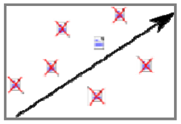
Modeling User Behavior and Interactions

Lecture 3: Improving Ranking with Behavior Data

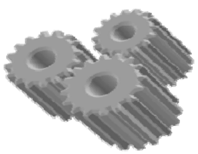
Eugene Agichtein

Emory University

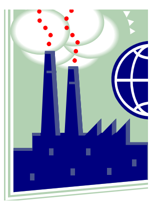
Lecture 3 Plan



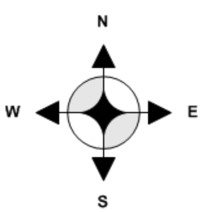
1. **Review: Learning to Rank**



2. **Exploiting User Behavior for Ranking:**
   - Automatic relevance labels
   - Enriching feature space



3. **Implementation and System Issues**
   - Dealing with Scale
   - Dealing with data sparseness



4. **New Directions**
   - Active learning
   - Ranking for diversity

Review: Learning to Rank

- Goal: instead of **fixed** retrieval models learn them:
  - Usually: **supervised** learning on **document/query** pairs **embedded in high-dimensional** feature space
  - **Labeled** by relevance of document to query
  - **Features**: provided by IR methods.
- Given training instances:
  - $(x_{q,d}, y_{q,d})$ for q = {1..N}, d = {1 .. N_q}
- Learn a ranking function
  - $f(x_{q,1}, \ldots x_{q,Nq})$

Ordinal Regression Approaches

- **Learn multiple thresholds:**

  Maintain T thresholds (b_1, … b_T), $b_1 < b_2 < \ldots < b_T$ => Learn parameters + (b_1, …, b_T)

  Chu & Keerthi, New Approaches to Support Vector Ordinal Regression ICML 05

- **Learn multiple classifiers:**

  Use T different training sets, train classifiers $C_1..C_T$ => Sum

  T. Qin et al., "Ranking with Multiple Hyperplanes." SIGIR 2007

- **Optimize pairwise preferences:**

  RankNet: Burges et al., Learning to Rank Using Gradient Descent, ICML 05

- **Optimize Rank-based Measures:**

  Directly optimize (n)DCG via local approximation of gradient

  LambdaRank: C. Burges, et al., "Learning to Rank with Non-Smooth Cost Functions." NIPS 2006

Learning to Rank Summary

- **Many** learning algorithms available to choose from
- Require training data (feature vectors + labels)
- **Where does training data come from?**
  - "Expert" human judges (TREC, editors, …)
  - Users: logs of user behavior
- **Rest of this lecture:**
  - Learning formulation and setup, to **train** and **use** learning to rank algorithms

Approaches to Use Behavior Data

- Use “clicks” as **new training examples**
  – Joachims, KDD 2002
  – Radlinski & Joachims, KDD 2005
- Incorporate behavior data as **additional features**
  – Richardson et al., WWW 2005
  – Agichtein et al., SIGIR 2006
  – Bilenko and White, WWW 2008
  – Zhu and Mishne, KDD 2009

Recap: Available Behavior Data

- Queries: queries sent to the search engine.
- Clicks on results: what was clicked.
- Dwell time: When clicks & queries happened.
- Browser buttons, printing, bookmarking, mousing: User interactions with web browser.
- Reformulations: The whole sequence of user actions.

Training Examples from Click Data

[Joachims 2002]

Assumption: If a user skips a link *a* and clicks on a link *b* ranked lower, then the user preference reflects $rank(b) < rank(a)$.

Example: $(3 < 2)$ *and* $(7 < 2)$, $(7 < 4)$, $(7 < 5)$, $(7 < 6)$

Ranking Presented to User:

1. Kernel Machines
   http://svm.first.gmd.de/
2. Support Vector Machine
   http://jbolivar.freeservers.com/
3. SVM-Light Support Vector Machine
   http://ais.gmd.de/~thorsten/svm light/
4. An Introduction to Support Vector Machines
   http://www.support-vector.net/
5. Support Vector Machine and Kernel ... References
   http://svm.research.bell-labs.com/SVMrefs.html
6. Archives of SUPPORT-VECTOR-MACHINES ...
   http://www.jiscmail.ac.uk/lists/SUPPORT...
7. Lucent Technologies: SVM demo applet
   http://svm.research.bell-labs.com/SVT/SVMsvt.html
8. Royal Holloway Support Vector Machine
   http://svm.dcs.rhbnc.ac.uk/

Loss Function

[Joachims 2002]

For two orderings r_a and r_b, a pair $d_i \neq d_j$ is

- *concordant*, if r_a and r_b agree in their ordering
  P = number of concordant pairs
- *discordant*, if r_a and r_b disagree in their ordering
  Q = number of discordant pairs

Loss function: [Kemeny & Snell, 62], [Wong et al, 88], [Cohen et al, 1999], [Crammer & Singer, 01], [Herbrich et al., 98] ...

$$l(r_a, r_b) = Q$$

Example:

$$r_a = (a, c, d, b, e, f, g, h)$$

$$r_b = (a, b, c, d, e, f, g, h)$$

=> discordant pairs (c,b), (d,b) => $l(r_a, r_b) = 2$

Learned Retrieval Function

[Joachims 2002]

Sort documents d_i by their "retrieval status value" $\mathrm{rsv}(q, d_i)$ with query q [Fuhr, 89]:

$$
\begin{aligned}
\mathrm{rsv}(q, d_i) = \quad & w_1 * \#(\text{of query words in title of } d_i) \\
& + w_2 * \#(\text{of query words in H1 headlines of } d_i) \\
& \ldots \\
& + w_N * \mathrm{PageRank}(d_i) \\
= \; & \vec{w}\, \Phi(q, d_i).
\end{aligned}
$$

Select F as:

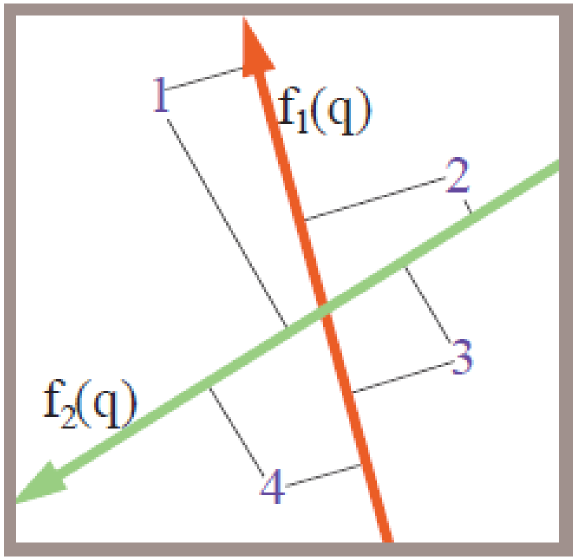
$$d_i > d_j$$

$$\Leftrightarrow$$

$$(d_i, d_j) \in f_{\vec{w}}(q)$$

$$\Leftrightarrow$$

$$\vec{w}\Phi(q, d_i) > \vec{w}\Phi(q, d_j)$$



Features

[Joachims 2002]

Query/Content Match:

- cosine between URL-words and query
- cosine between title-words and query
- query contains domain-name

Popularity-Attributes:

- length of URL in characters
- country code of URL
- domain of URL
- word "home" appears in title
- URL contains "tilde"
- URL as an atom

Results

[Joachims 2002]

weight	feature
0.60	cosine between query and abstract
0.48	ranked in top 10 from Google
0.24	cosine between query and the words in the URL
0.24	document was ranked at rank 1 by exactly one of the 5 search engines
...	
0.17	country code of URL is ".de"
0.16	ranked top 1 by HotBot
...	
-0.15	country code of URL is ".fi"
-0.17	length of URL in characters
-0.32	not ranked in top 10 by any of the 5 search engines
-0.38	not ranked top 1 by any of the 5 search engines

Extension: Query Chains

[Radlinski & Joachims, KDD 2005]

There is extra information in query reformulations.

Rank	Website
1	Ithaca Tompkins Regional Airport - Home
2	Ithaca Tompkins Regional Airport - Schedule
3	Ithaca Airport, Greece: Travel information

"ithaca airport"

↓

Rank	Website
1	Cornell Remote Sensing | Airline Service
2	Cheap Flights Ithaca - Discount Airfares
3	Ithaca Tompkins Regional Airport - The ...

"ithaca airline"

↓

Rank	Website
1	Cornell Remote Sensing | Airline Service
2	14850 Today: News for Ithaca, New York
3	14850 Today: Independence Air to serve ...

"new ithaca airline"

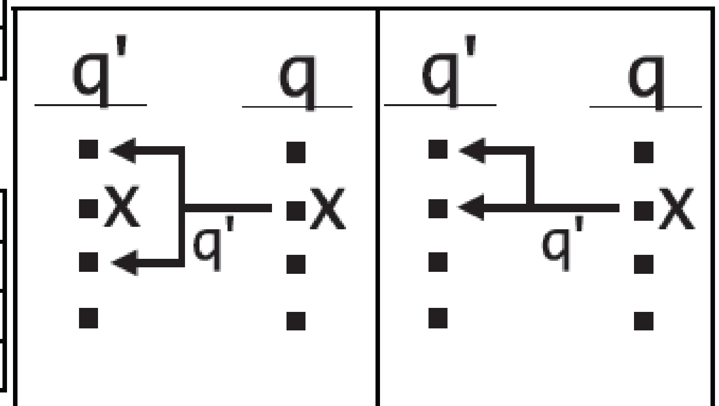
Query Chains (Cont'd)

[Radlinski & Joachims, KDD 2005]

Rank	Website
1	Ithaca Tompkins Regional Airport - Home
2	Ithaca Tompkins Regional Airport - Schedule
3	Ithaca Airport, Greece: Travel information

Rank	Website
1	Cornell Remote Sensing | Airline Service
2	Cheap Flights Ithaca - Discount Airfares
3	Ithaca Tompkins Regional Airport - The ...

Rank	Website
1	Cornell Remote Sensing | Airline Service
2	14850 Today: News for Ithaca, New York
3	14850 Today: Independence Air to serve ...



68.0 ± 8.4%	84.5 ± 6.1%

Inter-Judge Agreement: 86.4%

Baseline: 50.0%

Query Chains (Results)

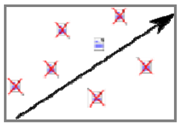
[Radlinski & Joachims, KDD 2005]

- Query Chains add slight improvement over clicks

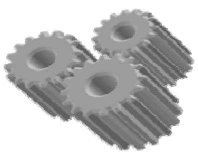
Evaluation Mode	User Prefers		
	Chains	Other	Indifferent
rel_{QC} vs. rel_0	392 (32%)	239 (20%)	579 (47%)
rel_{QC} vs. rel_{NC}	211 (17%)	160 (13%)	855 (70%)

Table 3: Results on Cornell Library search engine. rel_0 is the original retrieval function, rel_{QC} is that trained using query chains, and rel_{NC} is that trained without using query chains.

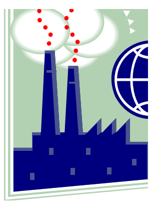
Lecture 3 Plan



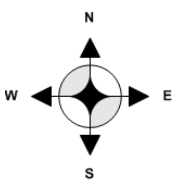
✓ **Review: Learning to Rank**



✓ **Exploiting User Behavior for Ranking:**
- ✓ Automatic relevance labels
- ➢ Enriching the ranking feature space



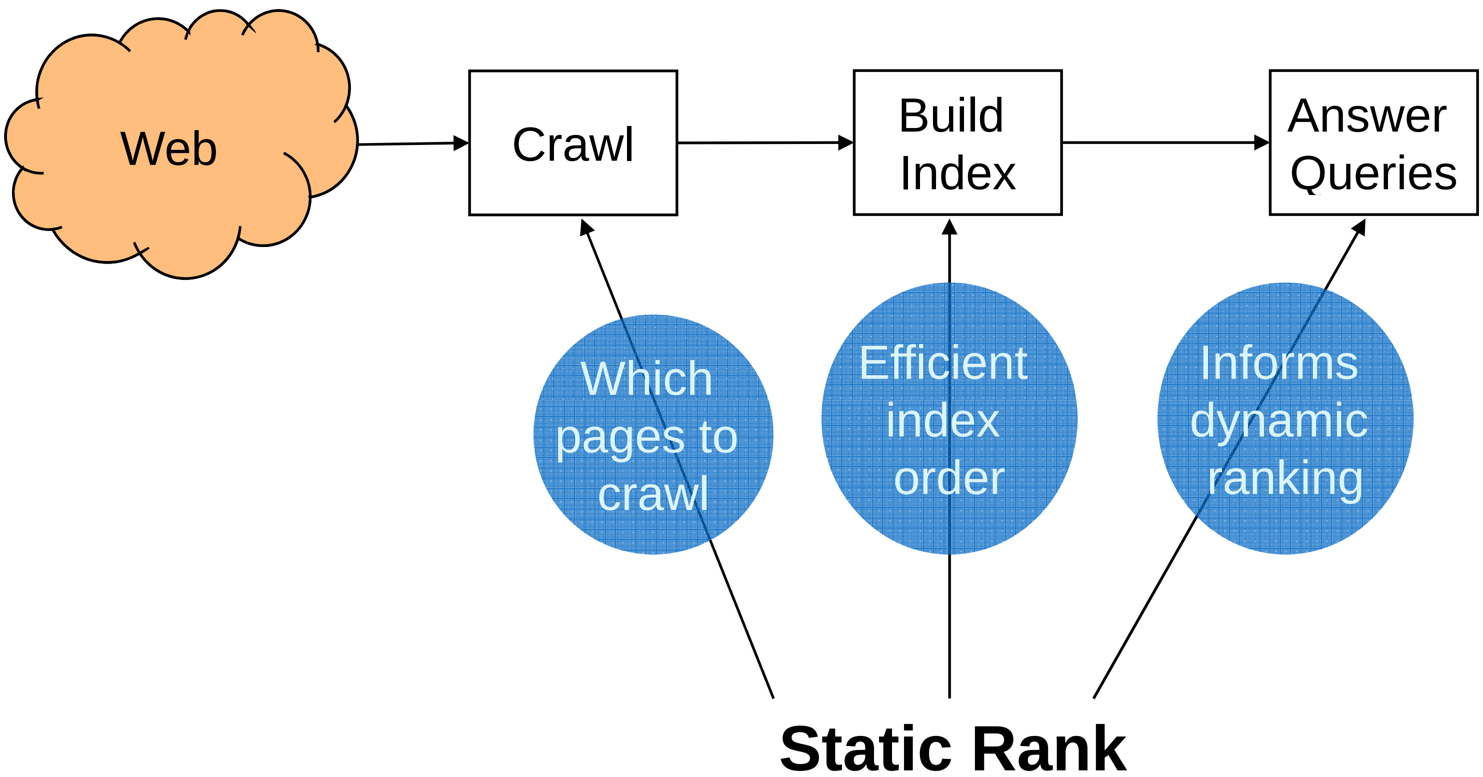
1. **Implementation and System Issues**
   – Dealing with Scale
   – Dealing with data sparseness



2. **New Directions**
   – Active learning
   – Ranking for diversity
   – Fun and games

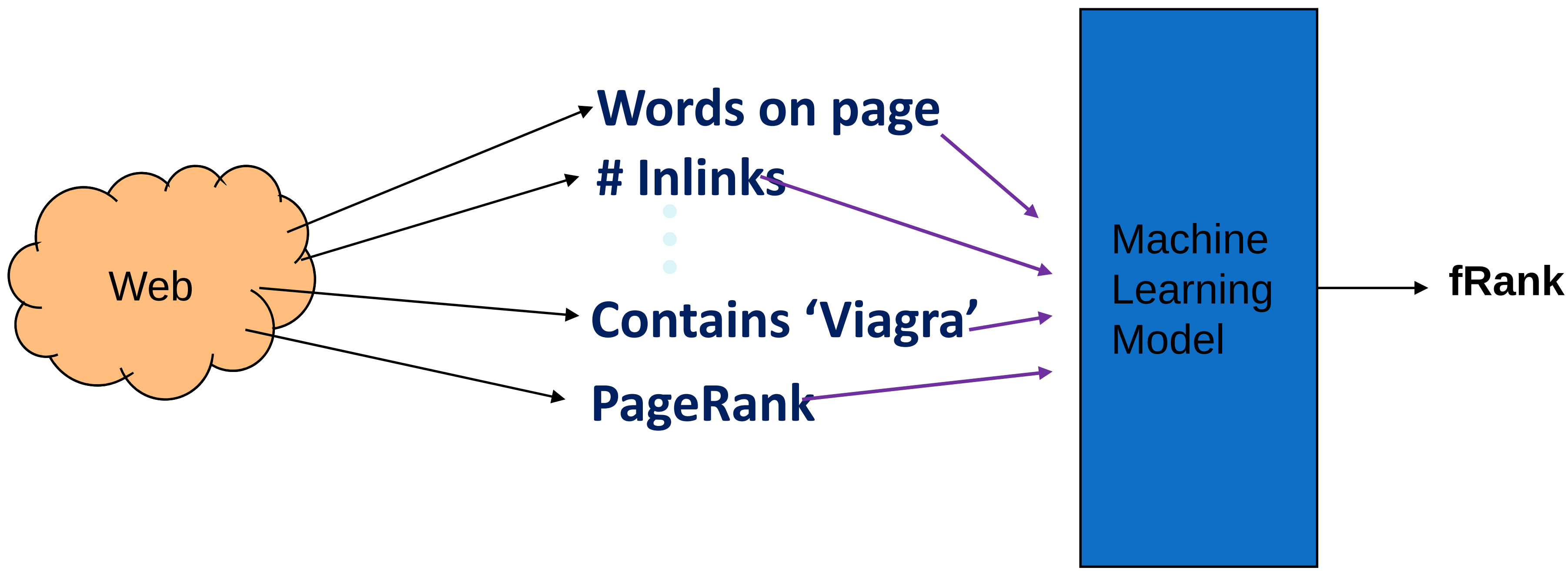
Incorporating Behavior for Static Rank

[Richardson et al., WWW2006]



fRank: Machine Learning for Static Ranking

[Richardson et al., WWW2006]



Features: Summary

[Richardson et al., WWW2006]

- **Popularity**
- Anchor text and inlinks
- Page
- Domain
- PageRank

Features: Popularity

[Richardson et al., WWW2006]

- Data from MSN Toolbar
- Smoothed

Function	**Example**
Exact URL	cnn.com/2005/tech/wikipedia.html?v=mobile
No Params	cnn.com/2005/tech/wikipedia.html
Page	wikipedia.html
URL-1	cnn.com/2005/tech
URL-2	cnn.com/2005
…	
Domain	cnn.com
Domain+1	cnn.com/2005
…	

Features: Anchor, Page, Domain

[Richardson et al., WWW2006]

- Anchor text and inlinks
  - Total amount of anchor text, unique anchor text words, number of inlinks, etc.
- Page
  - 8 Features based on page alone: Words in body, frequency of most common term, etc.
- Domain
  - Averages in domain: average #outlinks, etc.

Data

[Richardson et al., WWW2006]

- Human judgments
  1. Randomly choose query from MSN users
  2. Chose top URLs by search engine
  3. Rate quality of URL for that query
- 500k (Query,URL,Rating) tuples
- Judged URLs biased to good pages
  - Results apply to index ordering, relevance
  - Crawl ordering requires unbiased sample

Becoming Query Independent

[Richardson et al., WWW2006]

- (Query,URL,Rating) $\rightarrow$ (URL,Rating)
- Take maximum rating for each URL
  - **Good page if relevant for at least one query**
- Queries are common $\rightarrow$ likely correct index order and relevance order

Measure

[Richardson et al., WWW2006]

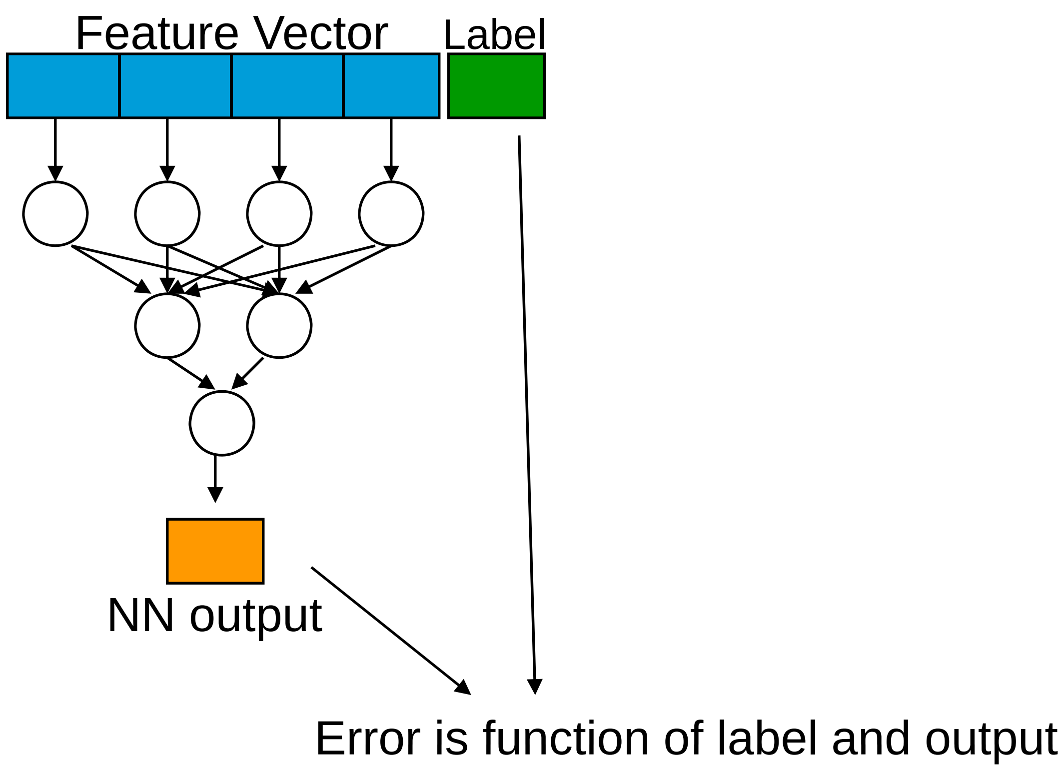
- Goal: Find static ranking algorithm that most correctly reproduces judged order

$$\text{pairwise accuracy} = \frac{|\mathbf{H_p} \cap \mathbf{S_p}|}{|\mathbf{H_p}|}$$

- Fraction of pairs that, when the humans claim one is better than the other, the static rank algorithm orders them correctly

RankNet, Burges et al., ICML 2005

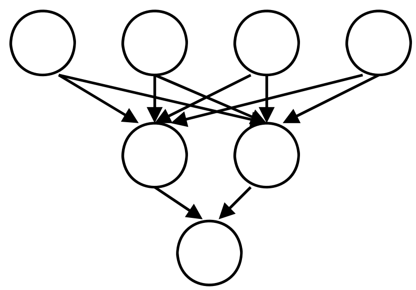
[Richardson et al., WWW2006]



RankNet [Burges et al. 2005]

[Richardson et al., WWW2006]

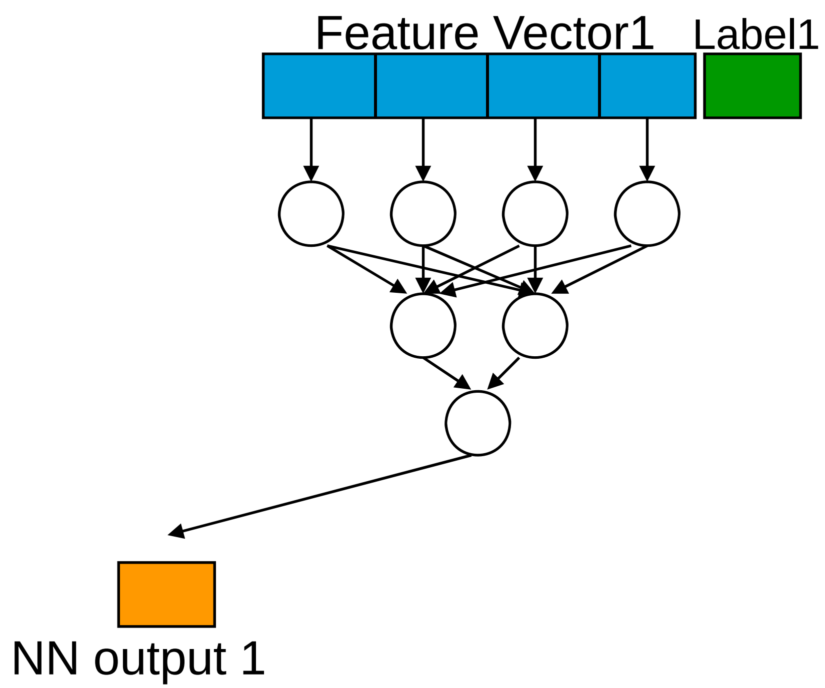
- Training Phase:
  - Present pair of vectors with label1 > label2



RankNet [Burges et al. 2005]

[Richardson et al., WWW2006]

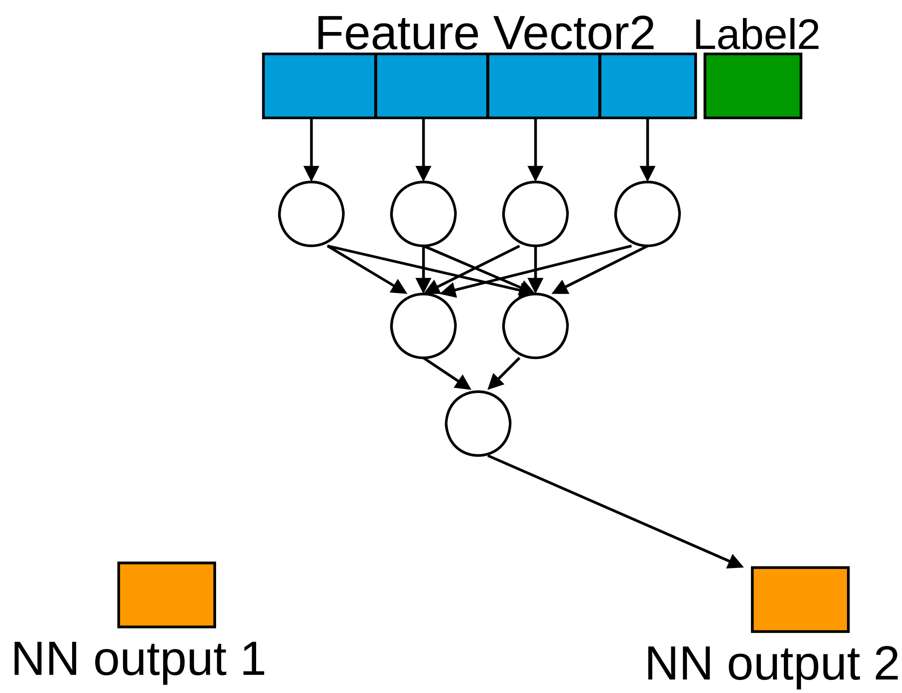
- Training Phase:
  - Present pair of vectors with label1 > label2



RankNet [Burges et al. 2005]

[Richardson et al., WWW2006]

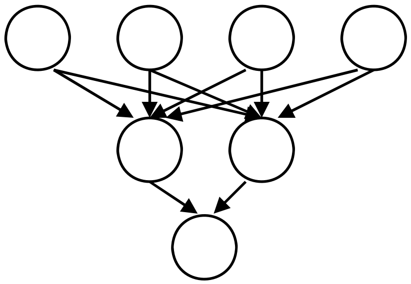
- Training Phase:
  – Present pair of vectors with label1 > label2

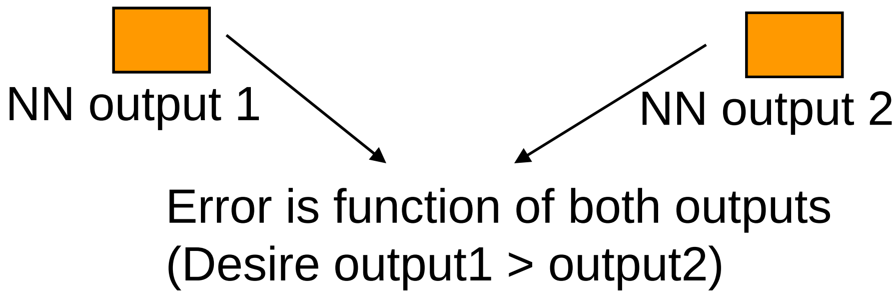


RankNet [Burges et al. 2005]

[Richardson et al., WWW2006]

- Training Phase:
  - Present pair of vectors with label1 > label2

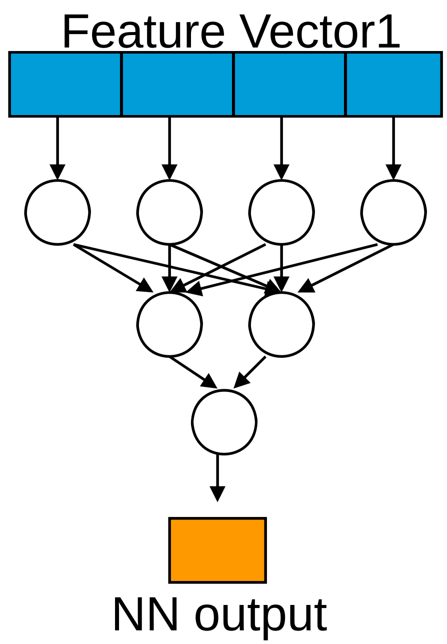




RankNet [Burges et al. 2005]

[Richardson et al., WWW2006]

- Test Phase:
  - Present individual vector and get score



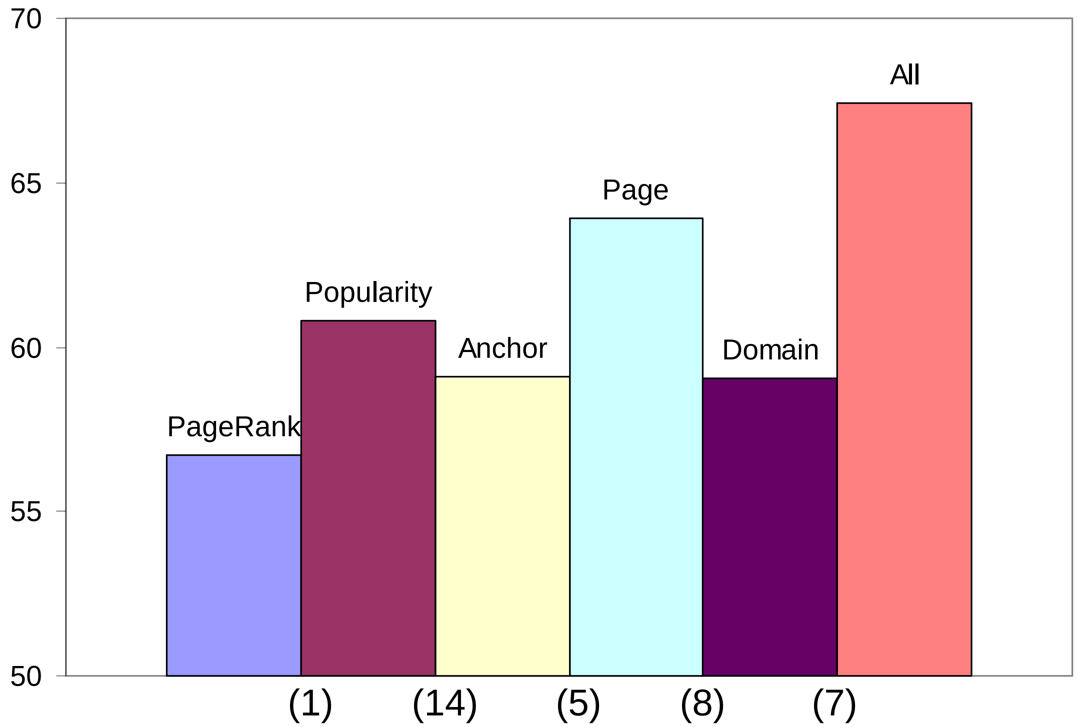
Experimental Methodology

[Richardson et al., WWW2006]

- Split ratings
  - – 84% training set
  - – 8% validation set
  - – 8% test set
- ➢ Training set: Train RankNet
- Validation set: Choose best net
- Test set: Measure pairwise accuracy

Accuracy of Each Feature Set

[Richardson et al., WWW2006]



Feature Set	*Accuracy (%)*
PageRank	56.70
Popularity	60.82
Anchor	59.09
Page	63.93
Domain	59.03
All Features	**67.43**

- Accuracy with only the given feature set
- Every feature set outperformed PageRank
- Best feature sets contain no link information

Qualitative Evaluation

[Richardson et al., WWW2006]

- Top ten URLs for PageRank vs. fRank

PageRank	*fRank*
google.com	google.com
apple.com/quicktime/download	yahoo.com
amazon.com	americanexpress.com
yahoo.com	hp.com
microsoft.com/windows/ie	target.com
apple.com/quicktime	bestbuy.com
mapquest.com	dell.com
ebay.com	autotrader.com
mozilla.org/products/firefox	dogpile.com
ftc.gov	bankofamerica.com

Technology Oriented   Consumer Oriented

Behavior for Dynamic Ranking

[Agichtein et al., SIGIR2006]

Presentation	
ResultPosition	Position of the URL in Current ranking
QueryTitleOverlap	Fraction of query terms in result Title
Clickthrough	
DeliberationTime	Seconds between query and first click
ClickFrequency	Fraction of all clicks landing on page
ClickDeviation	Deviation from expected click frequency
Browsing	
DwellTime	Result page dwell time
DwellTimeDeviation	Deviation from expected dwell time for query

Sample Behavior Features (from Lecture 2)

Feature Merging: Details

[Agichtein et al., SIGIR2006]

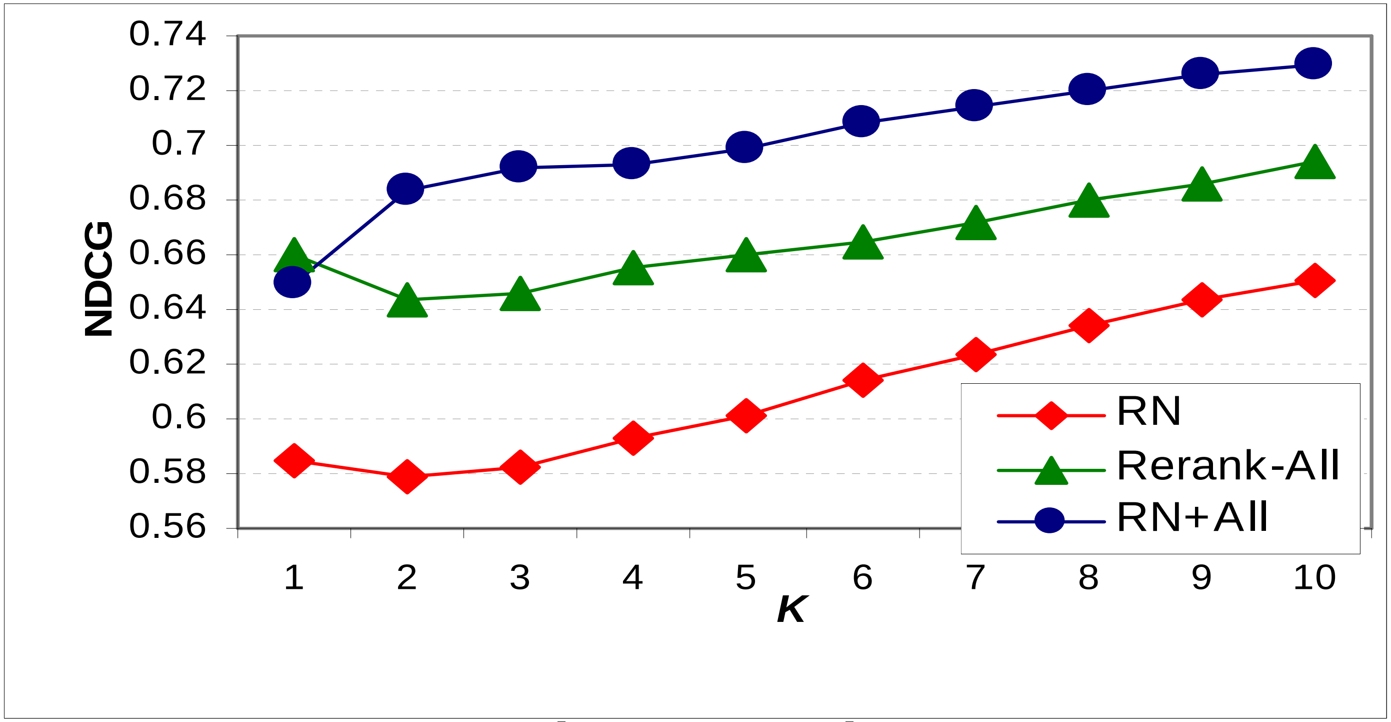
Query: SIGIR, fake results w/ fake feature values

Result URL	*BM25*	*PageRank*	...	*Clicks*	*DwellTime*	...
sigir2007.org	2.4	0.5	...	?	?	...
Sigir2006.org	1.4	1.1	...	150	145.2	...
acm.org/sigs/sigir/	1.2	2	...	60	23.5	...

- Value scaling:
  - Binning vs. log-linear vs. linear (e.g., μ=0, σ=1)
- Missing Values:
  - 0? (meaning for normalized feature values s.t. μ=0?)
- “real-time”: **significant** architecture/system problems

Results for Incorporating Behavior into Ranking

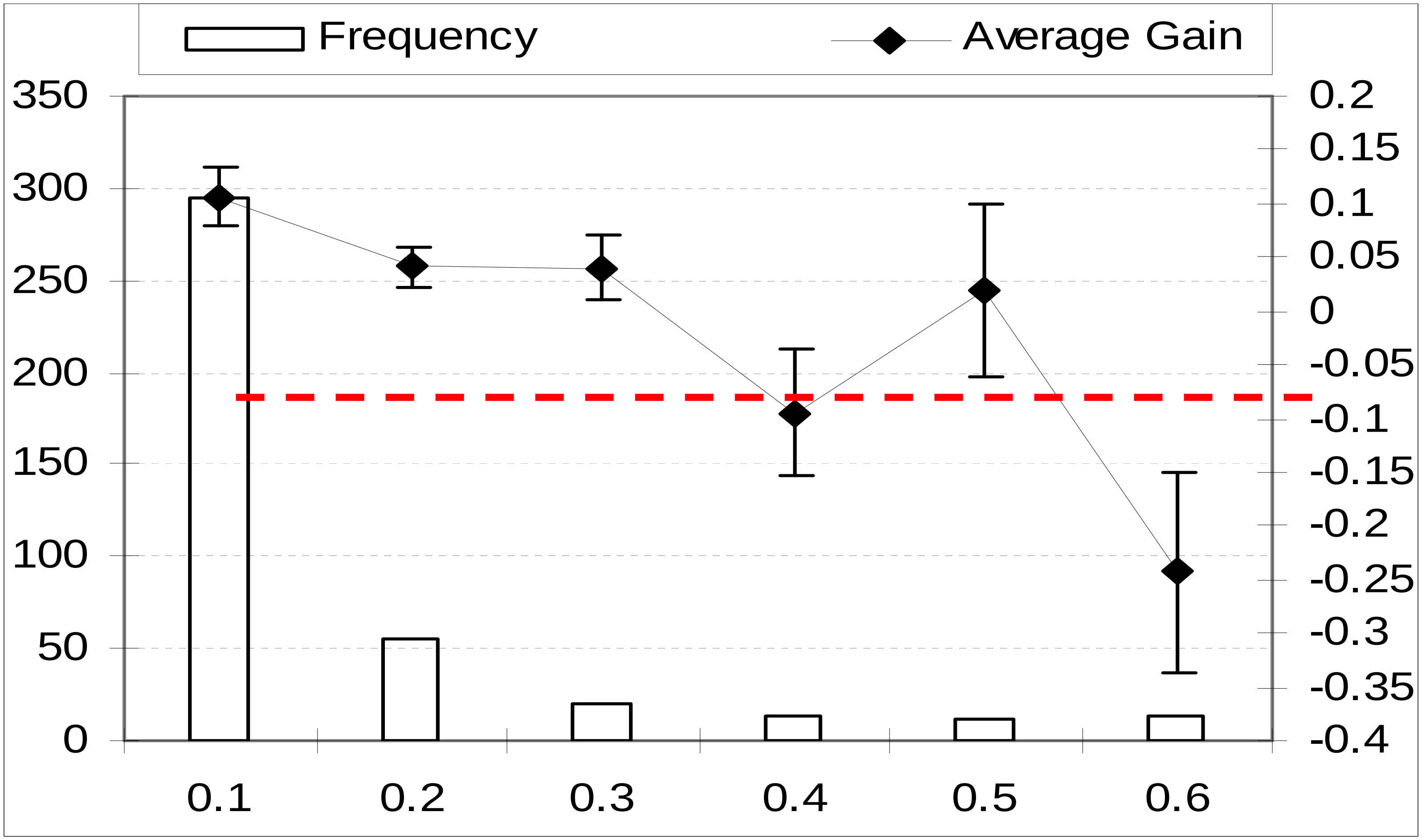
[Agichtein et al., SIGIR2006]



	MAP	Gain
RN	0.270	
RN+ALL	0.321	0.052 (19.13%)
BM25	0.236	
BM25+ALL	0.292	0.056 (23.71%)

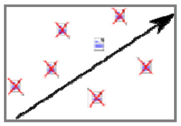
Which Queries Benefit Most

[Agichtein et al., SIGIR2006]

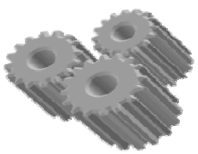


Most gains are for queries with poor original ranking

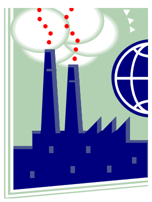
Lecture 3 Plan



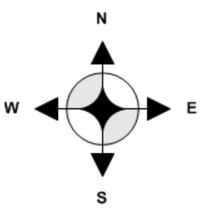
✓ **Review: Learning to Rank**



✓ **Exploiting User Behavior for Ranking:**
- ✓ Automatic relevance labels
- ✓ Enriching feature space



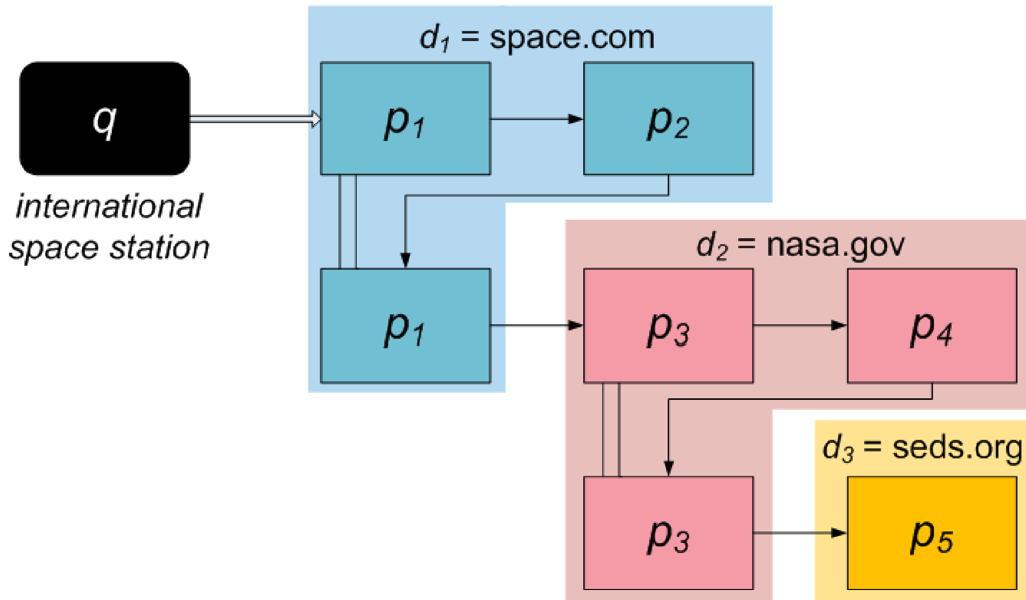
1. Implementation and System Issues
- ➢ Dealing with data sparseness
- – Dealing with Scale



2. New Directions
- – Active learning
- – Ranking for diversity
- – Fun and games

Extension to Unseen Queries/Documents: Search Trails

[Bilenko and White, WWW 2008]



- Trails start with a search engine query
- Continue until a terminating event
  - Another search
  - Visit to an unrelated site (social networks, webmail)
  - Timeout, browser homepage, browser closing

Probabilistic Model

[Bilenko and White, WWW 2008]

- IR via language modeling [Zhai-Lafferty, Lavrenko]

$$Rel(d_i, q) = p(d_i|q) = \sum_{t_j \in q} p(t_j|q)\, p(d_i|\, t_j)$$

- Query-term distribution gives more mass to rare terms:

$$p(t_j|q) = \frac{\exp\left(-p(t_j)\right)}{\sum_{t_k \in q} \exp\left(-p(t_k)\right)}$$

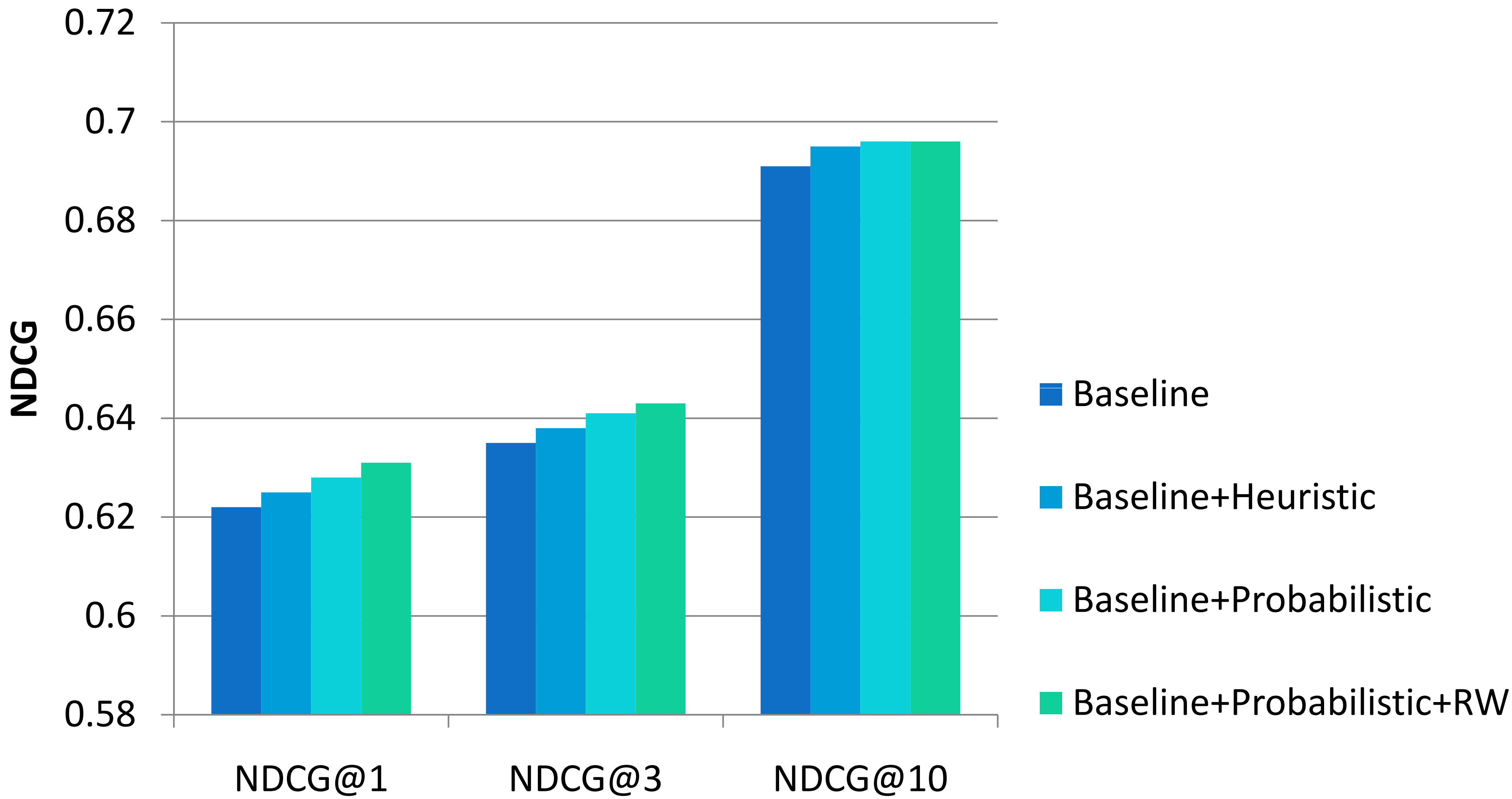
- Term-website weights *combine dwell time and counts*

$$f(d_i, t_j) = \sum_{\forall q' :\, t_j \in q';\, q' \to d_i} \log(time(q', d_i)) \qquad p(d_i|t_j) = \frac{f(d_i, t_j)}{\sum_{d_k \in D} f(d_k, t_j)}$$

Results: Learning to Rank

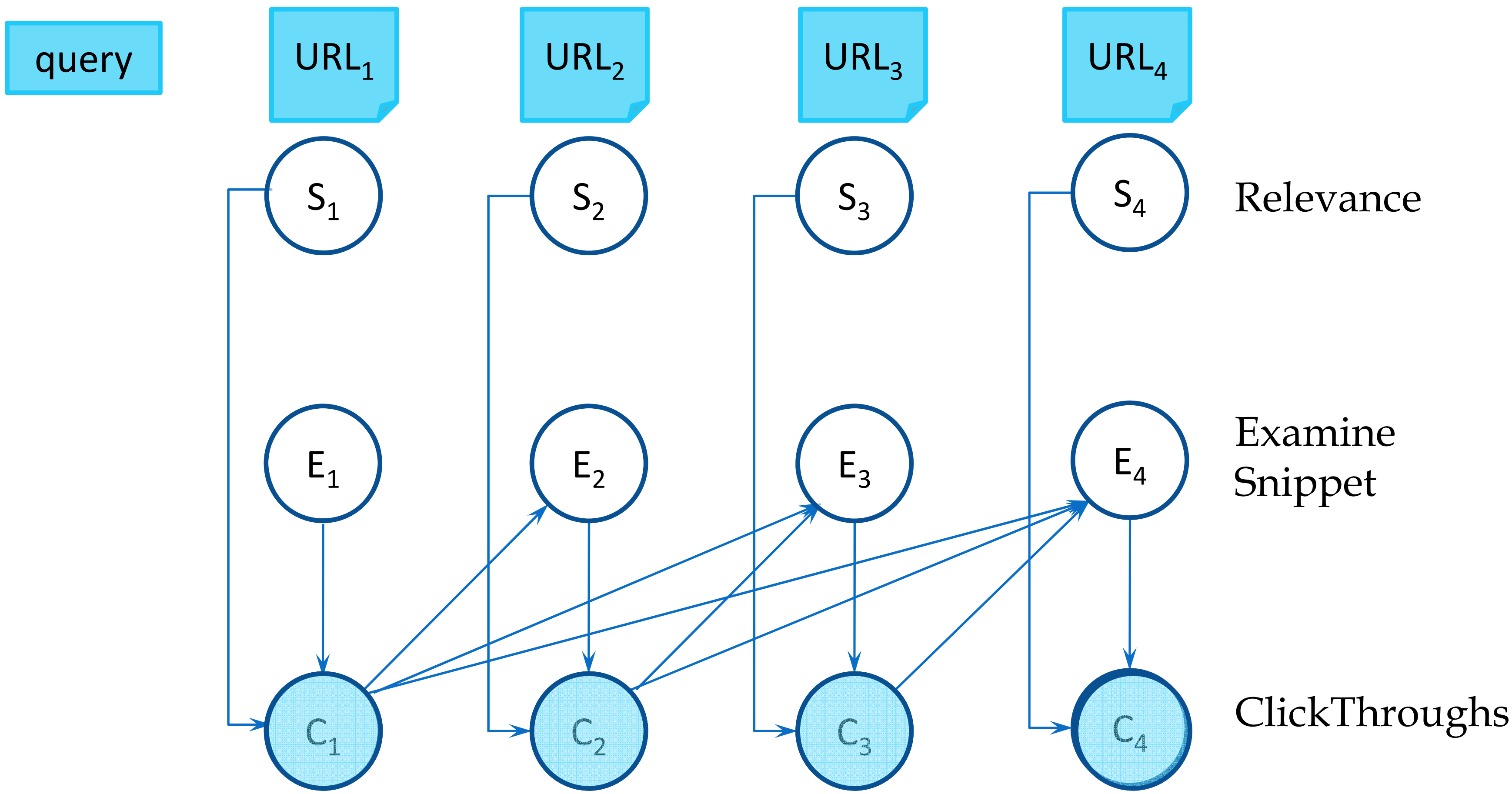
[Bilenko and White, WWW 2008]

Add *Rel*(*q*, d_i) as a feature to RankNet



Scalability: (Peta?)bytes of Click Data

BBM: Bayesian Browsing Model from Petabyte-scale Data, Liu et al, KDD 2009



BBM: Bayesian Browsing Model

Training BBM: One-Pass Counting

BBM: Bayesian Browsing Model from Petabyte-scale Data, Liu et al, KDD 2009

Algorithm 1 : LearnBBM($\mathbf{C}$, ϕ, $\mathcal{N}$)

Input: $\mathbf{C} = \{\mathbf{C}^1, \ldots, \mathbf{C}^n\}$: click data
$\phi = \{\phi_1, \ldots, \phi_n\}$: impression data

Output: $\mathcal{N} = \{N_j, \widetilde{N}_{j,r,d}\}$ for $j = 1, \ldots, N$ and $(r,d) \in \mathcal{T}$:
all the exponents in the posterior

initialize every value in $\mathcal{N}$ to 0;
for each search instance $k = 1, \ldots, n$
    initialize the preceding click position $r = 0$;
    for each position $i = 1, \ldots, M$
        set index for current document $\boxed{j = \phi_k(i);}$ Find R_j
        if $C_i^k == 1$
            N_j++;
            update the preceding click position $r = i$;
        else
            set the distance to the preceding click $d = i - r$;
            $\widetilde{N}_{j,r,d}++$;
        end
    end
end

Complexity: Time $O(nM)$, Space $O(NM^2)$.

$$p(\mathbf{R}|\mathbf{C}^{1:n}) \propto \prod_{j=1}^{N} R_j^{N_j} \prod_{(r,d)\in\mathcal{T}} \left(1 - \beta_{r,d} R_j\right)^{\widetilde{N}_{j,r,d}}$$

Training BBM on MapReduce

BBM: Bayesian Browsing Model from Petabyte-scale Data, Liu et al, KDD 2009

Algorithm 2 : `Map(I)` – Mapping a search instance I

Input: I: current search instance.
$I.qry$: returns the query,
$I.phi[i]$: gives the URL on the ith position,
$I.clk[i]$: indicates click on the ith position.
Output: $((q, u), val)$: intermediate (key, value) pairs for every position

$q = I.qry$; $r = 0$;
for each position $i = 1, \ldots, M$
  $u = I.phi(i)$;
  if $I.clk[i] == 1$
    $r = i$;
    $val = 0$;
  else
    $d = i - r$;
    $val = r(2M - r - 1)/2 + d$;
  end
  Emit$((q, u),\ val)$;
end

- Map: emit((q,u), idx)
- Reduce: construct the count vector

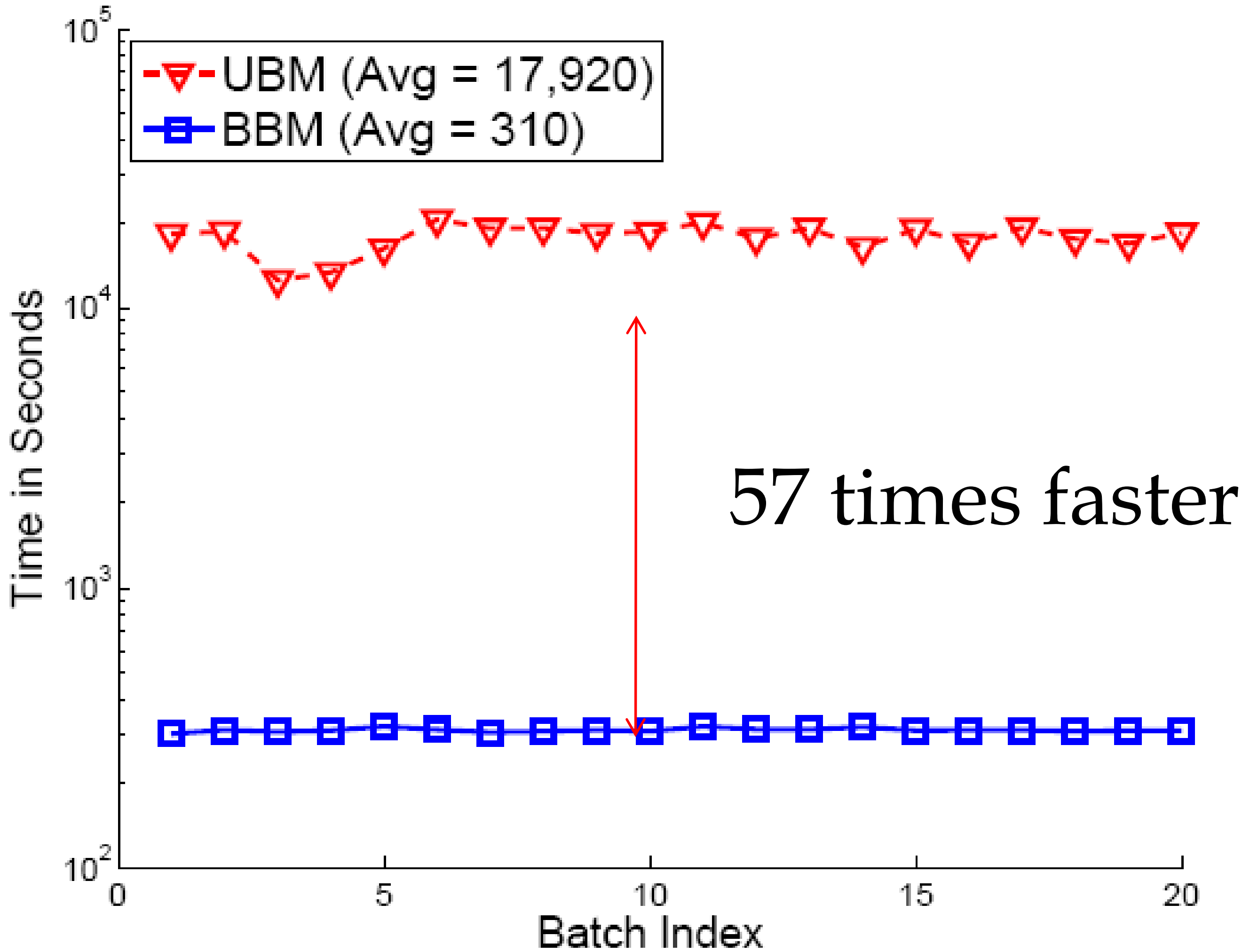
Algorithm 3 : `Reduce((q,u), valList)`

Input: (q, u): the intermediate key
$valList$: a list of values associated with (q, u)
Output: $((q, u), \mathbf{e})$: $\mathbf{e}$ is the exponent vector for (q, u)

$\mathbf{e} = \mathbf{0}$;
for each val in $valList$
  $\mathbf{e}[val] + +$
end
return $((q, u), \mathbf{e})$

Model Comparison on Efficiency

BBM: Bayesian Browsing Model from Petabyte-scale Data, Liu et al, KDD 2009



Large-Scale Experiment

BBM: Bayesian Browsing Model from Petabyte-scale Data, Liu et al, KDD 2009

Job Index	Input Size (TB)	# Query (10^6)	#Query-URL (10^6)
1	31.2	16.3	169.0
2	62.1	30.7	322.9
3	94.3	42.9	454.1
4	128.1	53.9	575.0
5	161.8	63.8	686.4
6	195.5	75.4	816.6
7	229.7	86.3	954.8
8	265.2	103.0	1,155.7

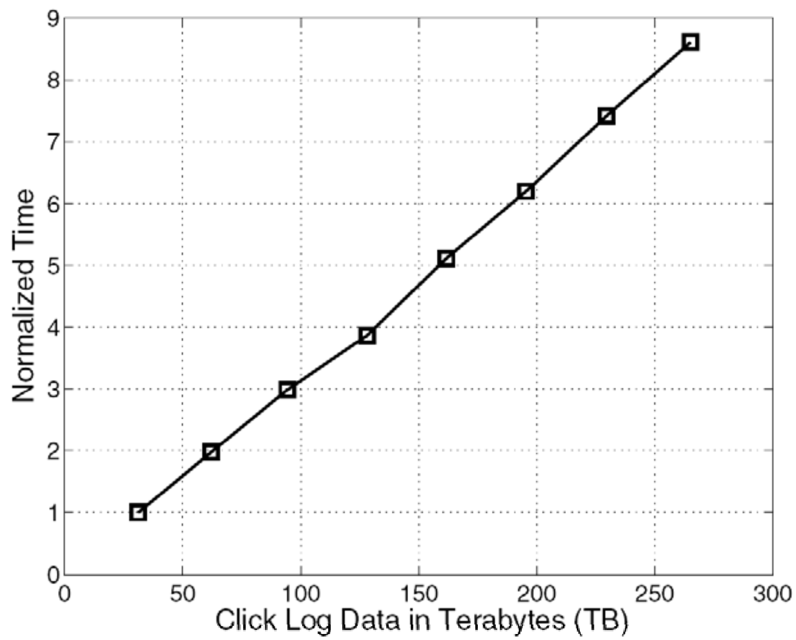
- Setup:
  - 8 weeks data, 8 jobs
  - Job k takes first k-week data

- Experiment platform
  - SCOPE: Easy and Efficient Parallel Processing of Massive Data Sets [Chaiken et al, VLDB'08]

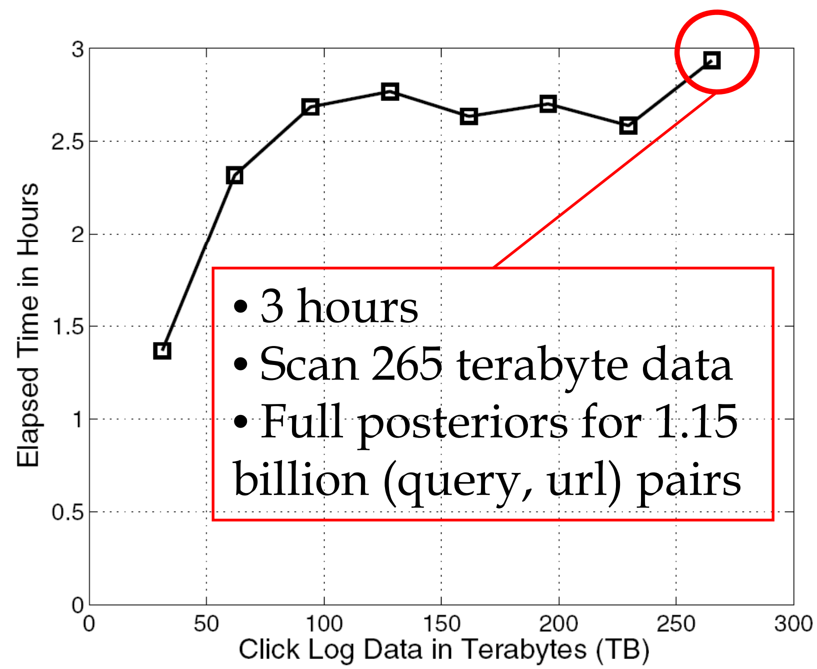
Scalability of BBM

BBM: Bayesian Browsing Model from Petabyte-scale Data, Liu et al, KDD 2009

- Increasing computation load
  - more queries, more URLs, more impressions
- Near-constant elapsed time

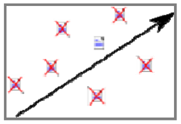


Computation Overload

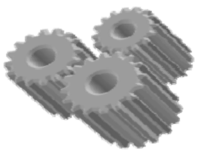


Elapse Time on SCOPE

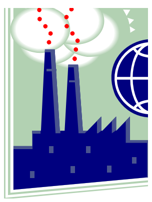
Lecture 3 Plan



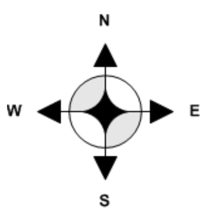
- ✓ **Review: Learning to Rank**



- ✓ **Exploiting User Behavior for Ranking:**
  - ✓ Automatic relevance labels
  - ✓ Enriching feature space



- ✓ **Implementation and System Issues**
  - ✓ Dealing with data sparseness
  - ✓ Dealing with Scale



- ➢ **New Directions**
  - ➢ Active learning
  - ➢ Ranking for diversity

New Direction: Active Learning

[Radlinski & Joachims, KDD 2007]

- Goal: Learn the relevances with as little training data as possible.
- Search involves a three step process:
  1. **Given relevance estimates, pick a ranking to display to users.**
  2. Given a ranking, users provide feedback: User clicks provide pairwise relevance judgments.
  3. Given feedback, update the relevance estimates.

Overview of Approach

[Radlinski & Joachims, KDD 2007]

- Available information:
  1. Have an estimate of the relevance of each result.
  2. Can obtain pairwise comparisons of the top few results.
  3. Do **not** have absolute relevance information.

- Goal: Learn the document relevance quickly.

- Will address four questions:
  1. How to represent knowledge about doc relevance.
  2. How to maintain this knowledge as we collect data.
  3. Given our knowledge, what is the best ranking?
  4. What rankings do we show users to get useful data?

1: Representing Document Relevance

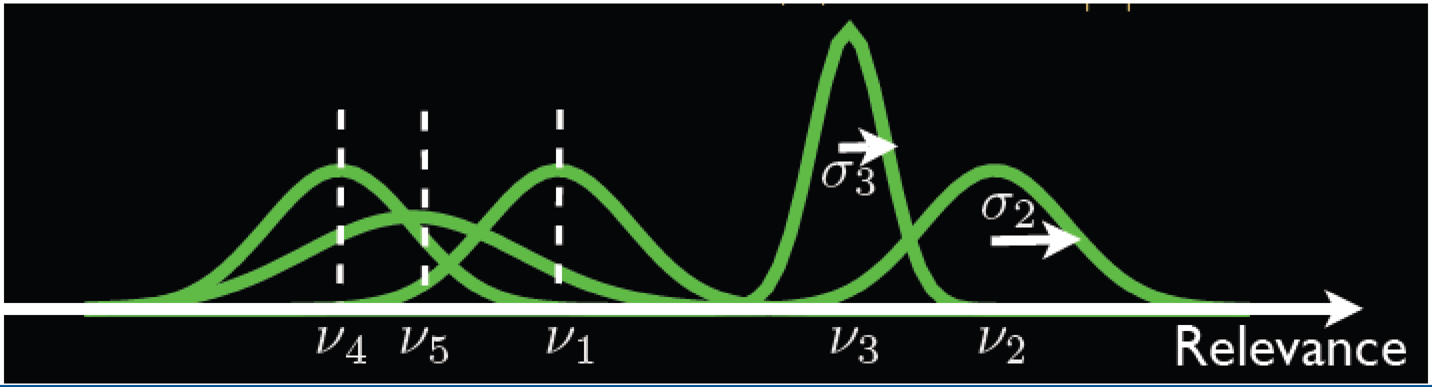
[Radlinski & Joachims, KDD 2007]

- Given a query, q , let $M^* = (\mu^*_1, \ldots, \mu^*_{|C|}) \in M$ be the true relevance values of the documents.
- Model knowledge of M^* with a Bayesian:

  $P(M|D) = P(D|M)\,P(M)/P(D)$

- Assume $P(M|D)$ is spherical multivariate normal:

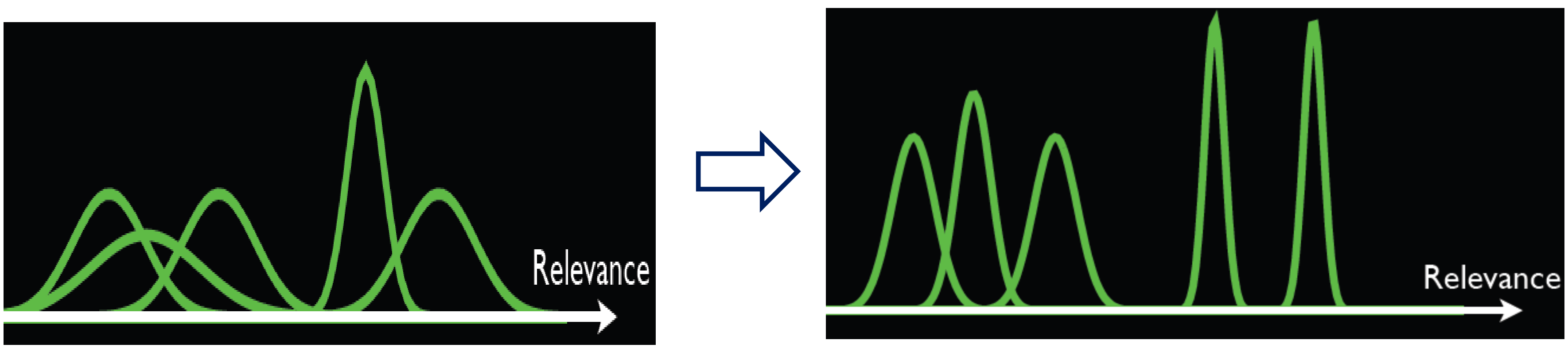
  $P(M|D) = N(\nu_1, \ldots, \nu_{|C|}; \sigma_1^2, \ldots, \sigma_{|C|}^2)$



1: Representing Document Relevance

[Radlinski & Joachims, KDD 2007]

- Given a fixed query, maintain knowledge about relevance as clicks are observed.
  - This tells us which documents we are sure about, and which ones need more data.



2: Maintaining P(M|D)

[Radlinski & Joachims, KDD 2007]

Model noisy pairwise judgments w [Bradley-Terry'52]

$$P(d_i \succ d_j) = \frac{rel(d_i)}{rel(d_i) + rel(d_j)}$$

Adding a Gaussian prior, apply off-the-shelf algorithm to maintain *Glicko Rating System*, commonly used for chess [Glickman 1999]

$$\nu_i \leftarrow \nu_i + \frac{q}{\frac{1}{\sigma_i^2} + \frac{1}{\delta^2}} g(\sigma_j^2)(s_i - E[s|\nu_i, \nu_j, \sigma_j^2])$$

$$\sigma_i^2 \leftarrow \left(\frac{1}{\sigma_i^2} + \frac{1}{\delta^2} \right)^{-1}$$

[Glickman, '99]

3: Ranking (Inference)

[Radlinski & Joachims, KDD 2007]

- Want to assign relevances $M = (\mu_1, \ldots, \mu_{|C|})$ such that $L(M, M^*)$ is small, but M^* is unknown.
- Minimize **expected** loss (pairwise):

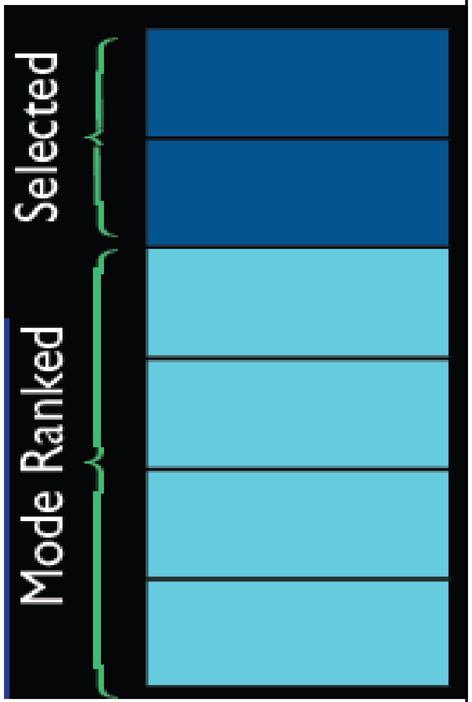
$$\sum_{i=1}^{|C|} \sum_{j=i+1}^{|C|} E_{M^* \sim P(M|\mathcal{D})} \left[\mathcal{L}^{pair}(M, M^*, i, j) \right]$$

4: Getting Useful Data

[Radlinski & Joachims, KDD 2007]

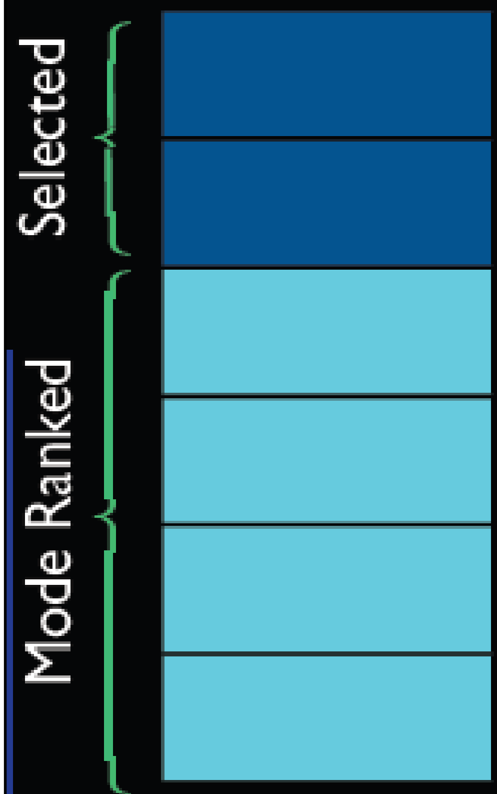
- **Problem**: could present the ranking based on **current** best estimate of relevance.
  – Then the data we get would always be about the documents already ranked highly.

- Instead, **optimize** ranking shown users:
  1. Pick top two docs to minimize **future loss**
  2. Append current best estimate ranking.



4: Exploration Strategies

[Radlinski & Joachims, KDD 2007]



Expected Loss:

$$\sum_{i=1}^{|\mathcal{C}|}\sum_{j=i+1}^{|\mathcal{C}|} E_{M^*\sim P(M|\mathcal{D})}\left[\mathcal{L}^{pair}(M,M^*,i,j)\right]$$

Strategies:

- **Passive:** Present the mode ranking.
- **Random:** Pick top two randomly.
- **Largest Expected Loss:** Select pair with largest contribution to the loss.
- **One Step Lookahead:** Select pair with largest expectation reduction in $\mathcal{L}^{pair}$

4: Loss Functions

[Radlinski & Joachims, KDD 2007]

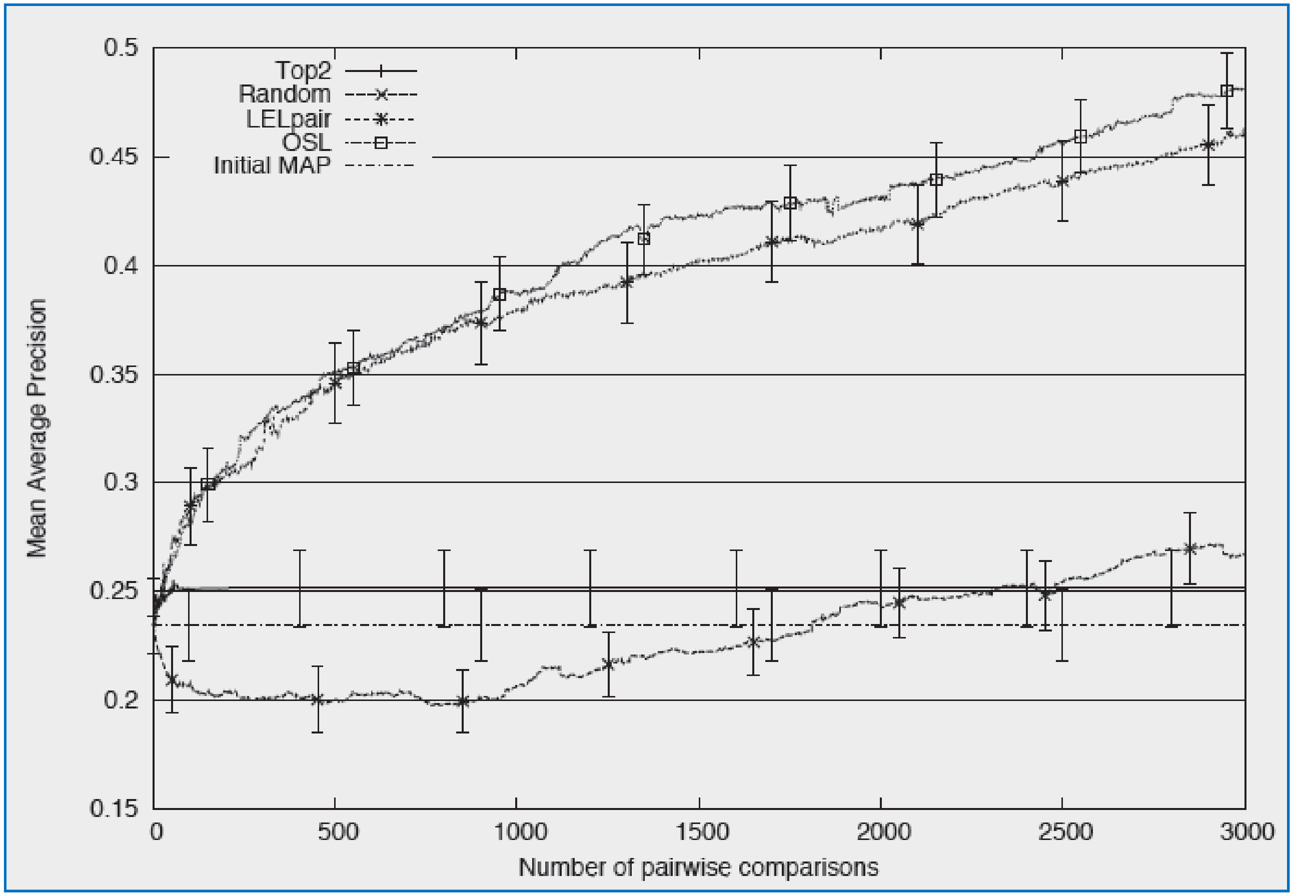
What loss function do we want to optimize for?

1. The loss for ranking a less relevant document above a more relevant document should be larger if the documents are presented higher.
2. The loss should be larger if error in relative relevance is larger.

$$\mathcal{L}^{pair} = \underbrace{e^{-r_{ij}}}_{(1)} \underbrace{\left((\mu_i - \mu_j) - (\mu_i^* - \mu_j^*)\right)^2}_{(2)} \underbrace{1_{misordered}}_{(\text{hinge; } 1)}$$

Results: TREC Data

[Radlinski & Joachims, KDD 2007]



Optimizing for ***relevance estimates*** better than for ***ordering***

Need for Diversity (in IR)

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]

- Ambiguous Queries
  - Users with different information needs issuing the same textual query (“Jaguar”)
- Informational (Exploratory) Queries:
  - User interested in “a specific detail or entire breadth of knowledge available” [Swaminathan et al., 2008]
  - Want results with high information diversity

Optimizing for Diversity

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]

- Long interest in IR community
- Requires **inter-document dependencies**

  → Impossible given current learning to rank methods

- Problem: no consensus on how to measure diversity.

  → Formulate as predicting diverse subsets

- Experiment:
  - Use training data with explicitly labeled subtopics (TREC 6-8 Interactive Track)
  - Use loss function to encode subtopic loss
  - Train using structural SVMs [Tsochantaridis et al., 2005]

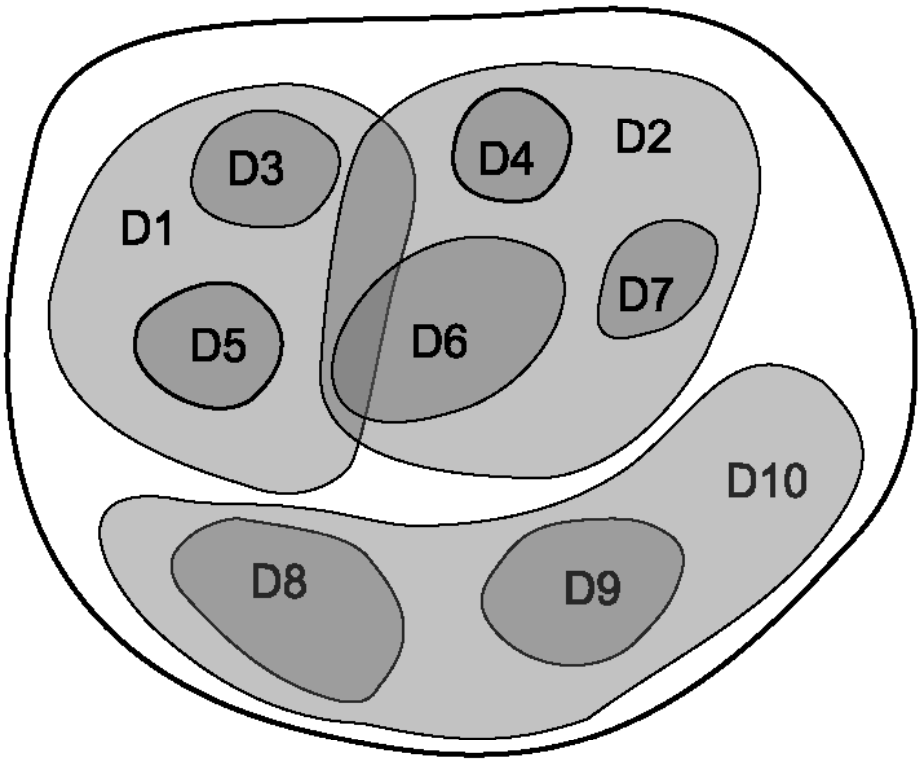
Representing Diversity

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]

- Existing datasets with manual subtopic labels
  - E.g., “Use of robots in the world today”
    - Nanorobots
    - Space mission robots
    - Underwater robots
  - Manual partitioning of the total information regarding a query
  - Relatively reliable

Example

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]



- Choose K documents with maximal information coverage.
- For K = 3, optimal set is {D1, D2, D10}

Maximizing Subtopic Coverage

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]

- **Goal:** select K documents which collectively cover as many subtopics as possible.
- Perfect selection takes n choose K time.
  - Set cover problem.
- Greedy gives (1-1/e)-approximation bound.
  - Special case of Max Coverage (Khuller et al, 1997)

Weighted Word Coverage

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]

- More distinct words = more information
  - Weight word importance
  - Does not depend on human labels
- **Goal:** select K documents which collectively cover as many distinct (weighted) words as possible
  - Greedy selection also yields (1-1/e) bound.
  - Need to find good weighting function (learning problem).

Example

[Predicting Diverse Subsets Using Structural SVMs, Y. Yue and Joachims, ICML 2008]

Document Word Counts

	V1	V2	V3	V4	V5
D1			X	X	X
D2		X		X	X
D3	X	X	X	X	

Word	Benefit
V1	1
V2	2
V3	3
V4	4
V5	5

Marginal Benefit

	D1	D2	D3	Best
Iter 1	12	11	10	D1
Iter 2				

Example (cont'd)

[Predicting Diverse Subsets Using Structural SVMs Y. Yue and Joachims, ICML 2008]

Document Word Counts

	V1	V2	V3	V4	V5
D1			X	X	X
D2		X		X	X
D3	**X**	**X**	X	X	

Word	Benefit
V1	1
V2	2
V3	3
V4	4
V5	5

Marginal Benefit

	D1	D2	D3	Best
Iter 1	12	11	10	D1
Iter 2	--	2	**3**	**D3**

Results: TREC data

[Predicting Diverse Subsets Using Structural SVMs Y. Yue and Joachims, ICML 2008]

- 12/4/1 train/valid/test split
  - Approx 500 documents in training set
- Permuted until all 17 queries were tested once
- Set K=5 (some queries have very few documents)
- SVM-div – uses term frequency thresholds to define importance levels
- SVM-div2 – in addition uses TFIDF thresholds

Results: TREC data

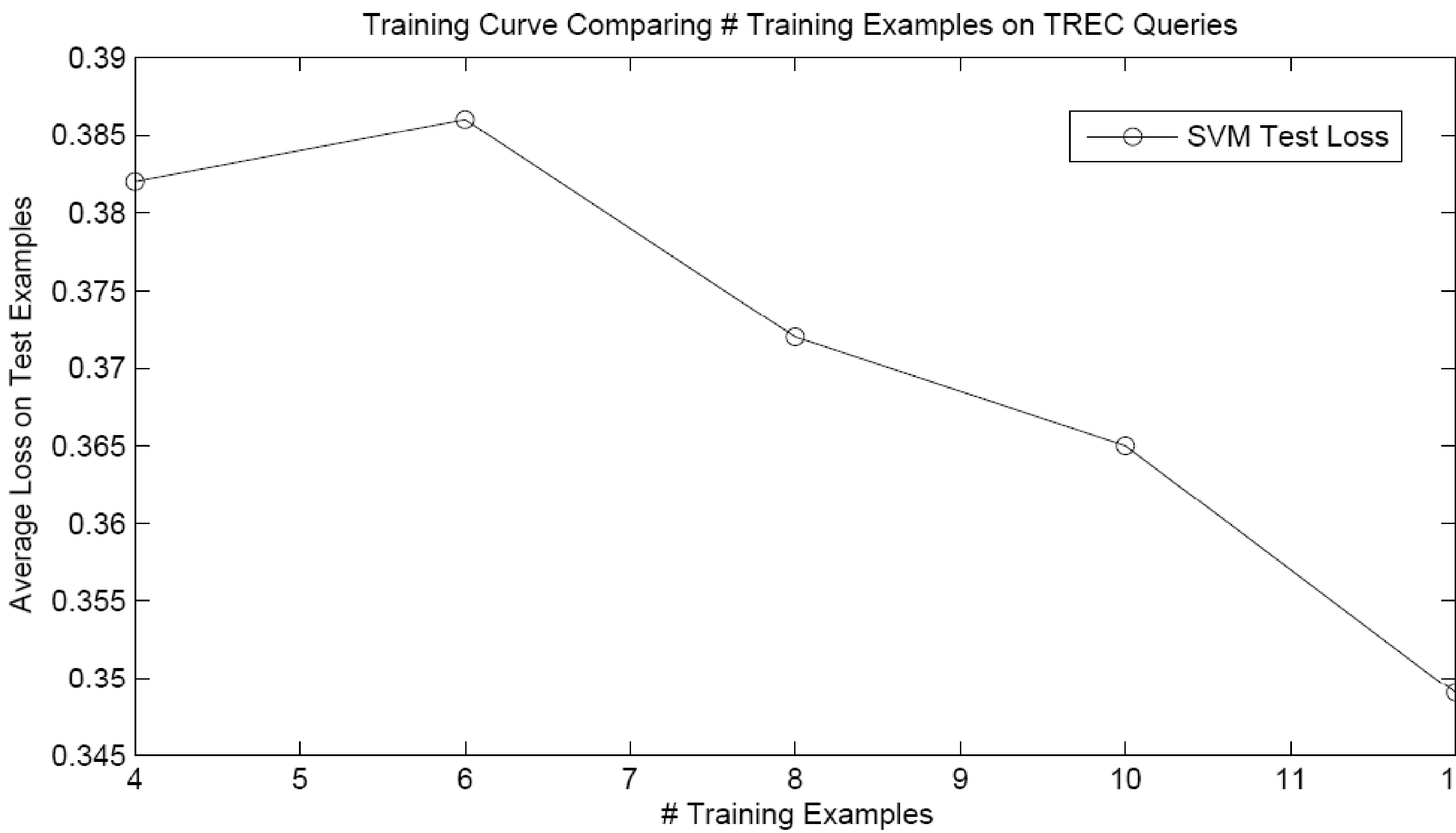
[Predicting Diverse Subsets Using Structural SVMs Y. Yue and Joachims, ICML 2008]

Method	Loss
Random	0.469
Okapi	0.472
Unweighted Model	0.471
Essential Pages	0.434
SVM-div	0.349
SVM-div2	0.382

Methods	W / T / L
SVM-div vs Ess. Pages	14 / 0 / 3 **
SVM-div2 vs Ess. Pages	13 / 0 / 4
SVM-div vs SVM-div2	9 / 6 / 2

Results: TREC data

[Predicting Diverse Subsets Using Structural SVMs Y. Yue and Joachims, ICML 2008]



Can expect further benefit from having more training data.

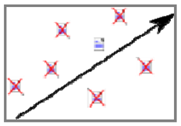
Summary

Predicting Diverse Subsets Using Structural SVMs Y. Yue and Joachims, ICML 2008

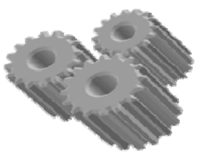
- Formulated diversified retrieval as predicting diverse subsets
  - Efficient training and prediction algorithms
- Used weighted word coverage as proxy to information coverage.
- Encode diversity criteria using loss function
  - Weighted subtopic loss

http://projects.yisongyue.com/svmdiv/

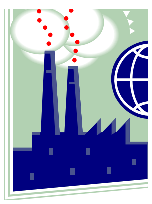
Lecture 3 Summary



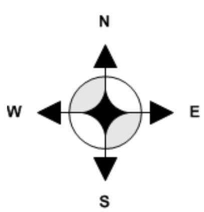
- ✓ **Review: Learning to Rank**



- ✓ **Exploiting User Behavior for Ranking:**
  - ✓ Automatic relevance labels
  - ✓ Enriching feature space



- ✓ **Implementation and System Issues**
  - ✓ Dealing with data sparseness
  - ✓ Dealing with Scale



- ✓ **New Directions**
  - ✓ Active learning
  - ✓ Ranking for diversity

Key References and Further Reading

Joachims, T. 2002. *Optimizing search engines using clickthrough data*, KDD 2002

Agichtein, E., Brill, E., Dumais, S. *Improving web search ranking by incorporating user behavior information*, SIGIR 2006

Radlinski, F. and Joachims, T. *Query chains: learning to rank from implicit feedback*, KDD 2005

Radlinski, F. and Joachims, T. *Active exploration for learning rankings from clickthrough data*, KDD 2007

Bilenko, M and White, R, *Mining the search trails of surfing crowds: identifying relevant websites from user activity.*, WWW 2008

Yue, Y and Joachims, *Predicting Diverse Subsets Using Structural SVMs*, ICML 2008