Social Search and Discovery using a Unified Approach

Einat Amitay
David Carmel
Nadav Golbandi
Nadav Har'El
Shila Ofek-Koifman
Sivan Yogev

IBM Haifa Research Lab
Web 2.0 data gives us

• New wealth of information (produced by ordinary users).

• New types of information – social information:
  – User-supplied metadata for documents (bookmarks, tags, ratings, comments).
  – Relationships between people and documents (who wrote a document, who tagged it, etc.).
  – Relationships between people and other people.
Outline of this talk

• Enterprise search.
• Unified search: document & person.
• Implementation of the unified search using **faceted search**.
• Adding tags to the equation(s).
• The system and its evaluation.
Enterprise search

First goal: use *social information* to improve search in an enterprise intranet (IBM).

- Improve the relevance of document results:
  - Add tags and comments to the searched text of the documents.
  - Important documents can be recognized by user activity around them (bookmarking, comments, etc.)
  - Allow search outside the enterprise boundaries.
Enterprise search

First goal: use *social information* to improve search in an enterprise intranet (IBM).

- Implementation: indexing only user-generated data from enterprise sources.
- Without using popularity – mediocre quality.
- When using popularity, precision is vastly improved over standard full-text search (P@10 between 0.7-0.8).
Unified search

- When in need of information,
  - Some people like to find a written document.
  - Some people like to find a person to ask.
  - Most people are between these extremes.
  - And each source is better in different situations.
Unified search

- Second goal: return more types of relevant information sources.

- Given a query, we want the search engine to return:
  - A ranked list of documents relevant to the query
  - A ranked list of people with interest in the query topic

- We also want to use people in queries:
  - “John Smith”
  - information retrieval “John Smith”
Person search

- A person is relevant to a query if he or she are related to documents relevant to the query. Therefore, given a query:
  - Find all documents relevant to this query,
  - Find people relevant to these documents.

- [McDonald & Ounis, Balog & de Rijke, 2006]

- But how to score?
Person search

• Returning to the Vector Space Model:
  – In VSM, documents define relevance matrix $D$, between documents and terms.

Can be viewed as a bipartite graph:
Person search

- Returning to the Vector Space Model:
  - In VSM, documents define relevance matrix $D$, between documents and terms.

A query is a vector of terms $q$

Search results: $D \times q$
Person search

• Returning to the Vector Space Model:
  – In VSM, documents define relevance matrix $D$, between documents and terms.

A query is a vector of terms $q$

Search results: $D \times q$
Person search

- Returning to the Vector Space Model:
  - Document-person relationships define relevance matrix \( P \) between documents and people.

Yet another bipartite graph:
And we get a multi-layer graph:
Person search

- Returning to the Vector Space Model:
  - $P^T \times D$ is the relevance matrix between terms and people.

$P^T \times D \times q$ are (scored) people search results.
Person search

- Returning to the Vector Space Model:
  - $P^T \times D$ is the relevance matrix between terms and people.

$P^T \times D \times q$ are (scored) people search results.
Person search

• Calculating \((P^T \times D) \times q\) seems like the obvious solution, but:
  – Keeping \(P^T \times D\) up-to-date is hard
  – Document and person search done using two different matrices \((D\) and \(P^T \times D\))
  – Lose non-VSM search engine features (phrase, etc)

• We therefore use the equivalent formula:
Person search

- Score for person i, \((P^T \times D \times q)_i = \sum_{\text{matching documents } d} \text{relation}(d, \text{person } i) \cdot \text{score}_q(d)\)

- Already proposed in Balog & de Rijke, with different (probabilistic) justification.

- But how to calculate the paths?
Faceted search

- Commonly used technique for adding navigation to a search engine.
- A **facet** is a single attribute of the document.
- In a camera search application, documents might have a “Brand” and “Price” facets.
- To each document, several *categories* are added. For example “Brand/Sony” or “Price Range/$90-$40”.
Faceted search

- Simplest faceted search *efficiently* goes over matching documents, counting for each category the number of documents:

  - **Price Range**
    - Below $90 (116)
    - $90 - $140 (106)
    - $140 - $170 (96)
    - $170 - $210 (105)
    - $210 - $260 (112)
    - $260 - $350 (117)
    - $350 - $650 (112)
    - Above $650 (112)

  - **Brand**
    - Canon (170)
    - Olympus (214)
    - Nikon (158)
    - Sony (169)
    - Panasonic (104)
    - Kodak (164)
    - Fuji (161)

  - **LCD Display Size**
    - Less than 1.5 in. (62)
    - 1.5 - 2.0 in. (1,262)
    - 2.0 - 2.4 in. (390)
    - More than 2.4 in. (754)
    - Select more than one
Faceted search

• In our application, we have “Related Person” facets.

• Categories like “Related Person/John Smith” attached to document, with a weight.

• Instead of just counting, *efficiently* aggregate expressions. For person i category:

\[
\sum_{\text{matching documents } d} \text{relation}(d, \text{person } i) \cdot \text{score}_q(d)
\]
Faceted search

- The actual index contains only one layer, Person-Document relationships.
- Document scores based on Lucene queries.
- Person scores are calculated through facets.
Tags search

- Tags are also added as facets to documents.
- Therefore tag scores can be calculated like person scores.
Faceted search

- More faceted search features we use:
  - Query-independent static score for categories (category boost).
    \[ ief(person) = \log\left(\frac{N}{N_{person}}\right) \]
  - Special query for “Facet f” returns all documents in this category, sorted by the category weight.
The Social Search Application

- Data from some of IBM's internal Web 2.0 sites:
  - Over 77,000 blog threads (thread = entry + comments)
    - Content: Blog entry, comments, tags
    - Person facet: author, commenter, bookmarker
  - Over 370,000 bookmarks to over 230,000 Web-pages
    - Content: Titles, user descriptions, tags
    - Person facet: bookmarker
  - Over 15,500 people who created that content
Main Features

• Queries can contain a combination of Lucene textual queries, people and tags.
• Result sets can contain a combination of web pages, people and tags.
• Easy navigation to narrow search, or find evidence to the current results.
• Good quality of results...
Search took 0.05 seconds.
Found 108 results. Showing results 1-10:

1. **Welcome to OpenID Enabled! — OpenID Enabled**
   Resource for OpenID community
   Bookmarked 2 times
   http://www.openidenabled.com/

2. **Sam Ruby: OpenID for non-SuperUsers**
   A well written article for novice users to get started with OpenID
   Bookmarked 4 times
   http://intertwingly.net/blog/2007/01/03/OpenID-for-non-SuperUsers

3. **OpenID: an actually distributed identity system**
   The first piece of the OpenID framework is authentication -- how you prove ownership of a URI. Today, websites require usernames and passwords to login, which means that many people use the same password everywhere.
   Bookmarked 24 times
   http://openid.net/

4. **OpenID Authentication 1.1**
   OpenID Authentication 1.1
   Bookmarked 1 time
   http://openid.net/specs/openid-authentication-1_1.html

5. **Microsoft and Google want to support OpenID - The Good, The Bad and The Ugly - BlogCentral**
   Blog entry by Germany/IBM, with 1 comments
   Bookmarked 1 time
   http://blogs.tap.ibm.com/weblogs/germany@de.ibm.com/entry/microsoft_and_google_want_to
Evaluation

- We return documents, people and tags for every query – need to evaluate precision.

- Document results evaluated using standard technique:
  - 50 real textual queries chosen from query logs
  - The top results judged by humans as being “relevant”, “very relevant” or “irrelevant”.
  - Very high precision demonstrated (P@10 ~ 0.8).
  - Much better than full-text enterprise search.
Evaluation

• “Related people” evaluation – large user study
  - 60 real textual queries chosen from query logs.
  - 100 related people retrieved for each query.
  - Each person was mailed listing 6-15 queries (some believed to be relevant and some irrelevant): Rate 1-5 whether the topic is relevant to you.
  - 612 people responded, from 116 IBM locations in 38 countries.
  - The ranked list of related people we generate are compared to these self-ratings using NDCG metric.
  - Compare full scoring formula to simpler ones.
Evaluation

- Evaluation results:

<table>
<thead>
<tr>
<th>Aggregation expression</th>
<th>NDC G 10</th>
<th>NDC G 20</th>
<th>NDC G 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count only “votes”</td>
<td>0.71</td>
<td>0.69</td>
<td>0.68</td>
</tr>
<tr>
<td>Sum of scores “CombSUM”</td>
<td>0.75</td>
<td>0.73</td>
<td>0.72</td>
</tr>
<tr>
<td>+relationship weights</td>
<td>0.76</td>
<td>0.74</td>
<td>0.73</td>
</tr>
<tr>
<td>+person static score: ief</td>
<td>0.77</td>
<td>0.76</td>
<td>0.74</td>
</tr>
</tbody>
</table>
Conclusions

- Web 2.0 data provides an excellent source for document and people search in an enterprise.
- Unified search can be easily realized using faceted search.
- In a 612-respondent study, the full scoring formula was shown better than simpler versions.
- Also strengthens previously published results by using a new data set and evaluation.
The real world...
Thank you!