Context Awareness and Adaptation in Recommendation

Bamshad Mobasher
DePaul University, Chicago, IL, USA

DMRS Workshop – Oct. 22-23, 2015 – Bolzano, Italy
Context in Recommendation
Context in Recommendation
Context-Aware Recommendation

Watching alone  Dislike

Watching with a kid  Like
Outline

• General views of context and their relevance to recommendation problem
  – Representational versus Interactional view
  – Background on Integrating context in recommender systems
  – Characterizing the environment for context aware recommendation

• Highlighted Approaches in Context Aware Recommendation
  – Similarity-Based Context-Aware Matrix Factorization
  – Context Adaptation with Dynamic Latent Variable Models
  – Context Adaptation with Exploration/Exploitation
• Yong Zheng

• Negar (Nikki) Hariri

• Robin Burke
Paul Dourish (2004) distinguished between two views of context:

- **Representational view:**
  - Context is information that can be described using a set of “appropriate” variables that can be observed and are distinguishable from features describing the underlying activity.

- **Interactional View of Context:**
  - The scope of contextual features is defined dynamically, and is occasioned rather than static.
  - Context gives rise to the activity and activity changes the context.
Representational View

• Context can be represented as an explicit, enumerated set of static attributes (i.e., it’s “extensional”)
  – Attributes are predefined based on the characteristics of the domain and environment
  – E.g., time, date, location, mood, device, etc.

• Implications:
  – Relevant contextual variables (and their structures) must be identified at the design stage
  – Must identify & acquire explicit contextual information before recommendations are made
Contextual Recommendation in a Representational Framework

<table>
<thead>
<tr>
<th>User</th>
<th>Movie</th>
<th>Time</th>
<th>Location</th>
<th>Companion</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>Titanic</td>
<td>Weekend</td>
<td>Theatre</td>
<td>Family</td>
<td>4</td>
</tr>
<tr>
<td>U2</td>
<td>Titanic</td>
<td>Weekday</td>
<td>Home</td>
<td>Family</td>
<td>5</td>
</tr>
<tr>
<td>U3</td>
<td>Titanic</td>
<td>Weekday</td>
<td>Theatre</td>
<td>Alone</td>
<td>4</td>
</tr>
<tr>
<td>U1</td>
<td>Titanic</td>
<td>Weekday</td>
<td>Home</td>
<td>Alone</td>
<td>?</td>
</tr>
</tbody>
</table>

- **Traditional RS**: Users × Items → Ratings
- **Contextual RS**: Users × Items × Context → Ratings
Representational Recommendation Frameworks

From Adomavicius & Tuzhilin, 2008
Interactional View

• Properties of Context
  – Context gives rise to a behavior that is observable, though context itself may not be (it’s “intensional”)
  • Exists (usually implicitly) in relation to the ongoing interaction of the user with the system
  • Can be inferred: a stochastic process with $d$ states $\{c_1, c_2, \ldots, c_d\}$ representing different contextual conditions

• Context aware recommendation
  – Explicit representation of context not as important as
  • recognizing behavior arising from the context
  • adapting to the needs of the user within the context
  • recognizing and adapting to context transitions
An Interactional Framework for Contextual Recommendation

Inspired by Atkinson and Shrifflin’s model of human memory

[Anand and Mobasher, 2007]
### Characterizing the Environment for CARS

<table>
<thead>
<tr>
<th>How Contextual Factors Change</th>
<th>Knowledge of the RS about the Contextual Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>Fully Observable</td>
</tr>
<tr>
<td></td>
<td>Partially Observable</td>
</tr>
<tr>
<td></td>
<td>Unobservable</td>
</tr>
<tr>
<td>Dynamic</td>
<td>Everything Known about Context</td>
</tr>
<tr>
<td></td>
<td>Partial and Static Context Knowledge</td>
</tr>
<tr>
<td></td>
<td>Latent Knowledge of Context</td>
</tr>
<tr>
<td></td>
<td>Context Relevance Is Dynamic</td>
</tr>
<tr>
<td></td>
<td>Partial and Dynamic Context Knowledge</td>
</tr>
<tr>
<td></td>
<td>Nothing Is Known about Context</td>
</tr>
</tbody>
</table>

Contextual Recommendation Algorithms

- **Extensions of standard collaborative filtering**
  - CF after Item / user splitting pre-filters
  - Differential Context Modeling

- **Heuristic distance-based approaches**
  - Extend items-item, user-user similarities to include contextual dimensions
    - Requires similarity/distance metrics for various contextual dimensions

- **Approaches based on matrix/tensor factorization**
  - Tensor = Users x Items x Contexts; then apply higher-order tensor factorization
    - Context-Aware Matrix Factorization
    - Factorization Machines

- **Probabilistic latent variable context models**

- **Models based on active learning, e.g., Exploration/Exploitation**
Highlighted Approach

Similarity-Based Context-Aware Matrix Factorization

**Pedigree:**
- Representational context
- Static environment
- Observable context information
Two general approaches for representational contextual modeling

1. Independent Contextual Modeling

Tensor Factorization, ACM RecSys 2010
Two general approaches for representational contextual modeling

2. Conditional (Dependent) Contextual Modeling

Deviation-Based Contextual Modeling
Baltrunas, et al., Context-aware Matrix Factorization, ACM RecSys 2011
Zheng, et al., Contextual Sparse Linear Method, ACM RecSys 2014

Similarity-Based Contextual Modeling
Zheng, et al., Similarity-Based Contextual Sparse Linear Method, UMAP 2015
Zheng, et al., Similarity-Based Contextual Recommendation, WISE 2015
Matrix Factorization of Ratings Data

- $p_u$ is the user-factor vector
- $q_i$ is the item-factor vector
- The latent factors may represent combinations of features or characteristics of movies and users that explain ratings

<table>
<thead>
<tr>
<th>User</th>
<th>HarryPotter</th>
<th>Batman</th>
<th>Spiderman</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>5</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>U2</td>
<td>?</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>U3</td>
<td>4</td>
<td>2</td>
<td>?</td>
</tr>
</tbody>
</table>

\[ \hat{r}_{ui} = p_u \cdot q_i \]
Matrix Factorization

Rating Prediction Function: $q_i^T p_u$

Goal: To learn the user and item vectors in order to minimize the regularized squared error on the known ratings in the data

The loss function is:

$$\min_{q^*, p^*} \sum_{(u,i) \in \mathcal{K}} (r_{ui} - q_i^T p_u)^2 + \lambda (\| q_i \|^2 + \| p_u \|^2)$$

Using gradient descent as the optimizer, the user and item vectors can be updated iteratively:

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$

$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$
Biased Matrix Factorization

Rating Prediction Function: 
\[ \hat{r}_{ui} = \mu + b_u + b_i + \mathbf{p}_u \cdot \mathbf{q}_i \]

Goal: To learn the user and item vectors in order to minimize the regularized squared error on the known ratings in the data

The loss function is: 
\[
\min_{p^*,q^*,b^*} \sum_{(u,i) \in \mathcal{K}} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda (\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2)
\]

Using gradient descent as the optimizer, the user and item vectors can be generated accordingly.
Context-aware MF (CAMF)

- CAMF was first proposed by Baltrunas et al., 2011

Basic MF:  \( \hat{r}_{ui} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \)

Biased MF:  \( \hat{r}_{ui} = \mu + b_u + b_i + \overrightarrow{p_u} \cdot \overrightarrow{q_i} \)

CAMF:  \( \hat{r}_{uick_1c_k2...c_kL} = \mu + b_u + \sum_{j=1}^{L} B_{ijck,j} + \overrightarrow{p_u} \cdot \overrightarrow{q_i} \)

CAMF replaced term \( b_i \) by \( \sum_{j=1}^{L} B_{ijck,j} \) which denotes the aggregated contextual rating deviation for a specific context and item pair
Similarity-Based CAMF

Basic MF: \[ \hat{r}_{ui} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \]

Biased MF: \[ \hat{r}_{ui} = \mu + b_u + b_i + \overrightarrow{p_u} \cdot \overrightarrow{q_i} \]

CAMF-Dev: \[ \hat{r}_{uic_{k,1}c_{k,2}...c_{k,L}} = \mu + b_u + \sum_{j=1}^{L} B_{ijc_{k,j}} + \overrightarrow{p_u} \cdot \overrightarrow{q_i} \]

CAMF-Sim: \[ \hat{r}_{uic_k} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \cdot \text{Sim}(c_k, c_E) \]

\( C_k \) denotes a context (e.g. \{Time=Morning, Location=Home\}) in which the item is rated;
\( C_E \) denotes the unknown or default contexts, e.g. \{Time=“”, Location=“”\};

The contextual rating prediction amounts to non-contextual predicted rating multiplied by the correlations between \( C_k \) and \( C_E \).
Similarity-Based CAMF

Similarity measures

1. Independent Context Similarity (ICS)
2. Latent Context Similarity (LCS)
3. Multidimensional Context Similarity (MCS)
Independent Context Similarity

For example:

$C_k = \{\text{Time} = \text{Weekend}, \text{Location} = \text{Home}\}$

$C_m = \{\text{Time} = \text{Weekday}, \text{Location} = \text{Office}\}$

The similarity between $C_k$ and $C_m$ is:

$\text{sim}(\text{Weekend, Weekday}) \times \text{sim}(\text{Home, Office})$

Contextual variables are assumed to be independent.
Latent Context Similarity

Each V is a vector over latent factors
- Learn latent factors for the whole context space
- Represent each context as a vector
- e.g. \(\text{sim}(\text{kids, family}) = V_{\text{kids}} \cdot V_{\text{family}}\)

\[\text{similarity}(c_{k,l}, c_{m,l}) = V_{c_{k,l}} \cdot V_{c_{m,l}}\]

Example:
Training: \(<\text{weekend, theater}> <\text{weekday, home}>\)
Testing: \(<\text{weekend, home}>\)
Multi-dimensional Context Similarity

Each contextual variable is represented by an axis in multidimensional space; each value in the variable is represented by a position in specific axis; a set of contextual conditions is thus denoted by a point in the space; the dissimilarity is the Euclidean distance between two points.

Need to learn the position of each contextual condition along its dimension
The general loss function in CAMF-Sim can be described as:

\[ \hat{r}_{uic_k} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \cdot \text{Sim}(c_k, c_E) \]

\[
\text{Minimize} \quad \frac{1}{2}(r_{uic_k} - \hat{r}_{uic_k})^2 + \frac{\lambda}{2}(\|\overrightarrow{p_u}\|^2 + \|\overrightarrow{q_i}\|^2 + \text{Sim}^2)
\]

Methods for each similarity measure

<table>
<thead>
<tr>
<th>ICS</th>
<th>LCS</th>
<th>MCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>The similarity (real valued) for each individual pair of context conditions</td>
<td>The vector representation (weights in factors) for each contextual condition</td>
<td>The positions (real values) for each contextual condition</td>
</tr>
</tbody>
</table>
Learning Process

\[ \hat{r}_{uic_k} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \cdot Sim(c_k, c_E) \]

Example (ICS):

\[ \text{sim}_1 = \text{similarity}(\text{“Time = Weekday”, “Time = N/A”}) \]
\[ \text{sim}_2 = \text{similarity}(\text{“Location = Home”, “Location = N/A”}) \]
\[ Sim(c_k, c_E) = \text{sim}_1 \times \text{sim}_2 \]

Minimize
\[ \frac{1}{2}(r_{uic_k} - \hat{r}_{uic_k})^2 + \frac{\lambda}{2}(\|\overrightarrow{p_u}\|^2 + \|\overrightarrow{q_i}\|^2 + Sim^2) \]

SGD Updates:

\[ err = r_{uic_k} - \hat{r}_{uic_k} \]
\[ \overrightarrow{p_u} = \overrightarrow{p_u} + \beta \cdot err \cdot \overrightarrow{q_i} \cdot Sim(c_k, c_E) \]
\[ \overrightarrow{q_i} = \overrightarrow{q_i} + \beta \cdot err \cdot \overrightarrow{p_u} \cdot Sim(c_k, c_E) \]
\[ \text{sim}_1 = \text{sim}_1 + \beta(\text{err} \cdot (\overrightarrow{p_u} \cdot \overrightarrow{q_i}) \cdot \text{sim}_2 - \alpha \cdot \text{sim}_1) \]
\[ \text{sim}_2 = \text{sim}_2 + \beta(\text{err} \cdot (\overrightarrow{p_u} \cdot \overrightarrow{q_i}) \cdot \text{sim}_1 - \alpha \cdot \text{sim}_2) \]
Data Sets and Metrics

- Context-aware data sets are usually limited and small.

<table>
<thead>
<tr>
<th>Rating Profiles</th>
<th>Restaurant</th>
<th>Music</th>
<th>Tourism</th>
</tr>
</thead>
<tbody>
<tr>
<td># of contextual dimensions, $F$</td>
<td>2</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td># of contextual conditions, $L$</td>
<td>7</td>
<td>34</td>
<td>67</td>
</tr>
<tr>
<td>Rating Scale</td>
<td>1-5</td>
<td>1-5</td>
<td>1-5</td>
</tr>
</tbody>
</table>

5-fold cross validation

Evaluation metrics:

- Precision: measuring the hit ratio of relevant items;
- MAP: taking the rankings of items into account.

$$MAP@N = \frac{1}{M} \sum_{i=1}^{M} ap@N/M,$$

$$ap@N = \frac{\sum_{k=1}^{N} P(k)}{\min(m, N)}$$
Experiments

• **Baseline:**
  – Tensor Factorization (TF)
  – Standard (Deviation-Based) CAMF

• **Our Approaches:**
  – CAMF-ICS
  – CAMF-LCS
  – CAMF-MCS
Results

X-axis denotes the number of recommended items
Findings

• **Similarity-based approach**: a more effective representation of contextual differences
  – compared to prior deviation-based approach

• **Some CAMF-Sim models always outperform TF and CAMF-Dev**

• **Representation of similarity matters**
  – CAMF-MCS is the often the best model; but computationally expensive

• **Sparsity is significant**
  – CAMF-LCS > CAMF-ICS
Dynamic Latent Variable Context Models

Pedigree:
• Interactional context
• Dynamic or partially dynamic environment
• Unobservable or partially observable context information
Latent Variable Context Models

• Generative approach to modeling user context

• Basic assumption:
  – users’ interactions involve a relatively small set of latent contextual states that can “explain” users’ behavior at different points during their interactions

• Have been used effectively in applications involving user’s performing informational or functional tasks

• Contexts correspond to sets or sequences of tasks/activities and are derived as latent factors from the observed user data
Static Latent Variable Models

Items → \( I_1, I_2, I_3, \ldots, I_n \)

Latent Factors → \( Z_1, Z_2, Z_k \)

User profiles → \( U_1, U_2, U_3, \ldots, U_m \)

\[
Pr(U_i, I_j) = \sum_{Z_k} Pr(U_i) \cdot Pr(Z_k | U_i) \cdot Pr(I_j | Z_k)
\]

I.e., The preference of a user for different items is encoded in the user’s membership in the latent classes
Inferring Latent Contexts From Sequences of User Interactions

- **Assumptions:**
  - Users’ interest on items is revealed sequentially
    - sequence of songs listened in a current playing session;
    - sequence of Web pages visited, etc.
  - Context is not explicit, but must be inferred from the activity of the users as they interact with the system

- **Example Domain: Music Recommendation**

- **Context may depend on many factors**
  - Types of user activity (exercising, relaxing, driving, dancing)
  - User’s moods or emotional states
  - Occasion or social setting
Context is reflected in the sequence of songs liked/disliked or played by the user in her current interaction with the system.
Example Usage Scenario:
Playlist Generation

1. The user selects an initial sequence of songs for the playlist.
2. The system *infers* user’s context and recommends a set of songs.
3. The user adds one of the recommendations (or a new song outside the recommendation set) to the playlist.
4. The system updates its knowledge about the user’s preferences before the next interaction.
LDA topic modeling to map user's interaction sequence to a sequence of latent topics

- Better at capturing more general trends in user's interests

The latent topics are generated from the top most frequent tags associated with songs

- Tags obtained from social tagging Web sites (e.g., last.fm)
- Tags characterize song features, user’s situation, mood, etc.
- For LDA, songs are taken as documents and tags as words
- After fitting the topic model for $K$ topics, the probability distribution over topics can be inferred for any given song
- For each song, a set of most dominant topics are selected
<table>
<thead>
<tr>
<th>Topic#1</th>
<th>Topic#2</th>
<th>Topic#3</th>
<th>Topic#4</th>
<th>Topic#5</th>
<th>Topic#6</th>
<th>Topic#7</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambient</td>
<td>latin</td>
<td>death</td>
<td>60s</td>
<td>chill</td>
<td>beautiful</td>
<td>electronic</td>
</tr>
<tr>
<td>instrumental</td>
<td>world</td>
<td>thrash</td>
<td>oldies</td>
<td>downtempo</td>
<td>sad</td>
<td>electronica</td>
</tr>
<tr>
<td>soundtrack</td>
<td>streamable</td>
<td>black</td>
<td>roll</td>
<td>chillout</td>
<td>mellow</td>
<td>house</td>
</tr>
<tr>
<td>classical</td>
<td>spanish</td>
<td>heavy</td>
<td>50s</td>
<td>christmas</td>
<td>melancholy</td>
<td>techno</td>
</tr>
<tr>
<td>beautiful</td>
<td>para</td>
<td>doom</td>
<td>rockabilly</td>
<td>lounge</td>
<td>acoustic</td>
<td>trance</td>
</tr>
<tr>
<td>age</td>
<td>bossa</td>
<td>brutal</td>
<td>top</td>
<td>electronic</td>
<td>chill</td>
<td>electo</td>
</tr>
<tr>
<td>chillout</td>
<td>fusion</td>
<td>melodic</td>
<td>500</td>
<td>trip-hop</td>
<td>soft</td>
<td>bass</td>
</tr>
<tr>
<td>experimental</td>
<td>musica</td>
<td>california</td>
<td>radio</td>
<td>electronica</td>
<td>slow</td>
<td>drum</td>
</tr>
<tr>
<td>movie</td>
<td>que</td>
<td>power</td>
<td>rolling</td>
<td>trip</td>
<td>melancholic</td>
<td>ambient</td>
</tr>
<tr>
<td>atmospheric</td>
<td>nova</td>
<td>progressive</td>
<td>1960s</td>
<td>ambient</td>
<td>favourite</td>
<td>beat</td>
</tr>
<tr>
<td>world</td>
<td>brazilian</td>
<td>gods</td>
<td>rhythm</td>
<td>hop</td>
<td>chillout</td>
<td>ambient</td>
</tr>
<tr>
<td>ethereal</td>
<td>african</td>
<td>seixas</td>
<td>time</td>
<td>easy</td>
<td>ballad</td>
<td>beat</td>
</tr>
<tr>
<td>chill</td>
<td>party</td>
<td>speed</td>
<td>elvis</td>
<td>cool</td>
<td>easy</td>
<td>idm</td>
</tr>
<tr>
<td>calm</td>
<td>brasil</td>
<td>swedish</td>
<td>soundtrack</td>
<td>sexy</td>
<td>life</td>
<td>experimental</td>
</tr>
<tr>
<td>electronic</td>
<td>espanol</td>
<td>old</td>
<td>american</td>
<td>radio</td>
<td>easy</td>
<td>club</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>minimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>party</td>
</tr>
</tbody>
</table>
## An Example Playlist (Mapped to Tags and Topics)

<table>
<thead>
<tr>
<th>Time</th>
<th>Popular Tags</th>
<th>Dominant Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Singer-songwriter, mellow, <em>relaxing</em>, chill, male vocalist, easy listening, <em>acoustic</em>, 00’s, <em>guitar</em>, <em>rock</em>, <em>happy</em></td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td><em>Alternative rock</em>, ballads, <em>calm</em>, beautiful, nice, soundtrack, favorites</td>
<td>6, 28</td>
</tr>
<tr>
<td>5</td>
<td><em>Electronic, electronica</em>, French, chill out, <em>trip-hop</em>, ambient, down-tempo, sexy, 90s, alternative, easy listening, guitar, mellow, <em>relax</em>, female vocal</td>
<td>7, 5</td>
</tr>
<tr>
<td>6</td>
<td>Soundtrack, 90s, <em>alternative</em>, atmospheric, female vocalist, <em>indie</em>, dreamy</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>Cover, Beatles cover, <em>rock</em>, 90s, soundtrack, brass , <em>pop rock</em>, <em>alternative</em>, <em>rock</em>, folk, brass</td>
<td>30, 18</td>
</tr>
</tbody>
</table>
An Example Playlist – Resulting Topic Sequences

• **Topic-based representation of the active session:**
  \[ h = [6, 6, 6, 20, 23, 6, 28, 7, 5, 23, 6, 25, 30, 18, 6, 20] \]

• **Assuming selected dominant topics for each song are independent, the active session is broken down into multiple sequences**
  \[ h_1 = [6, 6, 6, 6, 7, 23, 6, 30, 6] \]
  \[ h_2 = [6, 6, 20, 6, 7, 23, 6, 30, 6] \]
  \[ h_3 = [6, 6, 23, 6, 7, 23, 6, 30, 6] \]
  \[ h_4 = [6, 6, 6, 28, 7, 23, 6, 30, 6] \]
  \[ h_5 = [6, 6, 20, 28, 7, 23, 6, 30, 6] \]
  ...

Using a training set of playlists, sequential patterns are mined over the set of corresponding latent topic sequences

- Each pattern represents a frequent sequence of transitions between topics/contexts
- Given a user's current interaction (the sequence of last $w$ songs in the playlist), the discovered patterns are used to predict the context for the next song
Why Topic Level Aggregation?

- Mining SPs on topics instead of songs is useful in capturing user interests based on common characteristics of the current context.
- Makes it easier to track and detect changes in the users’ preferences due to changes in contextual states.
- Topic-based patterns are useful in managing the cold start problem: a new songs may still match topic-based patterns.
Song Recommendation Based on Contextual Post-filtering

- The predicted topics are used to contextualize the recommendations

\[
\text{contextScore}(h_u, s) = \frac{\sum_{t_i \in \text{predictedTopics}(h_u)} p(t_i | s)}{|\text{predictedTopics}(h_u)|}
\]

- \(\text{ContextScore}(h_u, s)\) represents the suitability of song \(s\) for the current context of user \(u\) (determined based on user’s active session, \(h_u\))

- Next, recommendations are re-ranked using the contextual information
  - Prediction score for a song \(s\): a linear combination of CF predicted rating and the context score for \(s\).
Evaluation

• **Dataset and Methodology:**
  – 28,963 user-contributed playlists from *Art of the Mix* website in January 2003
  – This dataset consists of 218,261 distinct songs for 48,169 distinct artists
  – Top tags were retrieved from the last.fm website for about 71,600 songs in our database
  – 48K songs with min. of 5 tags were used to build a 30-topic LDA model
  – The last $w = 7$ songs were selected as the user's active session, the last song was removed and the dominant topics associated with that song were used as target set
Topic Prediction Precision

(Playlist from *Art of the Mix*; Tags from last.fm)
Baseline Algorithms for Evaluation

• **User-based kNN**
• **Content-based recommender**
  – Attributes: artist, genre, era, and album
  – similarity of two songs calculated as the cosine similarity of their attribute vectors
  – Item-based kNN used to generate recommendations
• **BPRMF (Bayesian Personalized Ranking Matrix Factorization)**
  – Uses ranked pairs as training examples, so it optimizes for ranking rather than predicting a score
  – Avoids the problem of learning from only positive examples
Song Recommendation Performance

![Graph showing the performance of different song recommendation methods.](image-url)
Another Approach: Hierarchical Hidden Markov Models

- Use HHMM to learn common transitions between contextual states
- The model is used to predict the context for the next interaction with a user
- The predicted context is used to tailor the recommendations to match user’s current interests
- Pedigree: Interactional context; Dynamic or partially dynamic environment; Unobservable context information

Aghdam, Hariri, Mobasher, Burke, RecSys 2015
Recommendation Using a HHMM

Find most likely second-level state sequence

Find most likely first-level state sequence

Predict Next Context

Predict Next Song

Playlist
Recommendation Using a HHMM

1. Observation sequence, such as a playlist
   - Find most likely first-level state sequence
     - Find most likely second-level state sequence
       - Predict the next context
         - Generate Top-N recommendations

Viterbi algorithm

\[
P(T_0, ..., T_{M-1}) = \text{Max } P(\text{Playlist} \mid \text{First level State Sequence})
\]

\[
P(p_0, ..., p_{M-1}) = \text{Max } P(\text{1st level sequence} \mid \text{2nd level State Sequence})
\]

Given the most likely 2nd-level state sequence \((p_0, ..., p_{M-1})\), find the probability of each context (first-level state), \(T_i\), and select the most probable context:

\[
C(p_{M-1}, p_j) : \text{prob. of transition from state } p_{M-1} \text{ to } p_j.
\]

\[
D(p_j, T_i) : \text{prob. of observing } T_i \text{ at 2nd-level state } p_j.
\]

\[
\text{Max } P(T_i) = \sum_{j=0}^{k-1} C(p_{M-1}, p_j) \cdot D(p_j, T_i)
\]
Data Set

• Users’ listening activities for 5 months collected from Last.fm
  – Time-stamped sequence of artists
  – Training: first four months
  – Last month for evaluation
  – 837 users with at least one artist in the test and train partitions; Test data: 462 users
  – 51759 unique artists
## Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Top 5</th>
<th></th>
<th></th>
<th>Top 10</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
<td>F-Measure</td>
<td>Precision</td>
<td>Recall</td>
<td>F-Measure</td>
</tr>
<tr>
<td>HHMM</td>
<td>0.3272</td>
<td>0.00655</td>
<td>0.01284</td>
<td>0.27995</td>
<td>0.01125</td>
<td>0.02163</td>
</tr>
<tr>
<td>HMM</td>
<td>0.3644</td>
<td>0.0054</td>
<td>0.01064</td>
<td>0.3228</td>
<td>0.0102</td>
<td>0.01977</td>
</tr>
<tr>
<td>Sequential Pattern Mining</td>
<td>0.1036</td>
<td>3.56E-05</td>
<td>7.12E-05</td>
<td>0.0812</td>
<td>3.63E-05</td>
<td>7.26E-05</td>
</tr>
<tr>
<td>Item-based Markov Modeling</td>
<td>0.0342</td>
<td>8.69E-05</td>
<td>1.73E-04</td>
<td>0.0281</td>
<td>1.23E-04</td>
<td>2.45E-04</td>
</tr>
<tr>
<td>User-Based kNN</td>
<td>0.0350</td>
<td>0.0035</td>
<td>6.36E-03</td>
<td>0.0297</td>
<td>0.007</td>
<td>1.13E-02</td>
</tr>
<tr>
<td>BPRMF</td>
<td>0.0273</td>
<td>0.0047</td>
<td>8.02E-03</td>
<td>0.0247</td>
<td>0.006</td>
<td>9.65E-03</td>
</tr>
<tr>
<td>Most Popular</td>
<td>0.02188</td>
<td>0.006</td>
<td>9.42E-03</td>
<td>0.0179</td>
<td>0.008</td>
<td>1.11E-02</td>
</tr>
<tr>
<td>Random</td>
<td>0.0004</td>
<td>5.74E-06</td>
<td>1.13E-05</td>
<td>0.0004</td>
<td>2.76E-05</td>
<td>5.16E-05</td>
</tr>
</tbody>
</table>

HHMM has the highest recall and achieves the best overall F-score.
Popularity Bias

The items were sorted based on their overall frequencies in users profiles and grouped into $I = 10$ bins.
Context Adaptation in Interactive Recommendation

Pedigree:
• Interactional context
• Fully dynamic environment
• Unobservable context
Problem: Change of Context
New Context = New Utility Function

Like

Dislike
Detecting Context Changes

Context 1

How do we infer hidden context or change of context?

Context 2
Interactive Recommendation Scenario

- Estimate the utility of items based on a user’s profile
- Determine Effect of Context
- Update Utility Estimate
- Detect Change in Context

Recommendations

![Image of thumbs up and thumbs down icons]
Maximizing the Utility

- **Maximizing the utility for each step**
  - Recommendation = highest estimated expected utility
  - Reward = rating from user (or, item selection, click-through, etc.)

- **Maximizing reward over the interaction session**
  - **Exploitation**: choose the most profitable item
  - **Exploration**: choose other items to acquire more information (preferences, context)
  - Must trade off these behaviors
Multi-armed Bandit Algorithms

**Idea**
- Different choices (items) obtain different rewards
- Sample different items to find best reward
- Consider total reward over limited interactions

**For recommendation**
- Set of arms-> representation of candidate items
- Rewards-> Users’ feedbacks (e.g., ratings or click-through on recommended items)

**Solution approaches**
- E-greedy approach
- Upper Confidence Bound (UCB) algorithms
- Thompson Sampling
Adapting Thompson Sampling

- **Items ≠ Arms**
  - A item can be recommended only once per user
- ** Describe items by features**
  - Features constant during the interaction
  - Combination of collaborative and content-based data
- **Create reduced dimensionality representation**
  - Use PCA to represent each item in a k-dimensional space
  - Each “arm” is a point in this space
    - recommend items near that point
Thompson Sampling

- Item selected based on its probability of optimality
- Parameter $\theta$ characterizes the utility (reward) distribution
- $E_r(r|a, \theta)$: expected reward for item $a$ for the given $\theta$

```
D=\emptyset
for t = 1 to T do
    Draw $\theta^t \sim P(\theta|D)$
    Select $a_t = \text{argmax} E_r(r|a, \theta^t)$
    Observe reward $r_t$
    $D = D \cup (a_t, r_t)$
end for
```
Assumptions

- **Theta Distribution**
  - $\theta$ is drawn from an multivariate normal distribution
    - user’s preference function
    - location $\mu$, covariance $\Sigma$
    - $\mathcal{N}(\mu_t, \Sigma_t)$

- **Linearity**
  - Expected reward is a linear function of the item features
  - $r_x = f_x \cdot \theta$

- **Reward distribution** $P(r|\theta, a)$
  - linear transform of the $\theta$ distribution
Updating the User Model

- Prior and likelihood distributions:
  \[ p(\theta) = \mathcal{N}(\theta; \mu_\theta, \Sigma_\theta) \]
  \[ p(r|\theta) = \mathcal{N}(r; F\theta, \Sigma_r) \]

- Given a linear Gaussian system, the posterior is computed as follows:
  \[
  p(\theta|r) = \mathcal{N}(\mu_{\theta|r}, \Sigma_{\theta|r})
  \]
  \[
  \Sigma_{\theta|r}^{-1} = \Sigma_\theta^{-1} + F^T \Sigma_r^{-1} F
  \]
  \[
  \mu_{\theta|r} = \Sigma_{\theta|r}[F^T \Sigma_r^{-1}(r) + \Sigma_\theta^{-1} \mu_\theta]
  \]

Expected reward:
\[ r_x = f_x \cdot \theta \]
<table>
<thead>
<tr>
<th>Artist</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linkin park</td>
<td></td>
</tr>
<tr>
<td>Staind</td>
<td>100</td>
</tr>
<tr>
<td>Metallica</td>
<td>90</td>
</tr>
<tr>
<td>Green Day</td>
<td>90</td>
</tr>
<tr>
<td>Simple Plan</td>
<td>Skipped</td>
</tr>
<tr>
<td>Papa Roach</td>
<td>90</td>
</tr>
<tr>
<td>Nirvana</td>
<td>90</td>
</tr>
<tr>
<td>Foo fighters</td>
<td>0</td>
</tr>
<tr>
<td>Creed</td>
<td>Skipped</td>
</tr>
<tr>
<td>Sum 41</td>
<td>10</td>
</tr>
<tr>
<td>Incubus</td>
<td>0</td>
</tr>
<tr>
<td>Godsmack</td>
<td>Skipped</td>
</tr>
</tbody>
</table>
Change Detection

\[ W_{I(t-N)} = p(\theta|I_{(t-N)}) = \mathcal{N}(\mu_{t-N}, \Sigma_{t-N}) \]

\[ W_{I_t} = p(\theta|I_t) = \mathcal{N}(\mu_t, \Sigma_t) \]

Measure distance between two distribution: KL-divergence; Mahalanobis distance; Etc.
Detect Distribution Changes

\[ W_{I(t-N)} = \mathcal{N}(\mu_{t-N}, \Sigma_{t-N}) \quad \text{and} \quad W_{I_t} = \mathcal{N}(\mu_t, \Sigma_t) \]

\[ \Sigma = \frac{\Sigma_t + \Sigma_{t-N}}{2} \]

\[ \text{distance} = (\mu_t - \mu_{t-N})^T \Sigma^{-1} (\mu_t - \mu_{t-N}) \]

- How to measure distance between two distributions
  - KL-divergence
  - Mahalanobis distance
  - others
Sliding Window

Iteration

Computing Distance

Change Point Analysis Module

Change detected at time $t$

Updating the user’s profile
Evaluation

• **Simulating the change in a user’s behavior:**
  – Generating a hybrid user profile by switching between two random users in the test data.
Experiments

• Yahoo! Music ratings of musical artists version 1.0.
  – ~10M ratings of musical artists over the course of one month
  – ~2M users, ~100k artists.
  – Ratings: 0 to 100
• 5-fold cross validation
• Evaluation metric: Average obtained utility
  – user’s rating for each recommended item = utility
Results

- 60 iterations
- Simulated change at iteration 30
Results 2

- **CTI Data**
  - Server log data from the CS department at DePaul university
  - After pre-processing: 5319 users, and 2453 distinct pageviews
  - Number of Iterations: 20
  - Simulated change occurs at iteration 10
Findings

• **Thompson sampling**
  – effective implementation of interactive recommendations

• **Context-sensitivity**
  – Change detection enables recommender to recover more quickly when there is a new context
Future Work

• Integrating short- and long-term modeling
  – current study: context change is cold-start

• Characterize items/users using information about domain or users

• Realistic data set
  – current study: context change is artificial
    • grafting two different users’ playlists
  – user study
Questions

Thanks

?