Learning User Interaction Models for Predicting Web Search Result Preferences

Eugene Agichtein  Eric Brill  Susan Dumais  Robert Ragno

Presented by
JiaLei Wang  Seok Hyun Jin
Introduction

● Traditional approach to ranking for web search
   ○ Features that describe a candidate page
   ○ Supervised learning methods
   ○ Dependent on explicit relevance

● Use *implicit relevance feedback*
   ○ Clickthrough data
   ○ Scroll time
   ○ Reading time

● How can we model user’s behavior? Which implicit features correlate to explicit ratings?

● Given implicit feedback, how can we effectively use them to produce reliable preference?
Introduction:
Limitations of Existing Methods

● Don’t make extensive use of *implicit feedback*
  ○ Clickthrough, dwell time
  ○ Cheap and abundant
● Don’t necessarily generalize well for real-world web search
  ○ Web search is not controlled
    ■ “Users” may act irrationally, maliciously or may not even be human
    ● Not all users are “experts”
Introduction:
How can we address these limitations?

- How can we model user behavior? Which implicit features correlate to explicit ratings?
- Given implicit features, how can we effectively use them to determine preference?

- Use of a distributional model of user behavior
  - Aggregated behavior of large number of users
  - Allows self-correct for noise

- Extension of strategies to include richer set of features
  - Partial to more descriptive model of user behavior
    - Pre and Post-search user behavior
Learning User Behavior Model

- As we noted earlier, real web search user behavior can be "noisy".
- Hence, instead of treating each user as a reliable "expert", we use statistics to infer relevance information from many unreliable data of user inputs.
- Approach: Model user web search behavior as:

  \[
  \text{relevance information} + \text{background noise} = \text{user behavior}
  \]
Figure 3.1: Relative click frequency for top 30 result positions over 3,500 queries and 120,000 searches.
Learning User Behavior Model: Case study in click distribution

Figure 3.2: Relative click frequency for queries with varying PTR (Position of Top Relevant document).
Learning User Behavior Model

- Activity:
  - How do you interpret relevance result from previous distribution?

Figure 3.3: Relative corrected click frequency for relevant documents with varying PTR (Position of Top Relevant).
Learning User Behavior Model: Robust user behavior model

- Post-search activities are comprised of clicks, page dwell time, clicks from search, etc.
- We have just shown how the ‘relevance-driven’ click distribution can be recovered from the biased observed distribution.
- We conjecture that for other aspects of user behavior, we can do something similar. Observed value $o$ of a feature $f$ for query $q$ and result $r$ can be expressed as
  \[ o(q,r,f) = C(r,f) + \text{rel}(q,r,f) \]
- where $C(r,f)$ is the ‘background’ distribution
## Learning User Behavior Model:

### Features representing user behavior

<table>
<thead>
<tr>
<th>Query-text features</th>
<th>Derived feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>TitleOverlap</td>
<td>Fraction of shared words between query and title</td>
</tr>
<tr>
<td>SummaryOverlap</td>
<td>Fraction of shared words between query and summary</td>
</tr>
<tr>
<td>QueryURLOverlap</td>
<td>Fraction of shared words between query and URL</td>
</tr>
<tr>
<td>QueryDomainOverlap</td>
<td>Fraction of shared words between query and domain</td>
</tr>
<tr>
<td>QueryLength</td>
<td>Number of tokens in query</td>
</tr>
<tr>
<td>QueryNextOverlap</td>
<td>Average fraction of words shared with next query</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Browsing features</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeOnPage</td>
<td>Page dwell time</td>
</tr>
<tr>
<td>CumulativeTimeOnPage</td>
<td>Cumulative time for all subsequent pages after search</td>
</tr>
<tr>
<td>TimeOnDomain</td>
<td>Cumulative dwell time for this domain</td>
</tr>
<tr>
<td>TimeOnShortUrl</td>
<td>Cumulative time on URL prefix, dropping parameters</td>
</tr>
<tr>
<td>IsFollowedLink</td>
<td>1 if followed link to result, 0 otherwise</td>
</tr>
<tr>
<td>IsExactUrlMatch</td>
<td>0 if aggressive normalization used, 1 otherwise</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Derived feature</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IsRedirected</td>
<td>1 if initial URL same as final URL, 0 otherwise</td>
</tr>
<tr>
<td>IsPathFromSearch</td>
<td>1 if only followed links after query, 0 otherwise</td>
</tr>
<tr>
<td>ClicksFromSearch</td>
<td>Number of hops to reach page from query</td>
</tr>
<tr>
<td>AverageDwellTime</td>
<td>Average time on page for this query</td>
</tr>
<tr>
<td>DwellTimeDeviation</td>
<td>Deviation from overall average dwell time on page</td>
</tr>
<tr>
<td>CumulativeDeviation</td>
<td>Deviation from average cumulative time on page</td>
</tr>
<tr>
<td>DomainDeviation</td>
<td>Deviation from average time on domain</td>
</tr>
<tr>
<td>ShortURLDeviation</td>
<td>Deviation from average time on short URL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clickthrough features</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Position of the URL in Current ranking</td>
</tr>
<tr>
<td>ClickFrequency</td>
<td>Number of clicks for this query, URL pair</td>
</tr>
<tr>
<td>ClickRelativeFrequency</td>
<td>Relative frequency of a click for this query and URL</td>
</tr>
<tr>
<td>ClickDeviation</td>
<td>Deviation from expected click frequency</td>
</tr>
<tr>
<td>IsNextClicked</td>
<td>1 if there is a click on next position, 0 otherwise</td>
</tr>
<tr>
<td>IsPreviousClicked</td>
<td>1 if there is a click on previous position, 0 otherwise</td>
</tr>
<tr>
<td>IsClickAbove</td>
<td>1 if there is a click above, 0 otherwise</td>
</tr>
<tr>
<td>IsClickBelow</td>
<td>1 if there is click below, 0 otherwise</td>
</tr>
</tbody>
</table>
Learning User Behavior Model: Learning a predictive behavior model

- Instead of heuristics or insights, we use supervised learning to map features to user preferences.
  - Advantage: We can always mine more data instead of relying on intuition and limited lab evidence.
- Training data: query/URL pair, explicit label by expert.
- Training method: RankNet (Burges et al. 2005)
  - Scalable neural net training
  - Pairwise preference
  - Use gradient descent to rank
Predicting User Preferences:
Baseline Model

- Baseline Model ("current")
  - A state-of-the-art page ranking system currently used by a major web search engine.
  - The algorithm ranks results based on hundreds of features such as query to document similarity, query to anchor text similarity, and intrinsic page quality.
Predicting User Preferences: Clickthrough Model

- Clickthrough Model (Joachims et al. 2007)
  - Strategy SA (Skip Above):

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 1. | Kernel Machines  
   | [http://svm.first.gmd.de/](http://svm.first.gmd.de/) |
| 2. | Support Vector Machine  
| 3. | SVM-Light Support Vector Machine  
| 4. | An Introduction to Support Vector Machines  
| 5. | Support Vector Machine and Kernel ... References  
| 6. | Archives of SUPPORT-VECTOR-MACHINES ...  
   | [http://www.jiscmail.ac.uk/lists/SUPPORT...](http://www.jiscmail.ac.uk/lists/SUPPORT...) |
| 7. | Lucent Technologies: SVM demo applet  
| 8. | Royal Holloway Support Vector Machine  
   | [http://svm.dcs.rhbnc.ac.uk](http://svm.dcs.rhbnc.ac.uk) |
Predicting User Preferences: Clickthrough Model

- Clickthrough Model (Joachims et al. 2007)
  - Strategy SA+N (Skip Above + Skip Next):

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
Predicting User Preferences: Clickthrough Model

- Clickthrough Model with filtering
  - Strategy CD (deviation $d$): Given query, compute observed click frequency distribution $o(r,p)$
    \[ \text{dev}(r,p) = o(r,p) - C(p) \]
  - If $\text{dev}(r,p) > d$, retain the click as input to SA+N strategy

$\text{SA + N}$

$\text{SA + N}$
Predicting User Preferences: Clickthrough and General User Model

- Clickthrough Model with filtering
  - Strategy CDiff(margin m): For each pair of results \( r_i, r_j \) predict preference of \( r_i \) over \( r_j \) iff
    - \( \text{dev}(r_i, p_i) - \text{dev}(r_j, p_j) > m \)
  - Strategy CD + CDiff (deviation d, margin m): CDiff and CD are complimentary. CDiff is a generalization of the clickthrough frequency model of CD, while ignoring the positional information used in CD.

- General User Behavior Model
  - User Behavior Strategy: Supervised learning model based on direct & derived features described in previous slide.
Experimental Setup: Methods Compared and Datasets

- **Methods compared:**
  - Current
  - SA
  - CD
  - UserBehavior
  - SA+N
  - CDiff
  - CD+CDiff

- **3500 queries randomly sampled**
  - Top 10 results for each query manually rated by experts
  - Defined 3 subsets
    - **Q1**: Queries with *at least 1 click* (3500 queries)
    - **Q10**: Queries with *at least 10 clicks* (1300 queries)
    - **Q20**: Queries with *at least 20 clicks* (1000 queries)
Experimental Setup:
Evaluation Methodology and Metrics

- Evaluation based on pairwise agreement with explicit
- Query Precision \((q) = \frac{\#\{\text{pref} : \text{pref} \in \text{prediction}(q) \land \text{pref} \in \text{explicit}\}}{\#\text{prediction}(q)}\)
  - Fraction of pairs predicted that agree with human ratings
- Query Recall \((q) = \frac{\#\{\text{pref} : \text{pref} \in \text{prediction}(q) \land \text{pref} \in \text{explicit}\}}{\#\text{explicit}}\)
  - Fraction of human-rated preferences predicted correctly
- Average Query Precision/Recall for evaluation
Experimental Setup:
More on Metrics

Deviation: $\text{dev}(r, p) > d$

Margin: $\text{dev}(r_i, p_i) - \text{dev}(r_j, p_j) > m$

d and m as tradeoff between Query Precision and Recall

Activity 2:
What effect will changing d and m (both increase/decrease) have on query precision and query recall? Why?

- Query Precision(q) = $\frac{\#\{\text{pref} : \text{pref} \in \text{prediction}(q) \land \text{pref} \in \text{explicit}\}}{\#\text{prediction}(q)}$

- Query Recall(q) = $\frac{\#\{\text{pref} : \text{pref} \in \text{prediction}(q) \land \text{pref} \in \text{explicit}\}}{\#\text{explicit}}$
Experimental Setup: More on Metrics

Deviation: \( \text{dev}(r, p) > d \)

Margin: \( \text{dev}(r_i, p_i) - \text{dev}(r_j, p_j) > m \)

\( d \) and \( m \) as tradeoff between Query Precision and Recall

- \( d, m \) increase
  - Precision goes up
  - Recall goes down

- \( d, m \) decrease
  - Precision goes down
  - Recall goes up
Experimental Setup: Results

Figure 6.1: Precision vs. Recall of SA, SA+N, CD, CDiff, CD+CDiff, UserBehavior, and Current relevance prediction methods over the Q1 dataset.
Figure 6.3: Recall vs. Precision of CD+CDiff and UserBehavior for query sets Q1, Q10, and Q20 (queries with at least 1, at least 10, and at least 20 clicks respectively).
Figure 6.2: Precision vs. recall for predicting relevance with each group of features individually.
Conclusion

- Observed a wide range of strategies:
  - SA, SA+N
  - CD, CDiff
    - Considers “background noise”
  - UserBehavior
    - Richer features
- Accounting for the “background noise” before applying clickthrough strategies can improve accuracy.
- Using richer features that include user behavior before and after search lead to increased accuracy.