The Web II

IN4325 – Information Retrieval

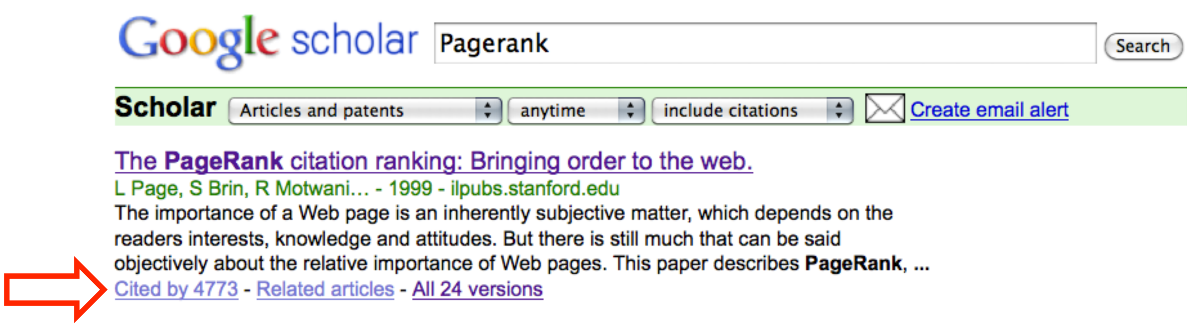
Assignment 5

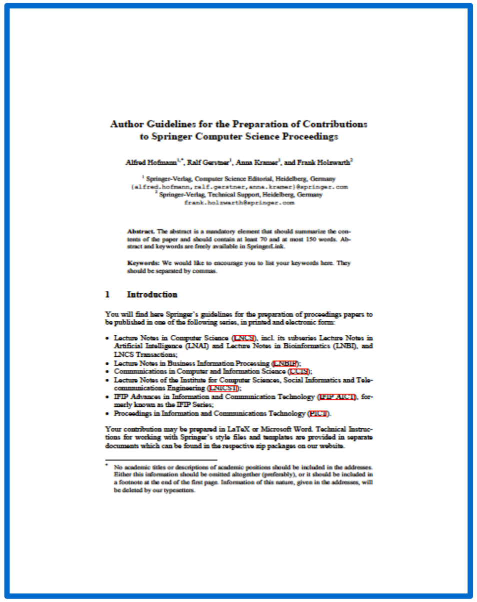
If you do not manage to find a topic, **email me** and I will assign you one!

- **Individual** work, 50% of your final grade
- Task: write a survey paper about an IR research topic
  - If you have an idea for a report that is not a survey (e.g. you want to implement an algorithm & evaluate it), check with me first!
- Deadline for the assignment: April 29, 2012
- You have a chance to hand in intermediate results
  - Topic description: by March 28, 2012 (up to half a page)
  - Outline: by April 4, 2012 (up to a page)
  - These two deadlines are voluntary & do not count towards your grade!

Assignment 5

- Use the LNCS proceedings template
  - Available for LaTeX and Word
  - http://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0
- Report length: 7-8 pages (including references)
- Minimum number of references: 6
  - Google Scholar is your friend





Author Guidelines for the Preparation of Contributions to Springer Computer Science Proceedings

Alfred Hofmann[1,*], Ralf Gerstner[1], Anna Kramer[1], and Frank Holzwarth[2]

[1] Springer-Verlag, Computer Science Editorial, Heidelberg, Germany
{alfred.hofmann,ralf.gerstner,anna.kramer}@springer.com
[2] Springer-Verlag, Technical Support, Heidelberg, Germany
frank.holzwarth@springer.com

Abstract. The abstract is a mandatory element that should summarize the contents of the paper and should contain at least 70 and at most 150 words. Abstract and keywords are freely available in SpringerLink.

Keywords: We would like to encourage you to list your keywords here. They should be separated by commas.

1 Introduction

You will find here Springer's guidelines for the preparation of proceedings papers to be published in one of the following series, in printed and electronic form:

- Lecture Notes in Computer Science (LNCS), incl. its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), and LNCS Transactions;
- Lecture Notes in Business Information Processing (LNBIP);
- Communications in Computer and Information Science (CCIS);
- Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (LNICST);
- IFIP Advances in Information and Communication Technology (IFIP AICT), formerly known as the IFIP Series;
- Proceedings in Information and Communications Technology (PICT).

Your contribution may be prepared in LaTeX or Microsoft Word. Technical Instructions for working with Springer's style files and templates are provided in separate documents which can be found in the respective zip packages on our website.

* No academic titles or descriptions of academic positions should be included in the addresses. Either this information should be omitted altogether (preferably), or it should be included in a footnote at the end of the first page. Information of this nature, given in the addresses, will be deleted by our typesetters.

Assignment 5

Examples often help!

- Important aspects
  - Show that you are capable of understanding a recent IR topic
  - Show that you are capable of formulating your own thoughts based on other people's work

- Suggested paper outline
  - Abstract (summary of the paper)
  - Introduction (explain the topic, the motivation, outline of the paper)
  - A section on the challenges
  - One or more sections that discuss an aspect/aspects of your topic
    - Questions to ask yourself: do the motivation/examples/data set/ evaluation/conclusions make sense?
  - Conclusions and future work

Assignment 5

- **Citations**: clearly mark sentences taken from other people's work
  - Use quotes "…."
  - Use sparringly
- Clearly distinguish your own thoughts and conclusions from those derived by others (**references**)
- Important IR conferences (have a look at their workshops too!)
  - SIGIR
  - CIKM
  - WWW
  - WSDM
  - ECIR

TREC

http://trec.nist.gov/ http://trecvid.nist.gov/

Text REtrieval Conference (1992-*)

- Conducted by the US National Institute of Standards and Technology, co-sponsored by DARPA
- Several “tracks” per year (a good way to learn about current work)

Ad-hoc retrieval task (1992)
Routing task (1992)
Interactive track (1995)
Multilingual track (1995)
Database merging track (1995)
Confusion track (1995)
Cross-Language track (1997)
Spoken document track (1997)
Question Answering track (1999)
Web track (1999)

Video track (2001)
Novelty track (2002)
Genomics track (2003)
Terabyte track (2004)
Enterprise track (2005)
Spam track (2005)
Blog track (2006)
Legal track (2006)
Million query track (2007)
Chemical IR track (2009)
Entity track (2009)
Microblog track (2011)

Assignment 5

- If you are looking for areas in IR not covered in this course
  - Quantum information retrieval
  - Cognitive perspectives of information retrieval
  - Information retrieval for specific user groups
    - E.g. children
  - Interactive information retrieval
  - Mobile search
  - Video & audio search
  - Search personalization
  - User interfaces and their influences on search
  - Novelty & diversity in search
  - Crowdsourcing
  - ….

Today

- Learning to rank
- Query logs

Learning to rank (LTR)

- Ranking: sort objects based on ‘some’ factor
  - So far in the lectures: sort documents based on their retrieval status value score (BM25, LM, VSM) with respect to a query

- Supervised approach to ranking
  - Training data: queries and the ground truth ranking of results
  - Goal: learn a ranking function that returns the best possible ranking
  - Instead of making assumptions (e.g. a PageRank document prior aids ad hoc retrieval), the data speaks for itself

- Highly active area of research in the IR & ML communities!!

LTR overview

- LTR approaches can be categorized as follows:
  - Pointwise: Regression/classification on single objects
  - Pairwise: Classification on object pairs
  - Listwise: Tackles the ranking problem directly

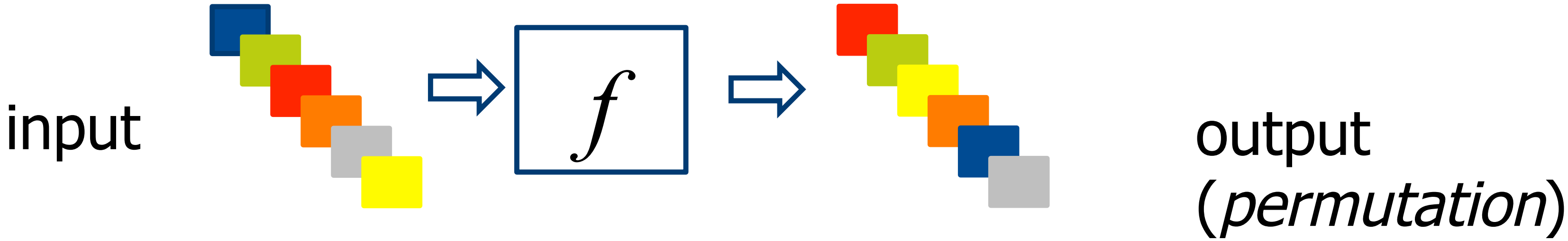
use standard ML techniques

- Standard classification/regression techniques were not developed for ranking, their loss functions do not directly link to the criteria used in the evaluation of ranking
  - Problematic: minimizing the loss function does not necessarily enhance the ranking performance
  - Thus: development of query-level loss functions

Listwise LTR: CosineRank

Qin et al., 2008 [1]

- Instances are ranked lists of documents
- Ranking function is trained through the minimization of a listwise loss function
  - Predicted list vs. ground truth list
- Advantage: natural expression of the IR ranking problem
- Several methods exist (here we only consider CosineRank)

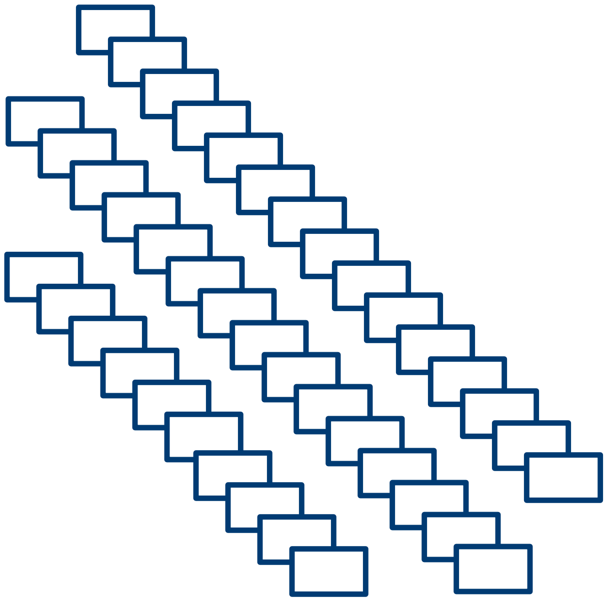


Listwise LTR: CosineRank

Qin et al., 2008 [1]

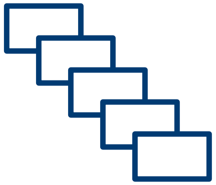
- Document-pair level loss vs. query-level loss

q_1



$$\frac{40 \times 39}{2} = 780 \ \ pairs$$

q_2



$$\frac{5 \times 4}{2} = 10 \ \ pairs$$

Listwise LTR: CosineRank

Qin et al., 2008 [1]

Document-pair level and query-level are the same if all queries are trained on the same number of document pairs *(not realistic)*

- Document-pair level loss vs. query-level loss

		Case 1	Case 2
Document pairs of q_1	Correctly ranked	770	780
	Wrongly ranked	10	0
	Accuracy	98.72%	100%
Document pairs of q_2	Correctly ranked	10	0
	Wrongly ranked	0	10
	Accuracy	100%	0%
Overall accuracy	Document-pair level	98.73%	98.73%
	Query-level	99.36%	50%

Listwise LTR: CosineRank

Qin et al., 2008 [1]

- Loss function terminology

$n(q)$ | $n(q)!$ | $q \in Q$ | $f \in \mathcal{F}$ | $\tau_g(q)$ | $\tau_f(q)$

#documents to be ranked for q

#possible ranking lists in total

space of all queries

space of all ranking functions

ground truth ranking list of q

ranking list generated by a ranking function f

(notation follows the paper)

Listwise LTR: CosineRank

Qin et al., 2008 [1]

- Query-level loss function: $L\left(\tau_g(q), \tau_f(q)\right) \geq 0$
- Wanted attributes
  1. Insensitive to the number of document pairs
  2. More important to rank the top results correctly than those at lower ranks

$$\tau_g(q) = \{d_1^{(1)} \succ \ldots \succ d_{i-j}^{(i-j)} \succ \ldots \succ d_i^{(i)} \succ \ldots \succ d_{i+j}^{(i+j)} \succ \ldots \succ d_{n(q)}^{(n(q))}\}$$

$$\tau_{f_1}(q) = \{d_1^{(1)} \succ \ldots \succ d_i^{(i-j)} \succ \ldots \succ d_{i-j}^{(i)} \succ \ldots \succ d_{i+j}^{(i+j)} \succ \ldots \succ d_{n(q)}^{(n(q))}\}$$

$$\tau_{f_2}(q) = \{d_1^{(1)} \succ \ldots \succ d_{i-j}^{(i-j)} \succ \ldots \succ d_{i+j}^{(i)} \succ \ldots \succ d_i^{(i+j)} \succ \ldots \succ d_{n(q)}^{(n(q))}\}$$

$$\Rightarrow L\left(\tau_g(q), \tau_{f_1}(q)\right) \geq L\left(\tau_g(q), \tau_{f_2}(q)\right)$$

in ad hoc (Web) retrieval, precision "reigns" over recall

  3. Existance of upper bound (loss function should not be biased by very difficult queries)

Listwise LTR: CosineRank

Qin et al., 2008 [1]

- RankCosine loss function adheres to all wanted attributes

$$0 \le L\left(\boldsymbol{g}(q), \boldsymbol{H}(q)\right) \le 1$$

$$L\left(\boldsymbol{g}(q), \boldsymbol{H}(q)\right) = \boxed{\frac{1}{2} \times \left(1\right.} - \boxed{\frac{\boldsymbol{g}(q)^T \boldsymbol{H}(q)}{\left\|\boldsymbol{g}(q)\right\| \times \left\|\boldsymbol{H}(q)\right\|}} \left.\right)$$

ground truth ranking list as a **vector**: ith element is the rating level of the ith document

output vector of the machine learner

cosine similarity

RankCosine

Qin et al., 2008 [1]

- Learning goal: minimize the total loss function over all training queries

$$L(\boldsymbol{H}) = \sum_{q \in Q} L(\boldsymbol{g}(q), \boldsymbol{H}(q))$$

document score

feature vector per document

$$\begin{array}{c} d_1 \\ d_2 \\ \cdots \\ d_{n(q)} \end{array} \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \cdots & \cdots & \cdots & \cdots \\ x_{n(q),1} & x_{n(q),2} & \cdots & x_{n(q),d} \end{pmatrix}_{n(q)\times d} \rightarrow \begin{pmatrix} \boxed{s_1} \\ s_2 \\ \cdots \\ s_{n(q)} \end{pmatrix}_{n(q)\times 1}$$

d features in total

- Ranking function: generalized additive model

$$\boldsymbol{H}(q) = \sum_{t=1}^{T} \alpha_t \boldsymbol{h}_t(q)$$

combination coefficient

weak learner: maps input matrix to an output vector

RankCosine

Qin et al., 2008 [1]

- Stage-wise greedy search strategy to train the parameters in the ranking function
- In the following slides, the idea of AdaBoost is described (instead of the specific derivation in [1])

$$L(\boldsymbol{H}_k) = \sum_q \frac{1}{2}\left(1 - \frac{\boldsymbol{g}(q)^{\mathrm{T}}(\boldsymbol{H}_{k-1}(q) + \alpha_k \boldsymbol{h}_k(q))}{\sqrt{(\boldsymbol{H}_{k-1}(q) + \alpha_k \boldsymbol{h}_k(q))^{\mathrm{T}}(\boldsymbol{H}_{k-1}(q) + \alpha_k \boldsymbol{h}_k(q))}}\right)$$

Setting the derivative of $L(\boldsymbol{H}_k)$ with respect to α_k to zero, with some relaxati
α_k as follows:

$$\alpha_k = \frac{\sum_q \boldsymbol{W}_{1,k}^{\mathrm{T}}(q)\boldsymbol{h}_k(q)}{\sum_q \boldsymbol{W}_{2,k}^{\mathrm{T}}(q)(\boldsymbol{h}_k(q)\boldsymbol{g}^{\mathrm{T}}(q)\boldsymbol{h}_k(q) - \boldsymbol{g}(q)\boldsymbol{h}_k^{\mathrm{T}}(q)\boldsymbol{h}_k(q))}$$

where $\boldsymbol{W}_{1,k}(q)$ and $\boldsymbol{W}_{2,k}(q)$ are two $n(q)$-dimension weight vectors with the

$$\boldsymbol{W}_{1,k}(q) = \frac{\boldsymbol{g}^{\mathrm{T}}(q)\boldsymbol{H}_{k-1}(q)\boldsymbol{H}_{k-1}(q) - \boldsymbol{H}_{k-1}^{\mathrm{T}}(q)\boldsymbol{H}_{k-1}(q)\boldsymbol{g}(q)}{\|\boldsymbol{H}_{k-1}(q)\|^{3/2}}$$

$$\boldsymbol{W}_{2,k}(q) = \frac{\boldsymbol{H}_{k-1}(q)}{\|\boldsymbol{H}_{k-1}(q)\|^{3/2}}$$

AdaBoost

Freund & Schapire, 1995

- Adaptive boosting
  - Meta-classifier (uses other classifiers)

- Weak classifier: a classifier that is a little bit better than random guessing
  - 'rules of thumb'
  - E.g. a small C4.5 decision tree

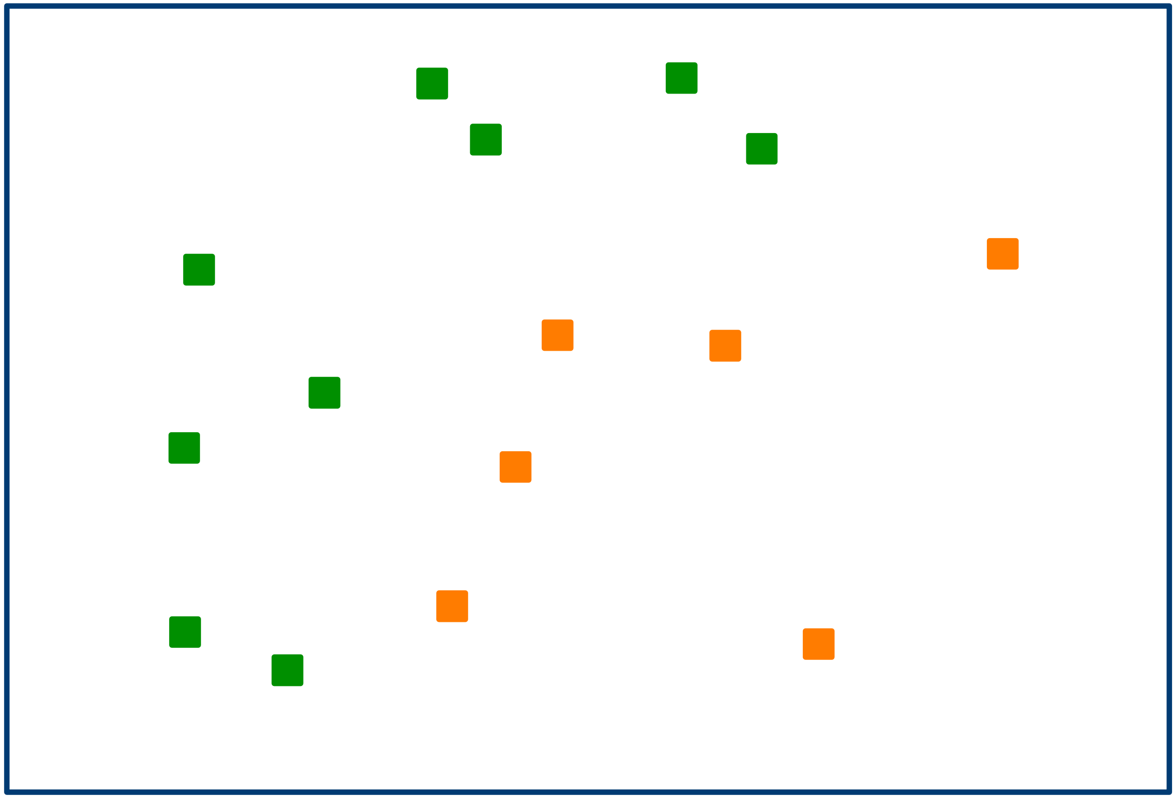
- Idea: combine many weak classifiers to get one 'strong' classifier
  - Adaptive: once a classifier is chosen, the next iteration is geared towards the miss-classified instances

- Advantage: less prone to overfitting

http://cseweb.ucsd.edu/~yfreund/adaboost/

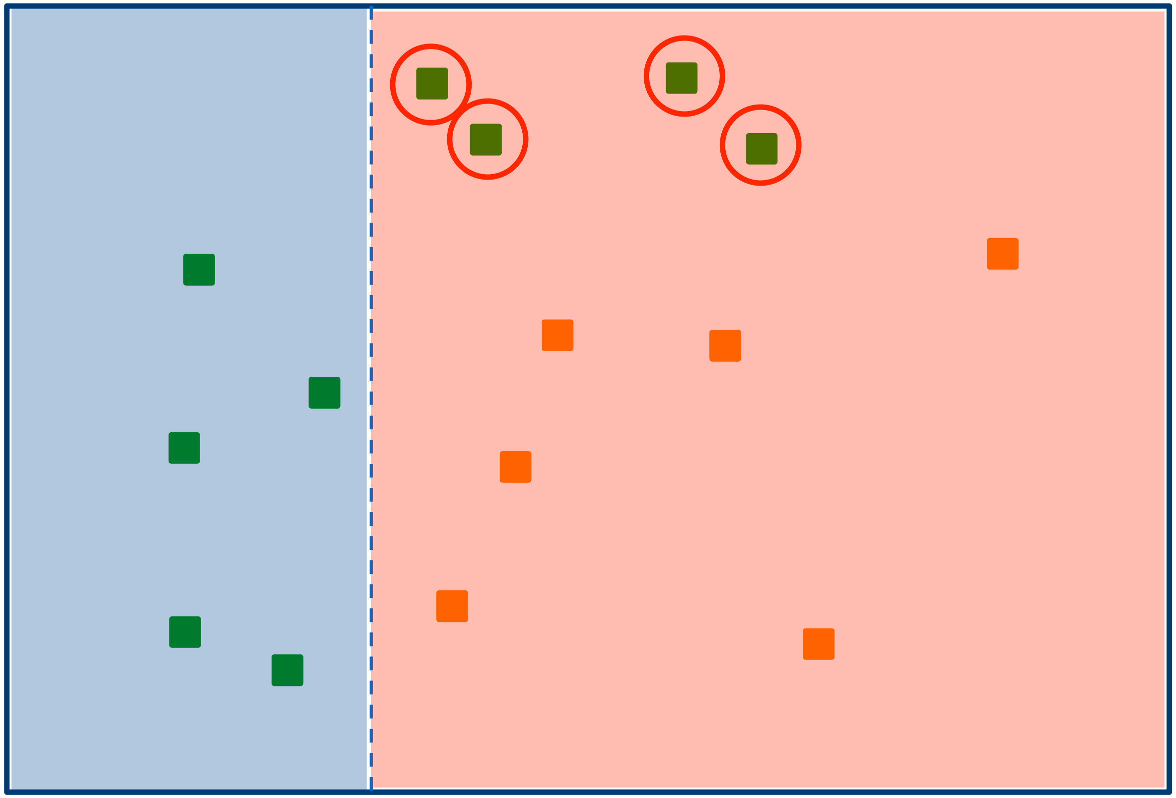
AdaBoost: example

training error: 1.0



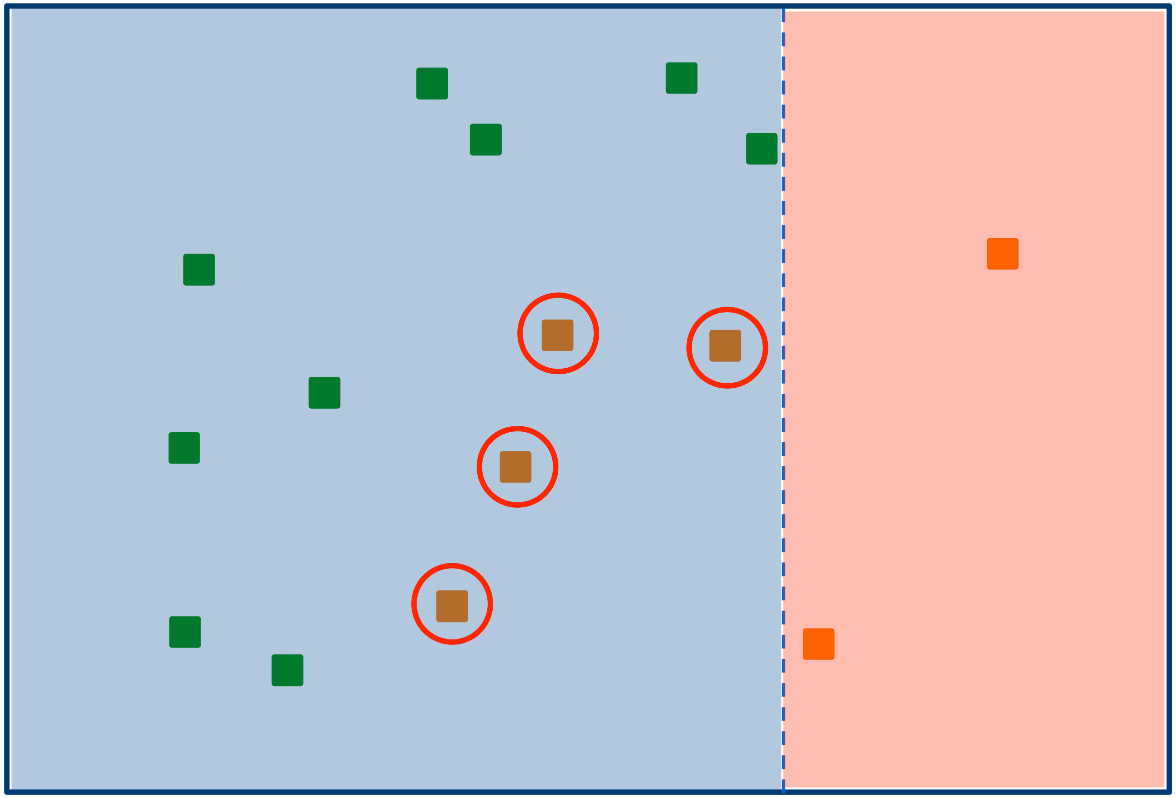
AdaBoost: example

training error: 0.2666



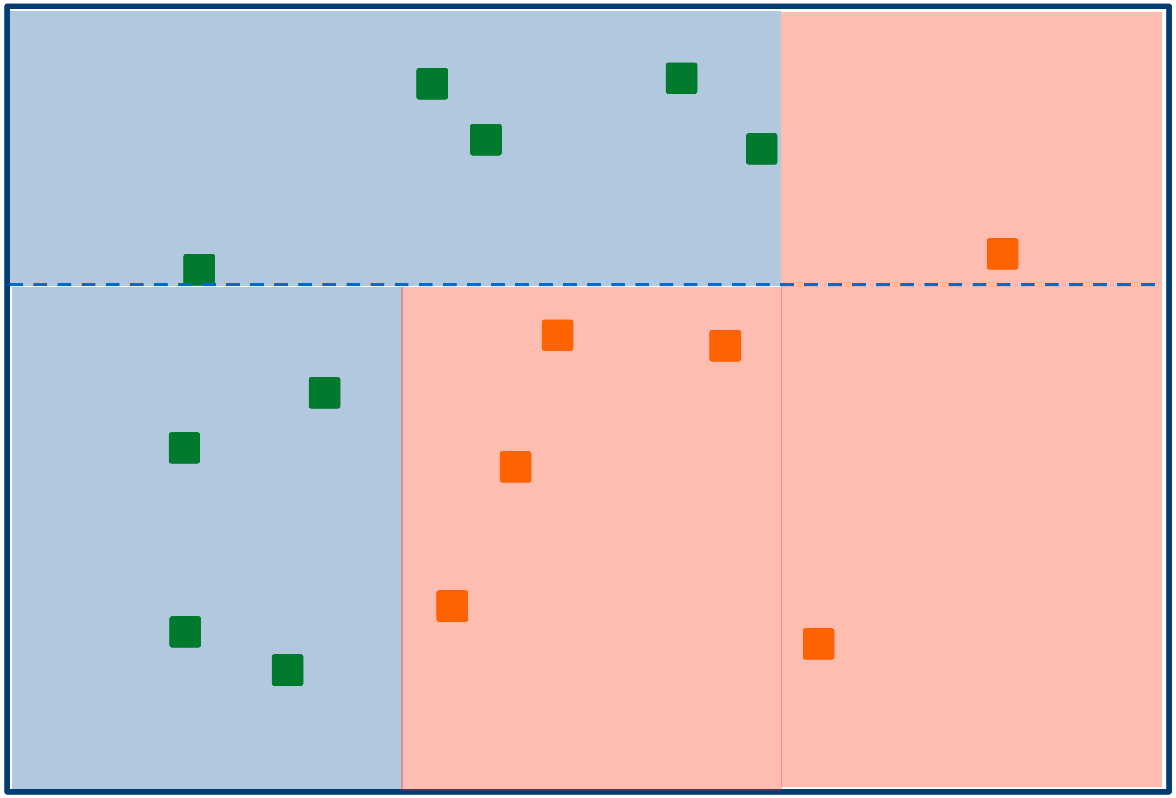
AdaBoost: example

training error: 0.2666



AdaBoost: example

training error: 0.00



AdaBoost algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1,1\}$

Initialize: $D_t(i) = 1/m$

For: $t = 1 \ldots T$

- get weak hypothesis $h_t : X \to \{-1,1\}$ from the set of weak classifiers with min. error wrt. to D_t

$$\varepsilon_t = \sum_{i=1}^{m} D_t(i) I\left(y_i \neq h_t(x_i)\right)$$

- choose: $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$

- update:

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha y_i h_t(x_i))}{Z_t}$$

correctly identified samples are down-weighted, incorrectly identified ones receive higher weigts

Output the final hypothesis:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

RankCosine

Qin et al., 2008 [1]

- Data set I
  - TREC Web track (1 million documents, .gov documents)
  - 50 queries (topic distillation task)
  - Binary relevance judgments
    - Number of relevant documents / query: between 1 and 86
  - 14 features per document
    - Content-based (e.g. BM25 score)
    - Web-structure based (e.g. PageRank)
  - 4-fold cross validation

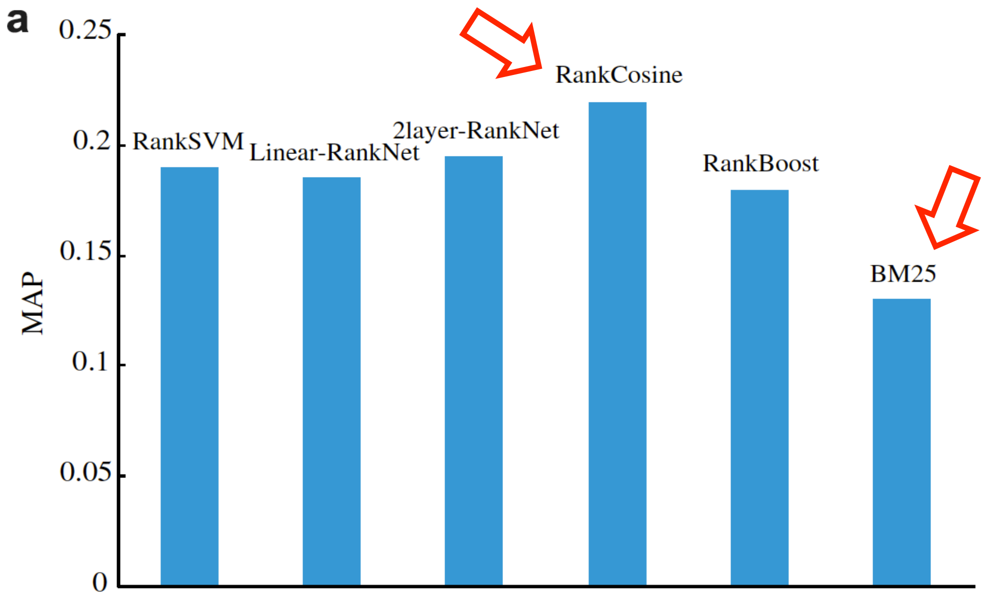
RankCosine

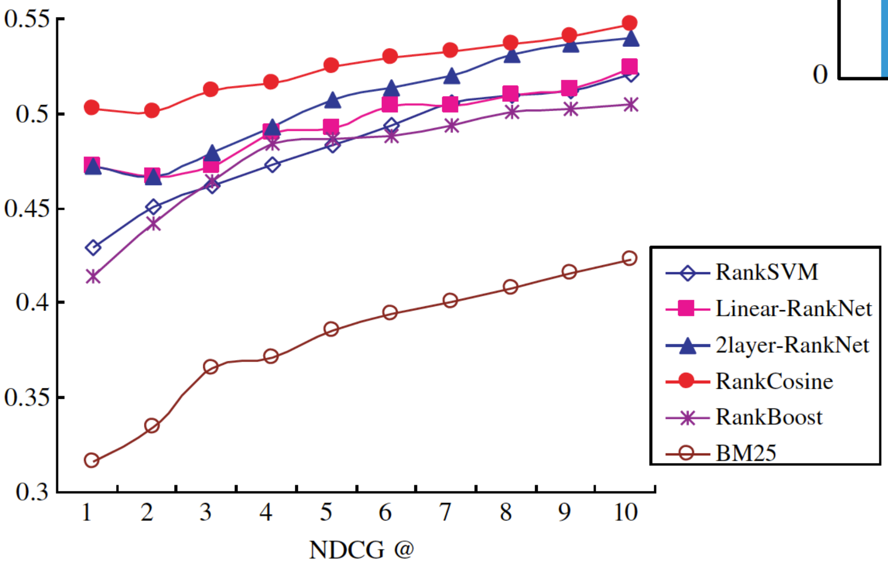
Qin et al., 2008 [1]

- Data set I
  - TREC Web track (1 million documents, .gov documents)
  - 50 queries (topic distillation task)
  - Binary relevance judgments
    - Number of relevant documents / query: 1-86
  - 14 features per document
    - Content-based (e.g. BM25 score)
    - Web-structure based (e.g. PageRank)
  - 4-fold cross validation

RankCosine

Qin et al., 2008 [1]





Graphs taken from [1]

RankCosine

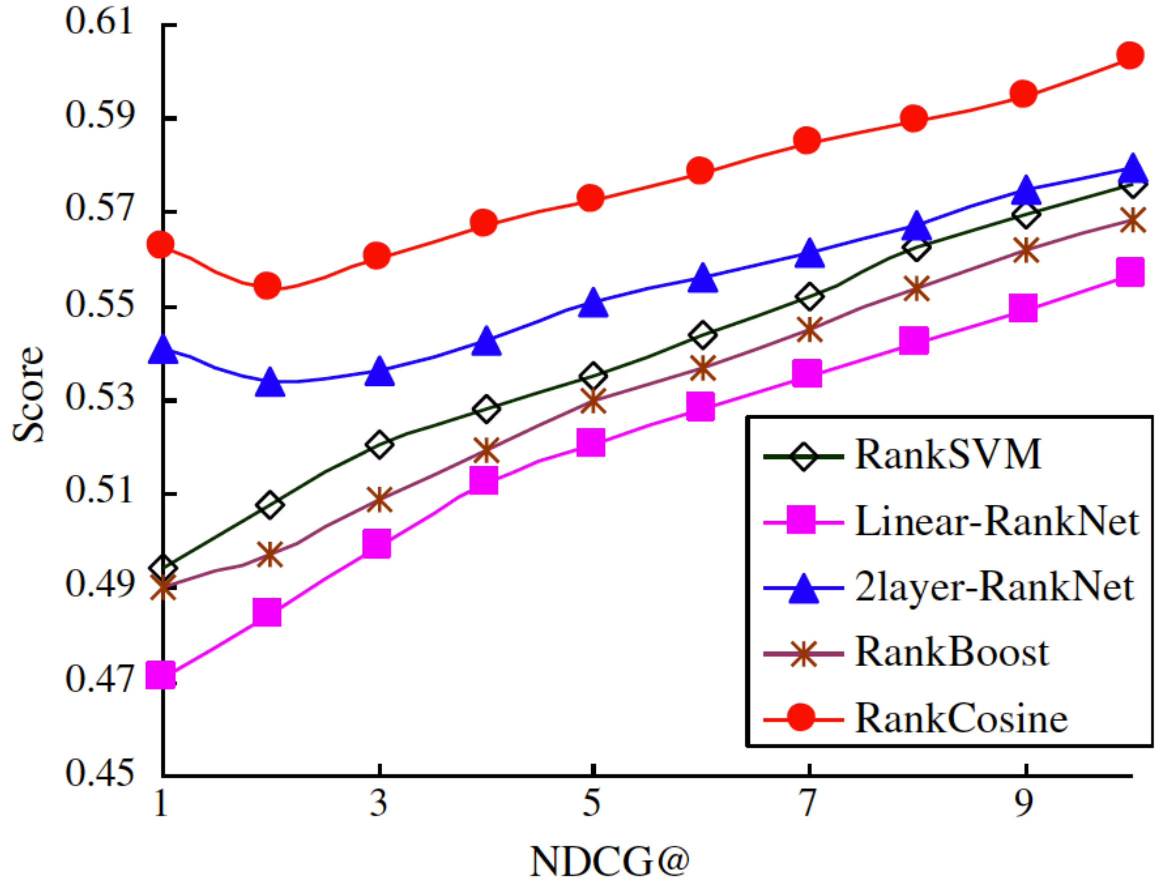
Qin et al., 2008 [1]

- Data set II: Web search data
  - ~2300 queries with human-labeled judgments for the top ranked documents in the result list
    - ~1300 training queries, ~1000 test queries
  - 5 levels of relevance: non-relevant (1) to definitely relevant (5)
    - Evaluation wrt. NDCG

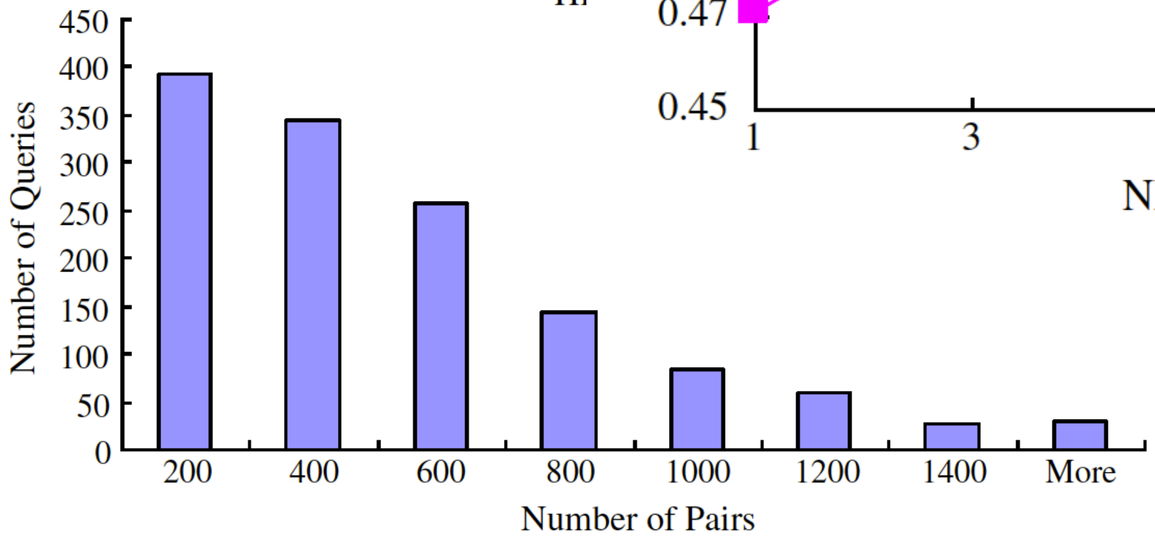
  - Number of search engine features: **334**
    - Query-dependent (term frequency in the anchor text, URL, title, body text, ….)
    - Query-independent (‘page quality’, number of hyperlinks, …)

RankCosine

Qin et al., 2008 [1]



Hi



Graphs taken from [1]

Query logs

Clickthrough data

- Search engines answer millions of queries a day & users leave a lot of traces on the Web

- Users
  - issue queries
  - follow links
  - click on ads
  - Spend time on pages
  - Reformulate their queries
  - Multi-task (browser tabs)
  - …

valuable source of information to tune and improve web search result rankings

Clickthrough data

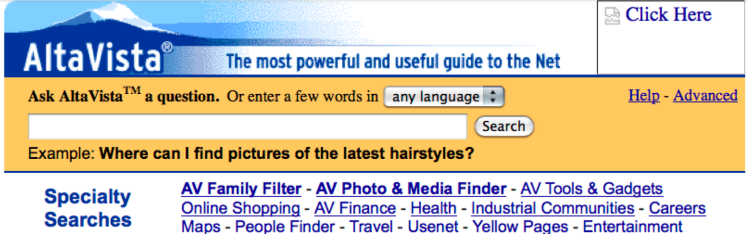
- Search engines answer millions of queries a day & users leave a lot of traces on the Web
- Users
  - issue queries
  - follow links
  - click on ads
  - Spend time on pa
  - Reformulate thei
  - Multi-task (brows
  - …

AEBE68B9618DF768	970916045759	http://www.tribnet,com/
AEBE68B9618DF768	970916045841	http://www.tribnet,com/ ipanema
AEBE68B9618DF768	970916045905	http://www.tribnet,com/ ipanema rio
AEBE68B9618DF768	970916045941	http://www.tribnet,com/ ipanema rio janeiro
F3ABB7F08275F45C	970916015655	
4D2B0109EDB9F6EE	970916192756	free beach
4D2B0109EDB9F6EE	970916192856	free beach
6F82D2C8FBDB32E1	970916114031	inductance calculations
6F82D2C8FBDB32E1	970916114113	inductance calculations
6F82D2C8FBDB32E1	970916114220	f. w. grover
B567BC7C324FC607	970916212905	tamron
B567BC7C324FC607	970916212914	tamron lens
B567BC7C324FC607	970916213036	tamron lens
B567BC7C324FC607	970916213107	
B567BC7C324FC607	970916213226	tamron lens
B567BC7C324FC607	970916213415	tamron lens
F6D568795FD49C6A	970916074751	avex huntsville
8DBB7BE1B9646A21	970916114829	roland camm-1 driver
8DBB7BE1B9646A21	970916114947	free roland camm-1 driver
8DBB7BE1B9646A21	970916115219	free download roland camm-1 driver

Example of a **simple** log file (user, time, query): Excite query log, 1999

Query log analysis

Silverstein et al., 1999 [5]



Wayback machine: April 29, 1999

- AltaVista search engine log
  - 1 billion search requests over 6 weeks
  - 285 million user sessions

- Search session: a series of queries submitted by a single user within a small range of time
  - Meant to capture a single user's attempt to answer an information need
  - Needs to be determined from the query log, e.g. by segmenting it into sessions according to time of inactivity (here: 5 minutes)

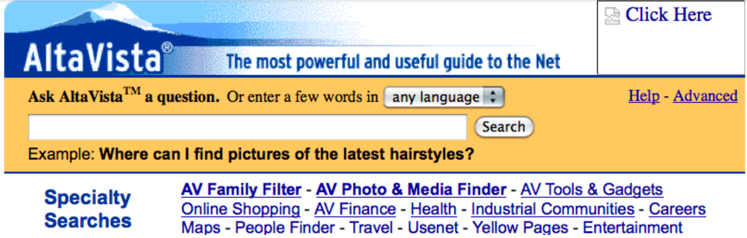
Query log analysis

Silverstein et al., 1999 [5]

- Number of terms per query
  - Average: 2.35 (std. deviation: 1.74)
  - Maximum: 393

- Number of advanced operators (+,-,AND,…) per query

#	%terms / query	%operators / query
0	20.6%	79.6%
1	25.8%	9.7%
2	15.0%	6.0%
3	12.6%	2.6%



Wayback machine: April 29, 1999

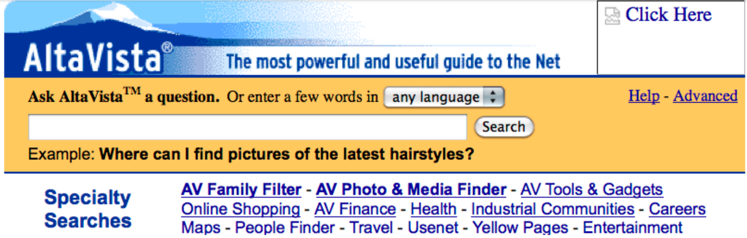
The 25 most often occurring queries

Query	Frequency
sex	1551477
applet	1169031
porno	712790
mp3	613902
chat	406014
warez	398953
yahoo	377025
playboy	356556
xxx	324923
hotmail	321267
[non-ASCII query]	263760
pamela anderson	256559
p****	234037
sexo	226705
porn	212161
nude	190641
lolita	179629
games	166781
spice girls	162272
beastiality	152143
animal sex	150786
SEX	150699
gay	142761
titanic	140963
bestiality	136578

Source: [5]

Query log analysis

Silverstein et al., 1999 [5]



Wayback machine: April 29, 1999

- Frequency of queries
  - Average: 3.97 (std. deviation: 221.31)
  - Maximum: 1.5 million

occurrence	%queries
1	63.7%
2	16.2%
3	6.5%

- Query modifications per session
  - Average: 2.02 (std. deviation: 123.4)
  - Maximum: 172325

queries/session	%sessions
1	77.6%
2	13.5%
3	4.4%

- Result pages per session
  - Average: 1.39 (std. deviation: 3.74)
  - Maximum: 78496

SERP/session	%sessions
1	85.2%
2	7.5%
3	3.0%

Hourly query log analysis

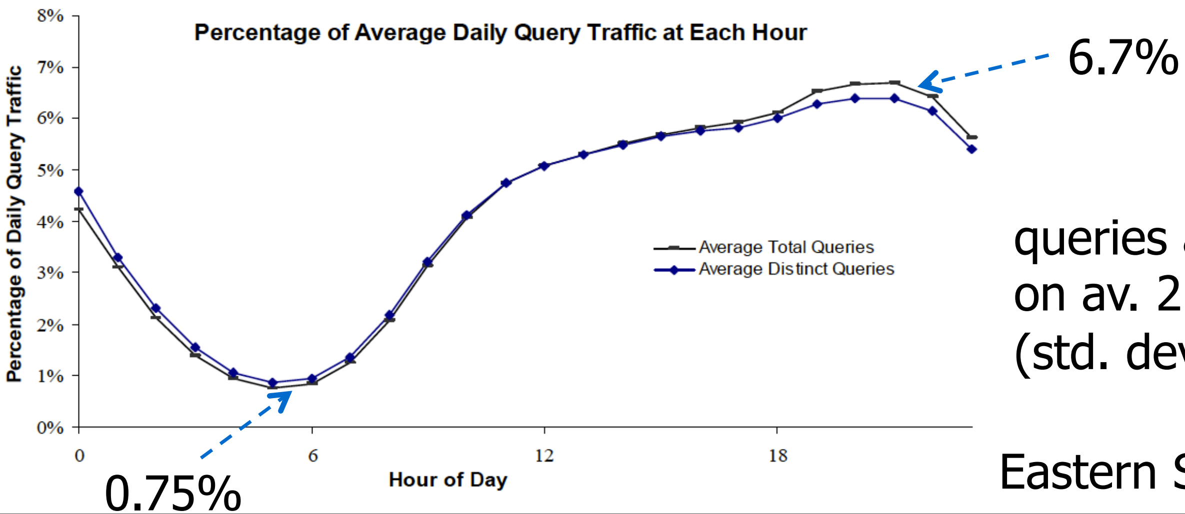
Beitzel et al., 2004 [6]

- How do queries change over time?
  - Time: hours of a day
- Goal: algorithms that predict the likelihood of a query being repeated during a day
- With accurate prediction
  - Impact on cache management and load balancing
  - Improved query disambiguation (information needs have different likelihoods during the day)

Hourly query log analysis

Beitzel et al., 2004 [6]

- Data: AOL query log
  - 1 week (December 2003), ~50 million users
- Average query length
  - Popular queries: 1.7, across all queries: 2.2
- 81% of the time users view the first result page only



queries are repeated on av. 2.14 times/hour (std. deviation 0.12)

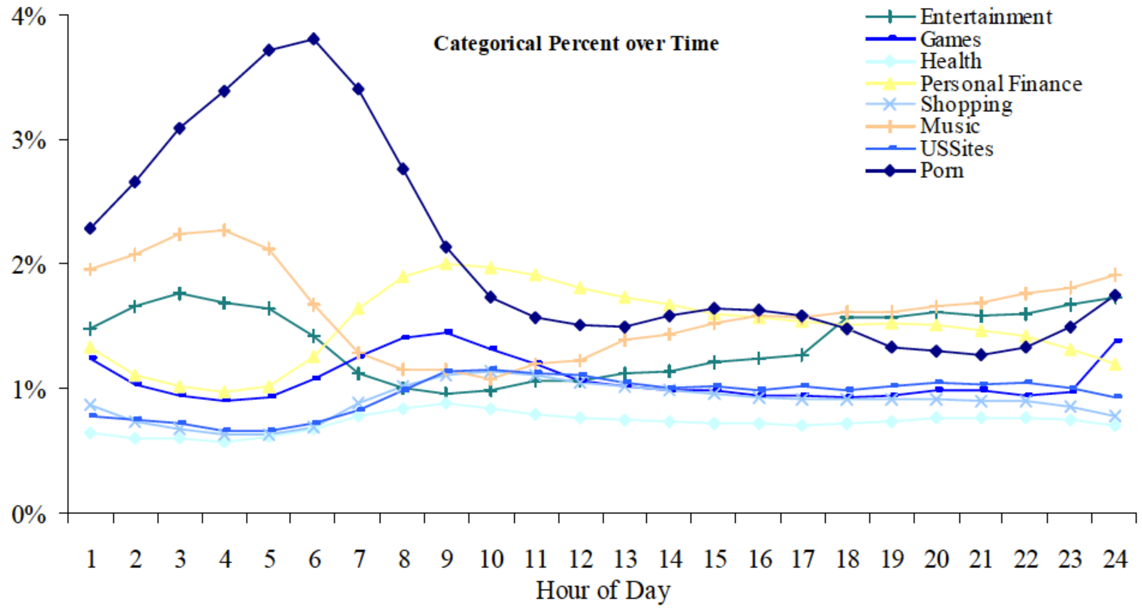
Eastern Standard Time

Source: [6]

Hourly query log analysis

Beitzel et al., 2004 [6]

- Query categories
  - Match queries to manually constructed 'topic lists'
  - 13% of queries match one or more categories



Some categories change more drastically in popularity during the day than others

Source: [6]

Query log clustering

Beeferman et al., 2000 [7]

- Recap: content-based document clustering
  - Documents as vectors in a high dimensional space
  - Documents are grouped according to their similarity in that space (e.g. cosine similarity)

- Clickthrough log based clustering
  - Clusters of related queries
  - Clusters of related URLs
  - Based on co-occurrence counts in the query log (no content analysis)

(query, clicked URL) ⇨

felony	jud13.flcourts.org/felony.html
missoula,+mt	missoula.bigsky.net/score/
feeding+infants+solid+foods	members.tripod.com/drlee90/solid.html
colorado+lotto+results	www.co-lotto.com/
northern+blot	www.invitrogen.com/expressions/1196-3.html
wildflowers	www.life.ca/nl/43/flowers.html

Source: [7]

Query log clustering

Beeferman et al., 2000 [7]

- Two observations
  1. Users with the same information need may phrase their queries differently but select the same URL from the result page
  2. After issuing the same query, users may visit two different URLs (evidence for their similarity)

- Usage scenarios
  - Rapid clustering capable of identifying late-breaking trends (in news)
  - Automatic ontology generation (ODP)
  - Bookmark organization
  - Search result clustering
  - User profile construction

Query log clustering

Beeferman et al., 2000 [7]

- Advantages over content-based clustering
  - Correlation between documents and queries can be computed efficiently
  - Text-free pages can be clustered
  - Pages with restricted access can be clustered
  - Pages with dynamic content can be clustered
- Iterative graph-based clustering; simultaneously find
  - Disjoint sets of queries (same/similar information need per cluster)
  - Disjoint sets of URLs (can be served for the same/similar information need per cluster)

Query log clustering

Beeferman et al., 2000 [7]

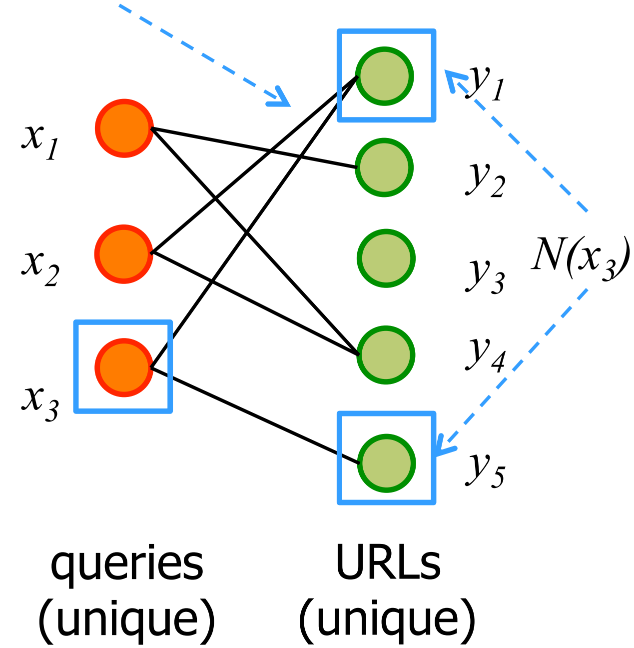
- Bipartite graph based on click log
  - Nodes in two separate partitions
  - Edges *never* exist between nodes of the same partition

- Intuitively: if the neighbourhoods $N(a)$ and $N(b)$ of two nodes a and b [in the same partition] have a large overlap, a and b can be considered similar

$$\sigma(a,b)=\begin{cases}\frac{N(a)\cap N(b)}{N(a)\cup N(b)}, & \text{if } |N(a)\cup N(b)|>0\\ 0, & \text{otherwise}\end{cases}$$

$$\sigma(a,b)\in[0,1]$$

(query, clicked URL) appeared in the query log

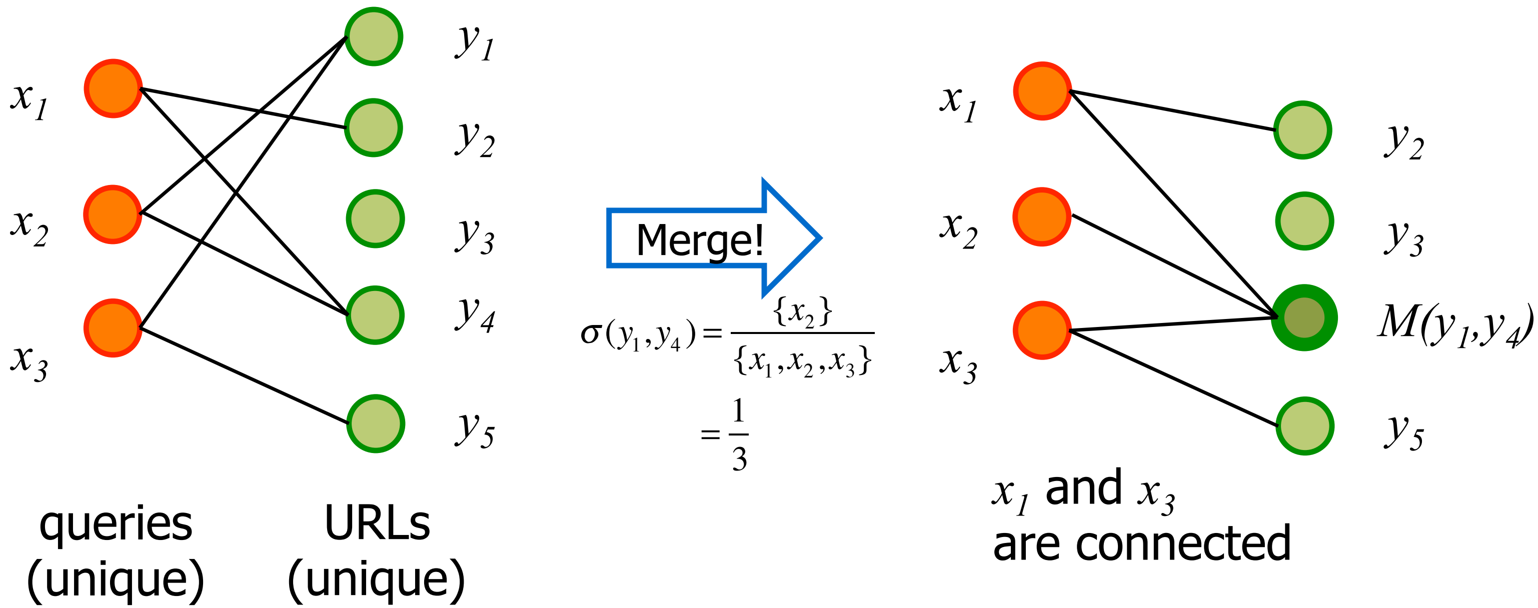


$$\sigma(x_1,x_2)=\frac{\{y_2,y_4\}\cap\{y_1,y_4\}}{\{y_2,y_4\}\cup\{y_1,y_4\}}=\frac{\{y_4\}}{\{y_1,y_2,y_4\}}=\frac{1}{3}$$

$$\sigma(x_1,x_3)=\frac{\{y_2,y_4\}\cap\{y_1,y_5\}}{\{y_2,y_4\}\cup\{y_1,y_5\}}=0$$

Query log clustering

Beeferman et al., 2000 [7]



➔perform iterative agglomerative clustering

Query log clustering

Beeferman et al., 2000 [7]

- Agglomerative iterative clustering
  - Input: bipartite graph G
  - Output: new bipartite graph G': each red (green) vertex of G' corresponds to one or more red (green) vertices of G

① Score all pairs of red vertices in G according to σ
② Merge the two red vertices x_i, x_j for which $\sigma(x_i, x_j)$ is largest
③ Score all pairs of green vertices y_i, y_j in G according to σ
④ Merge the two green vertices y_i, y_j for which $\sigma(y_i, y_j)$ is largest
⑤ Go to step (1) unless termination condition applies

- Stopping criterion: iterate until the graph consists of connected components with a single query and url

$$\max_{q_i, q_j \in Q} \sigma(q_i, q_j) \text{ and } \max_{u_i, u_j \in U} q(u_i, u_j) = 0$$

Query log clustering

Beeferman et al., 2000 [7]

unclustered query log | clustered query log

Source: [7]

Query log clustering

Beeferman et al., 2000 [7]

- Clustering evaluated within an application
  - Improved query suggestions in Web search

- Three systems
  - Baseline: standard (Lycos) query-suggestion approach
  - Full replacement: replace default suggestions with cluster-based suggestions
  - Hybrid: replace some of the original suggestions (the weakest ones) with the best cluster-based suggestions

- Evaluation: clickthrough rate
  - How often is each suggestion clicked by the user?

Query log clustering

Beeferman et al., 2000 [7]

- Results

Strategy	Impressions	Clicks	Clickthrough rate
Baseline	6,120,943	71,138	1.16%
hybrid	6,058,757	79,515	1.31%
Full replace.	5,985,997	61,377	1.03%

- Issues: long tail of the query log

Query log based query expansion

Cui et al., 2002 [8]

- Vocabulary gap (term mismatch) between authors and consumers, i.e. the users
- Augment the short Web queries by employing automatic query expansion (adding words and phrases)
- Approaches
  - Global analysis (co-occurrence)
  - Local analysis (relevance feedback)
  - Here: query log based
- Session: `<query> [clicked URLs]`

Query log based query expansion

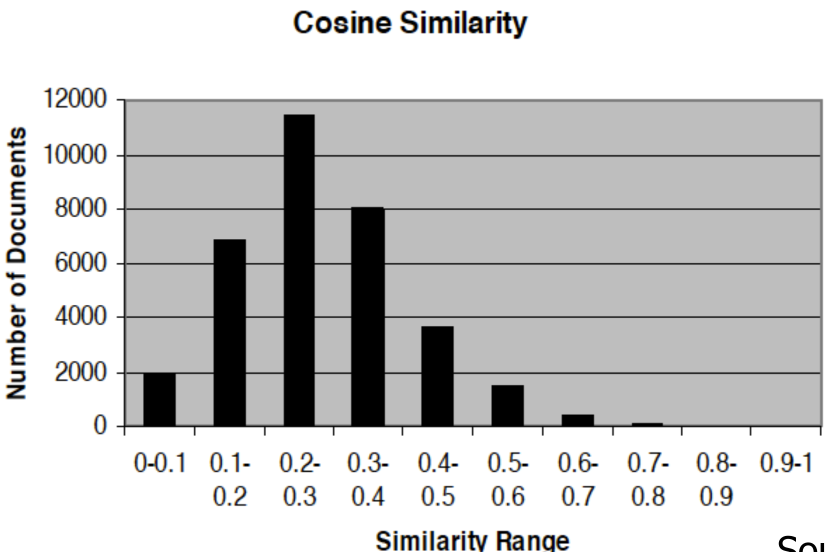
Cui et al., 2002 [8]

- Idea: if a set of documents is often clicked for the same queries, then the terms in these documents are related to the query terms
  - Connect query and document terms through the query log
  - Select high-quality expansion terms from the document space
- Assumption: clicked URLs are relevant to the query
- Replaces the query expansion approach based on relevance feedback (now: implicit relevance feedback)

Query log based query expansion

Cui et al., 2002 [8]

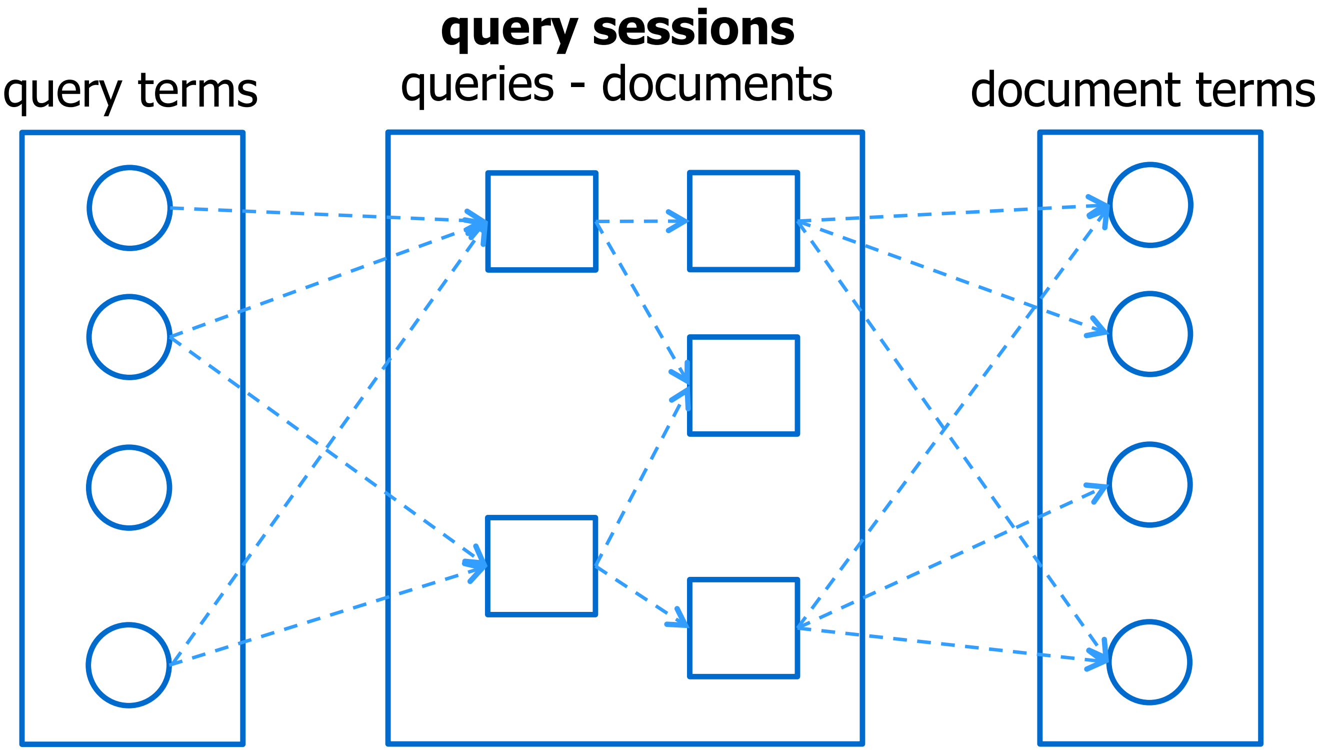
- The gap between the document and query space
  - Document as vector in the *document space*
  - Document as "virtual document" vector in the *query space* by collecting all queries with clicks on the document
  - Similarity: cosine
  - Average similarity: 0.28



Source: [8]

Query log based query expansion

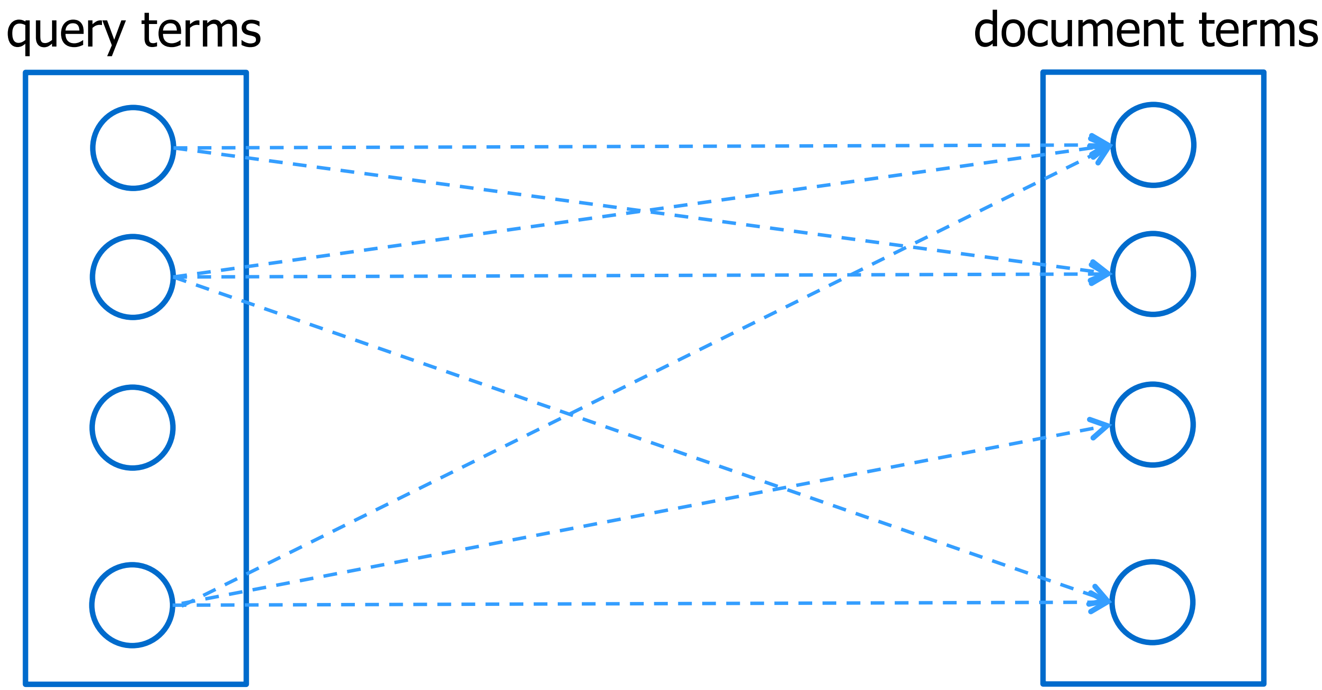
Cui et al., 2002 [8]



If there is at least one path from a query term, to a document term, a probabilistic link is established between them

Query log based query expansion

Cui et al., 2002 [8]



Query log based query expansion

Cui et al., 2002 [8]

- Degree of correlation between query and document terms based on conditional probabilities

$$P(w_j^{(d)} \mid w_i^{(q)}) = \frac{P(w_j^{(d)}, w_i^{(q)})}{P(w_i^{(q)})} = \frac{\sum_{\forall D_k \in S} P(w_j^{(d)} \mid D_k) \times \boxed{P(D_k \mid w_i^{(q)})} \times P(w_i^{(q)})}{P(w_i^{(q)})}$$

document term

query term

set of documents (documents that appear at least in one session with the query term)

probability of D_k being clicked if w_i appears in the query (query logs)

Query log based query expansion

Cui et al., 2002 [8]

- For a new query
  ① Extract the query terms
  ② For each query term, determine the document terms' conditional probabilities
  ③ Combine the probabilities for all query terms

$$P(w_j^{(d)} \mid Q) = \ln\left(\prod_i \left(P(w_j^{(d)} \mid w_i^{(q)}) + 1\right)\right)$$

  ④ Pick the top ranked document terms as expansion terms

Query log based query expansion

Cui et al., 2002 [8]

- Data set
  - Two-month Encarta query log with ~4.8 million sessions
  - Corpus: 42,000 Encarta documents
  - 30 test queries
  - Human assesors based relevance judgments
- Results
  - 50 expansion terms

	LC analysis	Log based	%change
Relevant terms (%)	23.27	30.73	+32.03

- E.g. Query "Steve Jobs" expanded with
  - "personal computer", "Apple computer", "CEO"

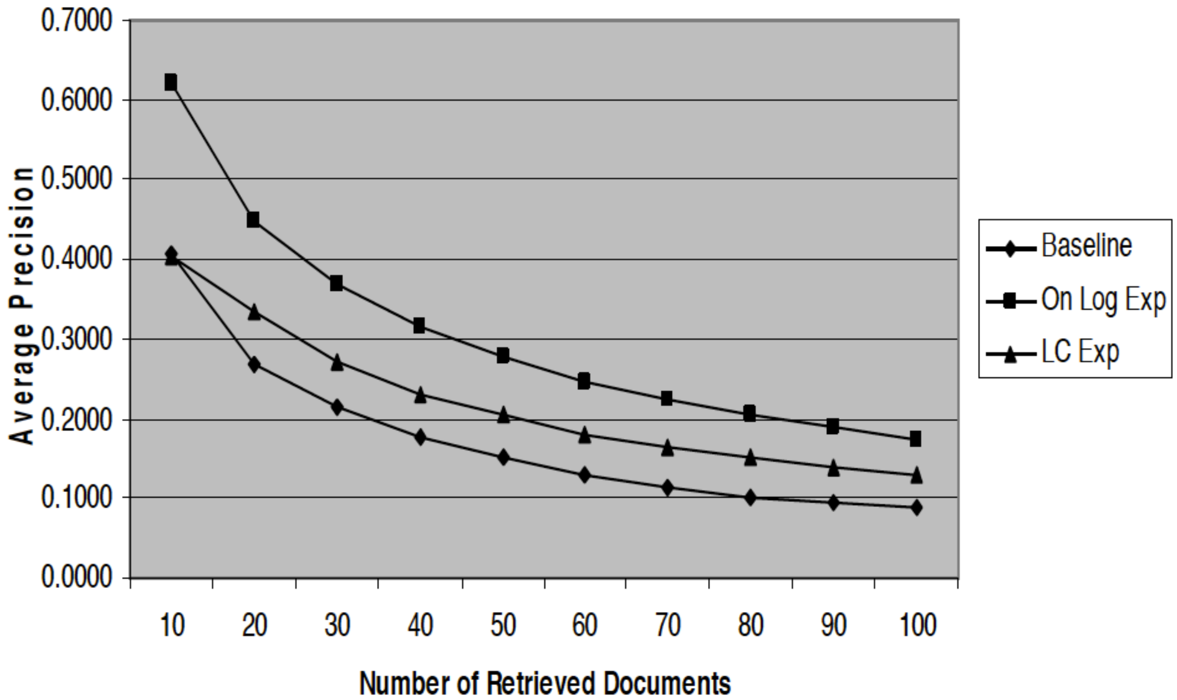
Source: [8]

1 Java computer 2 nuclear submarine
3 Apple computer 4 Windows 5 fossil fuel
6 cellular phone 7 search engine
8 Six Day War 9 space shuttle
10 economic impact of recycling tires
11 China Mao Ze Dong 12 atomic bomb
13 Manhattan project 14 Sun Microsystems
15 Cuba missile crisis 16 motion pictures
17 Steve Jobs 18 pyramids 19 what is Daoism
20 Chinese music 21 genome project
22 Apollo program 23 desert storm
24 table of elements 25 Toronto film awards
26 Chevrolet truck 27 DNA testing
28 Michael Jordan 29 Ford 30 ISDN

Query log based query expansion

Cui et al., 2002 [8]

- Results: system effectiveness in average precision



Source: [8]

Implicit relevance judgments

- Learning to rank, BM25, LM …
  - They all need a lot of training data to effectively learn the models' parameters
  - Usually asume explicit relevance judgments
- Training data in IR: relevance judgments
  - Pairs of (query,document) with relevance scores
- Extremely expensive to accumulate
  - TREC example: more than 700 assessor hours for 50 queries (assuming 30 seconds per document to be judged)

Clickthrough data

Agichtein et al., 2006 [4]

- How effective is implicit feedback in practice (i.e. in a large-scale operational environment)?
  - Web search engines use hundreds of features and are heavily tuned

- How can implicit feedback be combined with the existing ranking produced by the search system?

- Millions of interactions
  - Instead of treating a user as reliable "expert", aggregate information from multiple, unrealiable search session traces

Clickthrough data

Agichtein et al., 2006 [4]

- Incorporating implicit feedback as independent evidence
  - Retrieve an initial ranking
  - Assign an expected relevance/user satisfaction score based on previous interactions
  - Merge the rank orders of the initial and IF based ranking; order results by score S_{merge} (↓)

$$S_{merge}(d, I_d, O_d, w_I) = \begin{cases} w_I \times \frac{1}{I_d + 1} + \frac{1}{O_d + 1}, & \text{if implicit feedback exists for } d \\ \frac{1}{O_d + 1}, & \text{otherwise} \end{cases}$$

implicit rank of document d

original rank of d

influence of IF

if w_I *is* extremely high, clicked results are simply ranked over unclicked results

Clickthrough data

Agichtein et al., 2006 [4]

- Incorporating implicit feedback in the LTR algorithm
  - Derive a set of features from implicit feedback
  - At runtime, the search engine needs to fetch the implicit feedback features associated with each query-result URL pair
  - LTR needs to be robust to missing values
    - More than 50% of queries to Web search engines are unique

- Here: RankNet
  - Neural net based tuning algorithm that optimizes feature weights to best match explicitly provided pairwise user preferences
  - Has both train- and run-time efficiency
  - Aggregate (query,URL) pair features across all instances in the session logs

Clickthrough data

Agichtein et al., 2006 [4]

- Different types of user action features
- Directly observed vs. derived features
- Browsing behavior *after* the result has been clicked
- Snippet based features are included as users often determine relevance based on the snippet information

“Feature engineering” is the main issue!

Clickthrough features	
Position	P
ClickFrequency	N
ClickProbability	Probability of a click for this query and URL
ClickDeviation	Deviation from expected click probability
IsNextClicked	1 if clicked on next position, 0 otherwise
IsPreviousClicked	1 if clicked on previous position, 0 otherwise
IsClickAbove	1 if there is a click above, 0 otherwise
IsClickBelow	1 if there is click below, 0 otherwise
Browsing features	
TimeOnPage	Page dwell time
CumulativeTimeOnPage	Cumulative time for all subsequent pages after search
TimeOnDomain	Cumulative dwell time for this domain
TimeOnShortUrl	Cumulative time on URL prefix, no parameters
IsFollowedLink	1 if followed link to result, 0 otherwise
IsExactUrlMatch	0 if aggressive normalization used, 1 otherwise
IsRedirected	1 if initial URL same as final URL, 0 otherwise
IsPathFromSearch	1 if only followed links after query, 0 otherwise
ClicksFromSearch	Number of hops to reach page from query
AverageDwellTime	Average time on page for this query
DwellTimeDeviation	Deviation from average dwell time on page
CumulativeDeviation	Deviation from average cumulative dwell time
DomainDeviation	Deviation from average dwell time on domain
Query-text features	
TitleOverlap	Words shared between query and title
SummaryOverlap	Words shared between query and snippet
QueryURLOverlap	Words shared between query and URL
QueryDomainOverlap	Words shared between query and URL domain
QueryLength	Number of tokens in query
QueryNextOverlap	Fraction of words shared with next query

Table 4.1: Some features used to represent post-search navigation history for a given query and search result URL.

Source: [4]

Clickthrough data

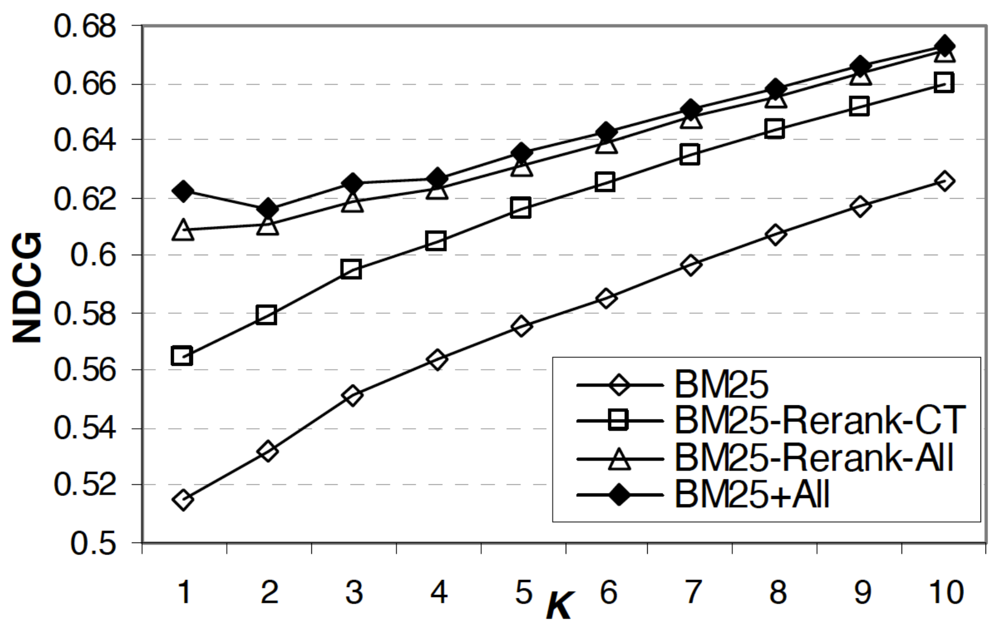
Agichtein et al., 2006 [4]

- Goal: improved retrieval effectiveness of the system
- Evaluation: "random sample of queries from web search logs of a major engine with associated results and traces for user actions"
  - 3000 queries (compare with TREC: 50-150)
  - Drawn uniformly at random, i.e. representative of the query distribution
  - On average, 30 results judged per query by human assessors (six point scale)
  - 8 weeks of user interactions with 1.2 million unique queries (sufficient interactions for ~50% of queries)

Clickthrough data

Agichtein et al., 2006 [4]

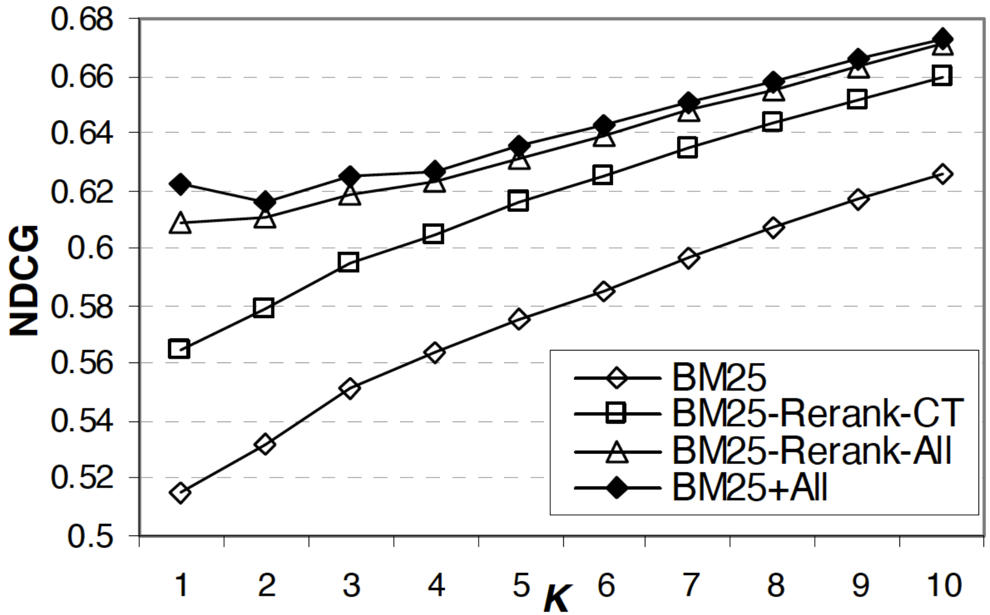
- Compared approaches
  - BM25F: content-based (fields) and query-independent link-based information (PageRank, URL depth, etc.)
  - BM25F-RerankCT: rerank
  - BM25F-RerankAll: rerank features (model of user p
  - BM25F+All: train RankNe



Source: [4]

Clickthrough data

Agichtein et al., 2006 [4]



Source: [4]

Clickthrough data

Agichtein et al., 2006 [4]

- Compared approaches II
  - RankNet: hundreds of features of a major Web search engine
  - RankNet+All: including IF features
  - BM25F: content-based (fields) and query-independent link-based information (PageRank, URL depth, etc.)
  - BM25F+All: train RankNet over the feature set of BM25F and IF

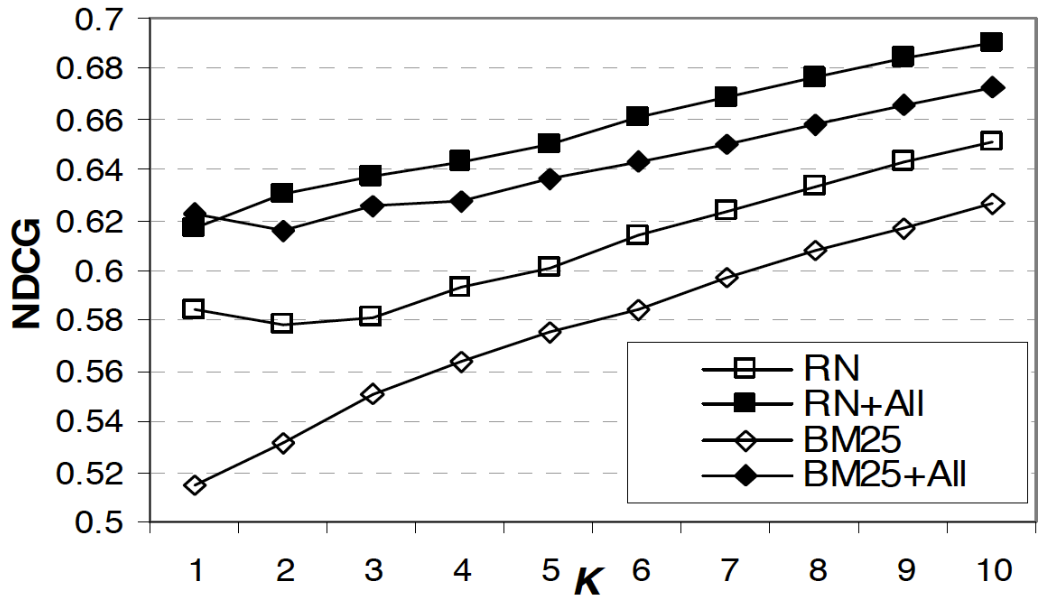
	MAP
BM25F	0.184
BM25F-RerankCT	0.215
BM25F-RerankAll	0.218
BM25F+All	0.222
RankNet	0.215
RankNet+All	**0.248**

Clickthrough data

Agichtein et al., 2006 [4]

- Compared approaches II
  - RankNet: hundreds of features of a major Web search engine
  - RankNet+All: including IF features
  - BM25F: content-based (fields) and query-independent link-based information (PageRank,
  - BM25F+All: train RankN

	MAP
BM25F	0.184
BM25F-RerankCT	0.215
BM25F-RerankAll	0.218
BM25F+All	0.222
RankNet	0.215
RankNet+All	**0.248**



Source: [4]

Implicit relevance judgments

Joachims et al., 2007 [3]

- Research question: how can training examples (qrels) be generated automatically from clickthrough data?
  - User behavior is 'for free'
- Advantages: cost effective, larger quantities, without burdening the user (no questionnaire, relevance feedback)
- Disadvantages: more difficult to interpret and noisy
- User study investigating users' interaction with SERP (Search Engine Result Page)
  - How does click behaviour relate to relevance judgments?
  - Eytracking study gives insights into users' subconscious behaviour

Implicit relevance judgments

Joachims et al., 2007 [3]

- Important to know what results a user actually views
  - Implicit relevance judgments need to be considered in this context (a result not viewed cannot be considered non-relevant)

- Early work assumed that each click represents an endorsement of the result (i.e. a click = a positive relevance judgment)

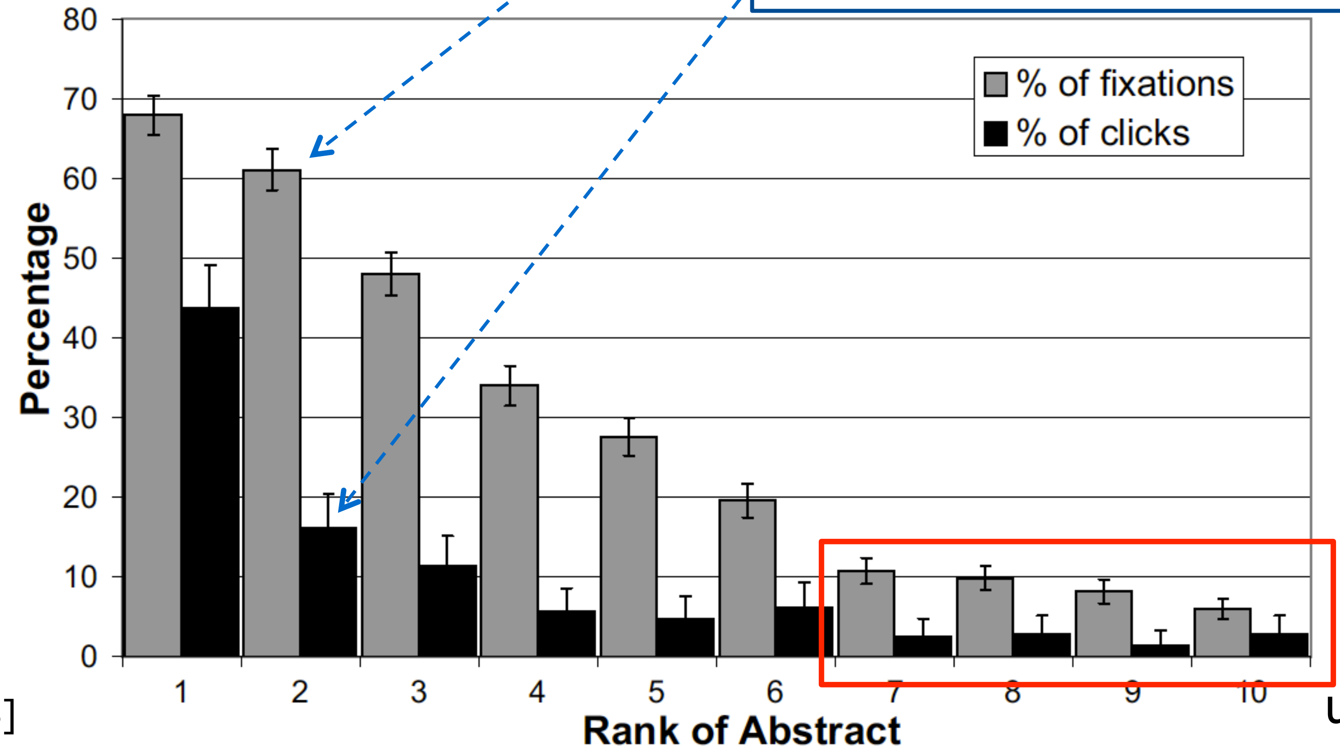
- User study with 3 experimental conditions
  - Normal (original Google ranking)
  - Swapped (top two Google results swapped)
  - Reversed (Google results in reverse order)

- Explicit relevance judgments collected as control

Implicit relevance judgments

Joachims et al., 2007 [3]

- Users mostly look at the top two results (less than 50% look beyond)
- Top ranked result twice as likely to be clicked as the second result (*though similar view rates*)



Source: [3]

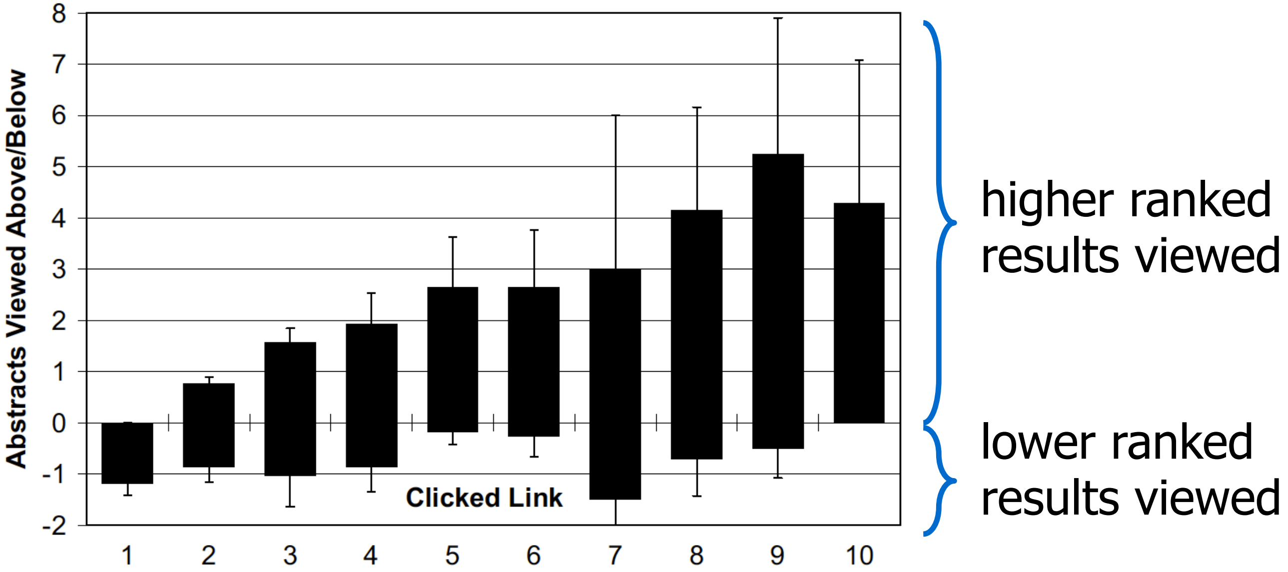
users had to scroll

Implicit relevance judgments

Joachims et al., 2007 [3]

- Users tend to scan the results from top to bottom
  - Results at rank 1 & 2 are viewed initially

- Which links do users evaluate before clicking?

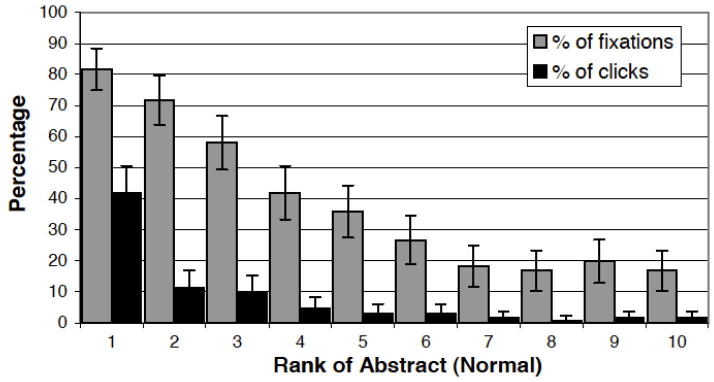


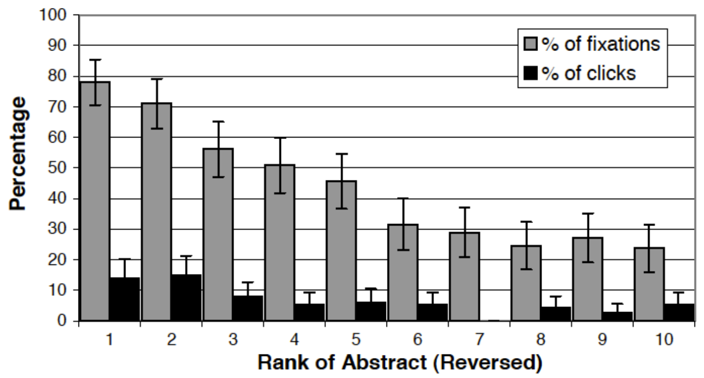
Source: [3]

Implicit relevance judgments

Joachims et al., 2007 [3]

- Does relevance influence user decisions?
  - So far: clicks considered independent of relevance
- reverse condition (degraded ranking)
  - Users view lower ranks more freq.
  - Users are less likely to click on result 1
  - Reverse: av. rank of a clicked result: 4 (compared to 2.7 in normal)
  - Quality-of-context bias: clicks are less relevant on average compared to the normal condition (clicks dependent on the overall quality of the system)





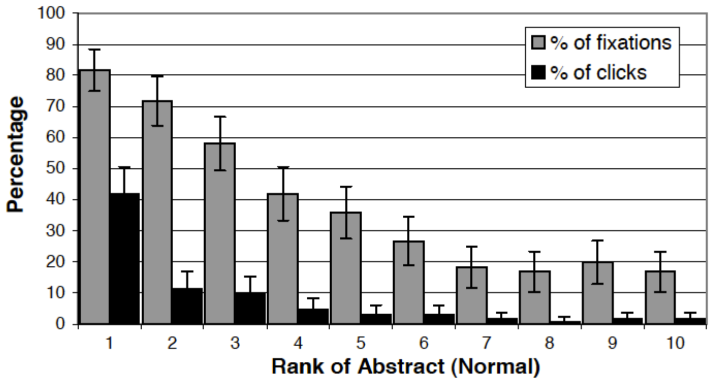
Source: [3]

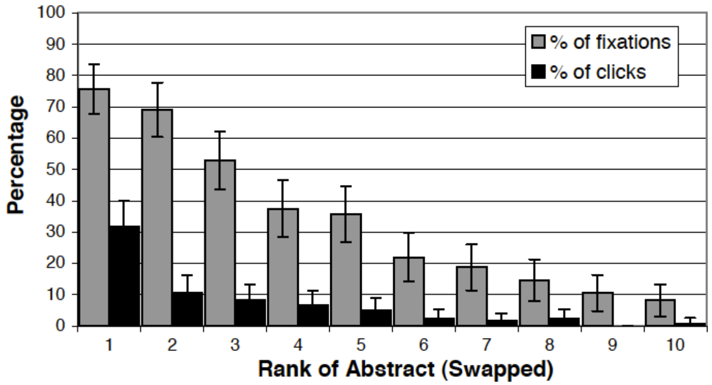
Implicit relevance judgments

Joachims et al., 2007 [3]

- Does relevance influence user decisions?
- Swapped condition
  - Trust bias (Google must be right!)
  - Users *are* influenced by result order
  - Decision to click influenced by result position

		click l_1, not l_2	click l_2, not l_1
Normal	$rel(l_1)>rel(l_2)$	19	1
	$rel(l_1)<rel(l_2)$	5	2
Swapped	$rel(l_1)>rel(l_2)$	15	1
	$rel(l_1)<rel(l_2)$	10	7





Source: [3]

Implicit relevance judgments

Joachims et al., 2007 [3]

- Thus: interpreting clicks as absolute relevance judgments is likely to fail
  - Accurate interpretations need to take the user's trust and the quality of the system into account (difficult to measure)

- However: clicks can be seen as *preference* statements
  - Exploit the fact that some results were *not* clicked
  - Example:

$$l_1^* \; l_2 \; l_3^* \; l_4 \; l_5^* \; l_6 \; l_7 \quad (*click)$$

    - l_3 is likely to be more relevant than l_2 (remember: users scan lists from top to bottom; user decided not to click l_2)
    - l_5 is likely to be more relevant than l_2 and l_4

Implicit relevance judgments

Joachims et al., 2007 [3]

- Example:

  $$l_1^* \; l_2 \; l_3^* \; l_4 \; l_5^* \; l_6 \; l_7 \quad (*click)$$

  - a relevance based ranking should return l_3 ahead of l_2 and l_5 ahead of l_2 and l_4 (*partial rankings*)

- Extracting preference feedback: **Click > Skip Above**

  For a ranking $(l_1, l_2, ...)$ and a set C containing the ranks of the clicked on results, extract a preference example $rel(l_i) > rel(l_j)$ for all pairs $1 \leq j < i$, with $i \in C$ and $j \notin C$.

  ➔takes trust bias and quality-of-context into account.

Implicit relevance judgments

Joachims et al., 2007 [3]

- Example:

$$l_1^* \; l_2 \; l_3^* \; l_4 \; l_5^* \; l_6 \; l_7 \quad (^*click)$$

  - a relevance based ranking should ret
    of l_2 and l_4 (*partial rankings*)

- Extracting preference feedback: **Clic**

  For a ranking $(l_1, l_2, ...)$ and a set
  of the clicked on results, extract a
  $rel(l_i) > rel(l_j)$ for all pairs $1 \leq j <$

  ➔takes trust bias and quality-of-cont

1	$C = \{2,5,7\}$
2	$rel(l_2) > rel(l_1)$
3	$rel(l_5) > rel(l_1)$
4	$rel(l_5) > rel(l_3)$
5	$rel(l_5) > rel(l_4)$
6	$rel(l_7) > rel(l_1)$
7	$rel(l_7) > rel(l_3)$
8	$rel(l_7) > rel(l_4)$
	$rel(l_7) > rel(l_6)$

Implicit relevance judgments

Joachims et al., 2007 [3]

- Extracting preference feedback: **Last Click > Skip Above**

  For a ranking $(l_1, l_2, ...)$ and a set C containing the ranks of the clicked on results, let $i \in C$ be the rank of the link that was clicked last. Extract a preference example $rel(l_i) > rel(l_j)$ for all pairs $1 \leq j < i$, and $j \notin C$.

- ... more strategies exist ....

Implicit relevance judgments

Joachims et al., 2007 [3]

- Extracting preference feedback: **Last Click > Skip Above**

For a ranking $(l_1, l_2, \ldots.)$ and a set C c
of the clicked on results, let $i \in C$ be
was clicked last. Extract a preference
for all pairs $1 \le j < i$, and $j \notin C$.

1
2
3
4
5
6
7
8

$C = \{2,5,7\}$

$rel(l_7) > rel(l_1)$

$rel(l_7) > rel(l_3)$

$rel(l_7) > rel(l_4)$

$rel(l_7) > rel(l_6)$

- ... more strategies exist ….

Implicit relevance judgments

Joachims et al., 2007 [3]

- Accuracy of implicit feedback compared to explicit feedback
    - Explicit: human assessors ranked the results according to their relevance

- **Click > Skip Above** yields 81% correct preferences
    - random baseline: 50% accuracy
    - Inter-rater agreement (human assessors): 90% accuracy (upper bound)

- **Last Click > Skip Above** yields 83% correct preferences

Implicit relevance judgments

Joachims et al., 2007 [3]

- Generated preferences: comparison between the results from the *same* query (within-query preferences)

- Too restrictive
  - Strategies only produce preferences between the top few results shown to the user
  - Typically users run query chains (query reformulations)
    - Between 1.5 and 3 queries on average per session

- Goal: generate accurate relative preference judgments between results from different queries within a chain of query reformulations (same information need)

Implicit relevance judgments

Joachims et al., 2007 [3]

- Generated preferences: com
  the *same* query (within-quer

oed $\Rightarrow l_1\ l_2\ l_3\ l_4\ l_5\ l_6\ l_7$

oxford english dictionary $\Rightarrow l'_1\ l'^{*}_2\ l'_3\ l'_4\ l'^{*}_5\ l'_6\ l'_7$

may be relevant to query "oed"

- Too restrictive
  - Strategies only produce preferences between the top few results shown to the user
  - Typically users run query chains (query reformulations)
    - Between 1.5 and 3 queries on average per session
- Goal: generate accurate relative preference judgments between results from different queries within a chain of query reformulations (same information need)

Implicit relevance judgments

Joachims et al., 2007 [3]

- Extracting preference feedback from query chains:
  Click > Skip Earlier QC

  For a ranking $(l_1, l_2, \ldots.)$ followed by ranking $(l_1^{'}, l_2^{'}, \ldots.)$ (not necessarily immediately) within the same query chain and sets C and $C^{'}$ containing the ranks of the clicked on results, extract a preference example $rel(l_i^{'}) > rel(l_j)$ for all pairs $i \in C^{'}$ and $j < \max(C)$, with $j \notin C$.

  - Accuracy depends on the presentation order
    - ~85% (normal) vs. ~55% (reversed)

- .... more strategies exist ....

Implicit relevance judgments

Joachims et al., 2007 [3]

- Extracting preference feedback from qu
  Click > Skip Earlier QC

  For a ranking $(l_1, l_2, \ldots.)$ followed by r
  (not necessarily immediately) within
  chain and sets C and C' containing th
  on results, extract a preference examp
  for all pairs $i \in C'$ and $j < \max(C)$, wi

  - Accuracy depends on the presentation o
    - ~85% (normal) vs. ~55% (reversed)

- .... more strategies exist ....

$$q_1 : l_{11}\ l_{12}\ l_{13}\ l_{14}\ l_{15}\ l_{16}\ l_{17}$$
$$q_2 : l^*_{21}\ l_{22}\ l^*_{23}\ l_{24}\ l^*_{25}\ l_{26}\ l_{27}$$
$$q_3 : l_{31}\ l^*_{32}\ l_{33}\ l_{34}\ l_{35}\ l_{36}\ l_{37}$$
$$q_4 : l^*_{41}\ l_{42}\ l_{43}\ l_{44}\ l_{45}\ l_{46}\ l_{47}$$

$$rel(l_{32}) > rel(_{22})$$
$$rel(l_{32}) > rel(l_{24})$$
$$rel(l_{41}) > rel(l_{22})$$
$$rel(l_{41}) > rel(l_{24})$$
$$rel(l_{41}) > rel(l_{31})$$

Implicit relevance judgments

Joachims et al., 2007 [3]

- Limitations:
  - Query chain approach requires accurately segmented search session
  - Training data is not independently identically distributed (assumed by ML algorithms)
  - "The participants in our study were young, well educated, and internet savy search-engine users."
  - Additional implicit feedback is not (yet) taken into account
    - Timing information
    - Behavior on pages clicked on the result page
    - Click spam (adversarial users)

Summary

- You can do A LOT with query logs!

Sources

1) Query-level loss functions for information retrieval. Qin et al. 2008.
2) Discriminative models for information retrieval. Nallapati. 2004.
3) Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search. Joachims et al. 2007
4) Improving Web search ranking by incorporating user behavior information. Agichtein et al. 2006.
5) Analysis of a very large web search engine query log. Silverstein et al. 1999.
6) Hourly analysis of a very large topically categorized web query log. Beitzel et al. 2004.
7) Agglomerative clustering of a search engine query log. Beeferman et al. 2000.