

LEARNING USER INTERACTION MODELS FOR PREDICTING WEB SEARCH PREFERENCES

Eugene Agichtein
Eric Brill
Susan Dumais
Robert Rango
Microsoft Research

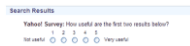
Jacob Bank and Christie Brandt

Predicting User Preferences

- Many successful supervised ranking methods...
- ...but they require labeled data
 - (e.g., pairwise preferences)

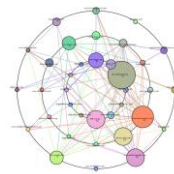
Problem: getting labeled data

- Explicit human ratings:
 - Expensive
 - Difficult to obtain
 - No effective way of getting explicit user feedback
- User interaction history:
 - “free” implicit feedback—millions each day
 - Click patterns, dwell time, mouse movement
- ...but how to model as pairwise preferences?



Implicitly Labeled Data

- Experiments with implicit ratings:
 - controlled text collections
 - selected queries/tasks
 - laboratory settings
- Real web:
 - Uncontrolled
 - Ill-defined queries/tasks
 - Automated bots
 - Noisy, non-expert users
 - Malicious
 - Irrational



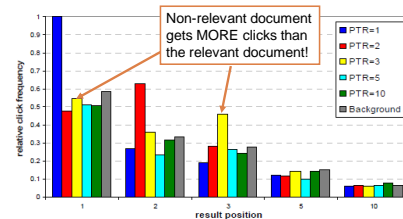
Main Questions

- Can explicitly accounting for “noisy” users provide more information?
- Can we automatically learn accurate user feedback interpretation models by representing user actions as a rich set of features?

Noisy Users

- Users click on non-relevant documents due to:
 - Visual appearance/ layout
 - User history/context
 - Presentation order (position)

(PTR: Position of Top Relevant Document)



Modeling Noisy Users

- 2 components to user behavior
- Relevance component
 - Query-specific user reaction
 - based on perceived true relevance of documents
- Background component:
 - Users clicking indiscriminately

q : query
 r : result
 p : position of r

$$clickthrough(q, r, p) = Expected p + relevance(q, r)$$

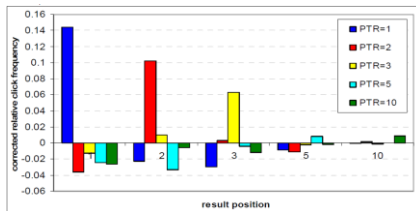
Calculating Background

- Calculate aggregated click frequency at position p :
 - Compute frequency of a click at p for each query q
 - (how often would a random click for query q land on p ?)
- Average frequencies across all queries

$$C(p) = \frac{1}{\#queries} \sum_{\forall queries q} \frac{\#clicks at p}{\#clicks in q}$$

Finding Relevance

- Find the expected behavior for each position over full dataset... and subtract it to get true relevance



Click Deviation

- Relevance: deviation from “expected behavior” at position p

r : result
 p : position

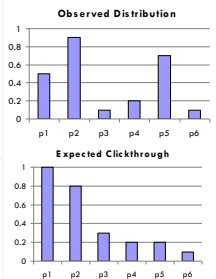
$$dev(r, p) = obs(r, p) - C(p)$$

Click deviation of result r at position p Observed click frequency at (r, p) Expected clickthrough at position p

Model 1: CD (Click Deviation)

- Filter out noisy clicks, then apply SA or SA+N strategies
 - For each result r_i at position p_i
 - Given a parameter d :
 - If $dev(r_i, p_i) > d$:
 - retain click as input for SA or SA+N strategies

Example: CD (Click Deviation)



Example: CD (Click Deviation)



- For a clicked result at position p :
- **SA (Skip Above):**
for all unclicked results $i < p$,
 $relevance(p) > relevance(i)$.
 $rel(d2) > rel(d1)$ $rel(d5) > rel(d4)$
 $rel(d5) > rel(d3)$ $rel(d5) > rel(d1)$
- **N (Skip Next):**
if the result $p+1$ is unclicked,
 $relevance(p) > relevance(p+1)$
 $rel(d2) > rel(d3)$ $rel(d5) > rel(d6)$
- **SA+N:**
combine both strategies

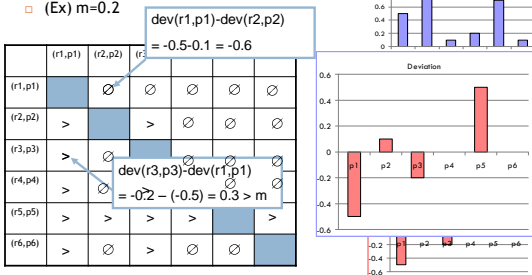
Model 2: Cdiff (Click Difference)

- Idea: when two results are compared, a result is “skipped” if it is clicked less than expected, “clicked” if more than expected.
- For each query q , calculate the deviation for each result-position pair
 - Compare every (r,p) -pair against every other:

$$dev(r_i, p_i) - dev(r_j, p_j) > m \Rightarrow rel(r_i) > rel(r_j)$$
- Ignores positional information
- Can compare events when both results clicked
 - Informational versus navigational queries

Example: Cdiff (Click Difference)

$dev(r_i, p_i) - dev(r_j, p_j) > m \Rightarrow rel(r_i) > rel(r_j)$
 □ (Ex) $m=0.2$



Precision/Recall Parameters

- d and m : tradeoff between precision and recall:
 - d, m large: higher precision, lower recall
 - d, m small: lower precision, higher recall

$$dev(r_i, p_i) > d$$

$$dev(r_i, p_i) - dev(r_j, p_j) > m$$

Beyond Clickthrough: General User Behavior Model

- Large set of features to represent user behavior *before* and *after* the click
- Automatically derive implicit feedback interpretation

Background: Richer Feature Set

- Time users spent reading Usenet news articles predicts user interest [Morita and Shinoda 1994]
- Page activity correlates with reader interest [Goecks and Shavlik 1999]
 - (small sample size, no testing against explicit measurements)
- Curious Browser—combined implicit measurements with explicit queries [Claypool et al. 2001]
 - Time spent on page + scrolling correlated with interest
 - Individual scrolling/mouse-clicks not correlated
- Rich (but query-independent) feature set: clickthrough most important, but adding dwell time improved accuracy [Fox et al. 2005]

General User Behavior Model

- Represent user actions as features—rich feature set
- Query-specific model (behavior deviates with query)
- Capture actions before and after query
 - ▣ Observed : relate directly to query/result pair
 - ▣ Distributional: deviations from “expected” behavior
 - Derived—measure deviation of feature for given search result from expected value for any result.

User Behavior Model

$$obs(q, r, f) = C(f) + rel(q, r, f)$$

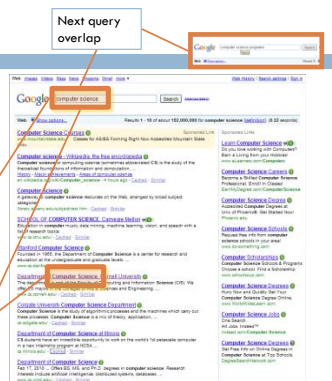
f : feature
 r : result
 q : query

Observed value of a feature with respect to result r and query q background Relevance-dependent component of behavior

(Observed feature values averaged across all search sessions and users for each query-result pair)

Features

- Query-text: text-based relations between query and document



- Title overlap, Summary overlap, Domain overlap

Features

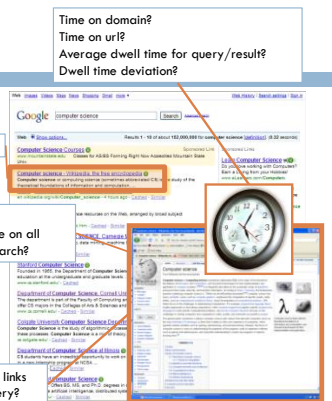
- Query-text: text-based relations between query and document
- Clickthrough: frequency, timing, order of clicks



- Are there any clicks before or after?

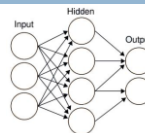
Features

- Query-text: text-based relations between query and document
- Clickthrough: frequency, order of clicks
- Browsing: user behavior after click (intra-query diversity of page browsing)



Learning User Behavior

- RankNet
 - ▣ Efficient
 - ▣ Scalable
 - ▣ Robust



- Train on pairs (r1, r2)
 - ▣ output: 1 if r1>r2, 0 otherwise
- Explicit boolean relevance judgments
- Gradient descent (multiple restarts) to set weights

Evaluation Metrics

- Evaluate based on pairwise agreement
- Query precision:

$$\frac{\#(\text{predicted} = \text{human judgement})}{\#(\text{predicted})}$$

- Query recall:

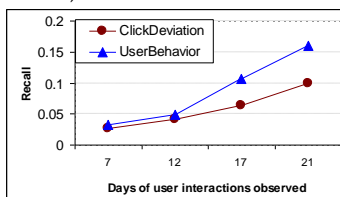
$$\frac{\#(\text{predicted} = \text{human judgement})}{\#(\text{human judgement})}$$

Strategies Compared

- Current:
 - a "state-of-the-art" ranking system from "a major websearch engine"
- SA
- SA+N
- CD
- CDiff
- CDiff+CD
- UserBehavior

Results: adding more data

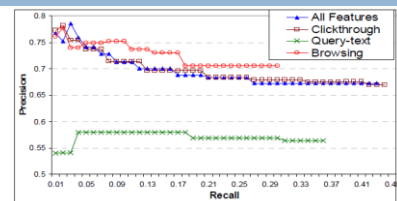
- Intelligent aggregation of large amounts of data improves precision (higher recall permitted)



Datasets

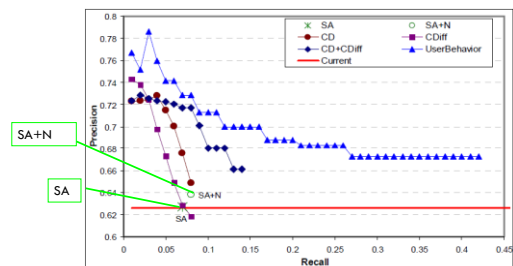
- "Orders of magnitude larger than any study yet reported in the literature"
- Explicit pairwise relevance judgements for top-10 results
 - Q1: at least 1 click for each query
 - (3500 queries, 28,093 query-URL pairs)
 - Q10: at least 10 clicks
 - (1300 queries, 18,728 query-URL pairs)
 - Q20: at least 20 clicks
 - (1000 queries, 12,922 query-URL pairs)
- Training/test for UB: train/validate on 75%, test on 25% (no query overlap)

Results: User Behavior Model Features

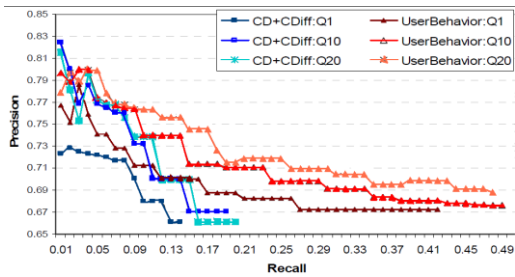


- Browsing features outperform combinations
- Query-text features by themselves perform badly

Results: Q1 (at least 1 click)



Results



Conclusions

- Explicitly accounting for “noisy” user behavior greatly improves accuracy
- New model presented which represents user actions as a rich set of features based on actions before and after search
- More extensive feature-based characterization of user behavior: dramatic improvement in accuracy over human-defined heuristics

Extensions?

- Targeting divergent access patterns (clustering)
- Modeling time-dependency of query distributions
- Automatically finding “reliable users”

Questions?