Seminars in Database

Prerequisites – 2nd
F. Ricci

Content

- Information Retrieval
- Recommender Systems
  - Collaborative Filtering
Basic Concepts in Information Retrieval

- Information Retrieval (IR) deals with the representation, storage and organization of unstructured data
- Its mission is to assist in information “discovery” (more exactly information search)
- 2 main discovery paradigms:
  - Search and Browse

Thanks to Aya Sofer and David Carmel (IBM Haifa) for these IR slides.

Basic Concepts

- The User Task
  - Retrieval
    - Search for particular information
    - Usually focused and purposeful
  - Browsing
    - General looking around for information
    - For example: Asia-> Thailand -> Phuket -> Tsunami
**Data- vs Information- Retrieval** (from CJ Risjbergen)

<table>
<thead>
<tr>
<th></th>
<th>DR</th>
<th>IR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matching</td>
<td>exact</td>
<td>partial/best</td>
</tr>
<tr>
<td>Inference Model</td>
<td>deduction</td>
<td>induction</td>
</tr>
<tr>
<td>Query Language</td>
<td>artificial</td>
<td>natural</td>
</tr>
<tr>
<td>Query Specification</td>
<td>complete</td>
<td>incomplete</td>
</tr>
<tr>
<td>Items wanted</td>
<td>matching</td>
<td>relevant</td>
</tr>
<tr>
<td>Error Response</td>
<td>sensitive</td>
<td>insensitive</td>
</tr>
</tbody>
</table>

**Web search**

- The Web is not a DB
  - no structure
  - can be seen as a flat collection of documents (What search engines did at the beginning, e.g., AltaVista)
- Yet the Web has a meta-structure:
  - it is a graph
  - using the information embedded in the graph led to a major breakthrough in IR (e.g., PageRank → Google)
Search: The Basic Concepts

- The user has an information need, that is expressed as a free-text query
- The query is a “document”, to be compared to a collection of documents
- **Effectiveness vs Efficiency**
  - How to compare documents? Similarity metrics needed!
  - How to avoid doing a sequential search? Can we search in parallel in a set of servers?

Basic Concepts

- Two Main Stages
  - Indexing process (build knowledge base)
    - Involves pre-processing and storing of information in a repository
  - Retrieval process (search knowledge base)
    - Involves issuing a query, accessing the repository to find candidate documents most similar to query, and browse them
    - Basic object is a document
**Search Preprocessing Stage - Indexing**

- Analyze the distribution of indexing units (words) within a collection of documents
- The **inverted index**, maps every indexing unit (word) to a list of postings (documents ids associated with frequency of occurrence, location information, etc...)

**Inverted Index**

- Given the texts $T_0 = "it is what it is", T_1 = "what is it"$ and $T_2 = "it is a banana"
- We have the following full inverted index:
  - "a": \{(2, 2)\}
  - "banana": \{(2, 3)\}
  - "is": \{(0, 1), (0, 4), (1, 1), (2, 1)\}
  - "it": \{(0, 0), (0, 3), (1, 2), (2, 0)\}
  - "what": \{(0, 2), (1, 0)\}
- In bold face the postings associated to document $T_1$
The Actual Search Process

- At retrieval time, the query is indexed. Postings are fetched, and analyzed to return the most similar documents
- Effectivity constraint: “The entire process should not take more than a second per query”

Classical IR Models

- **Boolean model** (Does the term occur in document?)
  - Simple model based on set theory
  - Queries are specified as Boolean expressions
- **Vector space model** (Similarity between vectors)
  - Query, Documents are represented as vectors in a N-dimensional space, N is the number of terms in the corpus
  - Find documents most similar to the query in that space
- **Probabilistic model** (Find probabilistically best answer)
  - Goal: model IR with probabilistic framework
  - Given a user’s query find out the document with the higher probability to be relevant to the query.
Text Preprocessing Techniques

- Goal: construct a canonical form of the document text called *document profile*
- Stages
  - Sentence splitting – sentence separation
  - Tokenization - word separation
  - Normalization – changing terms to a standard form (e.g., lowercase)
  - Stop-word filtering: ignores frequent terms (e.g. *is, the, as, I, at, in, to...*)
  - Lemmatization – reducing terms to their base form
  - Map between documents and indexing units

Retrieving – Query Evaluation, Ranking

**Query evaluation steps:**
1. Input query
2. Parse query
3. Lexicon lookup
4. Get posting from inverted index
5. Weights accumulator
6. Sort by weights
Evaluation Criteria

- **Effectiveness**
  - How precise is the answer set to the information needs?

- **Efficiency**
  - Retrieval time, indexing time, index size?

- **Usability and Personalization**
  - Learnability, novice use, expert use

Web IR - IR on the Web

- **First Generation**
  - Classical approach
  - Informational: IR/DB techniques on page content. E.g., Lycos, Excite, AltaVista

- **Second Generation**
  - Web as a graph
  - Navigational: use off-page Web specific data – links topology. E.g., Google

- **Third Generation**
  - Open research
  - A lot of business potential, “monetarization of infomediary role”, matching services
Problems with Using IR for Web

- Very large and heterogeneous collection
- Very short queries
- Unsophisticated users
- Difficult to judge relevance and to rank results
- Synonymy and ambiguity
- Authorship styles
- Search engine persuasion, keyword *stuffing* (a web page is loaded with keywords in the meta tags or in content)

Indexing/Retrieval

- Information Retrieval consists of 2 main stages
  - Indexing - pre-processing and storing of information into a repository (*an Index*)
  - Retrieval - issuing a query, accessing the index to find documents *RELEVANT* to the query
Typical IR system

Basic Concepts

- **Document**
  - any piece of information (book, article, database record, image)
  - usually textual data

- **Query**
  - a text representing the user’s information need

- **Relevance**
  - A relation (binary or numeric) between documents and queries \( R(d,q) \)
How to measure Document Relevance

- Although Relevance is a basic concept in IR, there is still no unique definition
- Difficulties
  - User dependent (experienced vs. novice)
  - Time dependent (e.g., news topics)
  - Geography dependent (Paris-FR vs. Paris-USA)
- Simplified (binary) assumption
  - $D$ – the set of all documents in the world,
  - $Q$ - the set of all queries in the world,
  - $R: D \times Q \rightarrow \{0,1\}$ is well defined:
    $d$ is “relevant” to $q$ if $R(d,q) = 1$

IR main task

- retrieve the “relevant” documents to the user’s query

![Diagram of documents and relevant documents]
Index building: Text Profiling

- Both documents and queries are profiled to generate a canonical representation.
- Profile is usually based on the set of indexing units in the text.
- Indexing units are generally representative terms in the text.
  - How to select representative terms?
  - For the moment, let’s say all the words in the given document/query.

Let us formalize a bit

- Given a collection of documents (a corpus).
  - All the terms in the collection are labeled as: $T = \{t_1, t_2, \ldots, t_N\}$.
  - Profile of a document $d_i$ is a vector of size $N$: $d_i \rightarrow (w_{i1}, w_{i2}, \ldots, w_{iN})$

$$w_{ij} = \begin{cases} 
0 & \text{if } t_i \text{ does not appear in } d_j \\
\text{num. of appears of } t_i \text{ in } d_j & \text{otherwise}
\end{cases}$$
Index Representation - sparse matrix

<table>
<thead>
<tr>
<th>A(t,d)</th>
<th>d_1</th>
<th>d_2</th>
<th>...</th>
<th>d_M</th>
</tr>
</thead>
<tbody>
<tr>
<td>t_1</td>
<td>w_{11}</td>
<td>w_{12}</td>
<td></td>
<td>w_{1M}</td>
</tr>
<tr>
<td>t_2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t_N</td>
<td>w_{N1}</td>
<td></td>
<td></td>
<td>w_{NM}</td>
</tr>
</tbody>
</table>

Using the index for Relevance Retrieval

- **Assumption**: a document not containing any query term is not relevant (*think about that! Is it true?*)
- Given a simple query of one term q={t_i}
- To find the relevant documents
  1. Retrieve all the documents d_j with w_{ij} > 0
  2. Sort them in decreasing order
  3. Return the sorted list of “relevant” documents to the user
- In general: given a user’s query q={t_1, t_2,...,t_k} return the sorted list of documents that contain at least one of the query terms.
The Boolean Model

- Simple model based on set theory
- Queries are specified as Boolean expressions
  - Indexing units are words
  - Boolean Operator: OR, AND, NOT
  - Example:
    \[ q = \text{“java” AND “compilers” AND (“unix” OR “linux“)} \]
- **Relevance:** A document is relevant to the query if it satisfies the Boolean expression of the query.

### Boolean Model - Example

<table>
<thead>
<tr>
<th>A(t,d)</th>
<th>d₁</th>
<th>d₂</th>
<th>d₃</th>
<th>d₄</th>
<th>d₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ q = a \text{ AND (b OR NOT c)) } \]
Search the other way around

- a -> D1, D2, D3, D5
- b -> D2, D4, D5
- c -> D3, D5

- a -> 1, 1, 1, 0, 1
- b -> 0, 1, 0, 1, 1
- NOT c -> 1, 1, 0, 1, 0

\[ q = a \land (b \lor (\neg c)) \]

Results: d1, d2, d5

Boolean Model – pros & cons

- **Pros:**
  - Fast (bitmap vector operations)
  - Binary decision (doc is “relevant” or not)
- **Cons:**
  - Binary decision – How to rank?
  - Query can fail (no result)
  - Who speaks Boolean?
  - Still very popular by commercial IR systems
Vector Model

- Documents are represented as vectors in a N-dimensional space
  - N is the number of terms in the corpus
- Query is a document like any other document
- Relevance – measured by similarity between vectors:
  - A document is relevant to the query if its vector is similar to the vector of the query.

Documents as Vectors

![Diagram](image_url)
Documents in 3D Space

Vector-space Model

- Represent the query as vector in the same document vector-space
- Relevance is measured by similarity
- Measure the cosines of the angle between doc-vectors and the query vector

\[
Sim(q,d) = \frac{q \cdot d}{|q| \cdot |d|} = \frac{\sum_i w_{iq} \cdot w_{id}}{\sqrt{\sum_i w_{iq}^2 \sum_i w_{id}^2}}
\]

for \(N=2\)

\[
= \frac{q_x d_x + q_y d_y}{\sqrt{(q_x^2 + q_y^2)(d_x^2 + d_y^2)}}
\]
Example

\[ Sim(q, d) = \frac{q \cdot d}{|q| \cdot |d|} = \frac{\sum w_{iq} w_{id}}{\sqrt{\sum w_{iq}^2 \sum w_{id}^2}} \]

\[ \text{sim}(q, d_i) = \frac{(0.4 \cdot 0.8) + (0.8 \cdot 0.3)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.8)^2 + (0.3)^2]}} \]

\[ = \frac{0.56}{\sqrt{0.58}} = 0.74 \]

\[ \text{sim}(q, d_j) = \frac{(0.4 \cdot 0.2) + (0.8 \cdot 0.7)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.2)^2 + (0.7)^2]}} \]

\[ = \frac{0.64}{\sqrt{0.42}} = 0.98 \]

Example 2

Query: (k1 k2 k3)

<table>
<thead>
<tr>
<th></th>
<th>k1</th>
<th>k2</th>
<th>k3</th>
<th>q • dj</th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>d2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>d3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>d4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>d5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>d6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>d7</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>q</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Binary weights for the sake of the simplicity
Example 3

<table>
<thead>
<tr>
<th>k1</th>
<th>k2</th>
<th>k3</th>
<th>q • dj</th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>d2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>d3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>d4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>d5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>d6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>d7</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

q 1 2 3

How to determine the w(t,d) weights?

- Binary weights:
  - \( w_{ij} = 1 \) if document \( d_j \) contains term \( t_i \), otherwise 0.
  - Actually it is the Boolean model

- Term Frequency (tf):
  - \( w_{ij} = \text{(number of occurrences of } t_i \text{ in } d_j) \)

- Inverse Document Frequency (idf):
  - E.g., q=“galaxy in space”.
  - Is the occurrence of the query term “in” in a document should contribute the same as the occurrence of the query term “galaxy”? 
How to determine the \( w(t,d) \) weights?

- \( tf \times idf \) weighting scheme (Salton 73)
  - \( tf \) – **term frequency**: a monotonic function of the term frequency in the document,
    - e.g., \( tf(t,d) = \log(freq(t,d) + 1) \)
    - e.g., \( tf(t,d) = \frac{freq(t,d)}{\text{Max}_{u \in d}\{freq(u,d)\}} \)
  - \( idf \) – the **inverse document frequency of a term**: a decreasing function of the term total frequency \( N_t \) = the number of documents in the corpus where \( t \) occurs
    - e.g., \( idf(t) = \log(N / N_t) \)
  - \( w_{ij} = tf(t_i,d_j) \times idf(t_i) \)

Computing TF-IDF -- An Example

- Given a document \( D \) containing terms (a, b, and c) with given frequencies:
  - \( \text{freq}(a,D)=3, \text{freq}(b,D)=2, \text{freq}(c,D)=1 \)
- Assume collection contains 10,000 documents and the term total frequencies of these terms are:
  - \( N_a=50, N_b=1300, N_c=250 \)
- Then:
  - a: \( tf = 3/3; \text{idf} = \log(10.000/50) = 5.3; \text{tf-idf} = 5.3 \)
  - b: \( tf = 2/3; \text{idf} = \log(10.000/1300) = 2.0; \text{tf-idf} = 1.3 \)
  - c: \( tf = 1/3; \text{idf} = \log(10.000/250) = 3.7; \text{tf-idf} = 1.2 \)
Vector-Space pros & cons

Pros:
- Terms weighting scheme improves retrieval effectiveness
- Approximate query matching
- Cosine similarity is a good ranking measure

Cons:
- Terms are not independent of all other terms
- Normalization makes incrementally difficult as recomputation is needed for each added term
- Considered superior to other models due to its simplicity and elegance.

Reminder

Free text -> indexing -> profile -> storing -> retrieving

Repository index

Free text

Indexing -> Retrieving

Retrieving
Inverted Index

- Purpose: to support efficient retrieval, avoid scanning the entire collection sequentially
- Retrieval should be proportional to the size of the query rather than to size of collection.

Inverted Index

- This is the primary data structure for text indexes
- Main Idea:
  - *Invert documents descriptions into a big index*
- Basic steps:
  - Make a “dictionary” of all the tokens in the collection
  - For each token, list all the docs it occurs in
Inverted Indexes

An Inverted File is a vector file “inverted” so that rows become columns and columns become rows

<table>
<thead>
<tr>
<th>docs</th>
<th>t1</th>
<th>t2</th>
<th>t3</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D7</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D8</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D9</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D10</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Terms

<table>
<thead>
<tr>
<th>Terms</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>D7</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Evaluation Criteria

- **Effectiveness**
  - How precise is the answer set to the information need

- **Efficiency**
  - Retrieval time, indexing time, index size

- **Usability**
  - Learnability, novice use, expert use
Effectiveness Evaluation

- **User-centered strategy**
  - Try several variations of the retrieval system
  - Measure which variation works the “best”
  - Given several users, and at least 2 retrieval systems
  - Have each user try the same task on both systems (within groups)
  - Or have each user try the same task on one system (between groups)
  - Measure which system works the “best” (for the “average” user)

IR Test Collection

- A Collection of Documents
  - Representative sources and quantity
- A Set of Topics
  - Used to form queries
- Relevance judgments
  - For each document, with respect to each topic
  - This is the expensive part!
Goal of the evaluation

- Given a (generic) user and an information retrieval system
- We want to measure how good is a user, using the system, in retrieving documents relevant to a set of given topics

Relevance Table

<table>
<thead>
<tr>
<th></th>
<th>d1</th>
<th>d2</th>
<th>....</th>
<th>d_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>topic1</td>
<td>+</td>
<td>-</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>topic2</td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>topic_M</td>
<td>-</td>
<td>+</td>
<td></td>
<td>+</td>
</tr>
</tbody>
</table>
### Traditional MOE

- **Precision**
  - How much of what was found is relevant?
  - Particularly for interactive searching

- **Recall**
  - How much of what is relevant was found?
  - Particularly important for law and patent searches

- **Fallout**
  - How much of what was irrelevant was rejected?
  - Useful when different size collections are compared

### The Contingency Table

<table>
<thead>
<tr>
<th>Doc</th>
<th>Action</th>
<th>Retrieved</th>
<th>Not Retrieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant</td>
<td>Relevant</td>
<td>Relevant Retrieved</td>
<td>Relevant Rejected</td>
</tr>
<tr>
<td>Not relevant</td>
<td>Irrelevant</td>
<td>Irrelevant Retrieved</td>
<td>Irrelevant Rejected</td>
</tr>
</tbody>
</table>

- **Precision** = \( \frac{\text{Relevant Retrieved}}{\text{Retrieved}} \)
- **Recall** = \( \frac{\text{Relevant Retrieved}}{\text{Relevant}} \)
- **Fallout** = \( \frac{\text{Irrelevant Rejected}}{\text{Not Relevant}} \)
Precision vs. Recall

\[
\text{Precision} = \frac{|\text{Rel Retrieved}|}{|\text{Retrieved}|}, \quad \text{Recall} = \frac{|\text{Rel Retrieved}|}{|\text{Rel in Collection}|}
\]

Why Precision and Recall?

Get as much good stuff while at the same time getting as little junk as possible.

High Recall – all the truth
High Precision – nothing but the truth
Retrieved vs. Relevant Documents

Very low precision, very low recall (0 in fact)

Retrieved vs. Relevant Documents

Very high precision, very low recall
Retrieved vs. Relevant Documents

High recall, low precision

Retrieved vs. Relevant Documents

High precision, high recall (at last!)
Precision and recall

A typical precision and recall curve

Recommender Systems
Original Definition of RS

- In everyday life we rely on recommendations from other people either by word of mouth, recommendation letters, movie and book reviews printed in newspapers ...
- In a typical recommender system people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients
  - Aggregation of recommendations
  - Match the recommendations with those searching for recommendations

[Resnick and Varian, 1997]

Recommender Systems

- A recommender system helps to make choices without sufficient personal experience of the alternatives
  - To suggest products to their customers
  - To provide consumers with information to help them decide which products to purchase
- They are based on a number of technologies: information filtering, machine learning, adaptive and personalized system, user modeling, ...
- Not clear separation from Information Retrieval – [Burke, 2002] claims that is the “individualized” and “interesting and useful” features that make the difference.
A Simplified Model of Recommendation

1. Two types of entities: Users and Items
2. A background knowledge:
   - A set of ratings: a map \( R: \text{Users} \times \text{Items} \rightarrow [0,1] \cup \{?\} \)
   - A set of “features” of the Users and/or Items
3. A method for eliminating all or part of the ‘?’ values for some (user, item) pairs – substituting ‘?’ with the true values
4. A method for selecting the items to recommend
   - Recommend to \( u \) the item \( i^* \) such that:
     \[ i^* = \arg \max_{i \in \text{Items}} \{R(u,i)\} \]

[Adomavicius et al., 2005]

“Core” Recommendation Techniques

**U** is a set of users

**I** is a set of items/products

<table>
<thead>
<tr>
<th>Technique</th>
<th>Background</th>
<th>Input</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaborative</td>
<td>Ratings from <strong>U</strong> of items in <strong>I</strong>.</td>
<td>Ratings from <strong>u</strong> of items in <strong>I</strong>.</td>
<td>Identify users in <strong>U</strong> similar to <strong>u</strong>, and extrapolate from their ratings of <strong>I</strong>.</td>
</tr>
<tr>
<td>Content-based</td>
<td>Features of items in <strong>I</strong>.</td>
<td><strong>u</strong>’s ratings of items in <strong>I</strong>.</td>
<td>Generate a classifier that fits <strong>u</strong>’s rating behavior and use it on <strong>I</strong>.</td>
</tr>
<tr>
<td>Demographic</td>
<td>Demographic information about <strong>U</strong> and their ratings of items in <strong>I</strong>.</td>
<td>Demographic information about <strong>u</strong>.</td>
<td>Identify users that are demographically similar to <strong>u</strong>, and extrapolate from their ratings of <strong>I</strong>.</td>
</tr>
<tr>
<td>Utility-based</td>
<td>Features of items in <strong>I</strong>.</td>
<td>A utility function over items in <strong>I</strong> that describes <strong>u</strong>’s preferences.</td>
<td>Apply the function to the items and determine <strong>I</strong>’s rank.</td>
</tr>
<tr>
<td>Knowledge-based</td>
<td>Features of items in <strong>I</strong>. Knowledge of how these items meet a user’s needs.</td>
<td>A description of <strong>u</strong>’s needs or interests.</td>
<td>Infer a match between <strong>I</strong> and <strong>u</strong>’s need.</td>
</tr>
</tbody>
</table>

[Burke, 2002]
The Collaborative Filtering Idea

- Trying to predict the opinion the user will have on the different items
- Be able to recommend the “best” items to each user based on the user’s previous likings and the opinions of other like minded users
- CF is a typical Internet application – it must be supported by a networking infrastructure
  - At least many users and one server
  - But we are thinking of using many servers
- There is no stand alone CF application.

Movie Lens

Welcome to MovieLens!
Free, personalized, non-commercial, ad-free, great movie recommendations. Have questions? Take the MovieLens Tour for answers.
Not a member? Join MovieLens now.
Need a gift idea? Try MovieLens QuickPick!

New to MovieLens?
Join today!
You get great recommendations for movies while helping us do research. Learn more:
- Try out QuickPick: Our Movie Gift Recommender
- Take the MovieLens Tour
- Read our Privacy Policy
- See our Browser Requirements
- Learn about Our Research

Hello MovieLens Users!
Please login:
Username:
Password:
Save login: 
Log into MovieLens
Forgot your password?
Next member? Join now

MovieLens is a free service provided by GroupLens Research at the University of Minnesota. We sometimes study how our members use MovieLens in order to learn how to build better recommendation systems. We promise to never give your personal information to anyone. See our privacy policy for more information.
Welcome to the new MovieLens!

Existing MovieLens users: We’d like to welcome you back to MovieLens, and let you know we have a new MovieLens FAQ you might want to read. We hope you like what you will see!

Take me to MovieLens!

New MovieLens users: Thank you for joining MovieLens! In order to generate personalized movie recommendations, we need to know a little about what movies you have already seen. MovieLens will now display several lists of movies. If you have seen any of the listed movies, please rate them using the rating scale shown below.

Ratings are on a scale of 1 to 5:

- ★★★★★ = Must See
- ★★★★☆ = Will Enjoy
- ★★★☆☆ = It’s OK
- ★★☆☆☆ = Fairly Bad
- ★☆☆☆☆ = Awful

Remember: the more movies you rate, the more accurate MovieLens' predictions will be.

To rate a movie, just click on the pulldown next to the title of a movie you have seen. Blue stars will appear to indicate that your rating has been received.

I'm ready to start rating!

So Far you have rated 0 movies. MovieLens needs at least 15 ratings from you to generate predictions for you. Please rate as many movies as you can from the list below.

<table>
<thead>
<tr>
<th>Your Rating</th>
<th>Movie Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>★★★</td>
<td>Austin Powers: International Man of Mystery (1997) Action, Adventure, Comedy</td>
</tr>
<tr>
<td>★★★★☆</td>
<td>Contact (1997) Drama, Sci-Fi</td>
</tr>
<tr>
<td>???</td>
<td>Eraser (1996) Action, Drama, Thriller</td>
</tr>
<tr>
<td>???</td>
<td>Maverick (1994) Action, Comedy, Western</td>
</tr>
<tr>
<td>★★★★☆</td>
<td>Philadelphia (1993) Drama</td>
</tr>
<tr>
<td>★★★</td>
<td>Penno, The (1993) Drama, Romance</td>
</tr>
<tr>
<td>???</td>
<td>Toy Story 2 (1999) Adventure, Animation, Children, Comedy, Fantasy</td>
</tr>
</tbody>
</table>

To get a new set of movies click the next> link.
Welcome ricci@itc.it (Log Out)
You've rated 16 movies.
You're the 298th visitor in the past hour.

35

New Forum Messages

- Message Subject: Best front page redesign thoughts
  - Author: robert, krati
  - Date: 2006-03-27
- Message: What's the best thing you watched and what did you rate it?
- Message: Upcoming films you are most looking forward to
  - Author: Viju
  - Date: 2006-03-27

Recently Applied Tags (tag your movies) (more about tags)
- My DVDs (121), revolution1 (1), Futureofmovies.com (136), 1994 (1), gumpak (1), john hurst (1), dry hung (1), most virginia (1), cannes pork (1), cannes (1).

Welcome ricci@itc.it (Log Out)
You've rated 16 movies.
You're the 298th visitor in the past hour.

You've searched for all titles.
Found 6273 movies, sorted by Prediction.
Genres: All | Include Genre: None
Order: All | Domains: All | Format: All | Languages: All
Show Print-friendly Page | Download Results | Suggest a Title
Tags Related to Your Search: In Netflix movies (178), Futureofmovies.com (134), My DVDs (121), Oscar (Best Cinematography) (98), Oscar (Best Picture) (85), (about tags)

Page 1 of 548 | Go to page: 1...10...218...357...406...545...last

Page 2:
- Cat Returns, The (tekko no enjokoro) (2006) DVD
  - Info: info, mid
  - Popular tags: anime, cats, to, tokyo, quack
- Immigrant, The (1917) DVD
  - Info: info, mid
  - Add tag
  - Popular tags: comedy, silent
- Experiment, The (Gas Experiment) (2001) DVD
  - Info: info, mid
  - Popular tags: drama, thriller, german
- Thais (Thais) (1956)
  - DVD
  - Info: info, mid
  - Add tag
  - Popular tags: drama, horror, thriller, spanish
  - DVD
  - Info: info, mid
  - Popular tags: adventure, animation, children, fantasy, romance, japan
- Why We Fight (2005)
  - Info: info, mid
  - Add tag
  - Popular tags: history, in, netflix, quest, controversy
Collaborative Filtering

The CF Ingredients

- List of m Users and a list of n Items
- Each user has a list of items he/she expressed their opinion about (can be a null set)
- Explicit opinion - a rating score (numerical scale)
- Sometime the rating is implicitly – purchase records
- Active user for whom the CF prediction task is performed
- A metric for measuring similarity between users
- A method for selecting a subset of neighbors for prediction
- A method for predicting a rating for items not currently rated by the active user.
Collaborative-Based Filtering

The collaborative based filtering recommendation techniques proceeds in these steps:
1. For a target/active user (the user to whom a recommendation has to be produced) the set of his ratings is identified.
2. The users more similar to the target/active user (according to a similarity function) are identified (neighbor formation).
3. The products bought by these similar users are identified.
4. For each one of these products a prediction - of the rating that would be given by the target user to the product - is generated.
5. Based on this predicted rating a set of top \( N \) products are recommended.

A Simplified Model of Recommendation

1. Two types of entities: Users and Items
2. A background knowledge:
   - A set of ratings: a map \( R: \text{Users} \times \text{Items} \rightarrow [0,1] \cup \{?\} \)
   - A set of "features" of the Users and/or Items
3. A method for eliminating all or part of the '?' values for some (user, item) pairs - substituting '?' with the true values:
   \[
   R(u, i) = \text{Average} \{R(su, i)\} \quad \text{su is similar to } u
   \]
4. A method for selecting the items to recommend
   - Recommend to \( u \) the item \( i^* = \arg \max_{i \in \text{Items}} \{R(u, i)\} \)

[Adomavicius et al., 2005]
Nearest Neighbor Collaborative Filtering

Current User

Users

User Model = interaction history

Current User Users Items

1st item rate 14th item rate

1st item rate 14th item rate

Hamming distance

5 6 6 5 4 8 Nearest Neighbor

Nearest Neighbor can be easily wrong

Current User

Users

User Model = interaction history

Current User Users Items

1st item rate

This is the only user having a positive rating on this product

1st item rate

Hamming distance

5 6 6 5 4 8 Nearest Neighbor
Matrix of ratings

Collaborative-Based Filtering

- A collection of user $u_i$, $i=1, \ldots, n$ and a collection of products $p_j$, $j=1, \ldots, m$
- A $n \times m$ matrix of ratings $v_{ij}$, with $v_{ij} = ?$ if user $i$ did not rate product $j$
- Prediction for user $i$ and product $j$ is computed as
  \[ v_{ij}^* = v_i + K \sum_{k \neq i, v_{ij} = ?} u_{ik} (v_{kj} - v_k) \]
- Where, $v_i$ is the average rating of user $i$, $K$ is a normalization factor such that the sum of $u_{ik}$ is 1, and
  \[ u_{ik} = \frac{\sum_j (v_{ij} - v_i)(v_{jk} - v_k)}{\sqrt{\sum_j (v_{ij} - v_i)^2 \sum_j (v_{jk} - v_k)^2}} \]
- Similarity of users $i$ and $k$
- Where the sum (and averages) is over $j$ s.t. $v_{ij}$ and $v_{kj}$ are not “?”.

[Matrix of ratings]

[Collaborative-Based Filtering diagram]
Example

Users' similarities: $u_{i5} = 0.5, u_{i8} = 0.5, u_{i9} = 0.8$

$$v^*_{ij} = v_i + K \sum_{v_{ik} \neq v_{ij}} u_{ik} (v_{kj} - v_k)$$

$$v^*_{ij} = 3.2 + 1/(0.5+0.5+0.8) \times [0.5 (4 - 4) + 0.5 (3 - 3.5) + 0.8 (5 - 3)]$$

$$= 3.2 + 1/1.8 \times [0 - 0.25 + 1.6] = 3.2 + 0.75 = 3.95$$