DePaul University

CENTER FOR WEB INTELLIGENCE

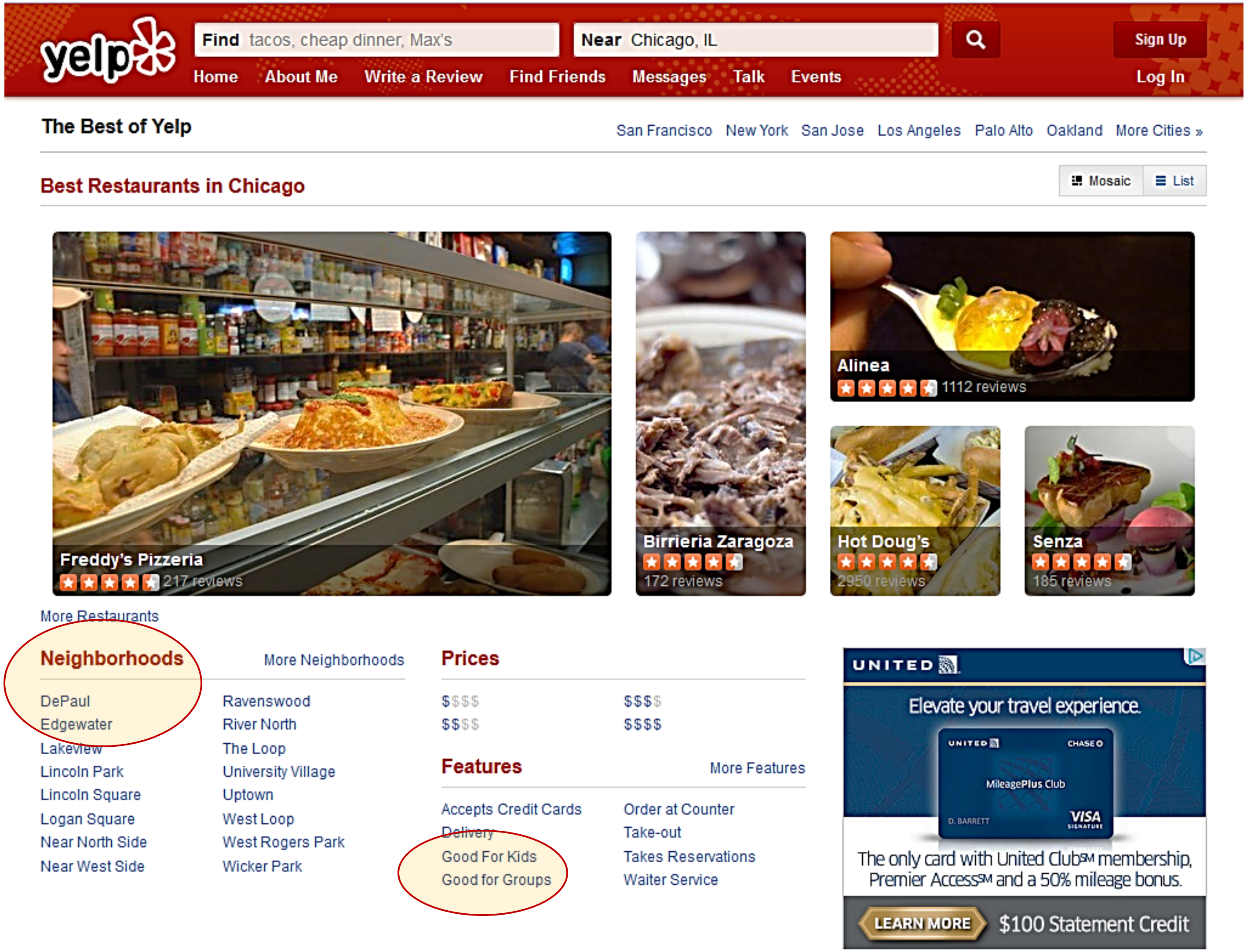
Context Awareness and Adaptation in Recommendation

Bamshad Mobasher

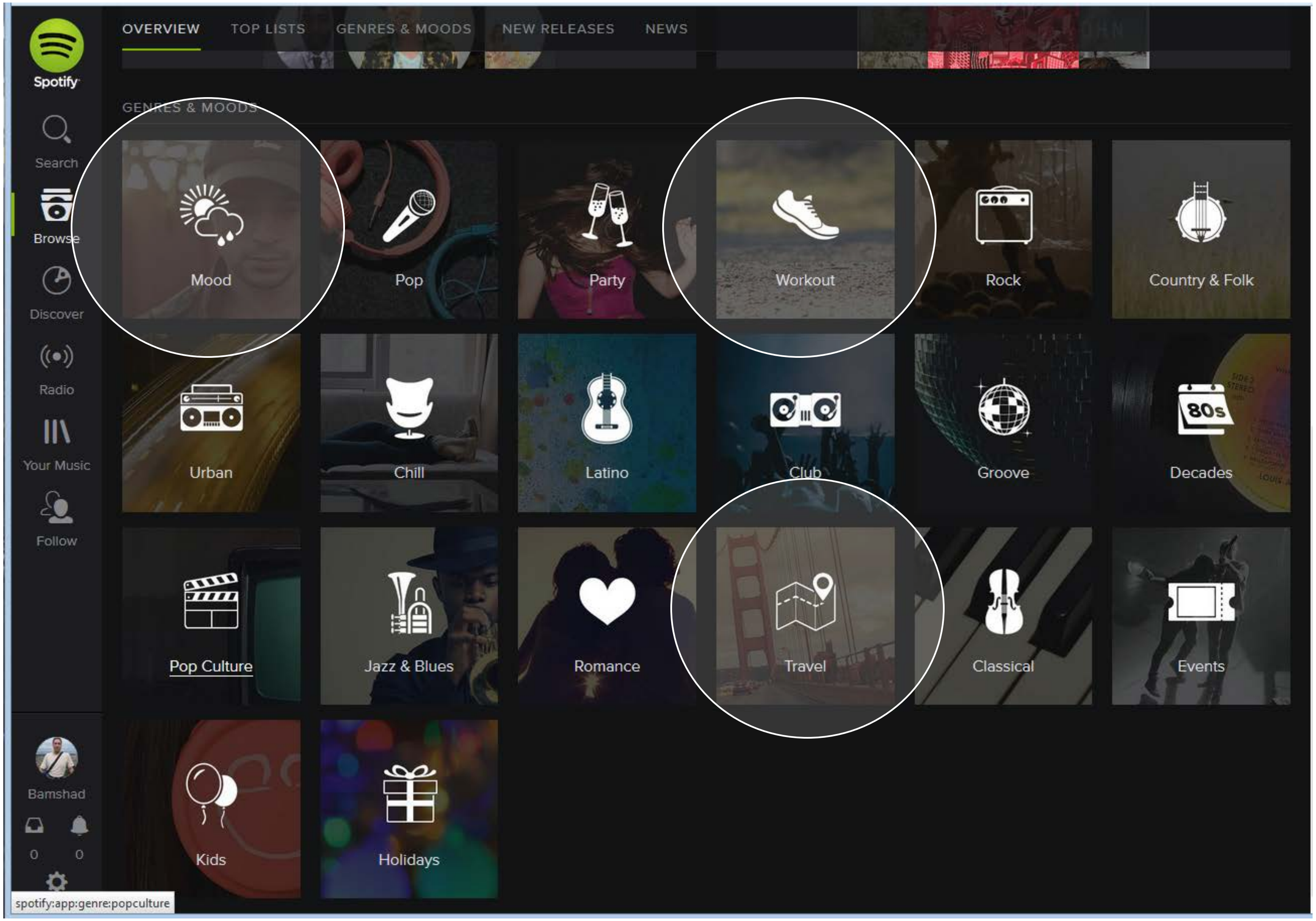
DePaul University, Chicago, IL, USA

DMRS Workshop – Oct. 22-23, 2015 – Bolzano, Italy

Context in Recommendation



Context in Recommendation



Context-Aware Recommendation

Watching alone

Watching with a kid

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**

Different Views of Context

- **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.

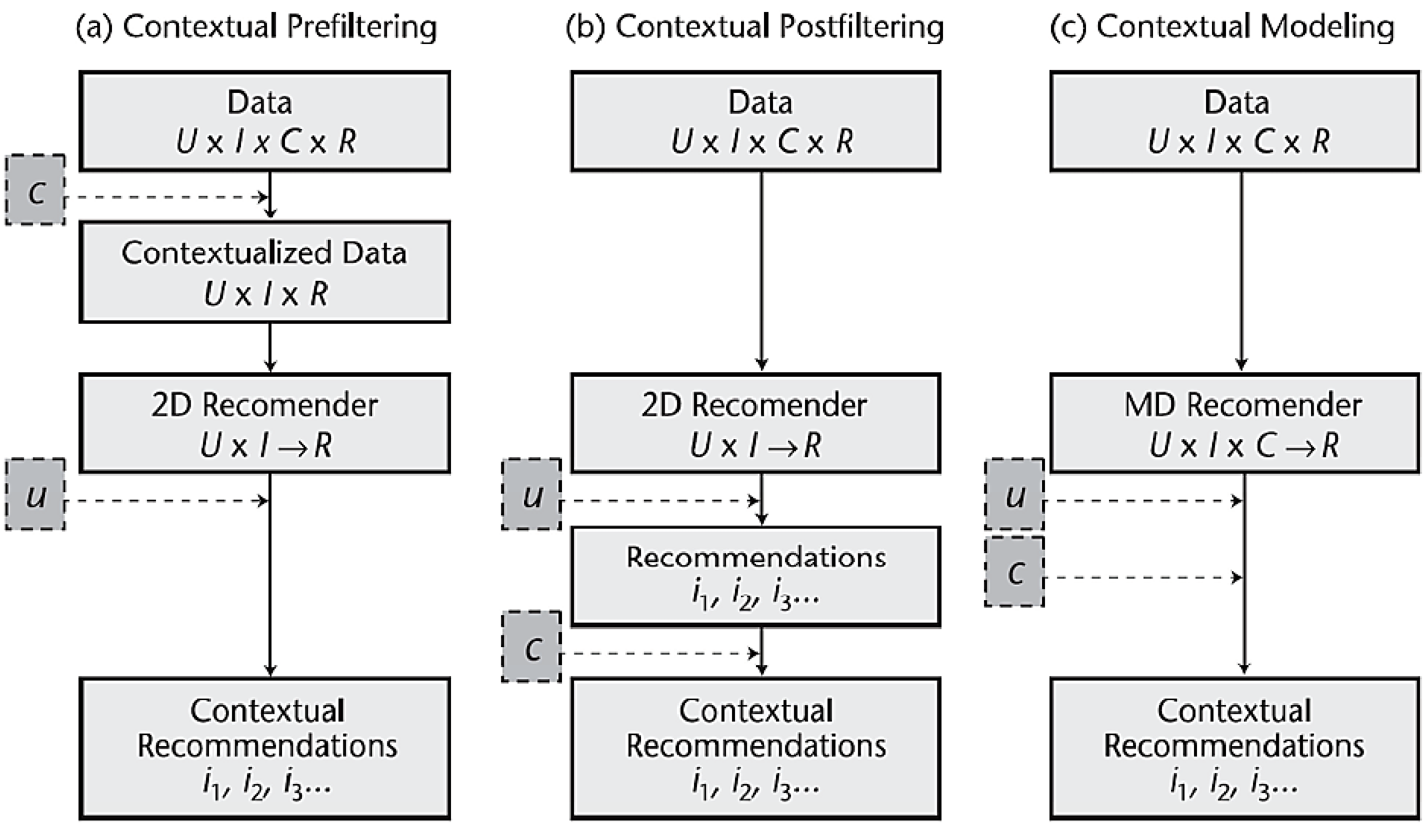
- **Implications:**
  - Relevant contextual variables (and their structures) must be identified at the design stage
  - Must identify & acquire explicit contextual information before recommendations are made

Contextual Recommendation in a Representational Framework

User	Movie	Time	Location	Companion	Rating
U1	*Titanic*	Weekend	Theatre	Family	4
U2	*Titanic*	Weekday	Home	Family	5
U3	*Titanic*	Weekday	Theatre	Alone	4
U1	*Titanic*	<u>Weekday</u>	<u>Home</u>	<u>Alone</u>	?

- **Traditional RS:** Users $\times$ Items → Ratings
- **Contextual RS**: Users $\times$ Items $\times$ Context → Ratings

Representational Recommendation Frameworks



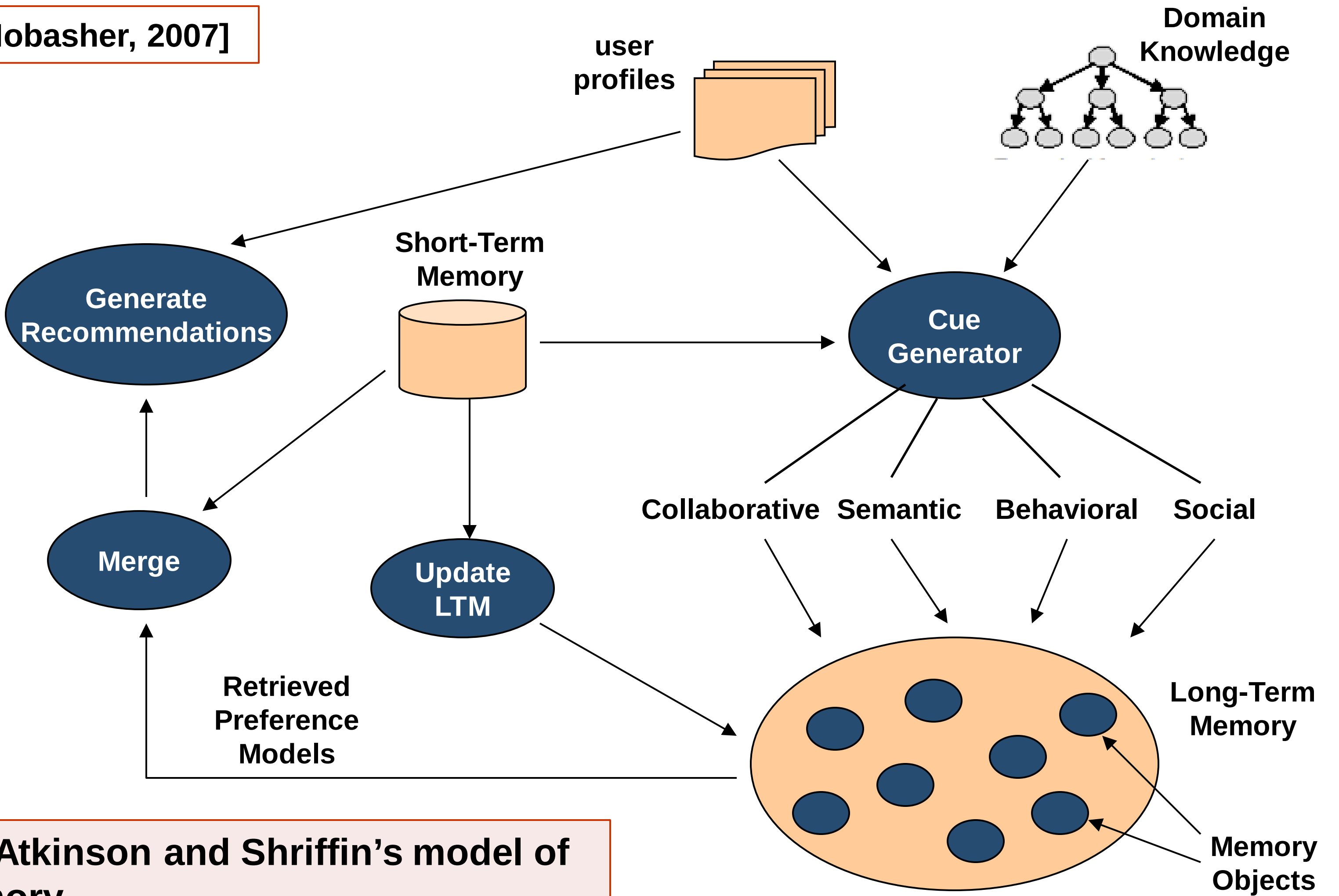
From Adomavicius & Tuzhilin, 2008

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

[Anand and Mobasher, 2007]



Inspired by Atkinson and Shriffin's model of human memory

Characterizing the Environment for CARS

How Contextual Factors Change	Knowledge of the RS about the Contextual Factors		
	Fully Observable	Partially Observable	Unobservable
Static	Everything Known about Context	Partial and Static Context Knowledge	Latent Knowledge of Context
Dynamic	Context Relevance Is Dynamic	Partial and Dynamic Context Knowledge	Nothing Is Known about Context

Adomavicius, Mobasher, Ricci, and Tuzhilin. *AI Magazine*, 2011

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

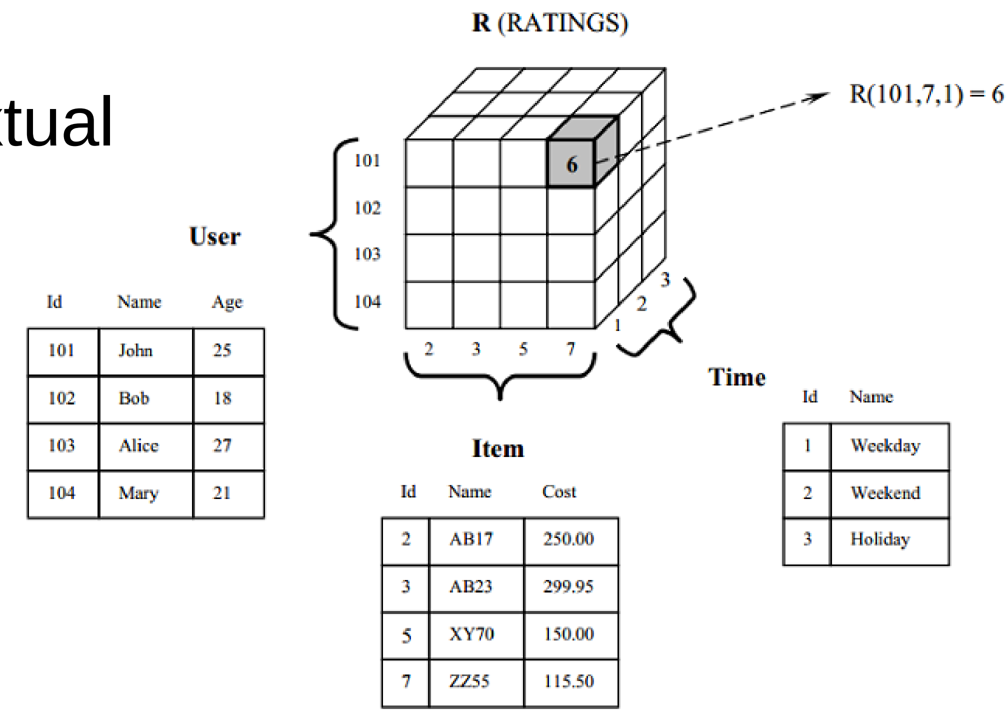
Similarity-Based Context-Aware Matrix Factorization

Pedigree:
- Representational context
- Static environment
- Observable context information

Contextual Modeling

Two general approaches for representational contextual modeling

1. Independent Contextual Modeling



Tensor Factorization, ACM RecSys 2010

Contextual Modeling

Two general approaches for representational contextual modeling

2. Conditional (Dependent) Contextual Modeling

Deviation-Based Contextual Modeling

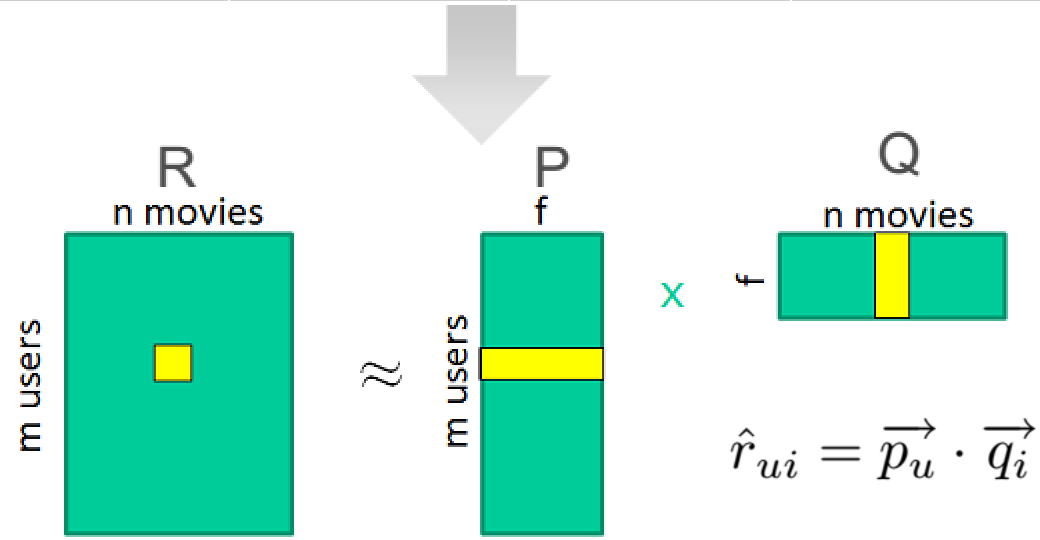
Baltrunas, et al., Context-aware Matrix Factorization, ACM RecSys 2011
Zheng, et al., Contextual Sparse Linear Method, ACM RecSys 2014

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

User	HarryPotter	Batman	Spiderman
U1	5	3	4
U2	?	2	4
U3	4	2	?



$$\hat{r}_{ui} = \overrightarrow{p_u} \cdot \overrightarrow{q_i}$$

- 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

Matrix Factorization

Rating Prediction Function: $q^T_i \, p_u$

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

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

Global average rating

User bias

Item bias

Rating Prediction Function: $\hat{r}_{ui} = \mu + b_u + b_i + \overrightarrow{p_u} \cdot \overrightarrow{q_i}$

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:

$$\min_{p*,q*,b*} \sum_{(u,i)\in\kappa} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2)$$

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: $\hat{r}_{uic_{k,1}c_{k,2}\ldots c_{k,L}} = \mu + b_u + \sum_{j=1}^{L} B_{ijc_{k,j}} + \overrightarrow{p_u} \cdot \overrightarrow{q_i}$

CAMF replaced term *b_i* by $\sum_{j=1}^{L} B_{ijc_{k,j}}$ 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: $\hat{r}_{uic_{k,1}c_{k,2}\ldots c_{k,L}} = \mu + b_u + \sum_{j=1}^{L} B_{ijc_{k,j}} + \overrightarrow{p_u} \cdot \overrightarrow{q_i}$

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

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 Ck and CE

Similarity-Based CAMF

Similarity measures

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

Independent Context Similarity

$$Sim(c_k, c_m) = \prod_{l=1}^{L} similarity(c_{k,l}, c_{m,l})$$

For example:

C_k = {Time = Weekend, Location = Home}

C_m = {Time = Weekday, Location = Office}

The similarity between C_k and C_m is:

sim(Weekend, Weekday) × sim(Home, Office)

Contextual variables are assumed to be independent

Latent Context Similarity

$$similarity(c_{k,l}, c_{m,l}) = V_{c_{k,l}} \bullet V_{c_{m,l}}$$

Each V is a vector over latent factors

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

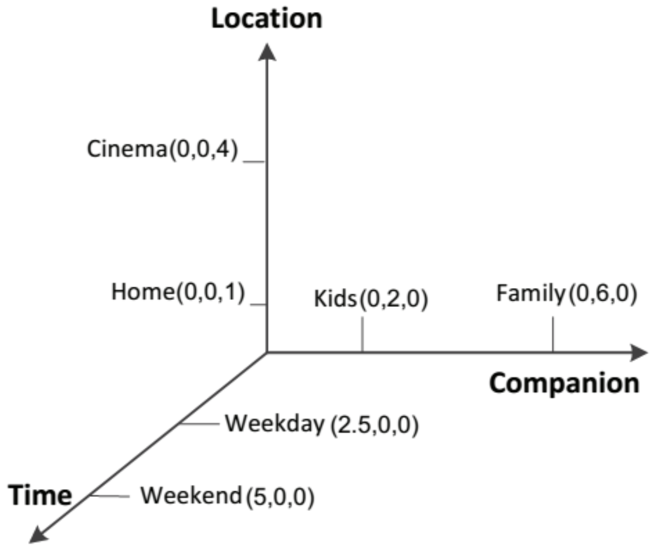
Alleviates Context Sparsity Problem

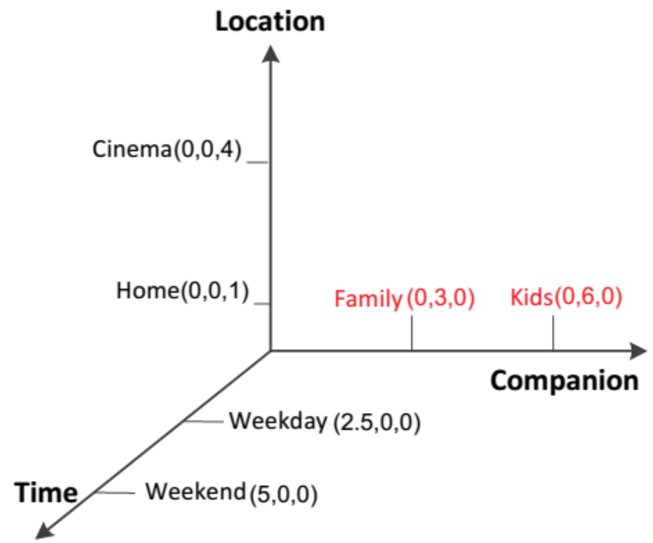
Example:

Training: <weekend, theater> <weekday, home>

Testing: <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

Learning Process

The general loss function in CAMF-Sim can be described as:

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

$$\underset{p,q,Sim}{Minimize} \frac{1}{2}(r_{uic_k} - \hat{r}_{uic_k})^2 + \frac{\lambda}{2}(\|\overrightarrow{p_u}\|^2 + \|\overrightarrow{q_i}\|^2 + Sim^2)$$

Methods for each similarity measure

ICS	LCS	MCS
The similarity (real valued) for each individual pair of context conditions	The vector representation (weights in factors) for each contextual condition	The positions (real values) for each contextual condition

Learning Process

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

Example (ICS):

$$sim_1 = similarity(\text{“}Time = Weekday\text{”}, \text{“}Time = N/A\text{”})$$
$$sim_2 = similarity(\text{“}Location = Home\text{”}, \text{“}Location = N/A\text{”})$$
$$Sim(c_k, c_E) = sim_1 \times sim_2$$

$$\underset{p,q,Sim}{Minimize} \frac{1}{2}(r_{uic_k} - \hat{r}_{uic_k})^2 + \frac{\lambda}{2}(\|\overrightarrow{p_u}\|^2 + \|\overrightarrow{q_i}\|^2 + Sim^2)$$

SGD Updates:

$$err = r_{uic_k} - \hat{r}_{uic_k}$$
$$\overrightarrow{p_u} = \overrightarrow{p_u} + \beta \cdot err \cdot \overrightarrow{q_i} \cdot Sim(c_k, c_E)$$
$$\overrightarrow{q_i} = \overrightarrow{q_i} + \beta \cdot err \cdot \overrightarrow{p_u} \cdot Sim(c_k, c_E)$$
$$sim_1 = sim_1 + \beta(err \cdot (\overrightarrow{p_u} \cdot \overrightarrow{q_i}) \cdot sim_2 - \alpha \cdot sim_1)$$
$$sim_2 = sim_2 + \beta(err \cdot (\overrightarrow{p_u} \cdot \overrightarrow{q_i}) \cdot sim_1 - \alpha \cdot sim_2)$$

Data Sets and Metrics

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

	Restaurant	**Music**	**Tourism**
Rating Profiles	50 users, 40 items 2314 ratings	40 users, 139 items 3940 ratings	25 users, 20 items 1678 ratings
# of contextual dimensions, F	2	8	14
# of contextual conditions, L	7	34	67
Rating Scale	1-5	1-5	1-5

5-fold cross validation

Evaluation metrics:

- Precision: measuring the hit ratio of relevant items;

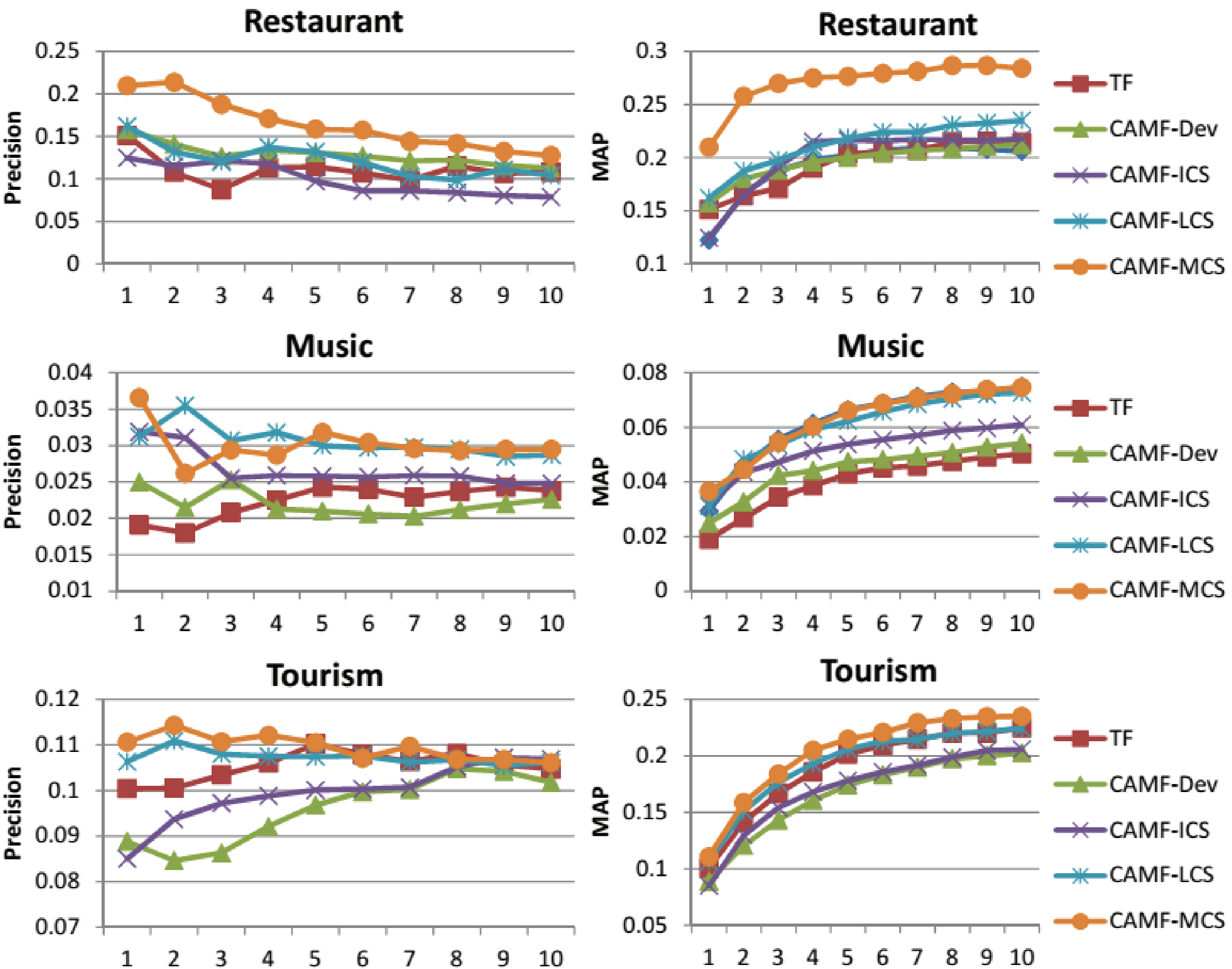
- MAP: taking the rankings of items into account.

$$MAP@N = \sum_{i=1}^{M} ap@N/M, \text{ where } ap@N = \frac{\sum_{k=1}^{N} P(k)}{\min(m, N)}$$

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

Highlighted Approach

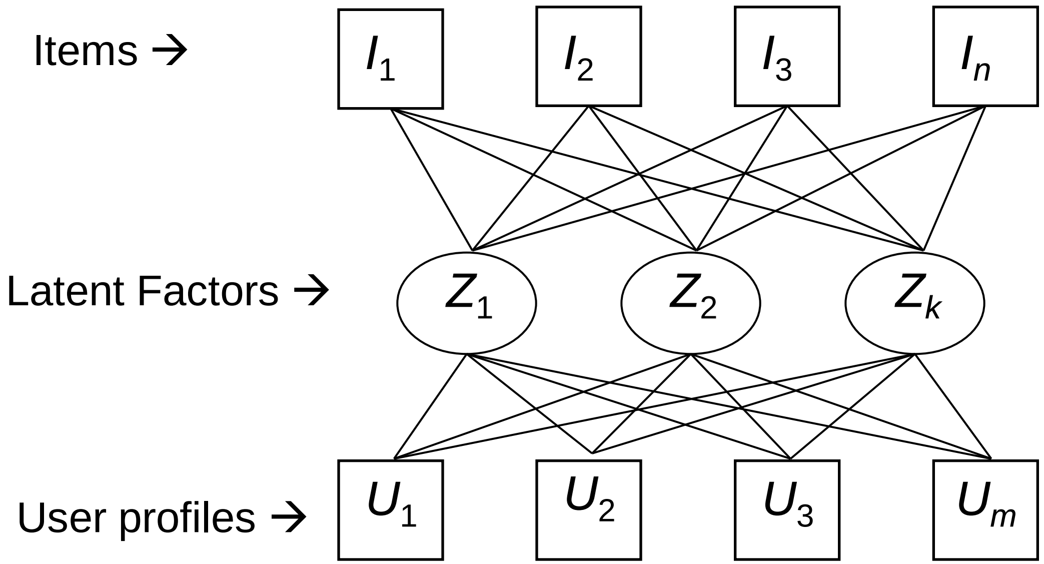
Dynamic Latent Variable Context Models

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



$$\Pr(U_i, I_j) = \sum_{z_k} \Pr(U_i) \bullet \Pr(Z_k \mid U_i) \bullet \Pr(I_j \mid Z_k)$$

I.e., The preference of a user for different items is encoded in the user's membership in the latent classes

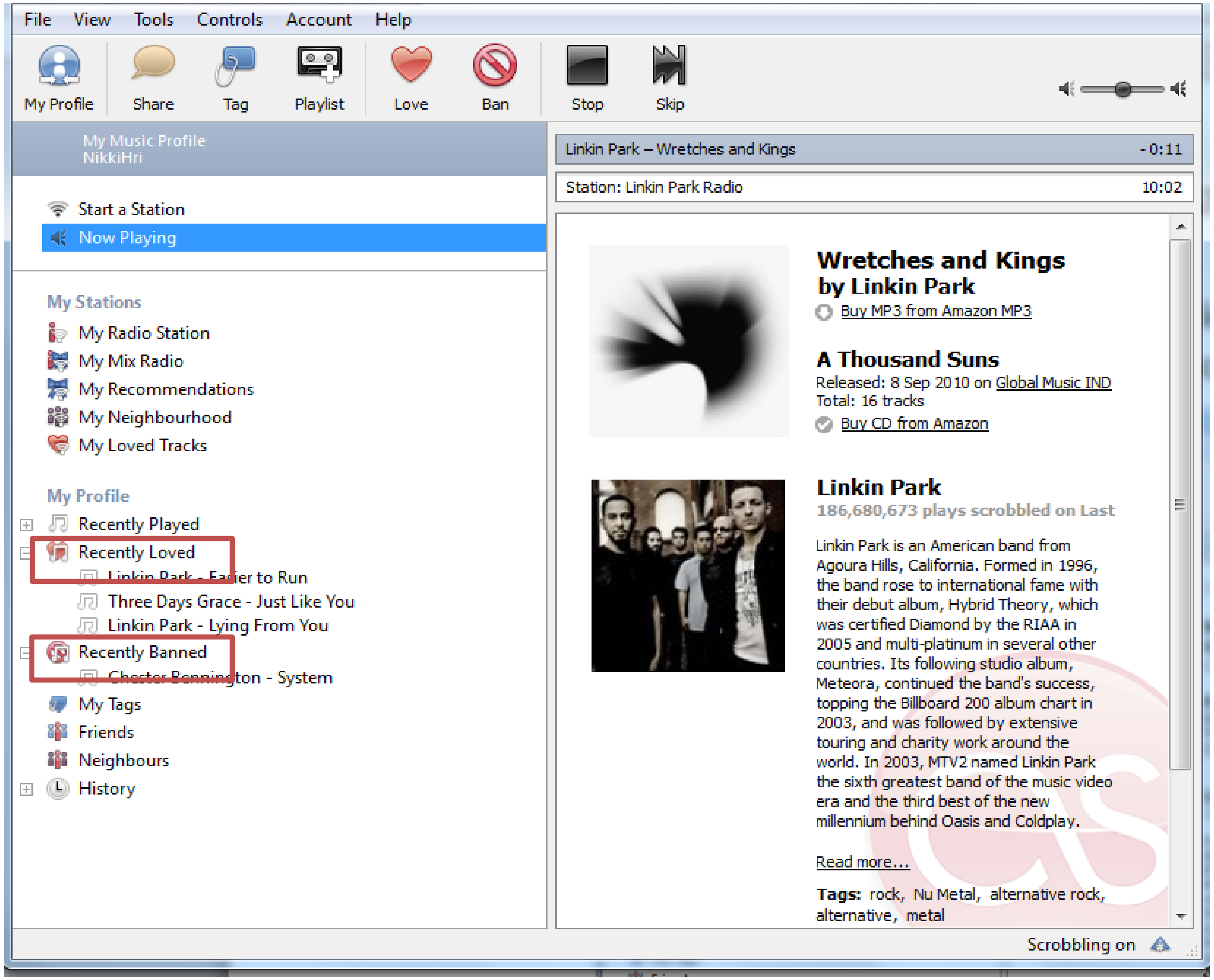
Inferring Latent Contexts From Sequences of User Interactions

- **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

Hariri, Mobasher, Burke, RecSys 2012

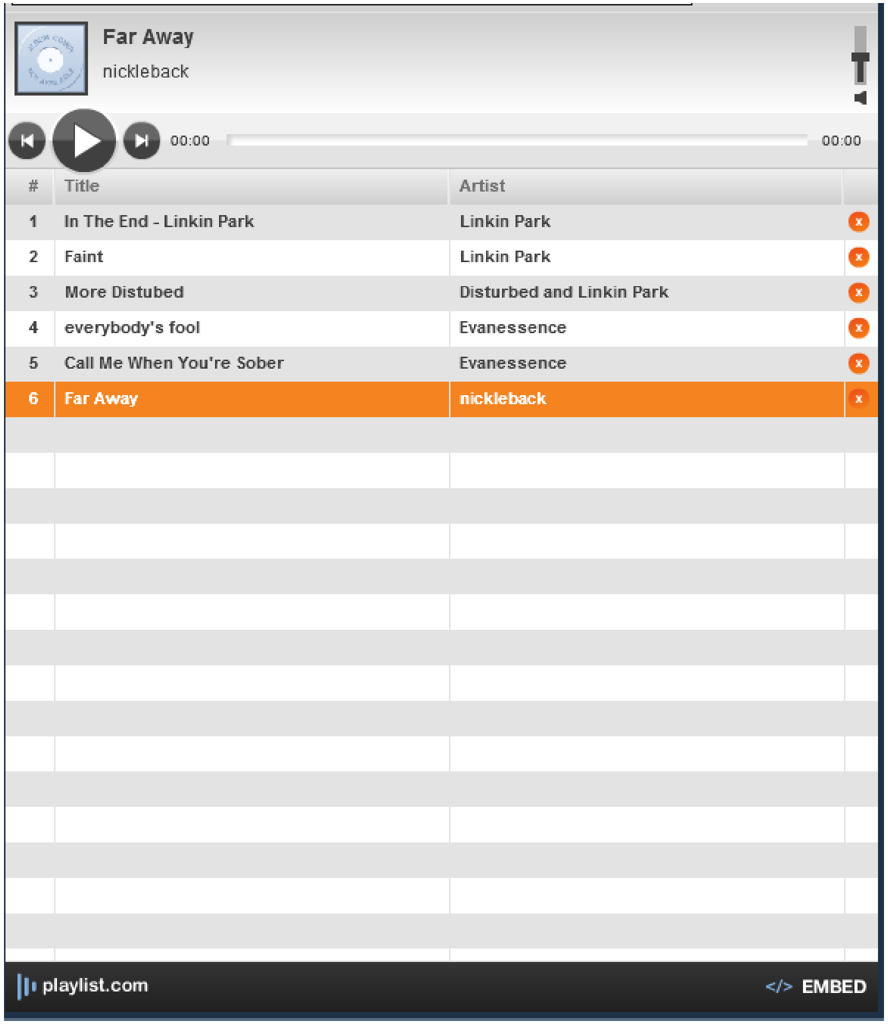
User Interactions

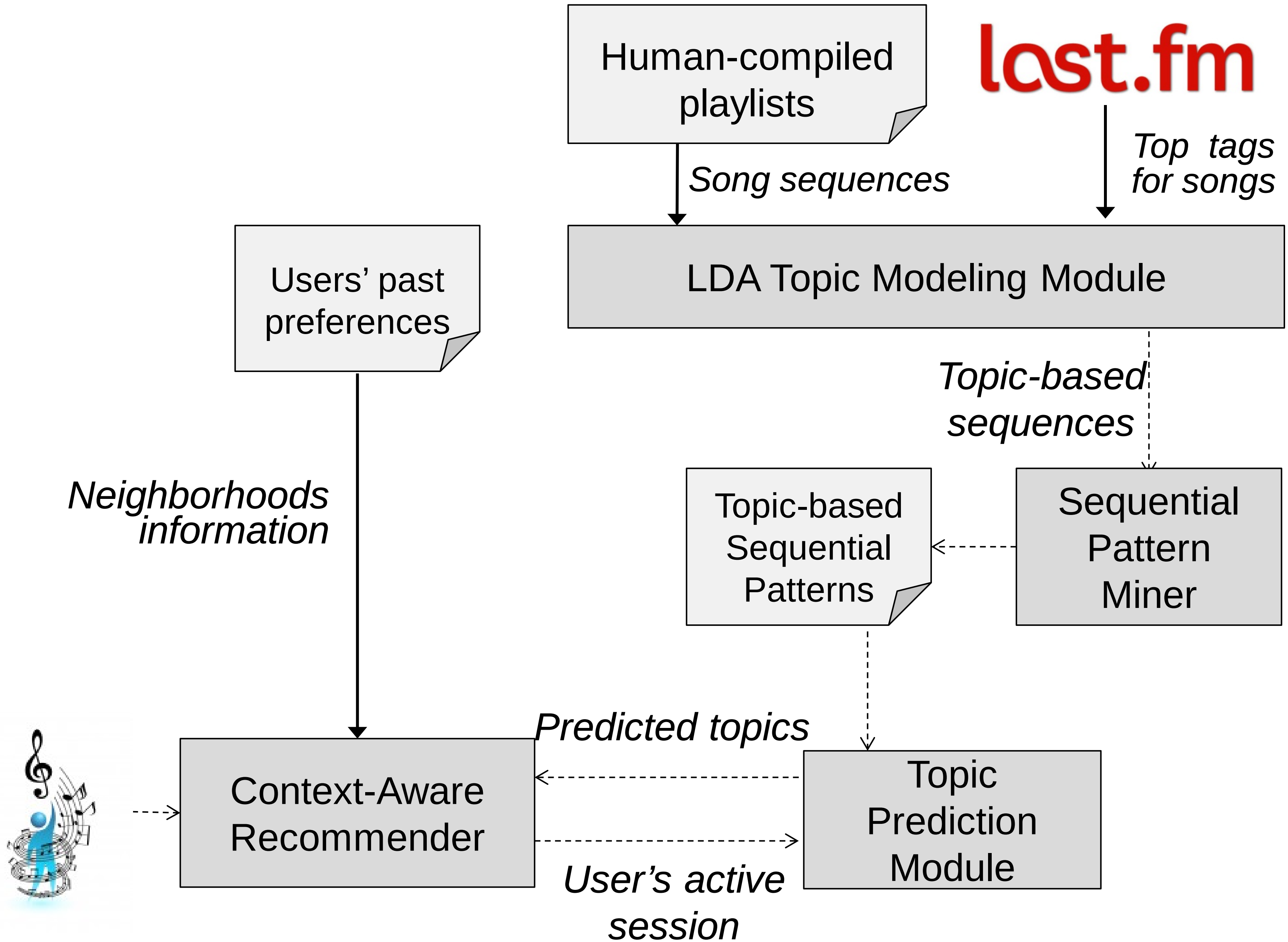
Context is reflected in the sequence of songs liked/disliked or played by the user in her current interaction with the system



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**




Human-compiled playlists
last.fm
Song sequences
Top tags for songs
Users' past preferences
LDA Topic Modeling Module
Topic-based sequences
Neighborhoods information
Topic-based Sequential Patterns
Sequential Pattern Miner
Predicted topics
Context-Aware Recommender
Topic Prediction Module
User's active session

Topic Modeling for Song Context Representation

- **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

Top Most Frequent Tags for a Sample of Topics

Topic#1	Topic#2	Topic#3	Topic#4	Topic#5	Topic#6	Topic#7
ambient	latin	death	60s	chill	beautiful	electronic
instrumental	world	thrash	oldies	downtempo	sad	electronica
soundtrack	streamable	black	roll	chillout	mellow	house
classical	spanish	heavy	50s	christmas	melancholy	techno
beautiful	para	doom	rockabilly	lounge	acoustic	tranc
age	bossa	brutal	top	electronic	chill	electro
chillout	fusion	melodic	500	trip-hop	soft	bass
experimental	musica	california	radio	electronica	slow	drum
movie	que	power	rolling	trip	melancholic	ambient
atmospheric	nova	progressive	1960s	ambient	favourite	beat
world	brazilian	gods	rhythm	hop	chillout	idm
ethereal	african	seixas	time	easy	ballad	experimental
chill	party	speed	elvis	cool	singer-songwriter	club
calm	brasil	swedish	soundtrack	sexy	life	minimal
electronic	espanol	old	american	radio	easy	party

An Example Playlist (Mapped to Tags and Topics)

Time	Popular Tags	Dominant Topics
1	Singer-songwriter, mellow, relaxing, chill, male vocalist, easy listening, acoustic, 00's, guitar, rock, happy	6
2	Singer-songwriter, chill, acoustic, mellow, rock, summer, surf, male vocalist, pop, relaxing, guitar, happy	6
3	singer,-songwriter, indie rock, folk, acoustic, mellow, chill out, relaxing, bittersweet, lo-fi	6, 20, 23
4	Alternative rock, ballads, calm, beautiful, nice, soundtrack, favorites	6, 28
5	Electronic, electronica, French, chill out, trip-hop, ambient, down-tempo, sexy, 90s, alternative, easy listening, guitar, mellow, relax, female vocal	7, 5
6	Soundtrack, 90s, alternative, atmospheric, female vocalist, indie, dreamy	23
7	Singer-songwriter, acoustic, chill, alternative, rock, male vocalist, easy listening, driving	6, 25
8	Cover, Beatles cover, rock, 90s, soundtrack, brass , pop rock, alternative, rock, folk, brass	30, 18
9	Indie, rock, acoustic, 90s, cover, mellow, pop, folk, dreamy, singer-songwriter, sad-core, summery, sweet, alternative rock, female vocalist	6, 20

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>>

  ...

Sequential Pattern Mining and Topic Prediction

- **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**

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

  – *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)

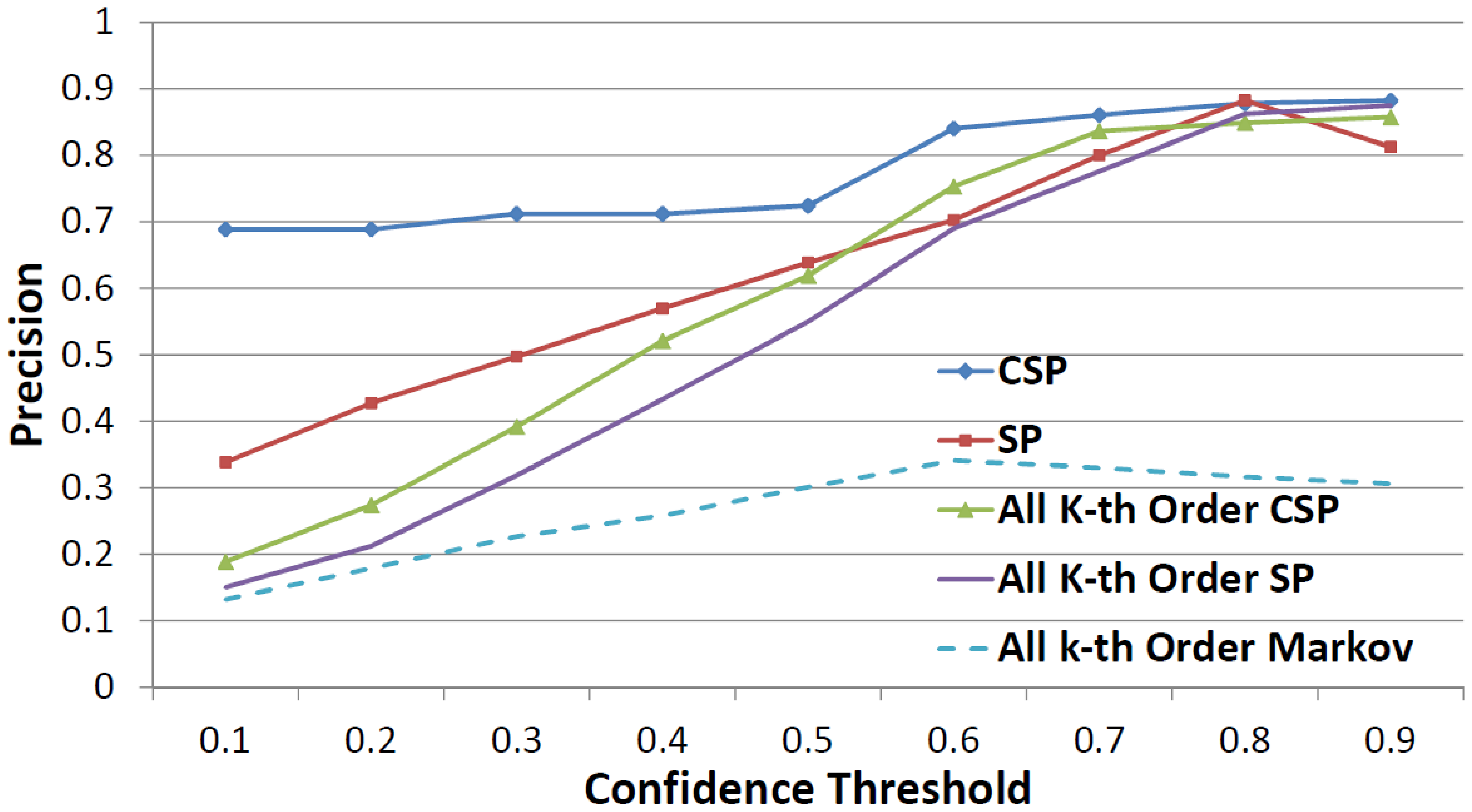
- **Next, recommendations are re-ranked using the contextual information**
  – Prediction score for a song *s*: a linear combination of CF predicted rating and the context score for *s*.

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

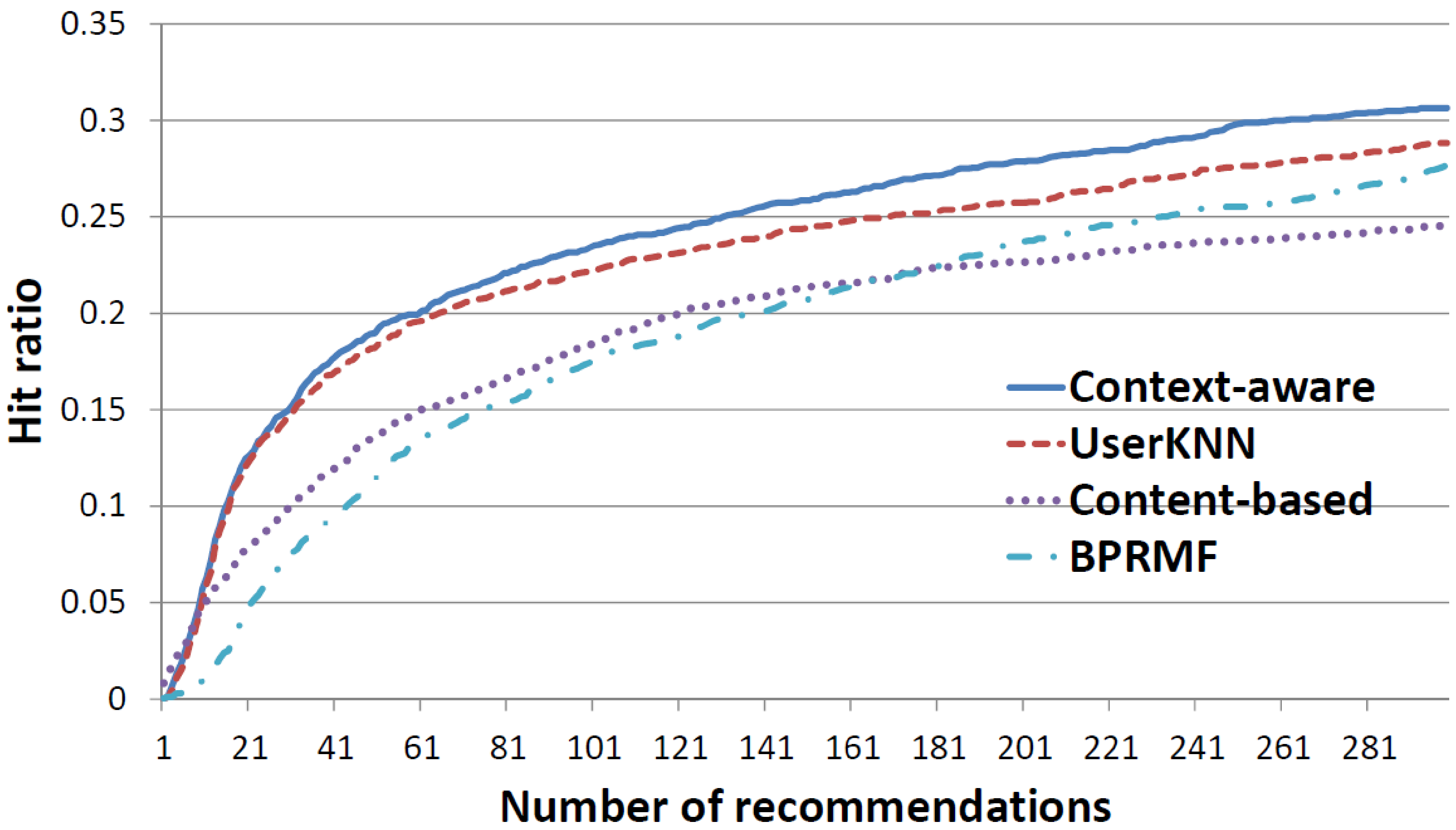
(Playlist from *Art of the Mix*; Tags from last.fm)



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

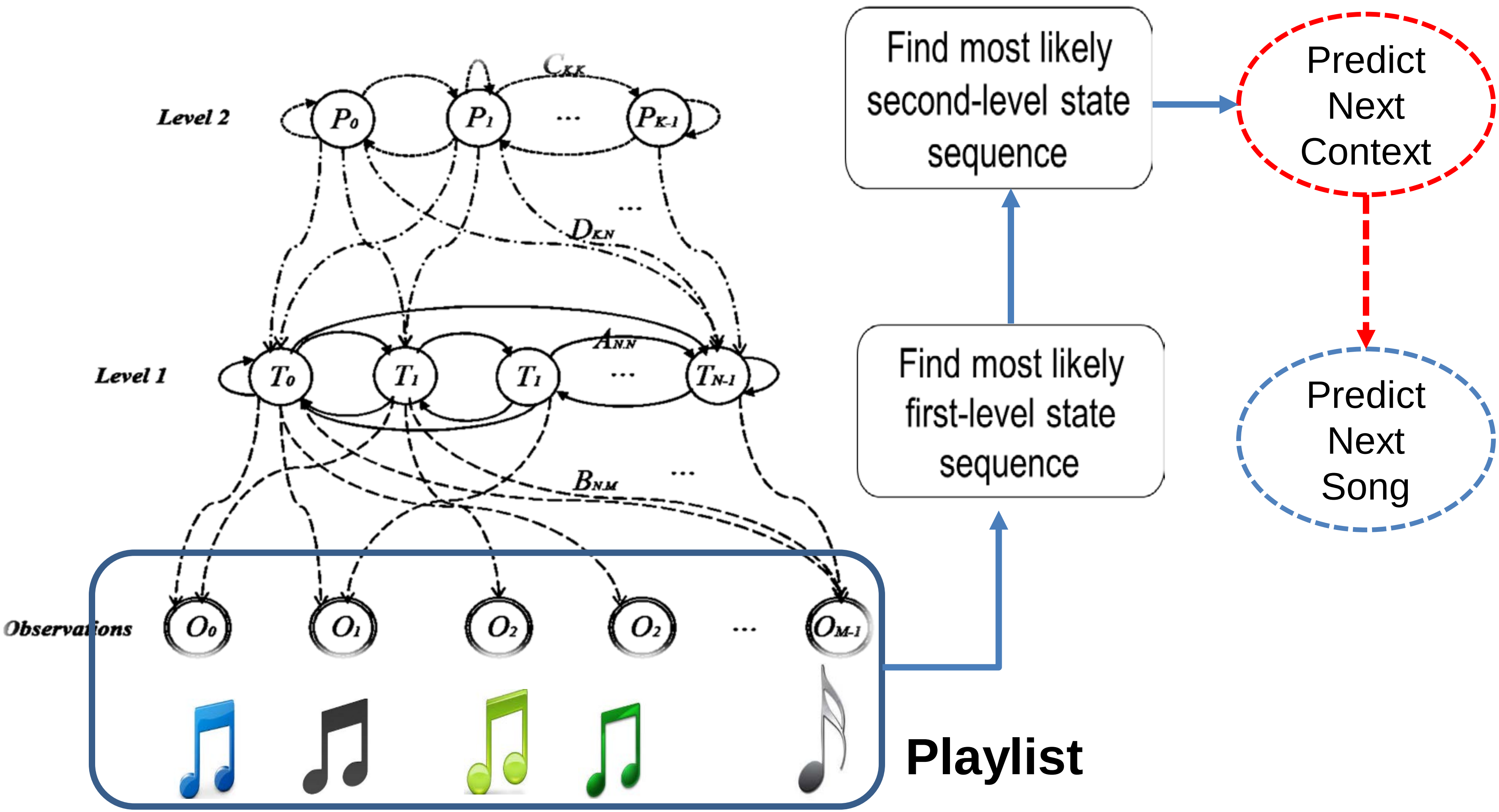


Another Approach: Hierarchical Hidden Markov Models

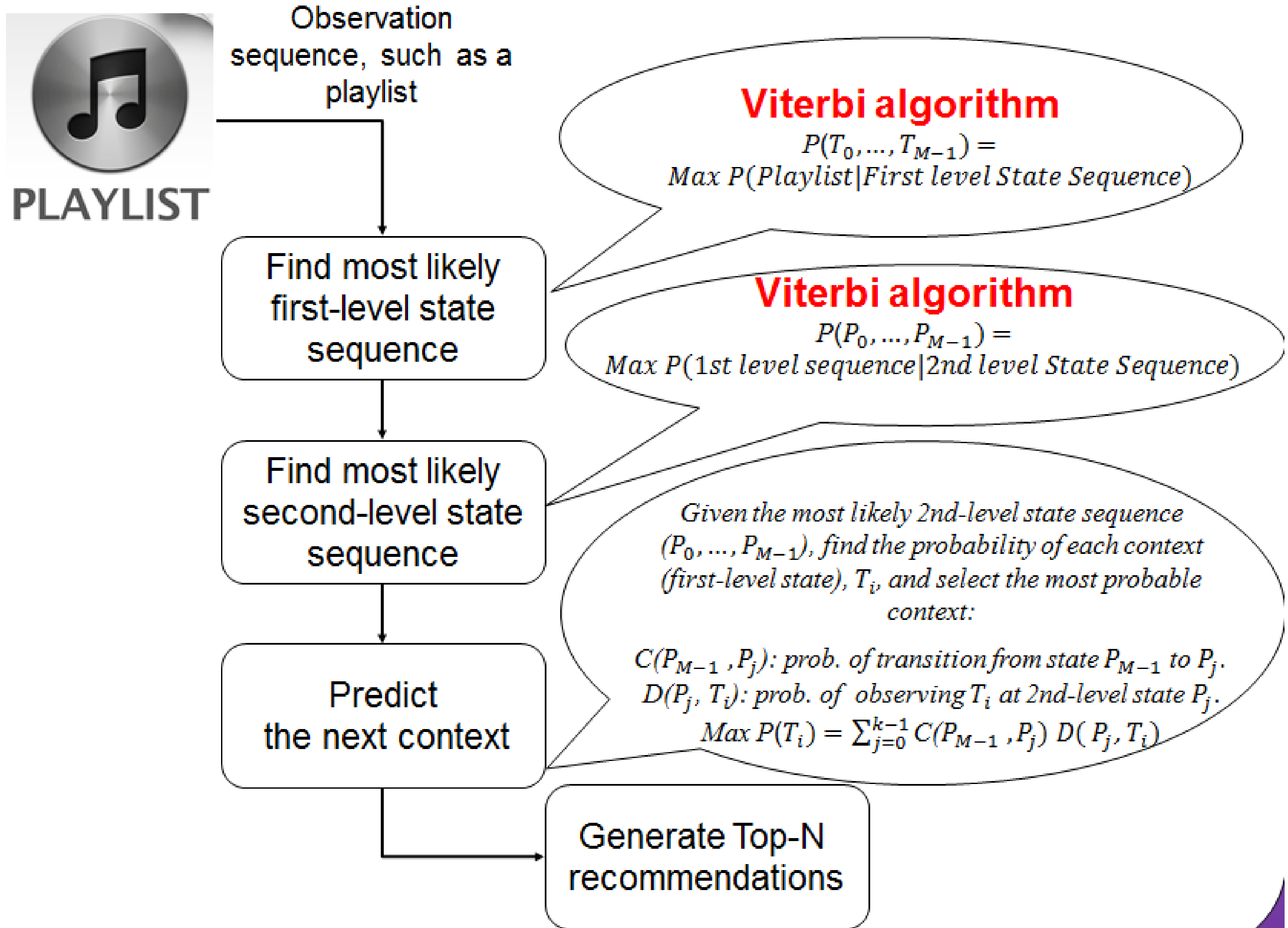
- **Use HHMM to learn common transitions between contextual states**
- **The model is used to predict the context for the next interaction with a user**
- **The predicted context is used to tailor the recommendations to match user’s current interests**
- **Pedigree: Interactional context; Dynamic or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

Aghdam, Hariri, Mobasher, Burke, RecSys 2015

Recommendation Using a HHMM



Recommendation Using a HHMM



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

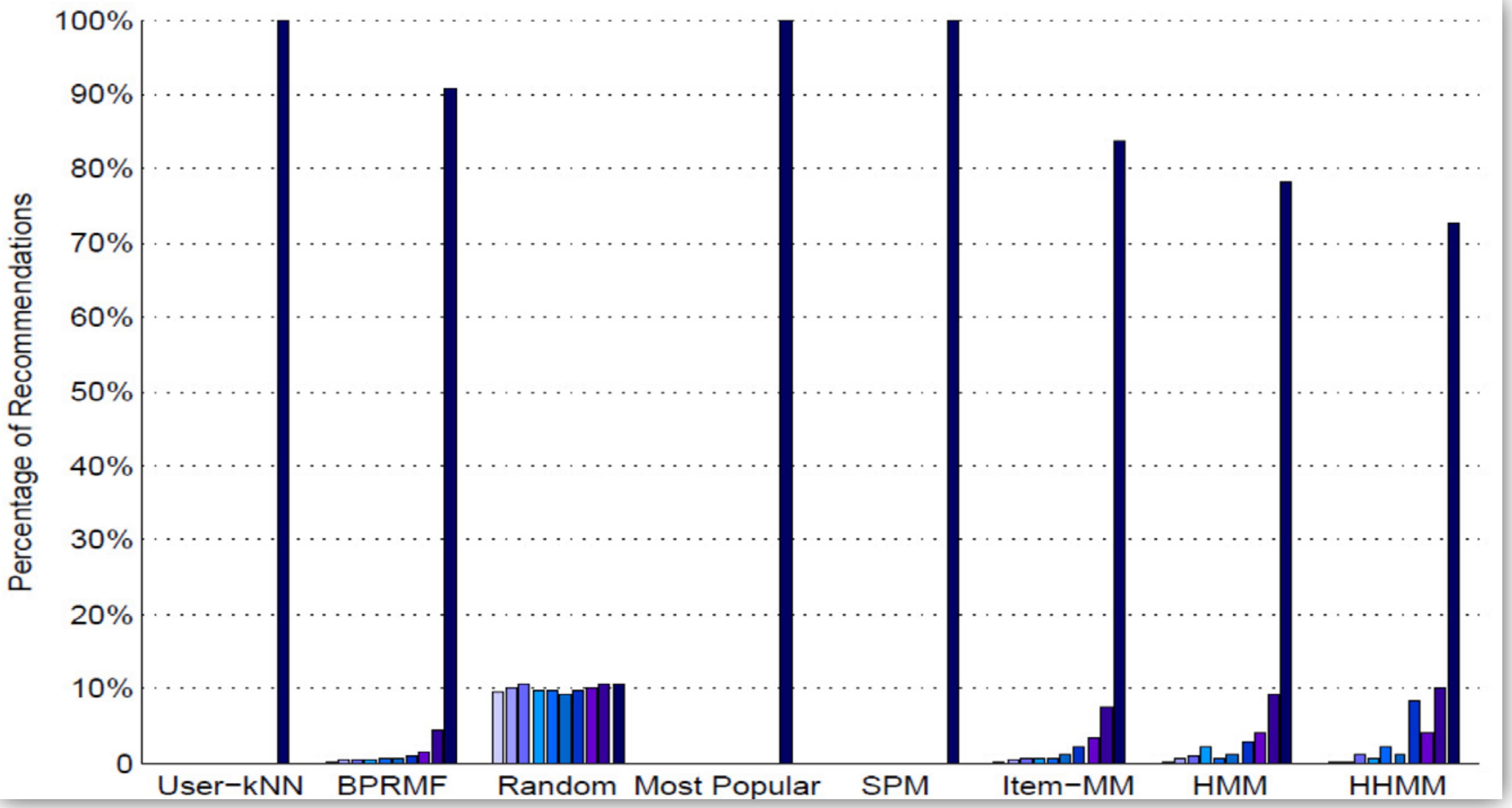
Results

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

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.



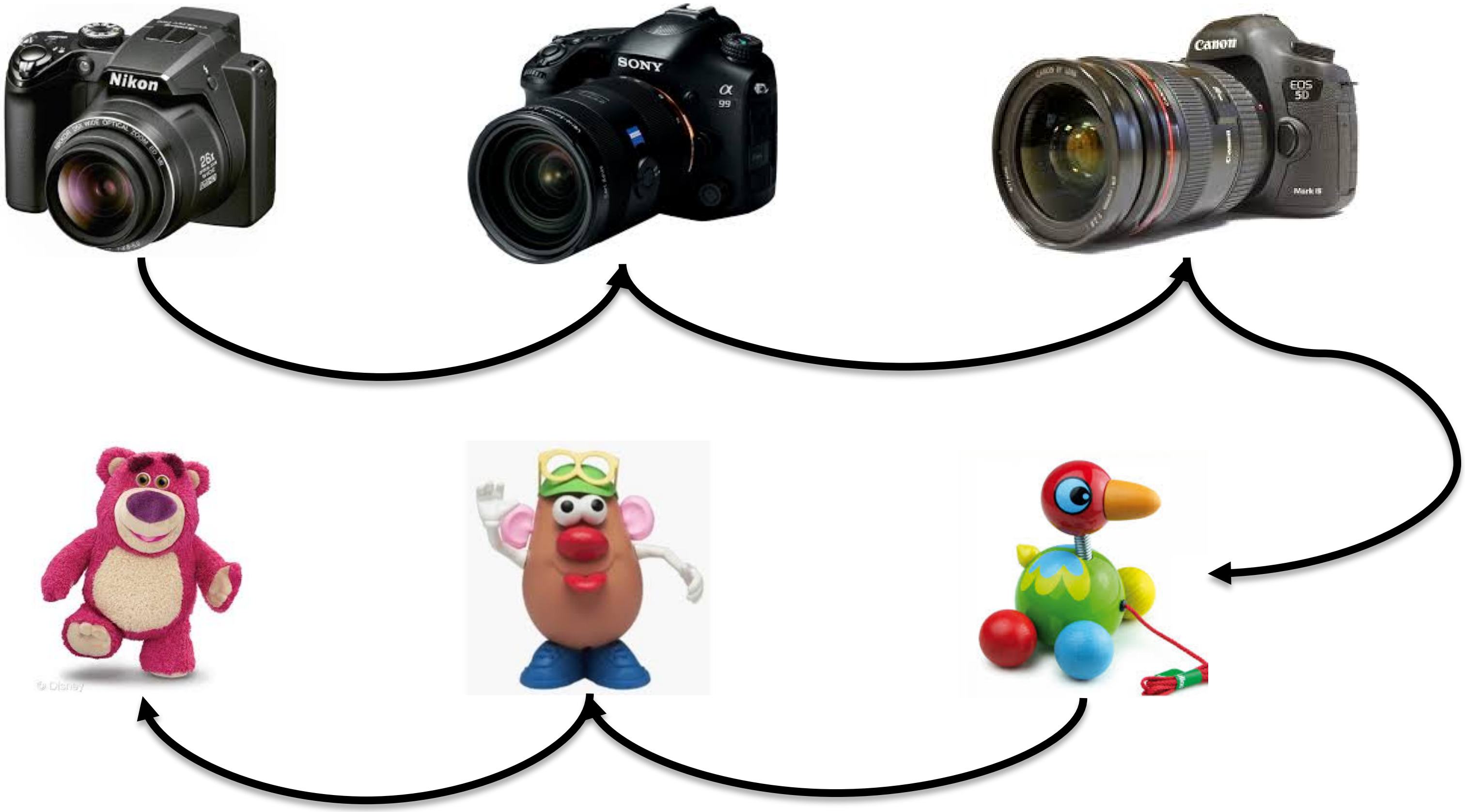
Highlighted Approach

Context Adaptation in Interactive Recommendation

Pedigree:

- Interactional context
- Fully dynamic environment
- Unobservable context

Problem: Change of Context



New Context = New Utility Function

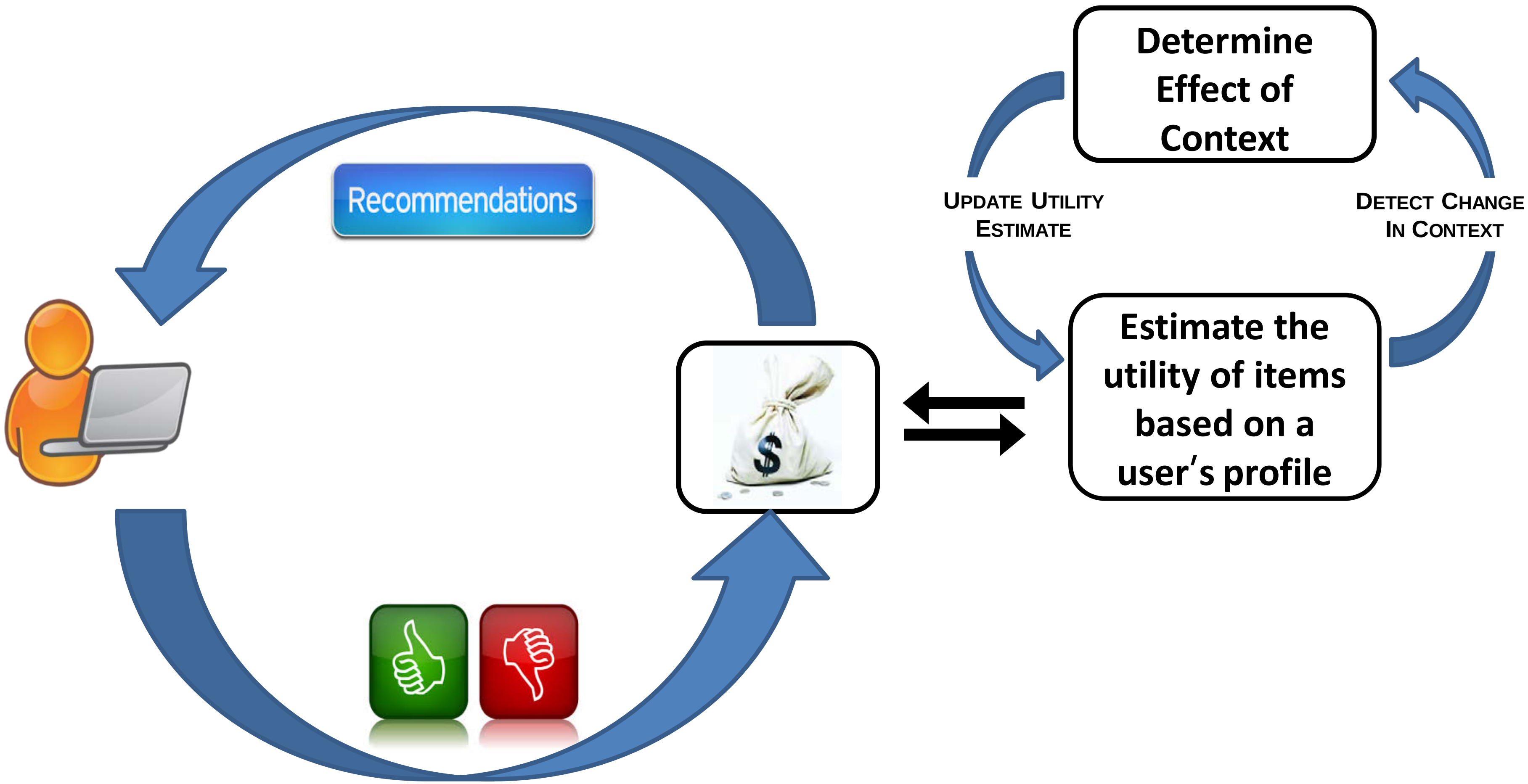
Detecting Context Changes

Context 1

Context 2

How do we infer hidden context or change of context?

Interactive Recommendation Scenario



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 θ characterizes the utility (reward) distribution**
- $E_r(r|a, \theta)$**: expected reward for item a for the given θ**

D=∅
for $t = 1$ to T **do**
  Draw $\theta^t \sim P(\theta|D)$
  Select $a_t = argmax E_r(r|a, \theta^t)$
  Observe reward r_t
  D $=$ D $\cup (a_t, r_t)$
end for

Assumptions

- **Theta Distribution**
  - θ is drawn from an multivariate normal distribution
    - user's preference function
    - location μ, covariance Σ
    - $\mathcal{N}(\mu_t, \Sigma_t)$
- **Linearity**
  - Expected reward is a linear function of the item features
  - $r_x = f_x \cdot \theta$
- **Reward distribution $\mathbf{P}(r|\theta, a)$**
  - linear transform of the θ 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:**

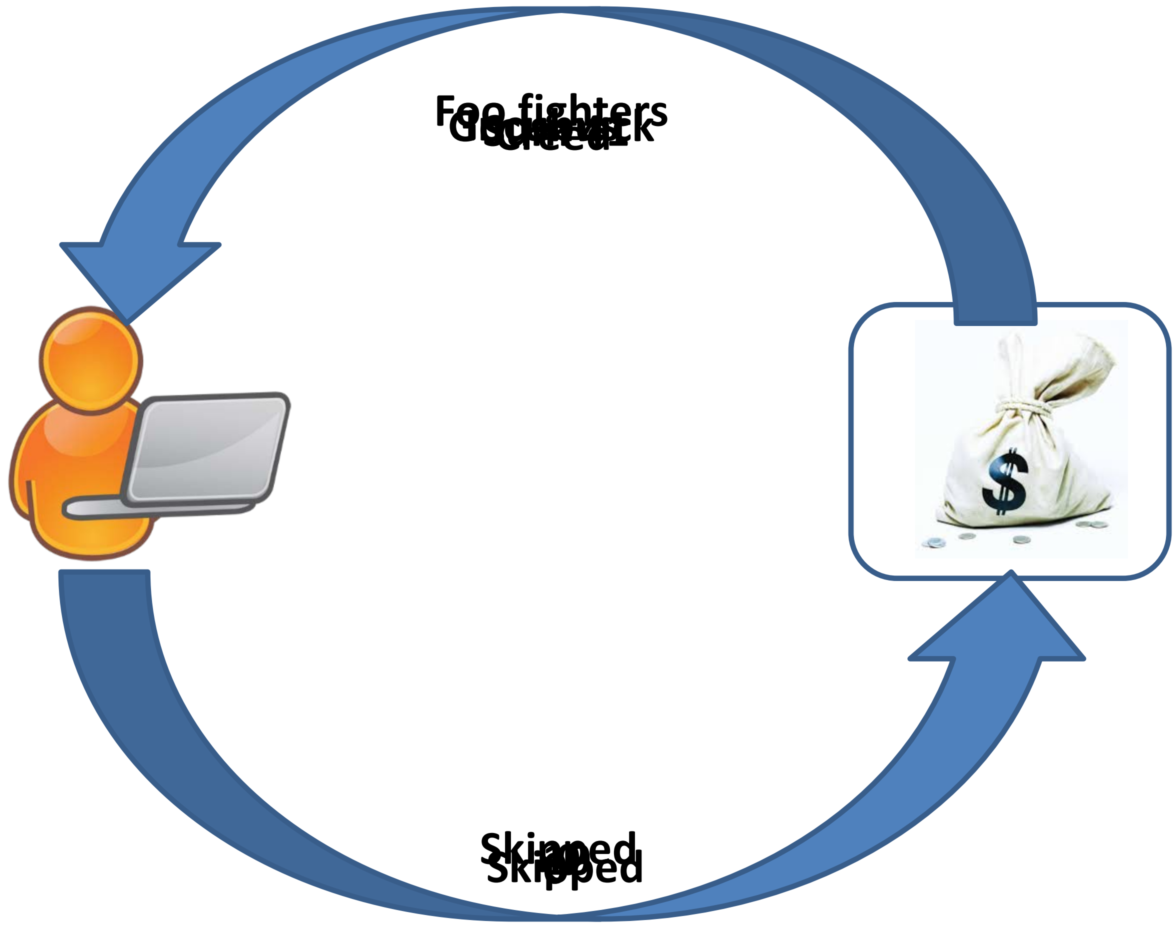
$$p(\theta|r) = \mathcal{N}(\mu_{\theta|r}, \Sigma_{\theta|r})$$
$$\Sigma_{\theta|r}^{-1} = \Sigma_\theta^{-1} + F^T \Sigma_r^{-1} F$$
$$\mu_{\theta|r} = \Sigma_{\theta|r}[F^T \Sigma_r^{-1}(r) + \Sigma_\theta^{-1}\mu_\theta]$$

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

Expected reward: $r_x = f_x \cdot \theta$

Change of Context

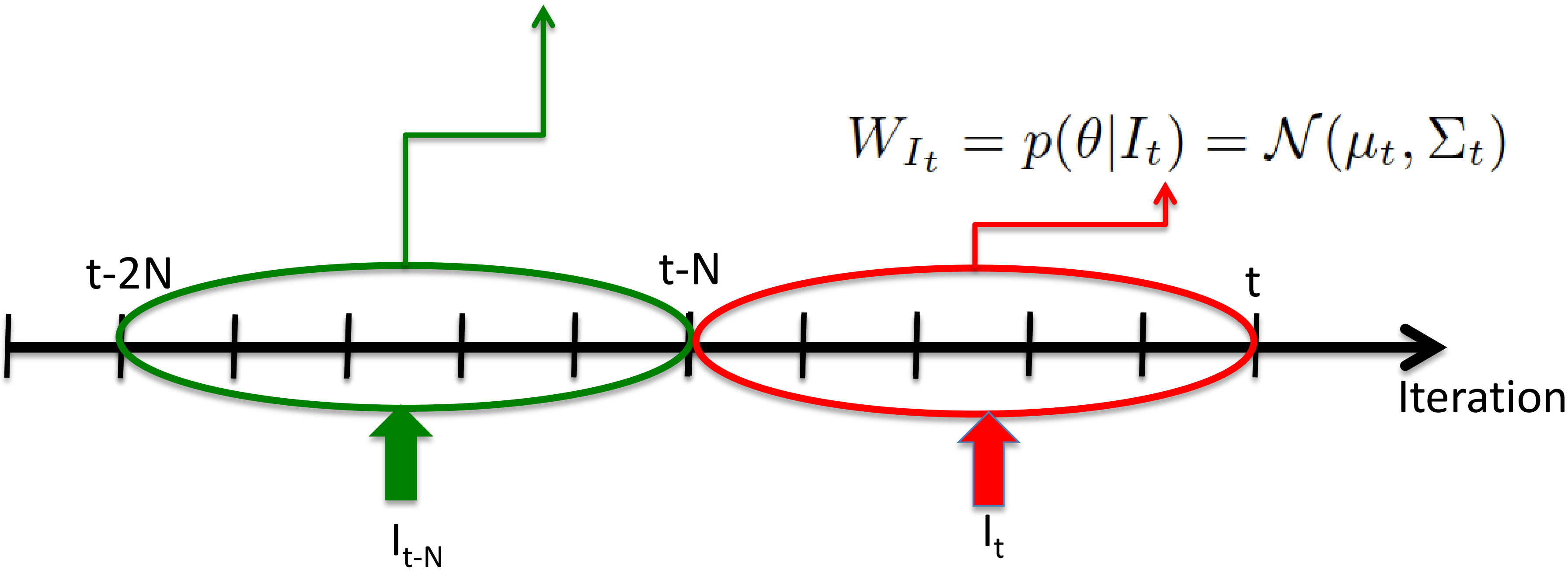
Artist	Score
Linkin park	
Staind	100
Metallica	90
Green Day	90
Simple Plan	Skipped
Papa Roach	90
Nirvana	90
Foo fighters	0
Creed	Skipped
Sum 41	10
Incubus	0
Godsmack	Skipped



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

$$W_{I_{(t-N)}} = \mathcal{N}(\mu_{t-N}, \Sigma_{t-N})$$

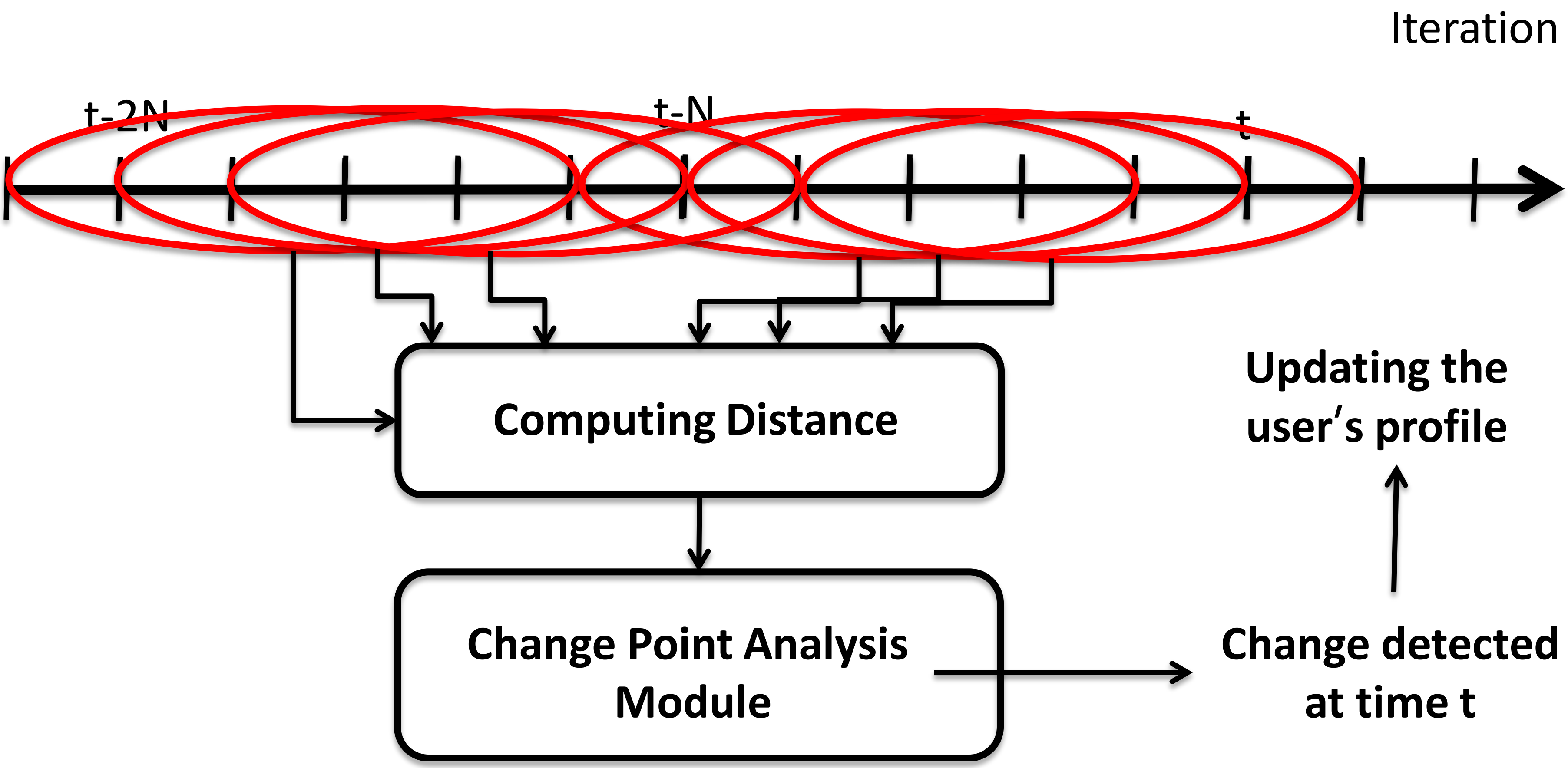
$$W_{I_t} = \mathcal{N}(\mu_t, \Sigma_t)$$

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

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

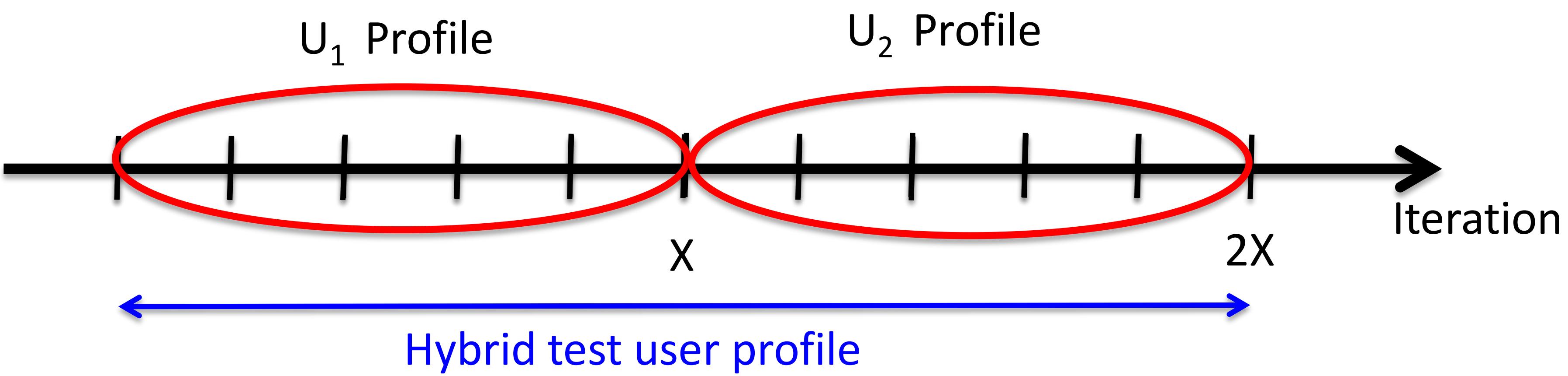
- **How to measure distance between two distributions**
  - KL-divergence
  - Mahalanobis distance
  - others

Sliding Window



Evaluation

- **Simulating the change in a user's behavior:**
  - Generating a hybrid user profile by switching between two random users in the test data.

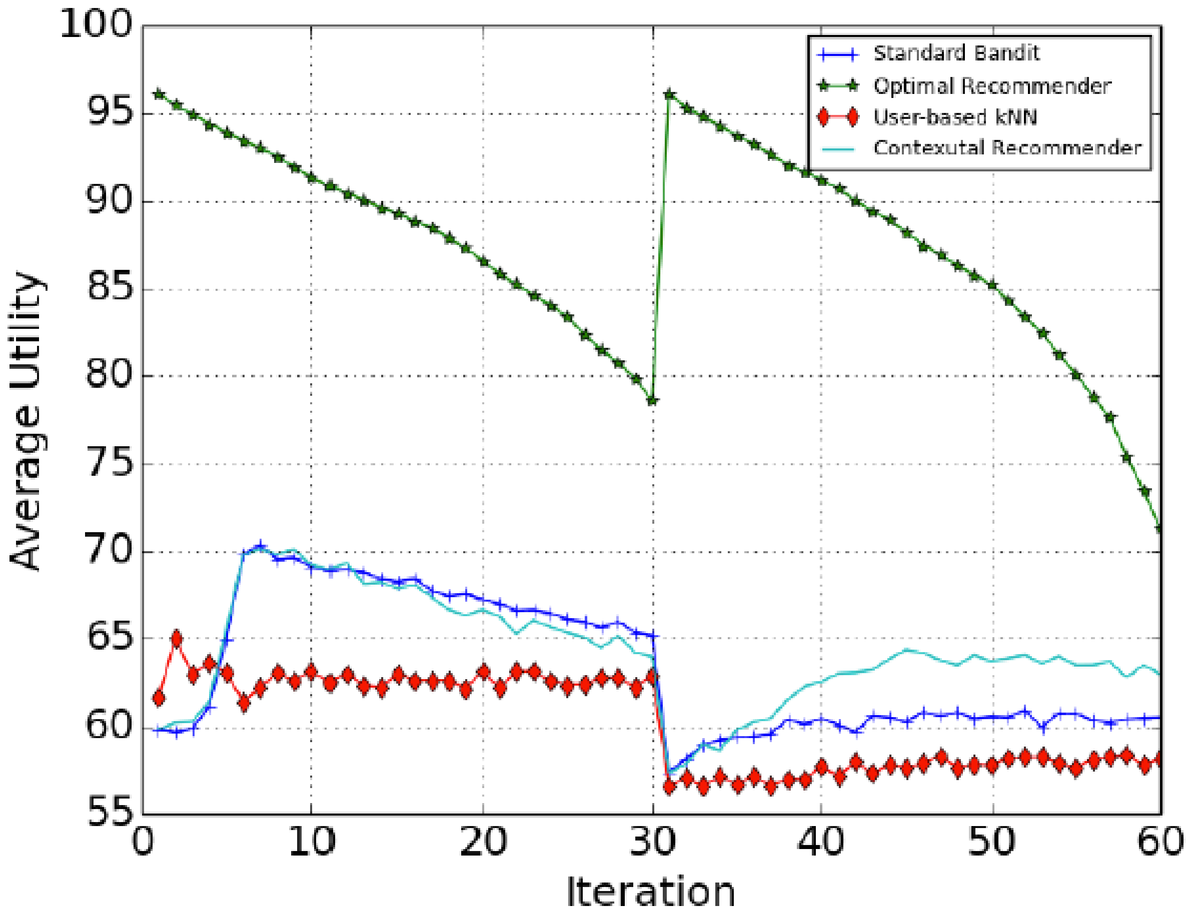


Experiments

- **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

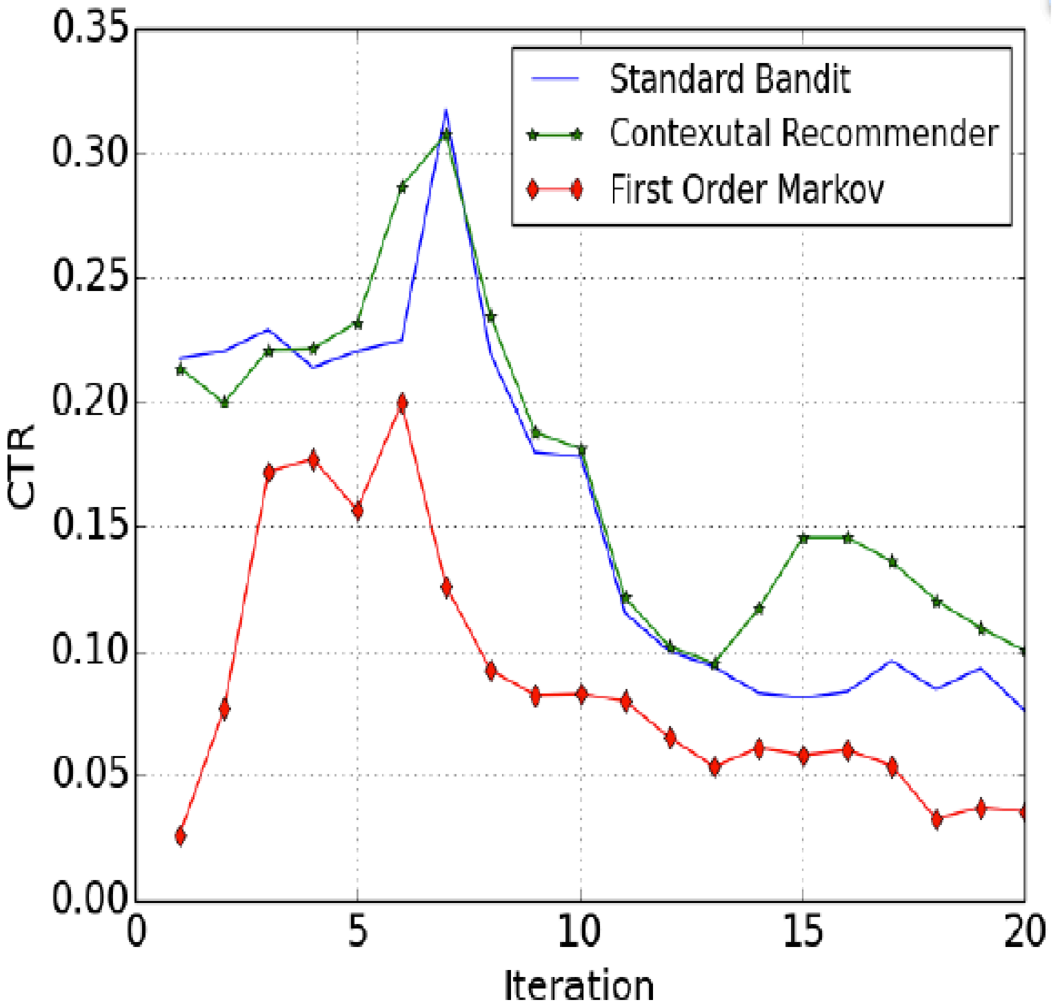
Results

- 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

?