**
Average		0.1811		**0.2756**	0.2714		**0.3527**
% change				**63.5%**			**31.9%**
Average-7		0.2161	0.2405	**0.3168**	0.2764	0.2950	**0.3612**
% change			13.1%	**56.0%**		7.1%	**33.0%**

Pause

- Any questions so far?

Ad Hoc Querying

- **Querying / Ranking is the automatic identification of those documents in a large document collection that are relevant to an explicitly-stated information need**



Simplifying Assumptions

- The document collection is static

- A document is relevant or it isn't

query

Steps in Basic Text Retrieval

- **At indexing time**
  - **Characterize each document in collection**
  - **Store characterizations on disk**

- **At query time**
  - **Characterize user’s query**
  - **Compare characterization of query against document characterizations**
  - **Return rank-ordered list of documents**

Common Term Assumption

- **Only documents that share features with the query can be relevant**
  - **We speak generally of *indexing terms;* for now, assume ordinary words are used.**
    - **Many, many variants exists**
      - **Terms can be weighted differently**
      - **Terms need not be simple words (e.g., two word phrases)**
- **Or, if a document and the query share no words in common, the document is not relevant**
  - **And should be given a low score**
  - **(or not even scored)**

query

Bag of Words Representation

- **Original Text**
- When in the Course of human Events, it becomes necessary for one People to dissolve the Political Bands which have connected them with another, and to assume among the Powers of the Earth, the separate and equal Station to which the Laws of Nature and of Nature’s God entitle them, a decent Respect to the Opinions of Mankind requires that they should declare the causes which impel them to the Separation.

- **Set of terms**
- a,among,and,another,assume,Bands,becomes,causes,connected,Course,decent,declare,dissolve,Earth,entitle,equal,Events,for,God,have,human,impel,in,it,Laws,Mankind,Nature,Nature's,necessary,of,one,Opinions,People,Political,Powers,requires,Respect,separate,Separation,should,Station,that,the,them,they,to,When,which,with

- **Bag of terms**
- a(1),among(1),and(3), ...

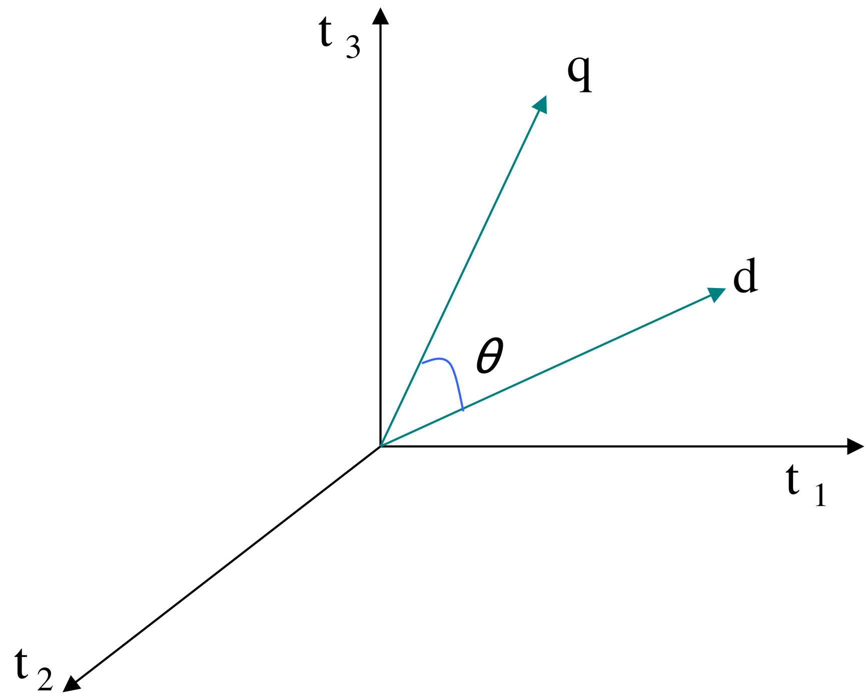
Vector-space Model

- **Binary 'weights' are too limiting, use term frequency information**
  - **Note on nomeclature: <u>term frequency</u> when used in the literature, indicates an ordinal count – how many times does a term occur in a given document or query**
  - **<u>relative term frequency</u> indicates the percentage**
- **Documents and queries are n-dimensional vectors**
  - **Components indicate the number of occurrences of the given term**
- **The framework is algebraic vector arithmetic**
  - **vectors have length, can be added together**
- **Documents are ranked against queries using a vector comparison**
  - **Sample metrics: Cosine (most common), Inner product, Dice**

query

Vector-space: Illustration

- Each axis represents one term
- Each document and each query is represented by a vector that describes the terms contained in the collection
- Various measures can be used to determine document similarity; cosine is a common measure
- 100,000 is a typical number of dimensions



Cosine:

$$Sim(d,q) = \frac{d \bullet q}{|d| \times |q|} = \frac{\sum_{i=1}^{t} w_{i,d} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w^2_{i,d}} \times \sqrt{\sum_{i=1}^{t} w^2_{i,q}}}$$

query

Vector Example

- **Vector Dims: [coffee, tea, milk, sugar, cup]**

- **Doc 1: “Would you like a cup of coffee?”**
  - **[1, 0, 0, 0, 1]**

- **Doc 2: “I take milk and sugar with coffee or tea.”**
  - **[1, 1, 1, 1, 0]**

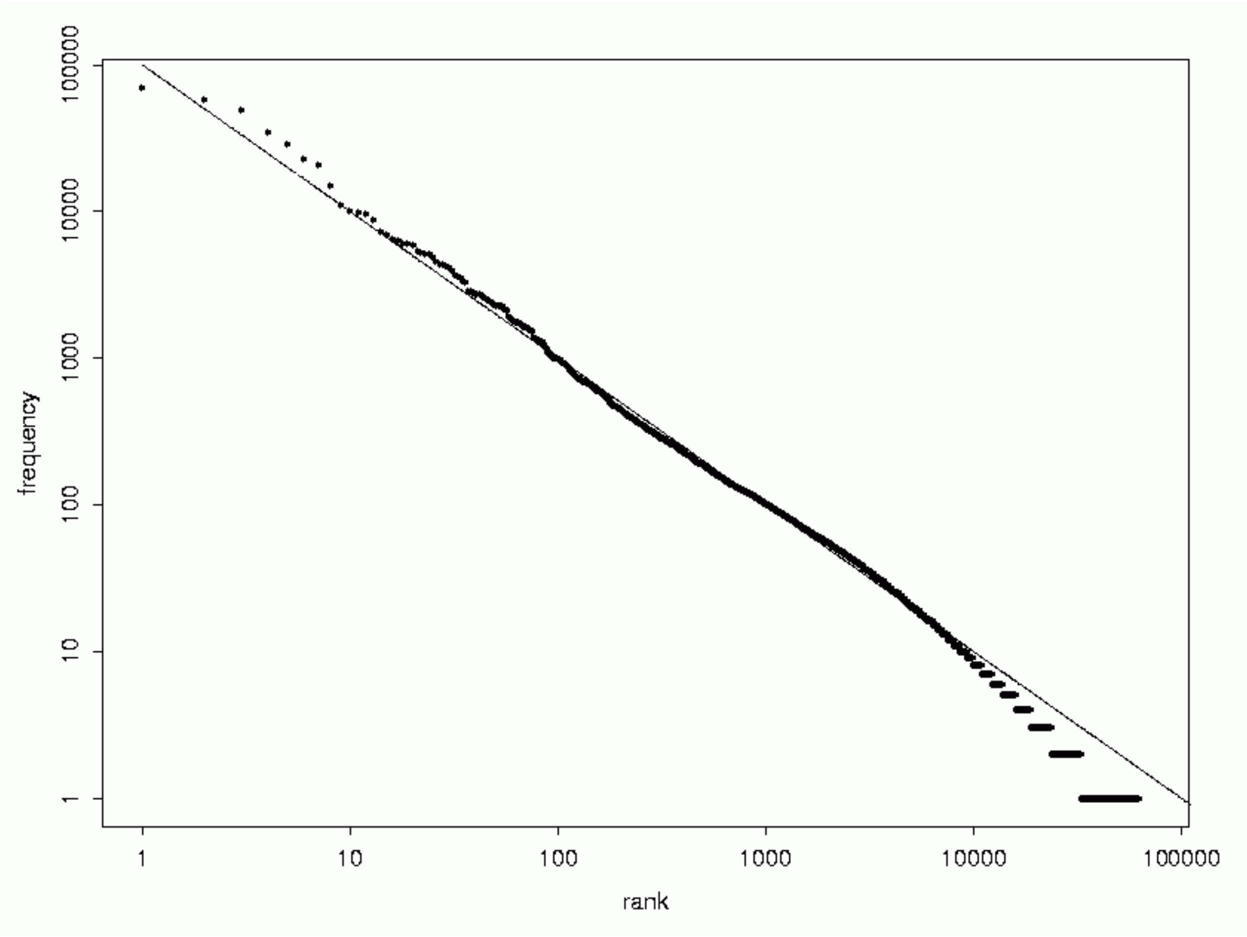
- **Doc 3: “The recipe uses a cup of milk and a cup of sugar”**
  - **[0, 0, 1, 1, 2]**

Assigning Weights to Terms

- Binary Weights
- Raw term frequency (= raw counts)
- 1+log(tf)
  - More occurrences better, but tapers off
- tf / idf or (tf x idf) or (tf – idf)
  - Zipfian distribution
  - Want to weight terms highly if they are
    - frequent in relevant documents … BUT ALSO
    - infrequent in the collection as a whole

Zipf's law

- **The *k*th most frequent term has frequency proportional to *1/k*.**



query

Frequency vs. Resolving Power

The most frequent words are not the most descriptive.

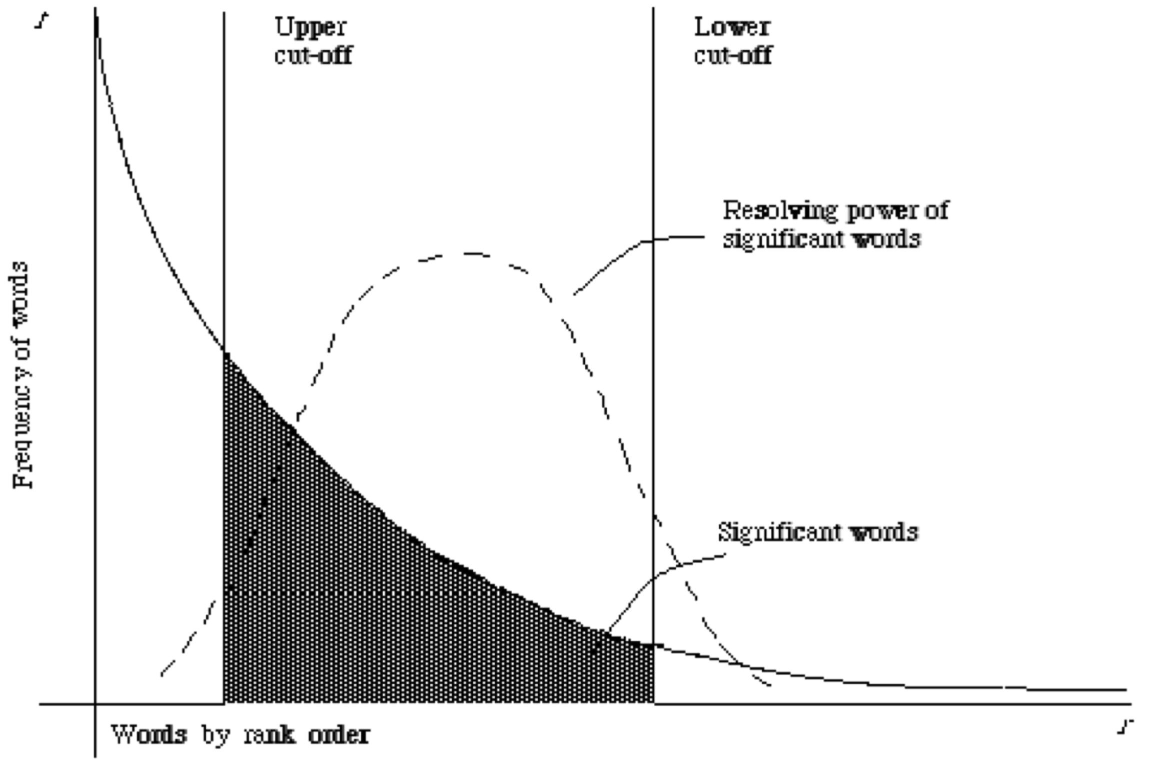


Figure 2.1. A plot of the hyperbolic curve relating f, the frequency of occurrence and r, the rank order (Adaped from Schultz[44] page 120)

Inverse Document Frequency

- **Document frequency is the number of documents a term occurs in**
  - ➢ **Its strictly a property of a term**
- **Medium document frequency terms appear to be the best for IR**
  - ➢ **Rare terms will only affect a few documents**
  - ➢ **Common terms don't discriminate**
- **IDF (inverse relative doc frequency)**
  - – **Log motivated by term distribution**
  - – **Several variants**
    - **Use base 2 logs**

$$IDF(t) = \log_2\left(\frac{N}{df(t)}\right)$$

query

Inverse Document Frequency

- IDF provides high values for rare words and low values for common words
- Thus, each dimension can be weighted differently
  - Terms that are too common are unimportant
  - Decrease the importance of "the" and increase the importance of "Kennedy"
  - Weight each term (dimension) by a multiplicative factor

$$\log\left(\frac{10000}{10000}\right) = 0$$

$$\log\left(\frac{10000}{5000}\right) = 1$$

$$\log\left(\frac{10000}{20}\right) = 8.96$$

$$\log\left(\frac{10000}{1}\right) = 13.2$$

tf x idf, tf/idf, tf-idf

$$w_{ik} = tf_{ik} * \log_2(N / df_i)$$

T_i = term i

tf_{ik} = frequency of term T_i in document D_k

idf_i = inverse document frequency of term T_i in C

N = total number of documents in the collection C

df_i = the number of documents in C that contain T_i

$$idf_i = \log_2\left(\frac{N}{df_i}\right)$$

query

Cosine Example

D1	D2	D3	D4	D5	D6	D7	D8	Q
apple	apple	apple	banana	apple	pineapple	kiwi	strawberry	apple
banana	kiwi	orange	kiwi	grape	pineapple	pineapple	watermelon	orange
grape	kiwi	orange	strawberry	grape		pineapple		
kiwi	orange	orange		orange				
orange								

Words	DF	IDF
apple	4	1
banana	2	2
grape	2	2
kiwi	4	1
orange	4	1
pineapple	2	2
strawberry	2	2
watermelon	1	3

TFxIDF	D1	D3	Query
apple	1	1	1
banana	2	0	0
grape	2	0	0
kiwi	1	0	0
orange	1	3	1
Sum-of-Squares	11	10	2
Length	3.3166	3.1623	1.4142
Dot product	2	4	2
Sim	0.4264	0.8944	1

$$CosineSim(d,q) = \frac{d \bullet q}{|d| \times |q|} = \frac{\sum_{i=1}^{t} w_{i,d} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w^2_{i,d}} \times \sqrt{\sum_{i=1}^{t} w^2_{i,q}}}$$

query

Cosine Example

D1	D2	D3	D4	D5	D6	D7	D8	Q
apple	apple	apple	banana	apple	pineapple	kiwi	strawberry	apple
banana	kiwi	orange	kiwi	grape	pineapple	pineapple	watermelon	orange
grape	kiwi	orange	strawberry	grape		pineapple		
kiwi	orange	orange		orange				
orange								

Words	DF	IDF
apple	4	1
banana	2	2
grape	2	2
kiwi	4	1
orange	4	1
pineapple	2	2
strawberry	2	2
watermelon	1	3

TFxIDF	D1	D3	Query
apple	1	1	1
banana	2	0	0
grape	2	0	0
kiwi	1	0	0
orange	1	3	1
Sum-of-Squares	11	10	2
Length	3.3166	3.1623	1.4142
Dot product	2	4	2
Sim	0.4264	0.8944	1

$$CosineSim(d,q) = \frac{d \bullet q}{|d| \times |q|} = \frac{\sum_{i=1}^{t} w_{i,d} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w^2_{i,d}} \times \sqrt{\sum_{i=1}^{t} w^2_{i,q}}}$$

Summary: Vector-space model

- Advantages
  - Achieves good performance
  - 40+ year standard approach
  - Ranks all documents wrt the query

- Disadvantages
  - Assumes orthogonal vector space
  - Dealing with document weights

- Extensions
  - Approximating cosine (efficiently)
  - Pruning postings lists without hurting rankings (much)

Statistical Language Models

- **Around 1998-2000 three groups developed a model based on statistical language modelling**
  - **Ponte and Croft, *(SIGIR-98)***
  - **Miller, Leek, and Schwartz, *(SIGIR-99)***
  - **Hiemstra and de Vries, (*CTIT Tech. Report*, May 2000)**

- **Appears to outperform vector cosine**

query

Language Modeling Retrieval Model

- A language model is a process that outputs strings in a language

The$_{.10}$ **purple**$_{.20}$ **green**$_{.20}$ **frog**$_{.50}$

- Build a language model for each document in collection

Reference: Ponte & Croft, 'A language modeling approach to information retrieval,' SIGIR '98, 275-281.

- **Calculate probability that each language model would produce query:**

$$P(Q \mid D) = \prod_{q \in Q} P(q \mid D) = \prod_{q \in Q} \frac{D_q}{|D|}$$

- **Rank documents according to these probabilities**
- **Requires smoothing for rare or non-existent terms:**

$$P(Q \mid D) = \prod_{q \in Q} \left[\alpha P(q \mid D) + (1-\alpha) P(q \mid C)\right] = \prod_{q \in Q} \left[\alpha \frac{D_q}{|D|} + (1-\alpha)\frac{C_q}{|C|}\right]$$

JOHNS HOPKINS
UNIVERSITY

query

1-Slide Example

- **Document collection (2 documents)**
  - **d_1: Xerox reports a profit but revenue is down**
  - **d_2: Lucent narrows quarter loss but revenue decreases further**
- **Model: MLE from documents; $\alpha = 1/2$**
- **Query: *revenue down***
  - $P(Q|d_1) = [(1/8 + 2/16)/2] \times [(1/8 + 1/16)/2]$
    $= 1/8 \times 3/32 = 3/256$
  - $P(Q|d_2) = [(1/8 + 2/16)/2] \times [(0 + 1/16)/2]$
    $= 1/8 \times 1/32 = 1/256$
- **Ranking: $d_1 > d_2$**

'Cover Density Ranking'

- Developed by Clarke et al. at U. Waterloo
- Like Coordination Level Ranking
  - But adds relative rankings within each level
- Key ideas
  - Documents that possess most of the query terms, together in close proximity, are likely to be relevant
  - Documents with many such spans are more likely to be relevant
- Requires a different kind of inverted file
  - Word positions must be stored for each word occurrence
- Suited for short queries
  - 4 words or fewer

Erosion[1]

query

It[2] took[3] the[4] sea[5] a[6] thousand[7] years,[8]
A[9] thousand[10] years[11] to[12] trace[13]
The[14] granite[15] features[16] of[17] this[18] cliff,[19]
In[20] crag[21] and[22] scarp[23] and[24] base.[25]

It[26] took[27] the[28] sea[29] an[30] hour[31] one[32] night,[33]
An[34] hour[35] of[36] storm[37] to[38] place[39]
The[40] sculpture[41] of[42] these[43] granite[44] seams,[45]
Upon[46] a[47] woman[48]'s[49] face.[50]

—*E.*[51] *J.*[52] *Pratt*[53] *(1882*[54]*–1964)*[55]

Superscripts indicate term positions. The term set

$$T' = \{\text{“}sea\text{”}, \text{“}thousand\text{”}, \text{“}years\text{”}\}$$

has the cover set

$$\mathscr{C}' = \{(5, 8), (10, 29)\}.$$

The extents (5, 11), (8, 29) and (1, 55) all satisfy T', but are not included in the cover set since they contain shorter extents that satisfy T'. Similarly, the term set

$$T'' = \{\text{“}granite\text{”}, \text{“}sea\text{”}\}$$

has the cover set

$$\mathscr{C}'' = \{(5, 15), (15, 29), (29, 44)\};$$

Cover Set Ranking

- A document is scored by summing the scores for each span in the cover set

$$S(\mathscr{C}) = \sum_{j=1}^{n} I(p_j, q_j),$$

- Each span is scored as:

$$I(p, q) = \begin{cases} \dfrac{\mathscr{K}}{q - p + 1} & \text{if } q - p + 1 > \mathscr{K}, \\ 1 & \text{otherwise.} \end{cases}$$

Relevance Feedback

- **Main Idea:**
  - **Modify existing query based on relevance judgments**
    - **Extract terms from relevant documents and add them to the query**
    - **and/or re-weight the terms already in the query**
  - **Manually**
    - **Users select relevant documents**
    - **Users/system select terms from an automatically-generated list**
  - **Automated (blind/pseudo) rel. feedback**
    - **Assume top *k* docs are relevant (e.g., 5 to 20)**

Relevance Feedback

- **Usually both:**
  - expand query with new terms
  - re-weight terms in query

- **There are many variations**
  - usually positive weights for terms from relevant docs
  - sometimes negative weights for terms from non-relevant docs
  - Remove terms ONLY in non-relevant documents

- **Performance Gains**
  - According to Salton, 10% to 40% improvement

Rocchio's Method

$$Q_1 = \alpha \ Q_0 + \frac{\beta}{n_1}\sum_{i=1}^{n_1} R_i - \frac{\gamma}{n_2}\sum_{i=1}^{n_2} S_i$$

where

Q_0 = the vector for the initial query

R_i = the vector for the relevant document i

S_i = the vector for the non - relevant document i

n_1 = the number of relevant documents chosen

n_2 = the number of non - relevant documents chosen

α, β and γ tune the importance of relevant and nonrelevant terms

(in some studies best to set β higher than γ)

eval

Evaluation

- **How do you know that one approach to retrieval is better than another?**
- **At least two requirements for a score-based method:**
  - **An answer key**
  - **A way to score a result set based on the answer key**

eval

Text REtrieval Conference (TREC)

- Annual bake-off for text retrieval systems
- Sponsored by NIST
- Roughly 2.5 gigabytes of text, newswire
  - 50 “topics” (queries)
  - Return top 1000 documents per topic (~80 groups)
  - Results judged by retired intelligence analysts
    - Documents are relevant or not
- Numerous tracks
  - Cross-Language
  - Spoken Documents
  - Question Answering
- http://trec.nist.gov/

Test Collections

- Collection of Documents
  - Must be releasable (copyright issues)
- Set of Topics
  - Need to be representative of real world
- Judgments
  - Exhaustive is best, but expensive
  - Pooled is still expensive, but practical
    - Useful if no systemic biases are introduced

eval

Sample TREC Topic

SGML Markup

Short Phrase

Sentence

<top>
<num> Number: 285
<title> Topic: World submarine forces
<desc> Description:
Determine the number of submarines, both nuclear-powered and conventional, presently in the inventories of all the countries in the world.
<narr> Narrative:
We are looking for a count of operable submarines in any country that currently has a navy with submarines. To be relevant a document should give a specific number of submarines, but not necessarily its entire fleet of submarines (although, that is our ultimate goal). A report of a French submarine suffering a mishap in the North sea would not be relevant. However, a report of a new submarine being built in Shanghai that contains other valuable information, such as “this is the third reported unit constructed at this base” would be relevant. Any information that would be considered useful as an intelligence tool in determining a country’s submarine order of battle would be relevant.
</top>

Paragraph

Precision and Recall

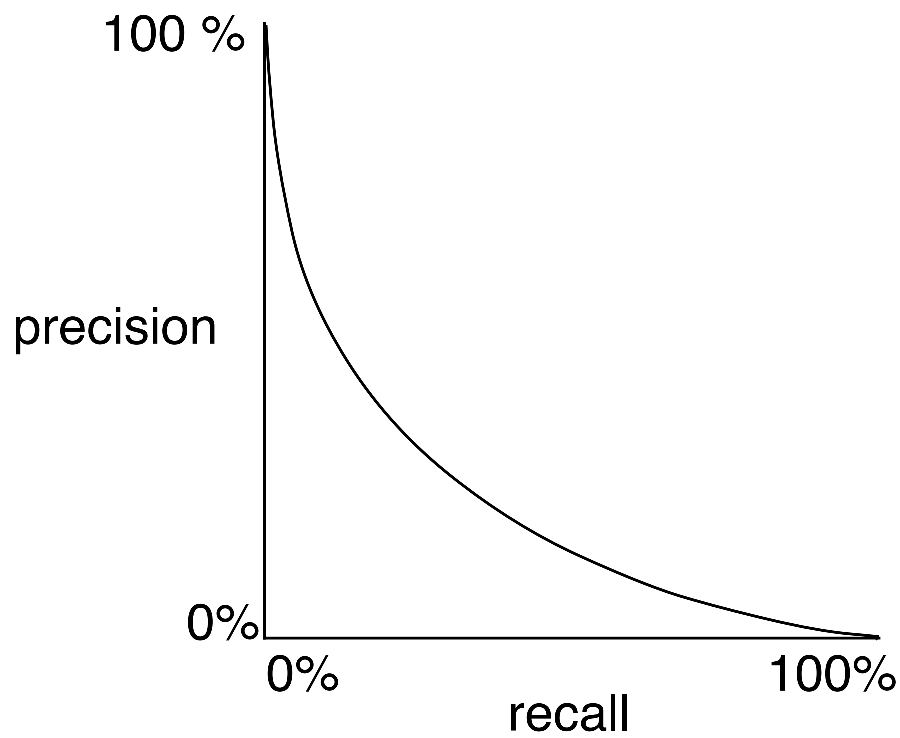
	relevant	not relevant
retrieved	A	B
not retrieved	C	D

“Type one errors” “Errors of commission” “False positives”

“Type two errors” “Errors of omission” “False negatives”

$$\text{precision} = \frac{A}{A + B}$$

$$\text{recall} = \frac{A}{A + C}$$



average precision = area under curve

Problems with Precision/Recall

- Can't know true recall value
  - except in small collections
- Precision/Recall measure different aspects of search quality
  - A combined measure sometimes is more appropriate
- Focused somewhat on set evaluation vs. ranked lists

eval

How Test Runs are Evaluated

$R_q=\{d_3,d_5,d_9,d_{25},d_{39},d_{44},d_{56},d_{71},d_{89},d_{123}\}$: 10 Relevant

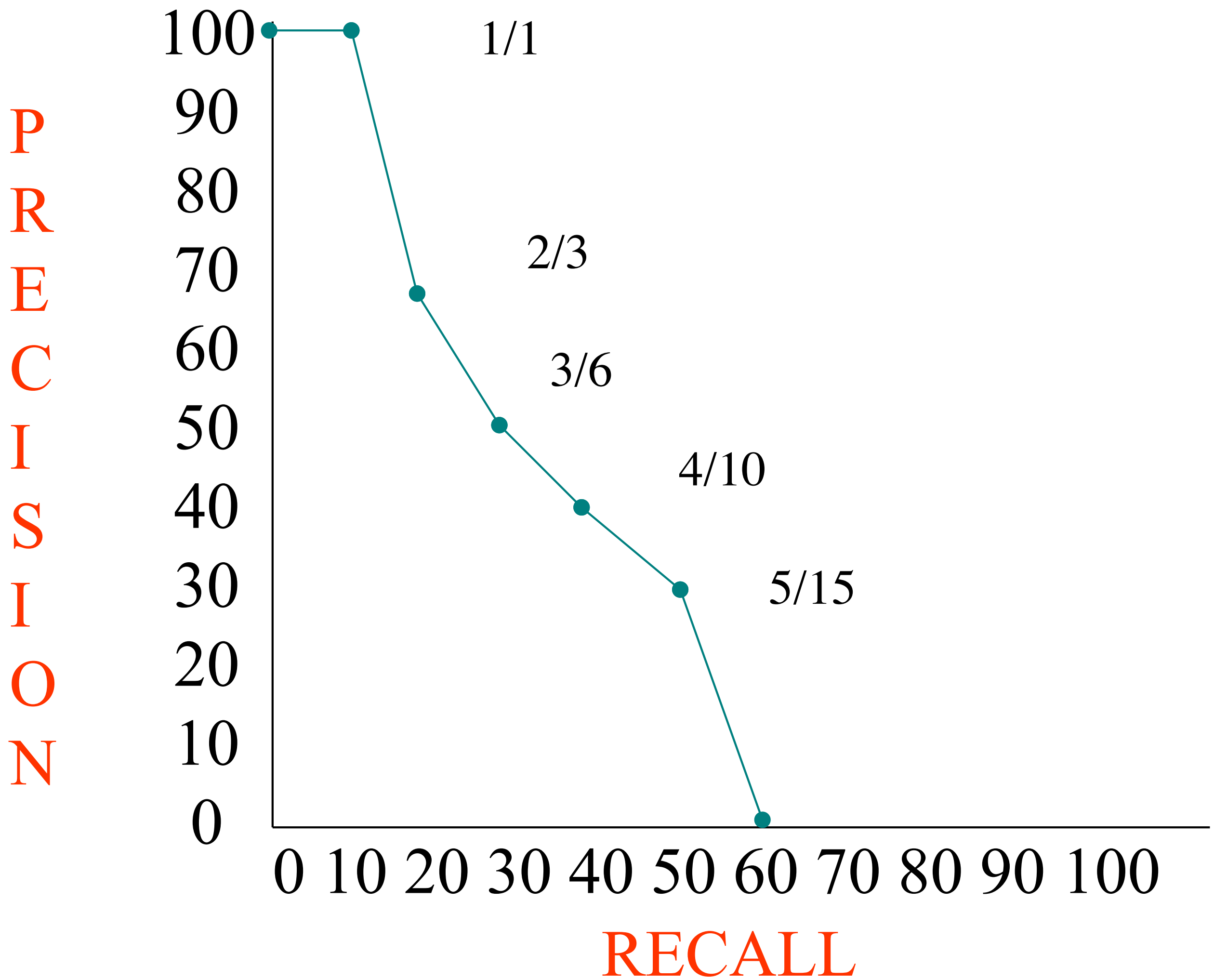
1. d_{123}*
2. d_{84}
3. d_{56}*
4. d_6
5. d_8
6. d_9*
7. d_{511}
8. d_{129}
9. d_{187}
10. d_{25}*
11. d_{38}
12. d_{48}
13. d_{250}
14. d_{113}
15. d_3*

- **First ranked doc is relevant, which is 10% of the total relevant. Therefore Precision at the 10% Recall level is 100%**
- **Next Relevant gives us 66% Precision at 20% recall level**
- **Etc….**

Graph for a Single Query

10 relevant: $R_q=\{d_3,d_5,d_9,d_{25},d_{39},d_{44},d_{56},d_{71},d_{89},d_{123}\}$

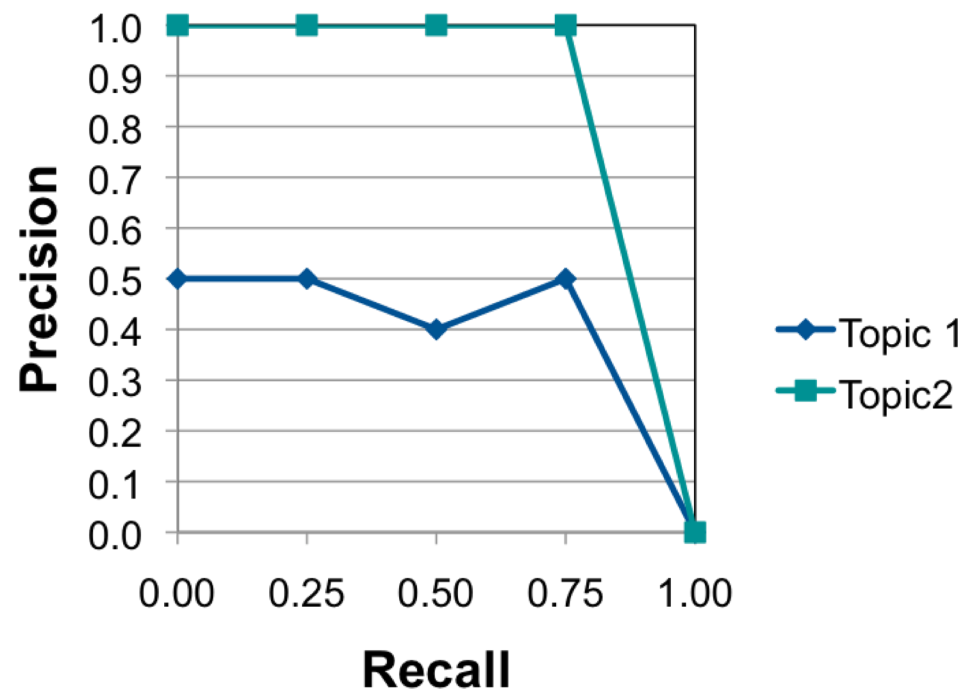
Ranked List: d_{123}, d_{84}, d_{56}, d_6, d_8, d_9, d_{511}, d_{129}, d_{187}, d_{25}, d_{38m}, d_{48}, d_{250}, d_{113}, d_3



Evaluation: Mean Average Precision

Documents are either Relevant or Not Relevant
Assume 4 Relevant Docs/Topic

Topic 1	Topic 2
No	Yes
Yes	Yes
No	Yes
No	No
Yes	No
Yes	No

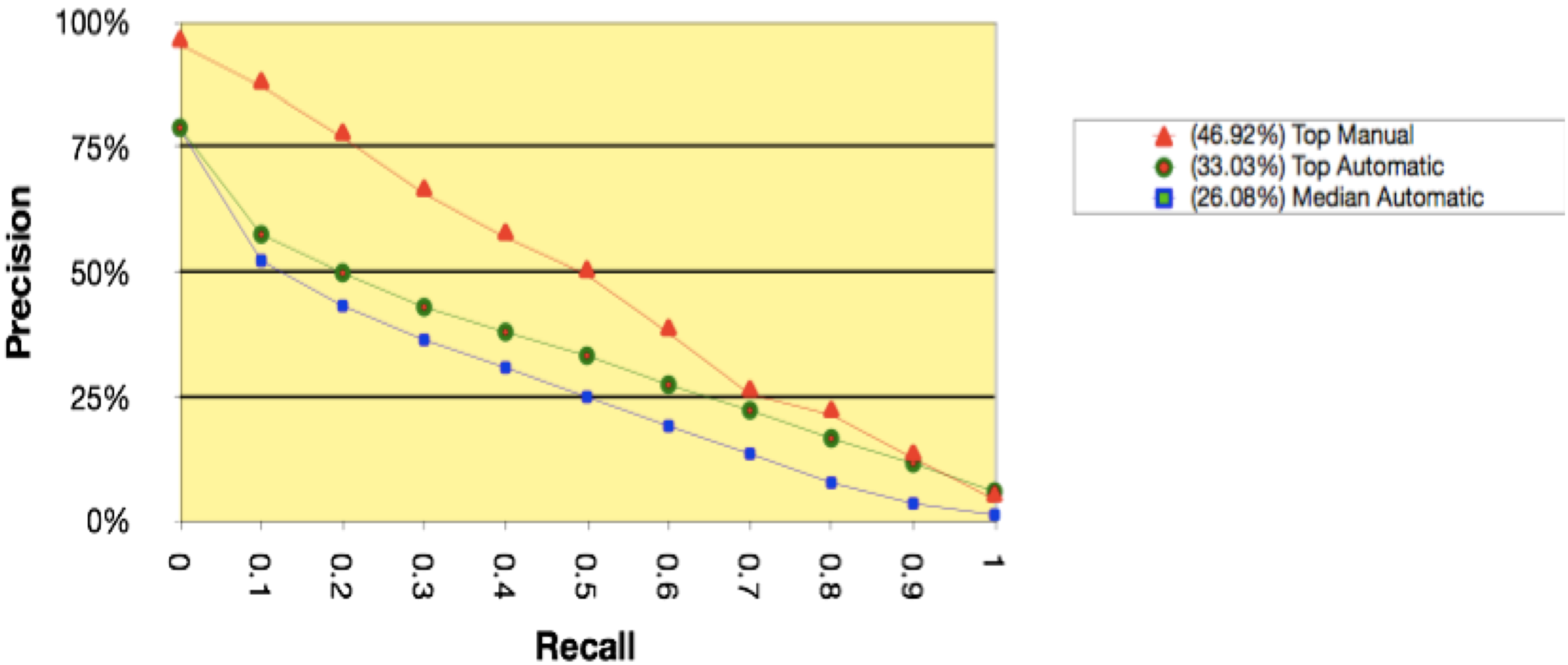


AP(T1) = (0.5 + 0.4 + 0.5) / 4 = 0.35
AP(T2) = (1 + 1 + 1) / 4 = 0.75

MAP = mean of AP over all topics
= (0.35 + 0.75) / 2 = 0.55

Average Precision approximates the area under the curve

TREC-8 Ad Hoc Retrieval Performance



Interpolated Recall/Precision Curves (multiple topics)

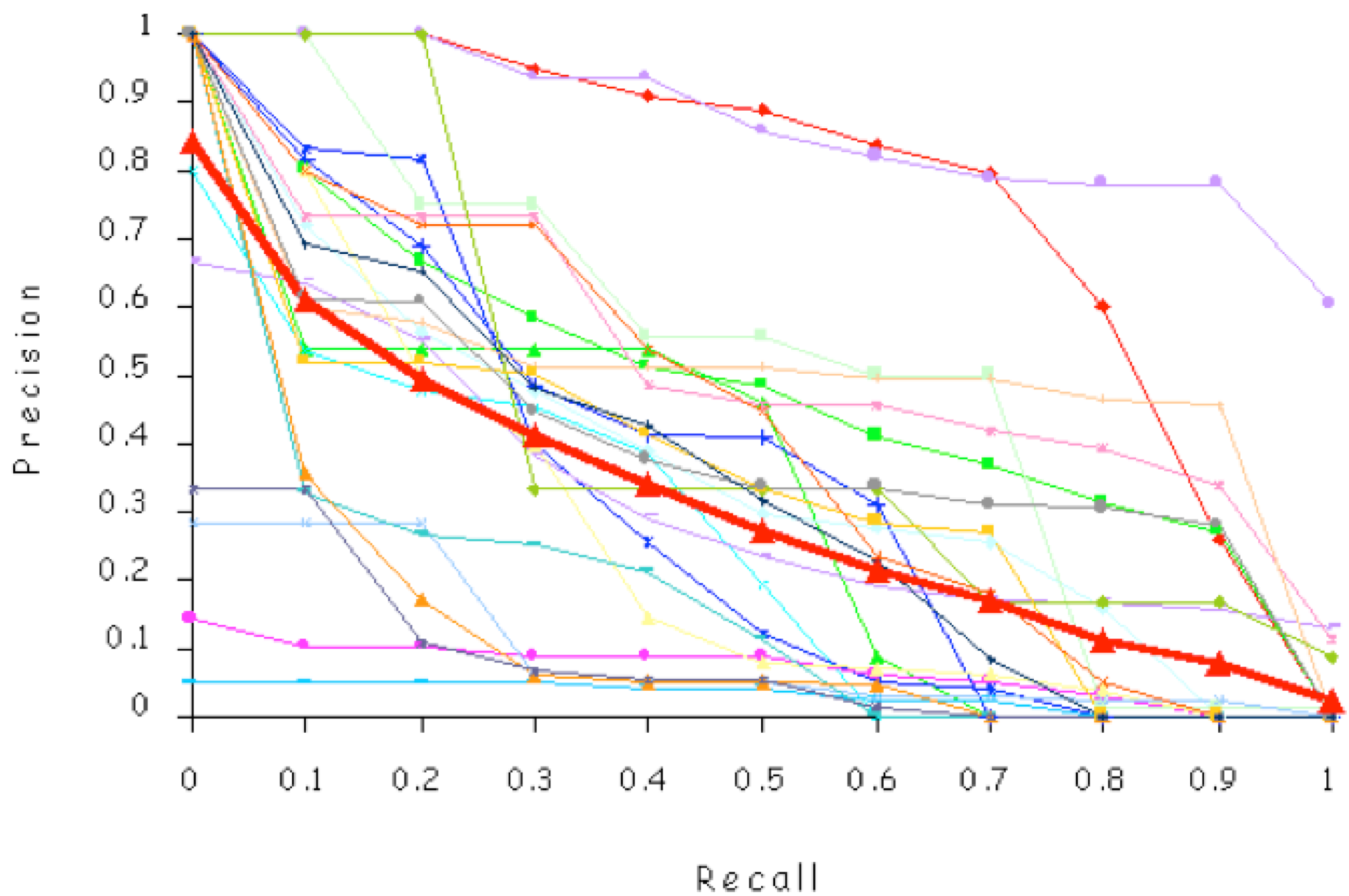


Figure: Dr. Ellen Voorhees (NIST)

Challenges of the Web

- **Distributed data**
  - Data exists on millions of decentralized servers
- **Volatile**
  - Perhaps 40% of Web changes monthly
- **Scale**
  - Growth is exponential
- **Lack of Structure**
  - Duplication (30%), lack of adherence to standards, naming
- **Quality**
  - No editorial review: false, poorly written, undesirable
- **Heterogeneous**
  - Many languages, many data formats

A singular advantage

- **The Web presents many challenges, but are there any benefits for IR?**
- **There is a particular kind of value-added annotation**

HTML

```
<A HREF="http://probablyirrelevant.org/">Probably Irrelevant</A>,
<A HREF="http://nlpers.blogspot.com/">NLPers</A>
<A HREF="http://www.thenoisychannel.com/">The Noisy Channel</A>,
<A HREF="http://www.searchenginewatch.com/">Search Engine Watch</A>,
<A HREF="http://www.dlib.org/">D-Lib Magazine</A>
Peter Norvig's <A HREF="http://norvig.com/spell-correct.html">tutorial on spelling correction</A>
John Sowa's <A HREF="http://www.jfsowa.com/logic/math.htm">Discrete Mathematics Primer</A>
```

Ranking Ideas for the Web

- **Exploit links**
  - **Possibly, words near a hyperlink are more important**
- **Currency**
  - **Assumes most recent data is best**
- **Popularity**
  - **Use estimates of what a large number of people think about a page or site**
  - **Estimate based on easy to obtain data**
    - **number of inbound links to ‘that’ page**
    - **called ‘backlink frequency’**
- **Authority**
  - **Harder to estimate than popularity**

Google’s Measure of Authority

- You may have heard that Google has a unique score for web pages that ranks their quality.
- PageRank is much in the spirit of Kleinberg’s HITS algorithm, though it is computed a bit differently
- In the mid/late 1990s search engines struggled to become large, but many indexes were filled with low quality pages, which sometimes yielded poor results
- Google’s Brin and Page became billionaires for commercializing PageRank
    - Kleinberg received the ~ $15,000 Nevanlinna prize

PageRank Fundamentals

- PageRank simulates a user browsing the Web.

- The user either jumps to a random page with probability q or follows a random hyperlink on the current page with probability $1 - q$

- This process can be modeled as a Markov chain, so that the stationary probability of ending on each page can be computed

PageRank

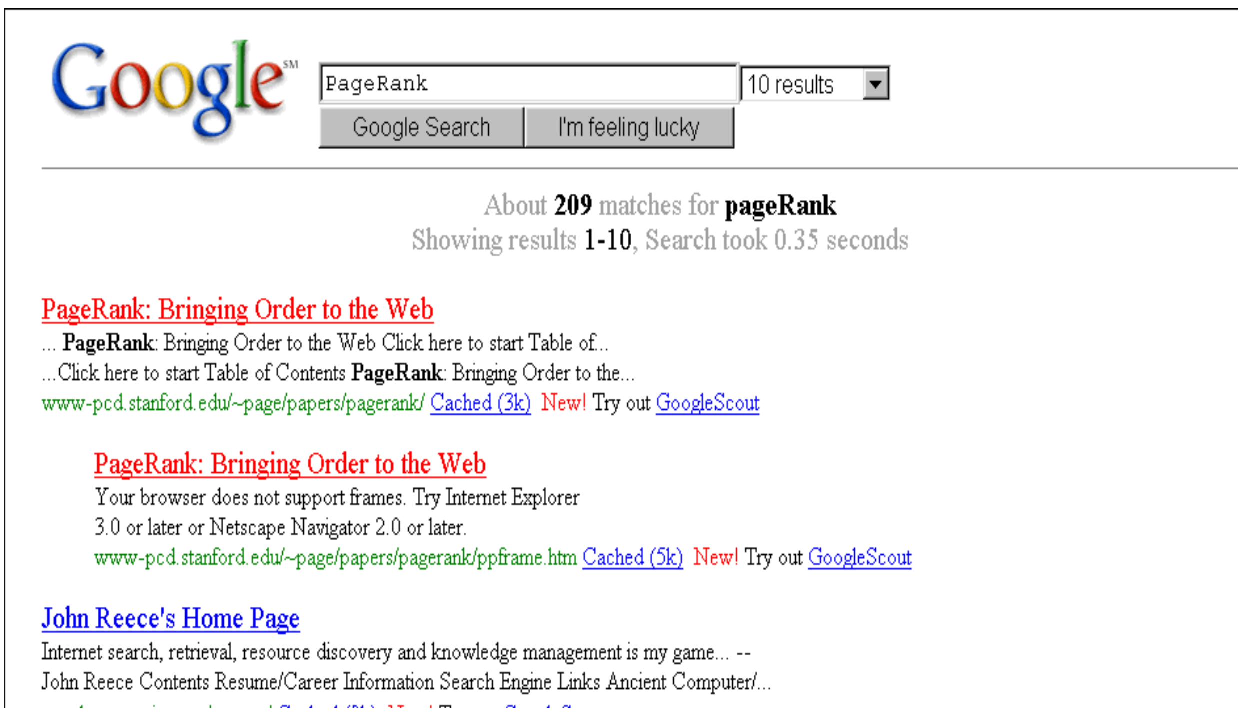
Let *C(a)* be the number of outgoing links of a page *a*

And suppose that a page *a* is pointed to by pages p_1 to p_n

N is the number of pages in the entire Web graph

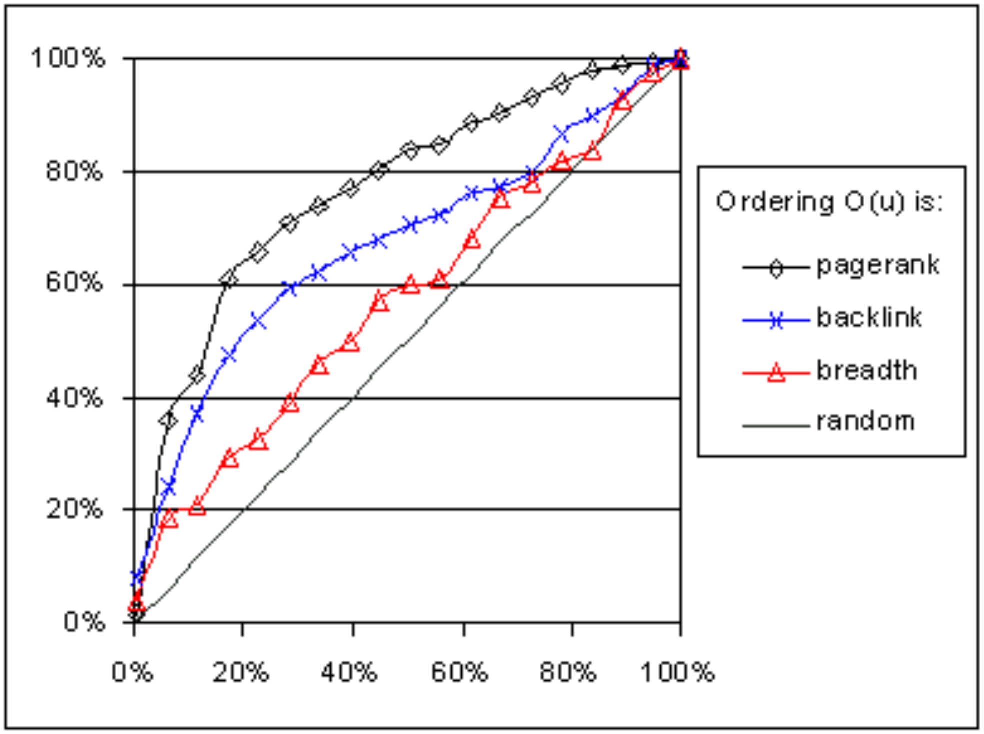
$$PR(a) = \frac{q}{N} + (1-q)\sum_{i=1}^{n}\frac{PR(p_i)}{C(p_i)}$$

typical q = 0.15



PageRank's advantages

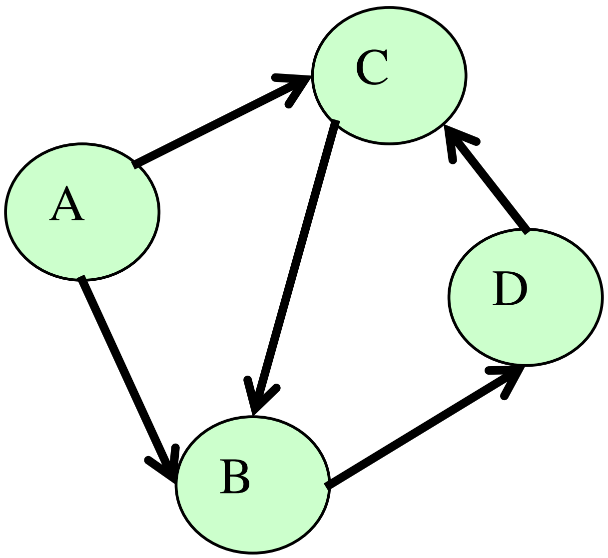
- **Google can rank unseen pages!**
  - ➢ **Corollary, Google can rank non-text content**
- **Estimates of page quality (for unseen pages) can be used for <u>crawl ordering</u>**



"Efficient crawling through URL ordering", Cho, Garcia-Molina, and Page, WWW-7.

web

PageRank Example



	A	B	C	D	Sum
t=0	0.25	0.25	0.25	0.25	1
t=1	0.05	0.35			

$$PR(a) = \frac{q}{N} + (1-q)\sum_{i=1}^{n}\frac{PR(p_i)}{C(p_i)}$$

Using teleport prob q of 0.20. Set PR(x) = ¼ since we have four pages.

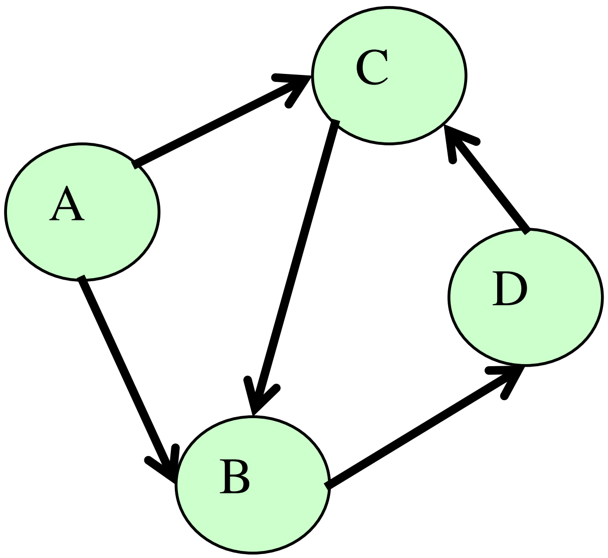
$PR(A, t_i) = 0.05 + 0$

$PR(B, t_i) = 0.05 + 0.80 * (PR(A, t_{i-1}) / 2 + PR(C, t_{i-1}) / 1)$

$PR(B, t_i) = 0.05 + 0.80 * (0.25 / 2 + 0.25 / 1)$

$PR(B, t_i) = 0.05 + 0.80 * (0.375) = \underline{0.35}$

PageRank Example



	A	B	C	D	Sum
t=0	0.25	0.25	0.25	0.25	1
t=1	0.05	0.35	0.35	0.25	1

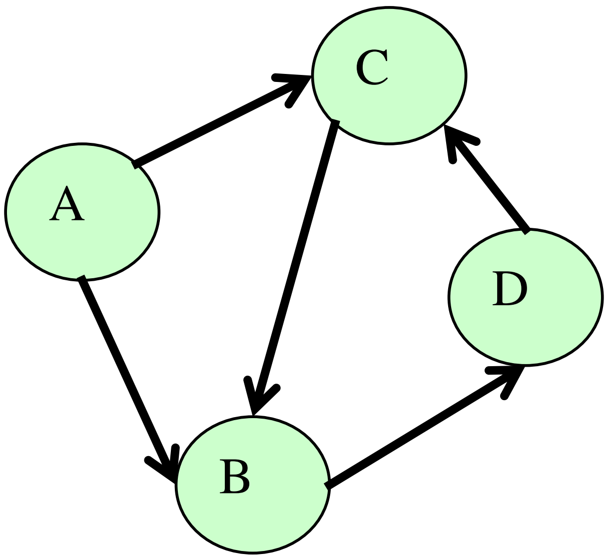
$$PR(a) = \frac{q}{N} + (1-q)\sum_{i=1}^{n} \frac{PR(p_i)}{C(p_i)}$$

Using teleport prob q of 0.20. Set PR(x) = ¼ since we have four pages.

PR(C, ti) = 0.05 + 0.80 * (PR(A, ti-1) / 2 + PR(D, ti-1) / 1) = 0.35

PR(D, ti) = 0.05 + 0.80 * (PR(B, ti-1) / 1) = 0.25

PageRank Example



	A	B	C	D	Sum
t=0	0.25	0.25	0.25	0.25	1
t=1	0.05	0.35	0.35	0.25	1
t=2	0.05	0.35	0.27	0.33	1
t=3	0.05	0.286	0.334	0.33	1
t=4	0.05	0.337	0.334	0.279	1
t=5	0.05	0.337	0.293	0.320	1
t=10	0.05	0.327	0.322	0.301	1

$$PR(a) = \frac{q}{N} + (1-q)\sum_{i=1}^{n}\frac{PR(p_i)}{C(p_i)}$$

What do user's want to find?

- https://trends.google.com/trends/topcharts

- **3/2003: Lycos top 50 (http://50.lycos.com/)**
  - KaZaA
  - IRS
  - Tattoos
  - 50 Cent
  - Joe Millionaire
  - Dragonball
  - Rhode Island Nightclub Fire
  - NASCAR
  - Taxes
  - t.A.T.u.

Possibly an edited list:

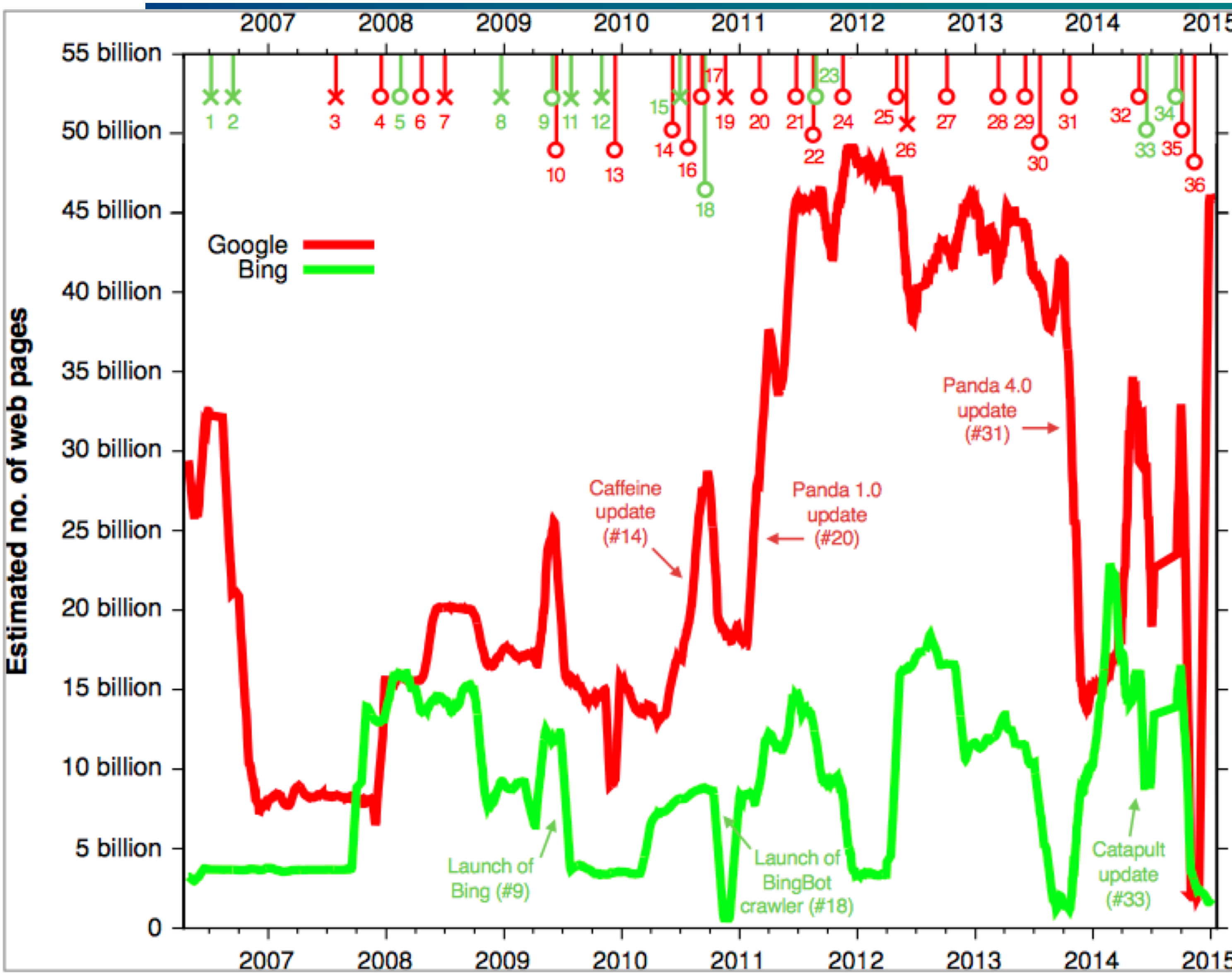
sex, guns, & weather are typical

web

Popular terms from AOL query log

web

Index Size



van den Bosch et al., 'A Longitudinal Analysis of Search Engine Index Size, Proc. ISSI 2015.

Taxonomy of Search Requests

- Andrei Broder (AV) characterized user's requests into three main categories:
  - Informational: Find information about X
  - Transactional: E.g., buying airline tickets
  - Navigational:
    - I know I saw a page on X last week but I didn't bookmark it
    - Or, where can I download Adobe Acrobat Reader from?

web

Self Promotion

Man lands job with $6 Google campaign

By **Lauren Indvik**

STORY HIGHLIGHTS

- Copywriter Alec Brownstein landed a job through a $6 Google marketing campaign
- Brownstein bought ads on the names of directors he wanted to work for, knowing they'd pop up when the directors "Google" themselves
- Since no one else was bidding, some of the ads cost him 15 cents
- In a couple of months, he got calls from all but one of the directors and job offers from two

RELATED TOPICS

- Google Inc.
- Online Advertising
- Jobs and Labor

(Mashable) -- By now, landing a job via social media is nothing new; we've perused the how-to guides and heard dozens of great success stories. There are, however, still plenty of creative opportunities for securing a job with a bit of clever online marketing.

Meet Alec Brownstein, senior copywriter at creative advertising shop Young & Rubicam (Y&R) New York.

Last summer, Alec was just another tired, 28-year-old copywriter at a large international ad agency who wanted nothing more than to work at "a really creative shop for really creative [creative directors]."

While Googling his favorite creative directors last summer, Brownstein noticed that there were no sponsored links attached to their names. Since Brownstein Googles himself "embarassingly frequently," he assumed that the creative directors did so as well, and thus he decided to purchase their names on Google AdWords.

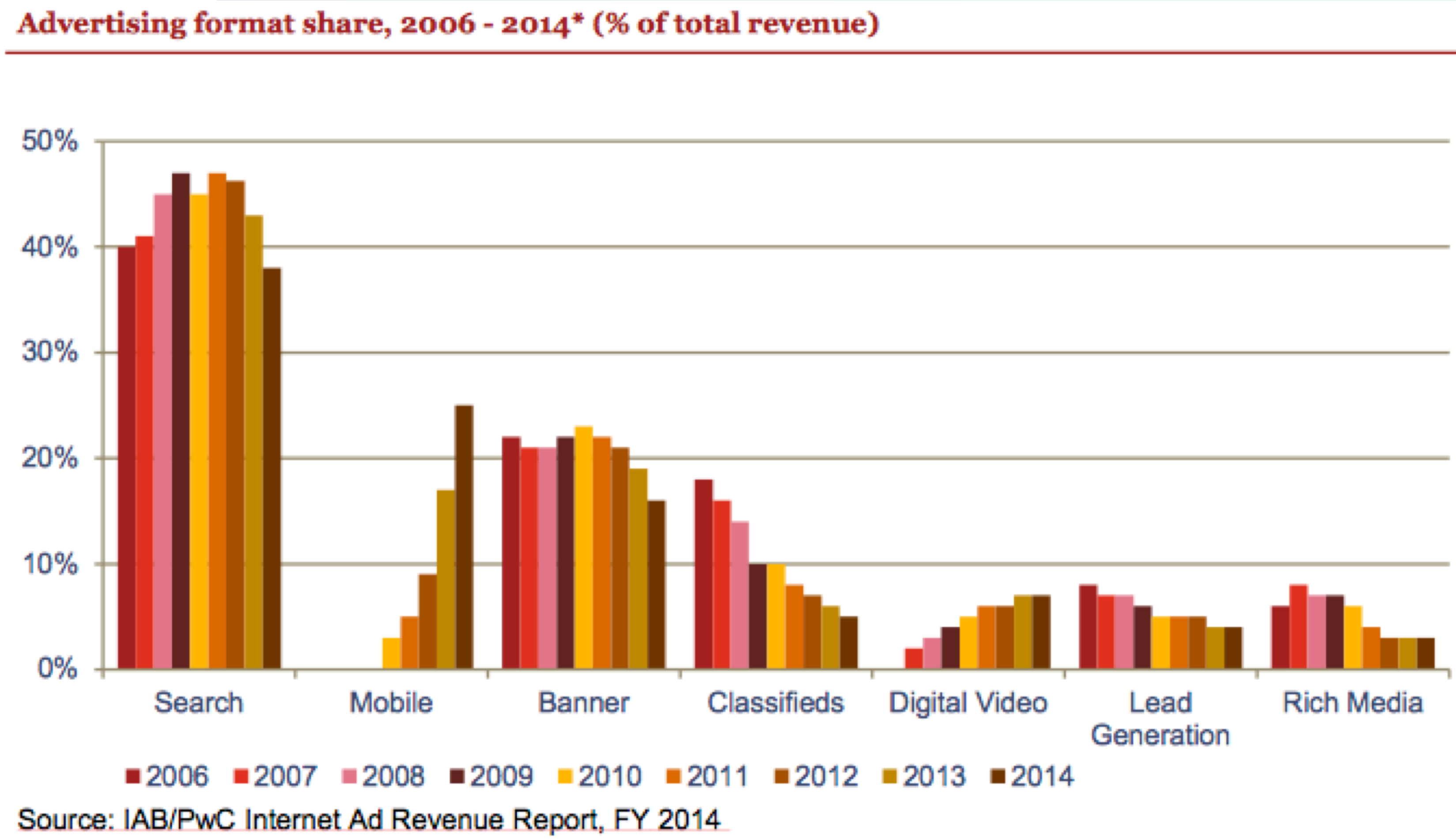
"Everybody Googles themselves," Brownstein explained. "Even if they don't admit it. I wanted to invade that secret, egotistical moment when [the creative directors I admired] were most vulnerable."

Since Brownstein was the only person bidding on the names of the five creative directors he most admired, he was able to get the top search spots for a mere 15 cents per click. Whenever someone ran a search for one of the creative directors' names, the following message appeared at the top of the page: "Hey, [creative director's name]: Goooogling [sic] yourself is a lot of fun. Hiring me is fun, too" with a link to Brownstein's website, alecbrownstein.com.

Over the next couple of months, Brownstein received calls from all but one of the creative directors whose names he had purchased. And finally, at the end of the year, he received a job offer from two: Scott Virtrone and Ian Reichenthal of Y&R New York.

The whole campaign cost him $6.

Online Advertising Revenues



- **2014: ~ $50 billion total**
  - **Search: ~ $20 billion/year (Google: 67%, Bing: 19%, Yahoo 10%)**

Resources

- **Book: Introduction to Information Retrieval (2008), Manning, Raghavan, and Schütze**
  - **http://nlp.stanford.edu/IR-book/information-retrieval-book.html**

- **Survey article by Zobel & Moffat: Inverted files for text search engines. ACM Computing Surveys, 38(2), 2006.**

Links to these and others at:

http://pmcnamee.net/ir.html

Other books:

- **IR: Implementing and Evaluating Search Engines (2010)**
  - **Buettcher, Clarke, and Cormack**
- **Managing Gigabytes, 2nd edition (1999)**
  - **Witten, Moffat, & Bell**

Research Software Systems

- **Lucene / ElasticSearch**
  - **Apache project (Java)**
- **Wumpus**
  - **U. Waterloo (Open source, C++)**
- **Terrier**
  - **Glasgow (Open source, Java)**
- **Lemur / Indri**
  - **Carnegie Mellon / UMass (C++ & Java bindings)**
- **SMART**
  - **Developed at Cornell University (C)**
- **mg**
  - **From the authors of *Managing Gigabytes* (C)**
- **INQUERY**
  - **Univ. Massachusetts (Amherst). Available???**