Chapter 10

Social Search
Social Search

- Social search
  - Communities of users *actively participating* in the search process
  - Goes beyond classical search tasks
- Key differences
  - Users interact with the system
  - Users interact with other users implicitly/explicitly
Web 2.0

- Social search includes, but is not limited to, the so-called social media site
  - Collectively referred to as “Web 2.0” as opposed to the classical notion of the Web (“Web 1.0”)

- Social media sites
  - User generated content
  - Users can tag their own and other’s content
  - Users can share favorites, tags, etc., with others

- Examples.
  - Digg, Twitter, Flickr, YouTube, Del.icio.us, CiteULike, MySpace, Facebook, and LinkedIn
Social Search Topics

- User tags
- Searching within communities
- Adaptive filtering
- Recommender systems
- Peer-to-peer and metasearch
User Tags & Manual Indexing

Then: Library card catalogs
- Indexing terms chosen with search in mind
- Experts generate indexing terms
- Terms are very high quality
- Terms chosen from controlled vocabulary

Now: Social media tagging
- Tags not always chosen with search in mind
- Users generate tags
- Tags can be noisy or even incorrect
- Tags chosen from *folksonomies*
Types of User Tags

- **Content-based**
  - car, woman, sky

- **Context-based**
  - new york city, empire state building

- **Attribute**
  - nikon (type of camera), black and white (type of movie), homepage (type of web page)

- **Subjective**
  - pretty, amazing, awesome

- **Organizational**
  - to do, my pictures, readme
Searching Tags

- Searching *user tags* is challenging
  - Most items have only a few tags
  - Tags are very short

- *Boolean, probabilistic, vector space* & language modeling will fail if use naively

- Must overcome the *vocabulary mismatch problem* between the query and tags
Tag Expansion

- Can overcome *vocabulary mismatch* problem by expanding *tag representation* w/ external knowledge.

- Possible external sources:
  - Thesaurus
  - Web search results
  - Query logs

- After *tags* have been expanded, can use standard retrieval models.
Tag Expansion Using Search Results

Age of Aquariums - Tropical Fish
Huge educational aquarium site for tropical fish hobbyists, promoting responsible fish keeping internationally since 1997.

The Krib (Aquaria and Tropical Fish)
This site contains information about tropical fish aquariums, including archived usenet postings and e-mail discussions, along with new ...

... 

Keeping Tropical Fish and Goldfish in Aquariums, Fish Bowls, and ... Keeping Tropical Fish and Goldfish in Aquariums, Fish Bowls, and Ponds at AquariumFish.net.

P(w | “tropical fish”)
Searching Tags

- Even with tag expansion, *searching tags* is challenging
- Tags are inherently *noisy* and *incorrect*
- Many items may *not* even be tagged!
- Typically *easier* to find *popular* items with many tags than *less* popular items with *few/no* tags
Inferring Missing Tags

- How can we automatically tag items with few or no tags?
- Uses of inferred tags
  - Improved tag search
  - Automatic tag suggestion
Methods for Inferring Tags

- **TF.IDF**
  - Suggest tags that have a high TF.IDF weight in the item
  - Only works for textual items

- **Classification**
  - Train binary classifier for each tag
  - Performs well for *popular tags*, but not as well for *rare tags*

- **Maximal marginal relevance**
  - Finds tags that are *relevant* to the item and novel with respect to existing tags

\[
MMR(t; T_i) = \left( \lambda Sim_{item}(t, i) - (1 - \lambda) \max_{t \in T_i} Sim_{tag}(t_i, t) \right)
\]
Browsing and Tag Clouds

- Search is useful for finding items of interest
- Browsing is more useful for exploring collections of tagged items

Various ways to visualize collections of tags
- Tag lists
- Tag clouds
- Alphabetical order
- Grouped by category
- Formatted/sorted according to popularity
Example Tag Cloud

animals architecture art australia autumn baby band barcelona beach berlin birthday black blackandwhite blue california cameraphone canada canon car cat chicago china christmas church city clouds color concert day dog england europe family festival film florida flower flowers food france friends fun garden germany girl graffiti green halloween hawaii holiday home house india ireland italy japan july kids lake landscape light live london macro me mexico music nature new newyork night nikon nyc ocean paris park party people portrait red river rock sanfrancisco scotland sea seattle show sky snow spain spring street summer sunset taiwan texas thailand tokyo toronto travel tree trees trip uk usa vacation washington water wedding
Searching with Communities

- What is an online community?
  - Groups of entities that interact in an online environment and share common goals, traits, or interests

- Examples
  - Baseball fan community
  - Digital photography community

- Not all communities are made up of humans!
  - Web communities are collections of web pages that are all about a common topic
Finding Communities

- What are the characteristics of a community?
  - Entities within a community are similar to each other
  - *Members* of a community are likely to interact more with other members of the community than those outside of the community

- Can represent interactions between a set of entities as a graph
  - *Vertices* are entities
  - *Edges* (directed or undirected) indicate interactions between the entities
Graph Representation

Node:

Vector:

\[
\begin{bmatrix}
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]
HITS

- **Hyperlink-induced Topic Search (HITS)** algorithm can be used to find communities
  - Link analysis algorithm, like PageRank
  - Each entity has a *hub* and *authority score*

- Based on a *circular* set of assumptions
  - Good *hubs* point to good *authorities*
  - Good *authorities* are pointed to by good *hubs*

- Iterative algorithm:
  \[ A(p) = \sum_{q \rightarrow p} H(q) \]
  \[ H(p) = \sum_{p \rightarrow q} A(q) \]
Algorithm 1 HITS

1: procedure HITS(G = (V, E), K)
2: \hspace{1em} A_0(p) \leftarrow 1 \ \forall p \in V
3: \hspace{1em} H_0(p) \leftarrow 1 \ \forall p \in V
4: \hspace{1em} \text{for } i = 1 \text{ to } K \text{ do}
5: \hspace{2em} A_i(p) \leftarrow 0 \ \forall p \in V
6: \hspace{2em} H_i(p) \leftarrow 0 \ \forall p \in V
7: \hspace{2em} Z_A \leftarrow 0
8: \hspace{2em} Z_H \leftarrow 0
9: \hspace{1em} \text{for } p \in V \text{ do}
10: \hspace{2em} \text{for } q \in V \text{ do}
11: \hspace{3em} \text{if } (p, q) \in E \text{ then}
12: \hspace{4em} H_i(p) \leftarrow H_i(p) + A_{i-1}(q)
13: \hspace{4em} Z_H \leftarrow Z_H + A_{i-1}(q)
14: \hspace{3em} \text{end if}
15: \hspace{2em} \text{if } (q, p) \in E \text{ then}
16: \hspace{3em} A_i(p) \leftarrow A_i(p) + H_{i-1}(q)
17: \hspace{3em} Z_A \leftarrow Z_A + H_{i-1}(q)
18: \hspace{2em} \text{end if}
19: \hspace{2em} \text{end for}
20: \hspace{1em} \text{end for}
21: \hspace{1em} \text{for } p \in V \text{ do}
22: \hspace{2em} A_i(p) \leftarrow \frac{A_i(p)}{Z_A}
23: \hspace{2em} H_i(p) \leftarrow \frac{H_i(p)}{Z_H}
24: \hspace{2em} \text{end for}
25: \hspace{1em} \text{end for}
26: \hspace{1em} \text{return } A_K, H_K
27: \text{end procedure}
Finding Communities

- **HITS**
  - Can apply HITS to entity interaction graph to find communities
  - Entities with large *authority scores* are the “core” or “authoritative” members of the community

- **Clustering**
  - Apply agglomerative or *K-means clustering* to entity graph
  - How to choose K?

- **Evaluating community finding algorithms** is hard

- Can use communities in various ways to improve search, browsing, expert finding, recommendation, etc.
Community Based Question Answering

- Some *complex information needs* can’t be answered by traditional search engines
  - Information from multiple sources
  - Human expertise
- Community based question answering tries to overcome these limitations
  - *Searcher* enters question
  - *Community members* answer question
Example Questions

What part of Mexico gets the most tropical storms?
How do you pronounce the French words, coeur and miel?
GED test?
Why do I have to pay this fine?
What is Schrödinger’s cat?
What’s this song?
Hi...can u ppl tell me sumthing abt death dreams??
What are the engagement and wedding traditions in Egypt?
Fun things to do in LA?
What lessons from the Tao Te Ching do you apply to your everyday life?
Foci of a hyperbola?
What should I do today?
Why was iTunes deleted from my computer?
Heather Locklear?
Do people in the Australian Defense Force (RAAF) pay less tax than civilians?
What’s a psp xmb?
If C(-3, y) and D(1, 7) lie upon a line whose slope is 2, find the value of y?
Why does love make us so irrational?
Am I in love?
What are some technologies that are revolutionizing business?
Community Based Question Answering

- **Pros**
  - Can find answers to *complex/obscure* questions
  - Answers are from *humans*, not algorithms
  - Can search *archive* of previous questions/answers

- **Cons**
  - Often takes *time* to get a response
  - Some questions *never* get *answered*
  - Answers may be *wrong*
Question Answering Models

- How can we effectively search an archive of question/answer pairs?
- Can be treated as a translation problem
  - Translate a question into a related question
  - Translate a question into an answer

- Translation-based language model:

\[
P(Q|A) = \prod_{w \in Q} \sum_{t \in V} P(w|t)P(t|A)
\]

- Enhanced translation model:

\[
P(Q|A) = \prod_{w \in Q} \frac{(1 - \beta) f_{w,A} + \beta \sum_{t \in V} P(w|t) f_{t,A} + \mu \frac{c_w}{c}}{|A| + \mu}
\]
Computing Translation Probabilities

- Translation probabilities are learned from a *parallel corpus*
- Most often used for learning *inter-language probabilities*
- Can be used for *intra-language probabilities*
  - Treat question / answer pairs are parallel corpus
- Various tools exist for computing translation probabilities from a parallel corpus
Example Question/Answer Translations

<table>
<thead>
<tr>
<th>everest</th>
<th>xp</th>
<th>search</th>
</tr>
</thead>
<tbody>
<tr>
<td>everest</td>
<td>xp</td>
<td>search</td>
</tr>
<tr>
<td>mountain</td>
<td>window</td>
<td>google</td>
</tr>
<tr>
<td>tallest</td>
<td>install</td>
<td>information</td>
</tr>
<tr>
<td>29,035</td>
<td>drive</td>
<td>internet</td>
</tr>
<tr>
<td>highest</td>
<td>computer</td>
<td>website</td>
</tr>
<tr>
<td>mt</td>
<td>version</td>
<td>web</td>
</tr>
<tr>
<td>ft</td>
<td>click</td>
<td>list</td>
</tr>
<tr>
<td>measure</td>
<td>pc</td>
<td>free</td>
</tr>
<tr>
<td>feet</td>
<td>program</td>
<td>info</td>
</tr>
<tr>
<td>mount</td>
<td>microsoft</td>
<td>page</td>
</tr>
</tbody>
</table>
Collaborative Search Scenarios

- Co-located Collaborative Searching
- Remote Collaborative Searching
Collaborative Search

- Challenges
  - How do users *interact* with system?
  - How do users *interact* with each other?
  - How is *data shared*?
  - What *data persists* across sessions?

- Very few commercial collaborative search systems

- Likely to see more of this type of system in the future
Document Filtering

- Ad hoc retrieval
  - *Document* collections and information *needs change* with time
  - *Results* returned when *query* is entered

- Document filtering
  - *Document* collections *change* with time, but information needs are *static* (long-term)
  - Long term information needs represented as a *profile*
  - *Documents* entering system that match the *profile* are delivered to the user via a *push* mechanism
Profiles

- Represents long term *information needs*

- Can be represented in different ways
  - Boolean or keyword query
  - Sets of *relevant* and *non-relevant* documents
  - Relational *constraints*
    - “published before 1990”
    - “price in the $10-$25 range”

- Actual representation usually depends on underlying *filtering model*

- Can be static (*static filtering*) or updated over time (*adaptive filtering*)
Document Filtering Scenarios

Static Filtering

Adaptive Filtering
Static Filtering

- Given a fixed profile, how can we determine if an incoming document should be delivered?

- Treat as an IR problem
  - Boolean
  - Vector space
  - Language modeling

- Treat as *supervised learning* problem
  - Naïve Bayes
  - Support vector machines
Adaptive Filtering

- In adaptive filtering, profiles are *dynamic*

- How can *profiles* change?
  - User can explicitly *update* the profile
  - User can provide (relevance) *feedback* about the documents delivered to the profile
  - Implicit user *behavior* can be *captured* and used to update the profile
Fast Filtering with Millions of Profiles

- Real filtering systems
  - May have thousands or even millions of profiles
  - Many new documents will enter the system daily

- How to efficiently filter in such a system?
  - Most profiles are represented as text or a set of features
  - Build an inverted index for the profiles
  - Distill incoming documents as “queries” and run against index
Evaluation of Filtering Systems

- Definition of “good” depends on the purpose of the underlying filtering system

<table>
<thead>
<tr>
<th></th>
<th>Relevant</th>
<th>Non-Relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieved</td>
<td>TP</td>
<td>FP</td>
</tr>
<tr>
<td>Not retrieved</td>
<td>FN</td>
<td>TN</td>
</tr>
</tbody>
</table>

- Generic filtering evaluation measure:

\[ U = \alpha \cdot TP + \beta \cdot TN + \delta \cdot FP + \gamma \cdot FN \]

\(\alpha = 2, \beta = 0, \delta = -1,\) and \(\gamma = 0\) is widely used
Collaborative Filtering

- In static and adaptive filtering, *users* and their *profiles* are assumed to be *independent* of each other.

- *Similar users* are likely to have *similar preferences*.

- *Collaborative filtering* exploits relationships between *users* to improve how items (documents) are *matched* to users (profiles).
Recommender Systems

- Recommender systems *recommend items* that a user may be interested in.

- **Examples**
  - Amazon.com
  - NetFlix

- Recommender systems use *collaborative filtering* to recommend items to users.
Recommender System Algorithms

- **Input**
  - \(<\text{user}, \text{item}, \text{rating}>\) tuples for *items* that the user has explicitly *rated*
  - Typically represented as a user-item *matrix*

- **Output**
  - \(<\text{user}, \text{item}, \text{rating}>\) tuples for *items* that the user has *not rated*
  - Can be thought of as filling in the *missing* entries of the user-item matrix

- Most algorithms *infer* missing ratings based on the *ratings of similar users*
Recommender Systems
Collaborative Searching

- Traditional search assumes single searcher
- Collaborative search involves a group of users, with a common goal - searching together in a collaborative setting

Example scenarios

- Students doing research for a history report
- Family members searching for information on how to care for an aging relative
- Team members working to gather information and requirements for an industrial project
Collaborative Search

- Two types of collaborative search settings depending on where participants are *physically located*

- Co-located
  - Participants in *same* location
  - *Co-Search* system

- Remove collaborative
  - Participants in *different* locations
  - *Search-Together* system
Static Filtering with Language Models

- Assume profile consists of $K$ relevant documents ($T_i$), each with weight $\alpha_i$

- Probability of a word given the profile is:

$$P(w|P) = \frac{(1 - \lambda)}{\sum_{i=1}^{K} \alpha_i} \sum_{i=1}^{K} \alpha_i \frac{f_{w,T_i}}{|T_i|} + \lambda \frac{c_w}{|C|}$$

- $KL$ divergence between profile and document model is used as score:

$$-KL(P||D) = \sum_{w \in V} P(w|P) \log P(w|D) - \sum_{w \in V} P(w|P) \log P(w|P)$$

- If $-KL(P||D) \geq \theta$, then deliver $D$ to $P$
  - Threshold ($\theta$) can be optimized for some metric
Adaptive Filtering Models

- **Rocchio**
  - Profiles treated as vectors
    \[ P' = \alpha P + \beta \frac{1}{|Rel|} \sum_{D_i \in Rel} D_i - \gamma \frac{1}{|Nonrel|} \sum_{D_i \in Nonrel} D_i \]

- **Relevance-based language models**
  - Profiles treated as language models
    \[ P(w|P) = \frac{1}{|Rel|} \sum_{D_i \in Rel} \sum_{D \in C} P(w|D)P(D_i|D) \approx \frac{1}{|Rel|} \sum_{D_i \in Rel} P(w|D_i) \]
## Summary of Filtering Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Profile Representation</th>
<th>Profile Updating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Boolean Expression</td>
<td>N/A</td>
</tr>
<tr>
<td>Vector Space</td>
<td>Vector</td>
<td>Rocchio</td>
</tr>
<tr>
<td>Language Modeling</td>
<td>Probability Distribution</td>
<td>Relevance Modeling</td>
</tr>
<tr>
<td>Classification</td>
<td>Model Parameters</td>
<td>Online Learning</td>
</tr>
</tbody>
</table>