Chapter 10

Social Search
Social Search

“Social search describes search acts that make use of social interactions with others. These interactions may be explicit or implicit, co-located or remote, synchronous or asynchronous”

Social search

- Search within a social environment
- Communities of users actively participating in the search process
- Goes beyond classical search tasks
- Facilitates the “information seeking” process [Evans 08]

Social vs. Standard Search

- Key differences
  - Users interact with the system (standard & Social)
  - Users interact with one another in an open/social environment implicitly (reading)/explicitly (writing) such as
    - Visiting *social media sites*, e.g., YouTube
    - Browsing through *social networking sites*, e.g., Facebook
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
  - Provide unique data resources for search engines

- Example.
  - YouTube, MySpace, Facebook, LinkedIn, Digg, Twitter, Flickr, Del.icio.us, and CiteULike
# Social Media/Networking Sites

<table>
<thead>
<tr>
<th>SM Services</th>
<th>Level of Collaboration</th>
<th>Content: Audio</th>
<th>Content: Video</th>
<th>Content: Image</th>
<th>Content: Text</th>
<th>Content: Aggregation</th>
<th>Provider Censorship</th>
<th>User Censorship</th>
<th>Privacy</th>
<th>Communication Type</th>
<th>Provides API</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook</td>
<td>HIGH</td>
<td>NONE</td>
<td>LOW</td>
<td>MEDIUM</td>
<td>HIGH</td>
<td>LOW</td>
<td>NONE</td>
<td>HIGH</td>
<td>BOTH</td>
<td>MANY-TO-MANY</td>
<td>GOOD</td>
</tr>
<tr>
<td>LinkedIn</td>
<td>LOW</td>
<td>NONE</td>
<td>NONE</td>
<td>LOW</td>
<td>HIGH</td>
<td>NONE</td>
<td>NONE</td>
<td>HIGH</td>
<td>BOTH</td>
<td>1-TO-MANY</td>
<td>NO</td>
</tr>
<tr>
<td>Twitter</td>
<td>MEDIUM</td>
<td>NONE</td>
<td>NONE</td>
<td>NONE</td>
<td>HIGH</td>
<td>HIGH</td>
<td>HIGH</td>
<td>NONE</td>
<td>HIGH</td>
<td>MANY-TO-MANY</td>
<td>GOOD</td>
</tr>
<tr>
<td>Delicious</td>
<td>LOW</td>
<td>NONE</td>
<td>NONE</td>
<td>NONE</td>
<td>LOW</td>
<td>HIGH</td>
<td>NONE</td>
<td>NONE</td>
<td>PUBLIC</td>
<td>MANY-TO-MANY</td>
<td>GOOD</td>
</tr>
<tr>
<td>Flickr</td>
<td>MEDIUM</td>
<td>NONE</td>
<td>HIGH</td>
<td>HIGH</td>
<td>LOW</td>
<td>NONE</td>
<td>NONE</td>
<td>HIGH</td>
<td>BOTH</td>
<td>MANY-TO-MANY</td>
<td>GOOD</td>
</tr>
<tr>
<td>YouTube</td>
<td>HIGH</td>
<td>NONE</td>
<td>HIGH</td>
<td>NONE</td>
<td>LOW</td>
<td>NONE</td>
<td>LOW</td>
<td>HIGH</td>
<td>BOTH</td>
<td>MANY-TO-MANY</td>
<td>GOOD</td>
</tr>
<tr>
<td>Skype</td>
<td>LOW</td>
<td>HIGH</td>
<td>HIGH</td>
<td>NONE</td>
<td>LOW</td>
<td>LOW</td>
<td>LOW</td>
<td>NONE</td>
<td>HIGH</td>
<td>PRIVATE</td>
<td>1-TO-1</td>
</tr>
<tr>
<td>Last.fm</td>
<td>LOW</td>
<td>HIGH</td>
<td>NONE</td>
<td>NONE</td>
<td>LOW</td>
<td>NONE</td>
<td>LOW</td>
<td>NONE</td>
<td>LOW</td>
<td>BOTH</td>
<td>MANY-TO-MANY</td>
</tr>
<tr>
<td>yelp.com</td>
<td>LOW</td>
<td>NONE</td>
<td>NONE</td>
<td>HIGH</td>
<td>HIGH</td>
<td>HIGH</td>
<td>NONE</td>
<td>NONE</td>
<td>HIGH</td>
<td>1-TO-MANY</td>
<td>OK</td>
</tr>
<tr>
<td>WikiAnswers</td>
<td>HIGH</td>
<td>NONE</td>
<td>NONE</td>
<td>LOW</td>
<td>HIGH</td>
<td>NONE</td>
<td>NONE</td>
<td>HIGH</td>
<td>PUBLIC</td>
<td>MANY-TO-MANY</td>
<td></td>
</tr>
<tr>
<td>World of Warcraft</td>
<td>MEDIUM</td>
<td>MEDIUM</td>
<td>HIGH</td>
<td>NONE</td>
<td>LOW</td>
<td>NONE</td>
<td>NONE</td>
<td>NONE</td>
<td>LOW</td>
<td>BOTH</td>
<td>NO</td>
</tr>
</tbody>
</table>
# Social Media/Network Sites

<table>
<thead>
<tr>
<th>Social Media/Network Sites</th>
<th>Facebook</th>
<th>LinkedIn</th>
<th>Delicious</th>
<th>Twitter</th>
<th>LibraryThing</th>
<th>Flickr</th>
<th>YouTube</th>
<th>MySpace</th>
<th>Last.fm</th>
<th>CiteULike</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Web Search</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>maybe</td>
<td>maybe</td>
<td>yes</td>
<td>maybe</td>
<td>maybe</td>
<td>Yes (e.g., Google Scholar)</td>
</tr>
<tr>
<td><strong>Recommendation</strong></td>
<td>friends, groups, ads, links, connections, jobs, ads, books, news articles, movies, videos, groups</td>
<td>books, links, news articles, location</td>
<td>books, groups, links</td>
<td>users, pics, travel, groups, links, videos</td>
<td>video, links, people,</td>
<td>friends, groups, ads, links, (provides rec based on search habits)</td>
<td>friends, music, concerts (the rec application exists)</td>
<td>articles, links, groups</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Filtering</strong></td>
<td>messages, activity, links, news articles,</td>
<td>links, news articles,</td>
<td>Messages (hashtag)</td>
<td>conversations, groups</td>
<td>pics</td>
<td>videos</td>
<td>messages, activity,</td>
<td>messages, music</td>
<td>new papers, articles</td>
<td></td>
</tr>
<tr>
<td><strong>Ads Suggestion</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>maybe</td>
<td>maybe</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Collaborative Searching/Filtering</strong></td>
<td>yes depending on topic</td>
<td>yes</td>
<td>yes</td>
<td>depending on topic</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>maybe</td>
<td>depending on topic</td>
<td></td>
</tr>
<tr>
<td><strong>User Similarity (profile)</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>maybe</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>maybe</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Personal Interest Identification</strong></td>
<td>yes</td>
<td>depending on domain</td>
<td>yes</td>
<td>yes</td>
<td>yes (but may not be comprehensive)</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>depending on topic</td>
<td></td>
</tr>
<tr>
<td><strong>Topic Identification</strong></td>
<td>maybe</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>No (or not really comprehensive)</td>
<td>maybe</td>
<td>maybe</td>
<td>maybe</td>
<td>maybe</td>
<td>yes (within area of study)</td>
</tr>
<tr>
<td><strong>Tag (matching/suggestions)</strong></td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>maybe</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
Social Search Topics

- Online *user-interactive* data, which provide a new and interesting search experience

  - **User tags:** users assign tags to data items, a *manual indexing* approach

  - **Searching within communities:** *virtual* groups of online users, who share *common interests*, *interact socially*, such as blogs and QA systems

  - **Recommender systems:** individual users are represented by their *profiles* (fixed queries – long-term info. need) such as CNN Alert Service, Amazon.com, etc.

  - **Peer-to-peer network:** querying a community of “nodes” (individual/organization/search engine) for an info. need

  - **Metasearch:** a special case of P2P – all the nodes are *SEs*
User Tags and Manual Indexing

Then: Library card catalogs
- *Indexing terms* chosen with search in mind
- *Experts generate* indexing terms *manually*
- Terms are very *high quality* based on the US Library of Congress (LOC) *Subject Headings* standardized by the LOC
- Terms chosen from *controlled/fixed vocabulary* and subject guides (a drawback)

Now: Social media tagging
- *Social media sites* allow *users* to *generate* own tags *manually* (+)
- Tags not always chosen with search in mind (-)
- Tags can be *noisy* or even *incorrect* and without quality control (-)
- Tags chosen from *folksonomies, user-generated taxonomies* (+)
Social Search Topics

Example. Some of the 116 million tags of LibraryThing, which archives 95 million book records w/ 1.93 million users (04/15)
Social Tagging

- According to [Guan 10]
  - Social tagging services allow users to annotate online resources with freely chosen keywords
  - Tags are collectively contributed by users and represent their comprehension of resources.
  - Tags provide meaningful descriptors of resources and implicitly reflect users’ interests.
  - Tagging services provide keyword-based search, which returns resources annotated by given tags.

Types of User Tags

- **Content-based**
  - Tags describe the *content* of an item, e.g., car, woman, sky

- **Context-based**
  - Tags describe the *context* of an item, e.g., NYC, empire bldg

- **Attribute-based**
  - Tags describe the *attributes* of an item, e.g., Nikon (type of camera), black and white (type of movie), etc.

- **Subjective-based**
  - Tags *subjectively* describe an item, e.g., pretty, amazing, etc.

- **Organizational-based**
  - Tags that organize items, e.g., to do, not read, my pictures, …
SearchingTags

- Searching collaboratively tagged items, i.e., *user tags*, is *challenging*
  
  - Most items have only a *few* tags, i.e., complex items are sparingly represented, e.g., “aquariums” ≠ “goldfish”, which is the *vocabulary mismatch problem*
  
  - Tags are very *short*

- *Boolean (AND/OR), probabilistic, vector space, and language modeling* will fail if use naïvely
  
  - *High* precision but *low* recall for conjunctive (AND) queries
  
  - *Low* precision but *high* recall for disjunctive (OR) queries
Tag Expansion

- Can overcome *vocabulary mismatch* problem, such as “aquariums” and “topical fish”, by *expanding* tag representation with *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

Example. Web search results enhance a tag representation, “tropical fish,” a query

A retrieved snippet

Pseudo-relevance feedback over related terms

\[
P(w | \text{“tropical fish”})
\]
Searching Tags

- Even with tag expansion, *searching tags* is challenging.
- Tags are inherently noisy (*off topic, inappropriate*) and incorrect (*misspelled, spam*).
- Many items may *not* even be tagged, which become virtually *invisible* to any search engine.
- Typically *easier* to find *popular* items with many tags than *less* popular items with *few/no* tags.
  - How can we automatically *tag items with few or no tags*?
    - Uses inferred tags to
      - Improve tag search
      - Automatic tag suggestion
Methods for Inferring Tags

- **TF-IDF**: \( wt(w) = \log(f_{w,D} + 1) \log(N / df_w) \)
  - Suggest tags that have a *high* TF-IDF weight in the item
  - Only works for textual items

- **Classification** (determines the appropriateness of a tag)
  - Train binary classifier for each tag, e.g., using SVM
  - Performs well for *popular tags*, but not as well for *rare tags*

- **Maximal marginal relevance**
  - Finds *relevant* tags to the item and *novel* with respect to others
  - Large, if \( t \) is very relevant to \( T_i \), but differs from other tags of \( T_i \)

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

  where \( Sim_{item}(t, i) \) is the *similarity* between tag \( t \) and item \( i \), i.e., \( T_i \)
  \( Sim_{tag}(t_i, t) \) is the *similarity* between tags \( t_i \) and \( t \)

  \( \lambda \) (\( \in 0..1 \)), a tunable parameter
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 clouds (show the popularity of tags based on sizes)
  - (Tags are) Alphabetically *ordered* and/or *weighted*
  - Formatted/sorted according to *popularity*
Sample 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
As defined in [Schrammel 09], tag clouds are

- Visual displays of set of words (tags) in which attributes of the text such as size, color, font weight, or intensity are used to represent relevant properties, e.g., frequency of documents linked to the tag

- A good visualization technique to communicate an “overall picture”

Searching with Communities

- What is an online community?
  - Groups of entities (i.e., users, organizations, websites) that *interact* in an online environment to share common goals, interests, or traits
  - Besides tagging, community users also *post* to newsgroups, blogs, and other forums
  - To improve the overall user experiences, web search engines should automatically find the *communities* of a user

- Example.
  - Baseball fan community, digital photography community, etc.

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

According to [Seo 09]

- Online communities are valuable information sources where knowledge is accumulated by interactions between people.
- Online community pages have many unique textual or structural features, e.g.,
  - A forum has several sub-forums covering high-level topic categories
  - Each sub-forum has many threads
  - A thread is a more focused topic-centric discussion unit and is composed of posts created by community members

Finding Communities

- How to design general-purpose algorithms for finding every possible type of on-line community?

- What are the criteria used for finding a community?
  - Entities (users) within a community are similar to each other
  - Members of a community are likely to interact more with one another of the community than those outside of the community

- Can represent interactions between a set of entities as a graph
  - Vertices (V) are entities
  - Edges (E), directed or undirected, denote interactions of entities
    - Undirected edges represent symmetric relationships
    - Directed edges represent non-symmetric or causal relationships
HITS

- Hyperlink-Induced Topic Search (HITS) algorithm can be used to find communities
  - A 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:

  \[ A(p) = \sum_{q \rightarrow p} H(q) \]
  \[ H(p) = \sum_{p \rightarrow q} A(q) \]

  Authority score of \( p \) is the sum of the hub scores of the entities pointing at \( p \)
  Hub score of \( p \) is the sum of the authority scores pointed at by \( p \)
Form community \((C)\)

- Apply the *entity interaction graph* to find communities
- Identify a subset of the entities \((V)\), called *candidate entities*, be members of \(C\) (based on common interest)
- Entities with large *authority scores* are the core or “authoritative” members of \(C\)
  - to be a strong authority, an entity must have many *incoming* edges, all with moderate/large *hub* scores, and
  - To be a strong hub, an entity must have many outgoing edges, all with moderate/large *authority* scores
- Vertices *not* connected with others have *hub* and *authority* scores of 0

**HITS**
Finding Communities

- Clustering
  - Community finding is an inherently unsupervised learning problem
  - Agglomerative or K-means clustering approaches can be applied to entity interaction graph to find communities
  - Use the vector representation to capture the connectivity of various entities
  - Compute the authority values based on the Euclidean distance

- Evaluating community finding algorithms is hard

- Can use communities in various ways to improve web search, browsing, expert finding, recommendation, etc.
**Graph Representation**

![Graph Diagram]

**Node:**

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Vector:**

1 2 3 4 5 6 7
Community Based Question Answering

- Some *complex information needs* can’t be answered by traditional search engines
  - No single webpage may exist that satisfies the information needs
  - Information may come from multiple sources
  - Human (non-)experts in a wide range of topics form a community-based question answering (CQA) group, e.g., Yahoo! Answers

- CQA tries to overcome these limitations
  - *Searcher* enters questions
  - *Community members* answer questions
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?
Whats 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**

- Users can find answers to *complex* or *obscure* questions with *diverse* opinions about a topic
- Answers are from *humans*, not algorithms, that can be interacted with who share common interests/problems
- Can search *archive* of previous questions/answers, e.g., Yahoo! Answers

**Cons**

- Some questions *never get answered*
- Often takes *time* (possibly days) to get a response
- Answers may be *wrong*, *spam*, or *misleading*
Community Based Question Answering

Yahoo! Answers, a community-driven question-and-answer site launched by Yahoo! on July 5, 2005
Question Answering Models

- How can we effectively search an archive of question/answer pairs databases?

- Can be treated as a translation problem
  - Translate a question into a related/similar question which likely have relevant answers
  - Translate a question into an answer: less desirable

- The vocabulary mismatch problem
  - Traditional IR models likely miss many relevant questions
  - Many different ways to ask the same question
  - Stopword removal and stemming do not help
  - Solution: consider related concepts (i.e., words)—the probability of replacing one word by another
Question Answering Models

- Translation-based language model (for finding related questions, then answers): translate $w$ (in $Q$) from $t$ (in $A$)

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

where $Q$ is a question
$A$ is a related question in the archive
$V$ is the vocabulary
$P(w|t)$ are the translation probability
$P(t|A)$ is the (smoothed) probability of generating $t$ given $A$

- Anticipated problem: a good (independent) term-to-term translation might not yield a good overall translation

- Potential solution: matches of the original question terms are given more weight than matches of translated terms
**Question Answering Models**

- *Enhanced* translation model, which extends the **translation-based language model** on ranking $Q$:

$$
P(Q|A) = \prod_{w \in Q} \frac{(1 - \beta) f_{w,A} + \beta \sum_{t \in V} P(w|t) f_{t,A} + \mu \frac{c_w}{|C|}}{|A| + \mu}
$$

where $\beta \in 0..1$ controls the influence of the translation probability

$\mu$ is a *smoothing* parameter

$|A|$ is the number of words in question $A$

$C_w$ is count of $w$ in the entire collection $C$, and

$|C|$ is the total number of *word occurrence* in $C$

- when $\beta \rightarrow 1$, the model becomes more similar to the translation-based language model

- when $\beta \rightarrow 0$, the model is equivalent to the original query likelihood model, without influence from the translation model
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 as parallel corpus
  - Translation probabilities are estimated from archived pairs \((Q_1, A_1), (Q_2, A_2), \ldots, (Q_N, A_N)\)

- **Drawbacks**
  - Computationally expensive: sum over the entire vocabulary, which can be very large
  - Solution: considering only a small number (e.g., 5) of (most likely) translations per question term
Sample 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>
Document Filtering

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

- Document filtering
  - Document collections change with time, but (long-term) information needs are static
  - Long term information needs represented as a profile
  - Documents entering system that match the profile are delivered to the user via a push mechanism
  - Must be efficient and effective (minimizes FPs and FNs)
Profiles

- Represents long-term *information needs* and personalizes the search experience

- Can be represented in different ways by including
  - A Boolean or keyword query
  - Sets of *relevant* and *non-relevant* documents
  - Social tags and named entities
  - Relational *constraints*
    - “Published before 1990”
    - “Price in the $10 - $25 range”

- Actual representation usually depends on the underlying *filtering model*

- Static (filtering) or updated over time (adaptive filtering)
Document Filtering Scenarios

Static Filtering
Easier to process, less robust

Adaptive Filtering
More robust, requires frequent updates
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

Require predefined threshold value
Static Filtering with Language Models

- Assume a profile $P$ consists of $K$ relevant documents $T_i (1 \leq i \leq K)$ each with weight $\alpha_i$

- Probability of a word $w$ given the profile $P$ is

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

  - $\alpha_i$ is the weight (important) associated with $T_i$
  - $f_{w,T_i}$ is the frequency of occurrence of $w$ in $T_i$
  - $\lambda$ is a smoothing parameter
  - $C_w$ is count of $w$ in the entire collection $C$, and
  - $|C|$ is the total number of word occurrence in $C$
Static Filtering with Language Models

- Probability of a word $w$ given a new document $D$ is

$$P(w | D) = (1 - \lambda) \frac{f_{w,D}}{|D|} + \lambda \frac{C_w}{|C|}$$

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

$$KL(P \| D) = \sum_{t \in V} P(t | P) \log \frac{P(t | P)}{P(t | D)}$$

Approximation Distribution

True Distribution

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

- If $-KL(P \| D) \geq \theta$, then deliver $D$ to $P$, where $\theta$ is some relevance threshold.
Adaptive Filtering

- In adaptive filtering, *profiles* are dynamic

- How can *profiles* change (from static to dynamic)?
  1. User can explicitly *update* the profile
  2. User can provide (relevance) *feedback* about the documents delivered to the profile
  3. Implicit user *behavior* can be *captured* and used to update the profile
Adaptive Filtering Models

- **Rocchio**
  - Profiles treated as vectors

\[
P' = \alpha P + \beta \frac{1}{|\text{Rel}|} \sum_{D_i \in \text{Rel}} D_i - \gamma \frac{1}{|\text{Nonrel}|} \sum_{D_i \in \text{Nonrel}} D_i
\]

- **Relevance-based (profiles) language models**

\[
P(w|P) = \frac{1}{|\text{Rel}|} \sum_{D_i \in \text{Rel}} \sum_{D \in C} P(w|D)P(D_i|D)
\]

\[
\approx \frac{1}{|\text{Rel}|} \sum_{D_i \in \text{Rel}} P(w|D_i)
\]

where \( C \) is the set of documents in the collection \( P(D_i \mid D) \), the probability that \( D_i \) is generated from \( D \)’s LM, which is very close to 1 if \( D_i = D \); 0, otherwise
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
  - Do not produce ranking of documents for each profile
  - Evaluation measures, such as Precision@n and MAP, are non-relevant; precision, recall, and F-measure are computable

- Generic filtering evaluation measure:

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

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

- Static and adaptive filtering are not social tasks; profiles (and their users) are assumed to be independent of each other.

- Similar users are likely to have similar preferences or profiles.

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

  - If A is similar to B and A judged a document D is relevant, then it is likely that D is also relevant to B.

  - Often used as a component of recommender system.
Collaborative Filtering

According to [Ma 09], there are two widely-used types of methods for collaborative filtering:

- **Neighborhood-based methods**
  - Include user-based approaches, which predict the ratings of active users based on the computed information of items similar to those chosen by the active user.
  - Suffer from data sparsity and scalability problems

- **Model-based methods** use the observed user-item ratings to train a compact model that explains the given data so that ratings can be predicted

Rating using User Clusters

- Clustering can be used to find groups of similar users

- Measure *user/user similarity* using rating correlation:

\[
\frac{\sum_{i \in I_u \cap I_{u'}} (r_u(i) - \hat{r}_u) \cdot (r_{u'}(i) - \hat{r}_{u'})}{\sqrt{\sum_{i \in I_u \cap I_{u'}} (r_u(i) - \hat{r}_u)^2 \sum_{i \in I_u \cap I_{u'}} (r_{u'}(i) - \hat{r}_{u'})^2}}
\]

- Use *average rating* of other users within the same cluster to rate *unseen* items

\[
\hat{r}_u(i) = \frac{1}{|\text{Cluster}(u)|} \sum_{u' \in \text{Cluster}(u)} r_{u'}(i)
\]
Collaborative Filtering

- **Example.** Predicts the missing values in the user-item matrix [Ma 09]

![User-item Matrix](image)

![Predicted User-item Matrix](image)