Context-Aware User Modeling for Recommendation

Bamshad Mobasher

Center for Web Intelligence

School of Computing
DePaul University, Chicago

mobasher@cs.depaul.edu

UMAP 2013, Rome, Italy

Special Thanks To:

- **Yong Zheng, DePaul University**
  - Especially for help with material related to item/user splitting, differential context modeling, and statistics relates to context-aware recommendation in related conferences
  - **Make sure you go to his talk: Wednesday afternoon session**
- **Negar Hariri, DePaul University**
  - Especially for material related to context-aware song recommendation approach and query-driven context-aware recommendation
- **Francesco Ricci, Free University of Bozen, Bolzano**
  - Especially for material related to InCarMusic recommender systems, and other assorted material

Context in Recommendation

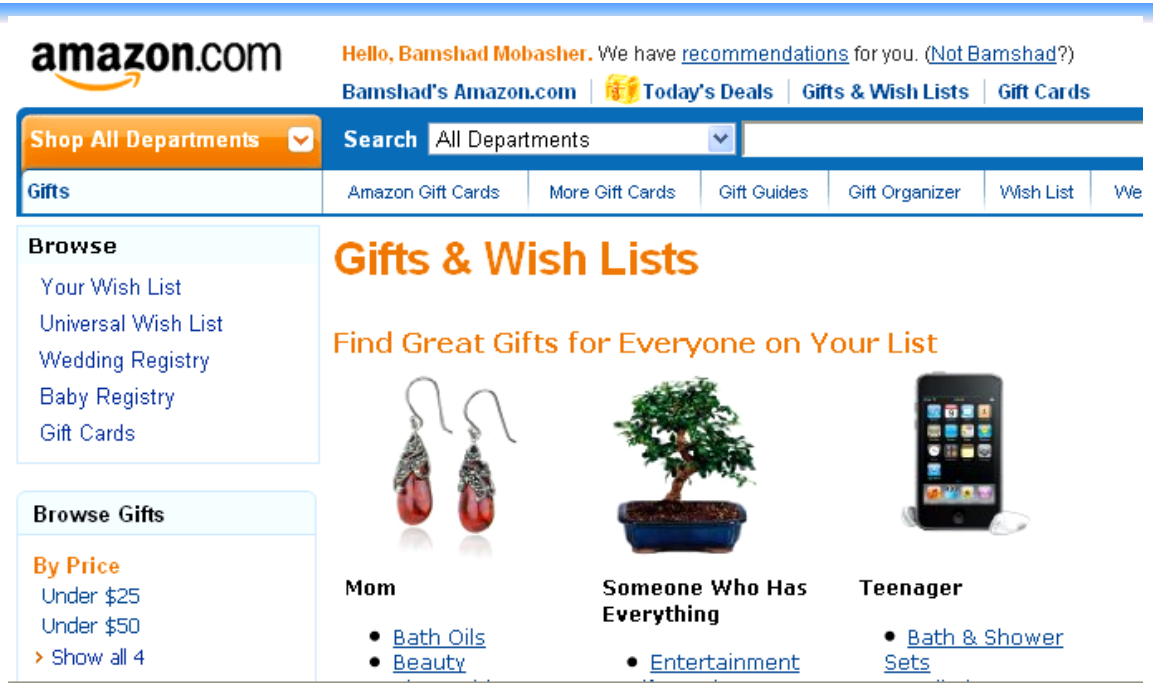
- **Recommendation Scenario**
  - Steve's purchases on Amazon:
    - mystery-detective fiction "Da Vinci Code" (for himself)
    - "Python Programming" (for work)
    - "Green Eggs and Ham" (gift for his daughter)
  - How should we represent Steve's *interest in books*?
  - System needs to know the difference between children books and computer books, i.e., the contexts in which Steve interacts with the system.
  - What should be recommended if Steve is reading reviews for a book on Perl Scripting?

Anatomy of a Famous Example

- **Jack buys a book on pregnancy for his friend Jill who is expecting**
- **The purchase becomes part of Jack's profile and in subsequent visits, he gets recommendations about baby cloths, child rearing, etc.**

- **Amazon' approach: distinguish between the task of gift buying versus buying items for oneself.**

Anatomy of a Famous Example



- **Is this a problem of context?**
- **Or, a problem in user profiling?**
- **Are they the same thing?**

- **Goal of identifying gifts, is to exclude them from profile not to change context**
- **Once excluded, no context is used for the actual user**

- **Even if "gift" were to be taken as a context, it would have to be handled differently for each recipient**

Context in Recommendation

Some forms of recommendation may be more contextual than others

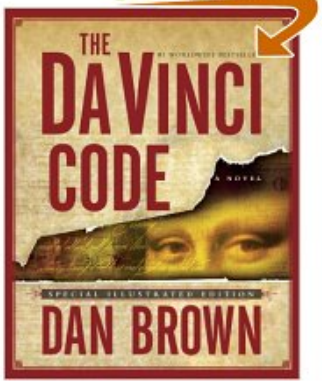
amazon.com | Bamshad's Store | Books | See All 32 Product Categories | Your Account | Cart | Your Lists | Help

Advanced Search | Browse Subjects | Bestsellers | The New York Times® Best Sellers | Magazines | Corporate Accounts | Amazon Shorts | AmazonConnect | Bargain Books | Textbooks

Search Books GO Find Gifts Web Search

Join Amazon Prime and ship Two-Day for free and Overnight for $3.99.

SEARCH INSIDE!™



Share your own customer images

Search inside another edition of this book

The Da Vinci Code: Special Illustrated Edition : A Novel (Paperback)

by Dan Brown "ROBERT LANGDON awoke slowly..." (more)

Explore: Books on Related Topics | Concordance | Text Stats | SIPs | CAPs

Browse: Front Cover | Copyright | Excerpt | Back Cover | Surprise Me!

List Price: ~~$22.95~~

Price: **$14.92** & eligible for **FREE Super Saver Shipping** on orders over $25. Details

You Save: $8.03 (34%)

Availability: Usually ships within 24 hours. Ships from and sold by Amazon.com.

Want it delivered Monday, April 24? Order it in the next 16 hours and 40 minutes, and choose **One-Day Shipping** at checkout. See details

42 used & new available from $13.90

Avg. Customer Review: (230 customer reviews)

Rate this item

I Own It

Quantity: 1

Add to Shopping Cart

or

Sign in to turn on 1-Click ordering.

A9.com users save 1.57 Amazon. Learn how.

More Buying Choices

42 used & new from $1

Available for in-store pi now from: $22.95

Price may vary based on availability

Enter your ZIP Code:

Have one to sell? Sell yours

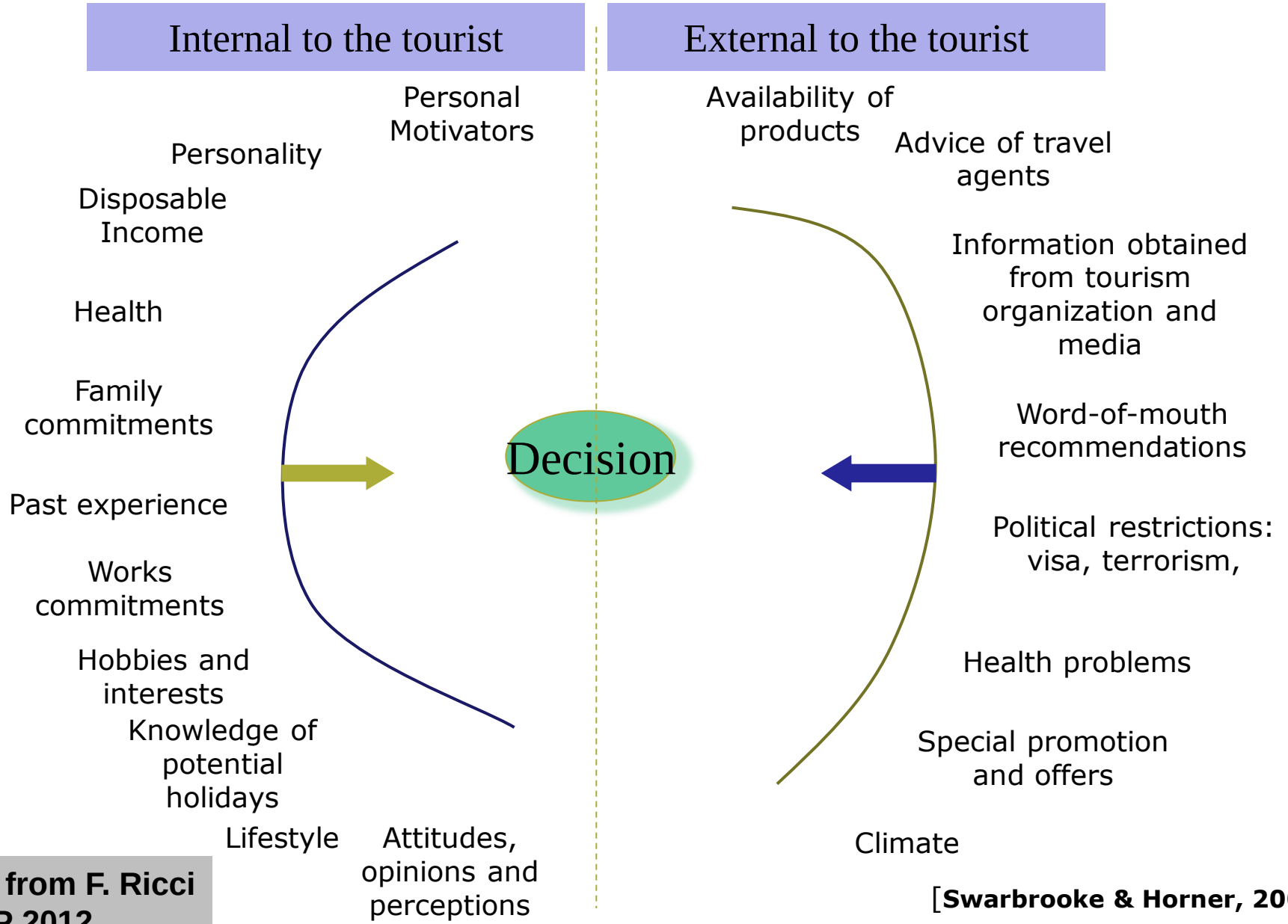
Customers who bought this item also bought

Angels & Demons by *Dan Brown*

Holy Blood, Holy Grail by *Michael Baigent*

Secrets of the Code: The Unauthorized Guide to the Mysteries Behind The Da Vinci Code by *Dan Burstein*

Factors influencing Holiday Decision



[**Swarbrooke & Horner, 2006**]

Slide from F. Ricci UMAP 2012

Relevant Questions

- "... it is difficult to find a relevant definition satisfying in any discipline. Is context a frame for a given object? Is it the set of elements that have any influence on the object? Is it possible to define context a priori or just state the effects a posteriori? Is it something static or dynamic? Some approaches emerge now in Artificial Intelligence. In Psychology, we generally study a person doing a task in a given situation. Which context is relevant for our study? The context of the person? The context of the task? The context of the interaction? The context of the situation? When does a context begin and where does it stop? What are the real relationships between context and cognition?”

- Bazire and Brezillon, 2005

Context in Recommendation

- **Recommendation Scenario - Revisited**
  - Steve's purchases on Amazon:
    - mystery-detective fiction "Da Vinci Code" (for himself)
    - "Python Programming" (for work)
    - "Green Eggs and Ham" (gift for his daughter)
  - Context seems to be tied to a particular interaction of user with the system
  - Utility of an item in a given context may be in conflict with overall preferences of the user
  - Context may change during one visit or interaction (e.g., when context is tied to task, location, etc.)
  - System needs to distinguish between different longer-term preferences and possible short-term interests

Outline

- **General views of context and their relevance to recommendation problem**
  - Representational versus Interactional Views
  - Representation and acquisition of context
  - System knowledge about context
- **Key Concepts in Contextual Aware Recommendation**
  - General Background on Recommender Systems
  - Architectures for integrating context in recommender systems
  - Highlighted Approaches
    - Item / User Splitting
    - Approaches based on Matrix Factorization
    - Differential Contextual Modeling
  - Implementation Frameworks & Example Implementations
    - Representational → Multi-dimensional recommendation
    - Interactional → A Framework based on human memory

What is Context?

- **By example**
  - Location, time, identities of nearby users …
- **By synonym**
  - Situation, environment, circumstance
- **By dictionary [WordNet]**
  - the set of facts or circumstances that surround a situation or event
- **Problems:**
  - New situations don’t fit examples
  - How to use in practice?

Types of Context

[Fling, 2009]

- **Physical context**
  - time, position, and activity of the user, weather, light, and temperature ...
- **Social context**
  - the presence and role of other people around the user
- **Interaction media context**
  - the device used to access the system and the type of media that are browsed and personalized (text, music, images, movies, …)
- **Modal context**
  - The state of mind of the user, the user's goals, mood, experience, and cognitive capabilities.

Defining Context

- **Entities interact with their environment through "situated actions"**
  - "Any information that can be used to characterise the situation of entities." (Dey et al., 2001)
- **Context of an entity exist independently and outside of the entity's actions**
  - Everything that affects computation except its explicit input and output." (Lieberman and Selker, 2000)
- **Intensionality versus Extensionality**

Different Views of Context

- **Dourish (2004) distinguishes between two views of context:**
  - representational view and the interactional view
- **Representational view, makes four key assumptions:**
  - Context is a form of information, it is delineable, stable and separable from the activity
- **Implications:**
  - Context is information that can be described using a set of “appropriate” attributes that can be observed
  - These attributes do not change and are clearly distinguishable from features describing the underlying activity undertaken by the user within the context
  - No “situated action”

Different Views of Context

- **Interactional View of Context (Dourish, 2004)**
  - Contextuality is a relational property, i.e. some information may or may not be relevant to some activity
  - The scope of contextual features is defined dynamically, and is occasioned rather than static
  - Rather than assuming that context defines the situation within which an activity occurs, there is a cyclical relationship between context and activity:
    - Context gives rise to the activity and activity changes the context

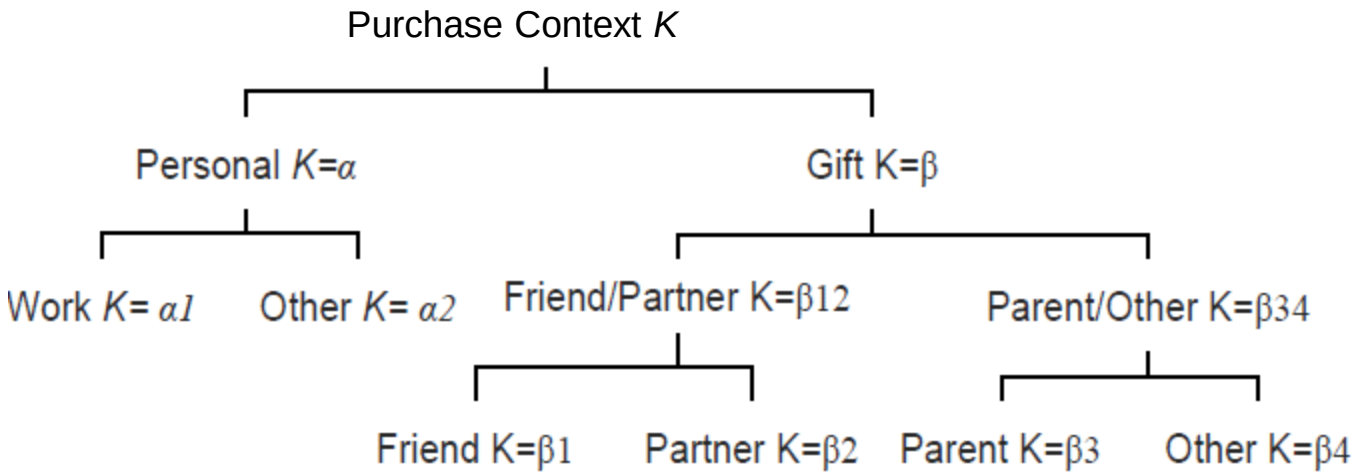
Representational View: Assumptions & Implications

- **Context can be represented as an explicit, enumerated set of static attributes (i.e., it's "extensional")**
  - Typically attributes are predefined based on the characteristics of the domain and environment
    - E.g., time, date, location, mood, task, device, etc.
  - Contextual variable can have associated structure
    - E.g., Sunday < Weekend
- **Implications:**
  - Must identify and acquire contextual information as part of data collection before actual recommendations are made.
  - Relevant contextual variables (and their structures) must be identified at the design stage.
- **Drawback**
  - The "qualification problem" – similar to the outstanding problem from AI.

Example of Contextual Variables

- Temporal context: a temporal hierarchy with multiple temporal relationships of varying granularity, e.g.,
  - Time (2008.10.19 11:59:59pm) → Date (2008.10.19) → Year (2008)
  - Time (2008.10.19 11:59:59pm) → Hour (11pm) → TimeOfDay (evening)
  - Date (2008.10.19) → Month (October) → Season (Fall)
  - Date (2008.10.19) → DayOfWeek (Sunday) → TimeOfWeek (Weekend)
- Purchase Context:

Adomavicius, Tuzhilin, 2008



Representing Contextual Variables

- Formally, contextual information can be defined as a *vector* of contextual variables $c = (c_1,\ldots,c_n)$, where $c_i \in C_i$
  - Sometimes we refer to c as a *contextual condition*
- $C = C_1 \times \ldots \times C_n$ denotes the space of possible values for a given context
  - Each component C_i may have additional structure (e.g., a tree): it can be defined as a hierarchical set of nodes (concepts)
  - If $c_i \in C_i$, then c_i represents one of the nodes in the hierarchy C_i
- Example:
  - C = PurchaseContext × TemporalContext
  - c = (work, weekend), i.e., purchasing something for work on a weekend is the contextual condition

Interactional View: Assumptions & Implications

- **Properties of Context**
  - Context gives rise to a behaivor that is observable, though context itself may not be observable (it's "intensional")
    - Context exists (usually implicitly) in relation to the ongoing interaction of the user with the system
  - not static
    - Can be derived: a stochastic process with d states $\{c_1, c_2, \ldots, c_d\}$ representing different contextual conditions
- **Context aware recommendation**
  - Explicit representation of context may not be as important as
    - recognizing behavior arising from the context
    - adapting to the needs of the user within the context
- **Drawback:** Ability to explain recommendations

Obtaining Context

- Explicitly specified by the user
  - E.g., “I want to watch a movie at home with my parents”
- Observed or deduced by the system
  - Time (from system clock)
  - Location (from GPS)
  - Deduced from user’s behavior (e.g., shopping for business or pleasure)
  - Etc.
- Significant research literature on obtaining, inferring, and predicting context (e.g., for mobile computing)
- We will only discuss as part of the description of some highlighted implementations

Relevance of Contextual Information

- Not all contextual information is relevant for generating recommendations
- E.g., which contextual information is relevant when recommending a book?
  - For what purpose is the book bought? (Work, leisure, …)
  - When will the book be read? (Weekday, weekend, …)
  - Where will the book be read? (At home, at school, on a plane, …)
  - How is the stock market doing at the time of the purchase?
- Determining relevance of contextual information:
  - *Manually*, e.g., using domain knowledge of the recommender system's designer
  - *Automatically*, e.g., using feature selection procedures or statistical tests based on existing data

What the system knows about context

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

See: Adomavicius, Mobasher, Ricci, and Tuzhilin. Context Aware Recommender Systems. *AI Magazine*, Fall 2011

Recommender Systems Basics

The Recommendation Task

- **Basic formulation as a prediction problem**

Given a profile $\boldsymbol{P_u}$ for a user $\boldsymbol{u}$, and a target item $\boldsymbol{i_t}$, predict the preference score of user $\boldsymbol{u}$ on item $\boldsymbol{i_t}$

- **Typically, the profile P_u contains preference scores by u on some other items, $\{i_1, \ldots, i_k\}$ different from i_t**
  - preference scores on $i_1, \ldots, i_k$ may have been obtained explicitly (e.g., movie ratings) or implicitly (e.g., time spent on a product page or a news article)

Recommendation as Rating Prediction

- Two types of entities: *Users* and *Items*
- Utility of item *i* for user *u* is represented by some rating *r* (where $r \in Rating$)
- Each user typically rates a *subset* of items
- Recommender system then tries to estimate the unknown ratings, i.e., to extrapolate rating function *R* based on the known ratings:
  - $R: Users \times Items \rightarrow Rating$
  - I.e., two-dimensional recommendation framework
- The recommendations to each user are made by offering his/her highest-rated items

Traditional Recommendation Approaches

- **Collaborative Filtering**
  - Give recommendations to a user based on preferences of “similar” users
- **Content-Based Filtering**
  - Give recommendations to a user based on items with “similar” content in the user’s profile
- **Rule-Based (Knowledge-Based) Filtering**
  - Provide recommendations to users based on predefined (or learned) rules
  - age(x, 25-35) and income(x, 70-100K) and children(x, >=3) → recommend(x, Minivan)
- **Combined or Hybrid Approaches**

Content-Based Recommenders

- **Predictions for unseen (target) items are computed based on their similarity (in terms of content) to items in the user profile.**
- **E.g., user profile P_u contains**

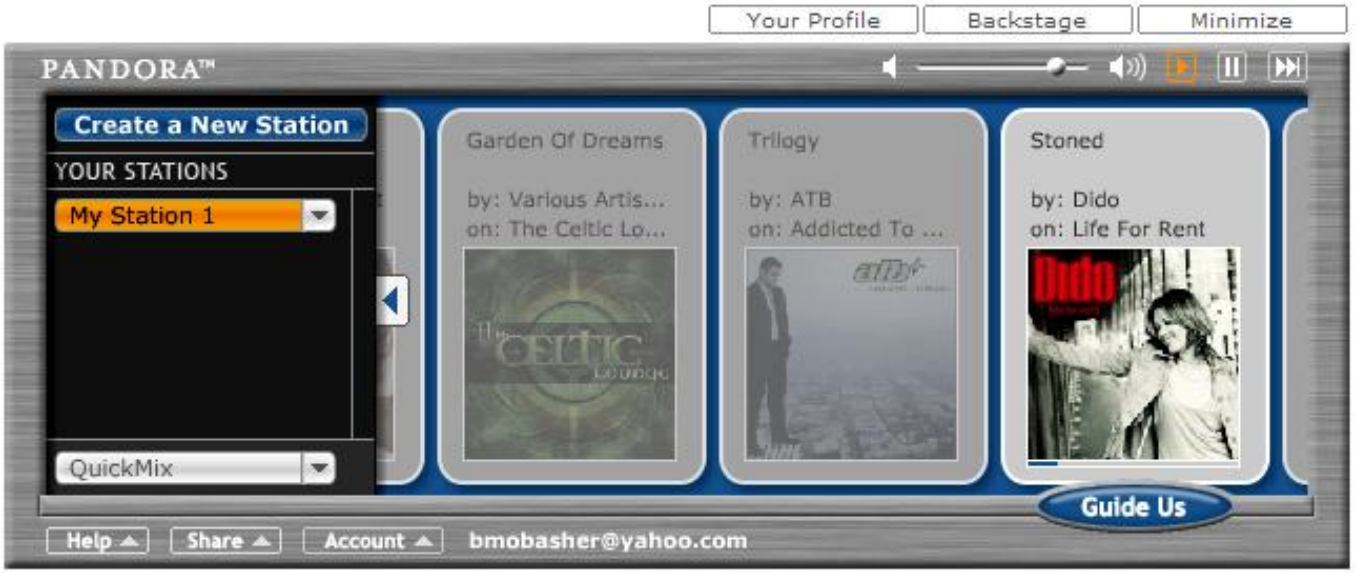
recommend highly: 

and recommend "mildly": 

Content-Based Recommenders

:: more examples

- **Music recommendations**
- **Play list generation**



Example: Pandora

Collaborative Recommenders

- **Predictions for unseen (target) items are computed based the other users' with similar ratings on items in user *u*'s profile**
  - i.e. users with similar tastes (aka "nearest neighbors")
  - requires computing correlations between user u and other users according to interest scores or ratings
  - *k*-nearest-neighbor (*k*nn) strategy
  - User-based collaborative filtering
- **Item-based Collaborative Filtering**
  - item-item similarities are computed in space of ratings
    - two items are similar if they are rated similarly by many users
  - prediction for target item is computed based on user u's own ratings on the most similar items

Example Collaborative System

	Item1	Item 2	Item 3	Item 4	Item 5	**Item 6**	Correlation with Alice
Alice	5	2	3	3		**?**	
User 1	2		4		4	1	-1.00
User 2	2	1	3		1	2	0.33
User 3	4	2	3	2		1	.90
User 4	3	3	2		3	1	0.19
User 5		3		2	2	2	-1.00
User 6	5	3		1	3	2	0.63
User 7		5		1	5	1	-1.00

Prediction

Best match

Using k-nearest neighbor with k = 1

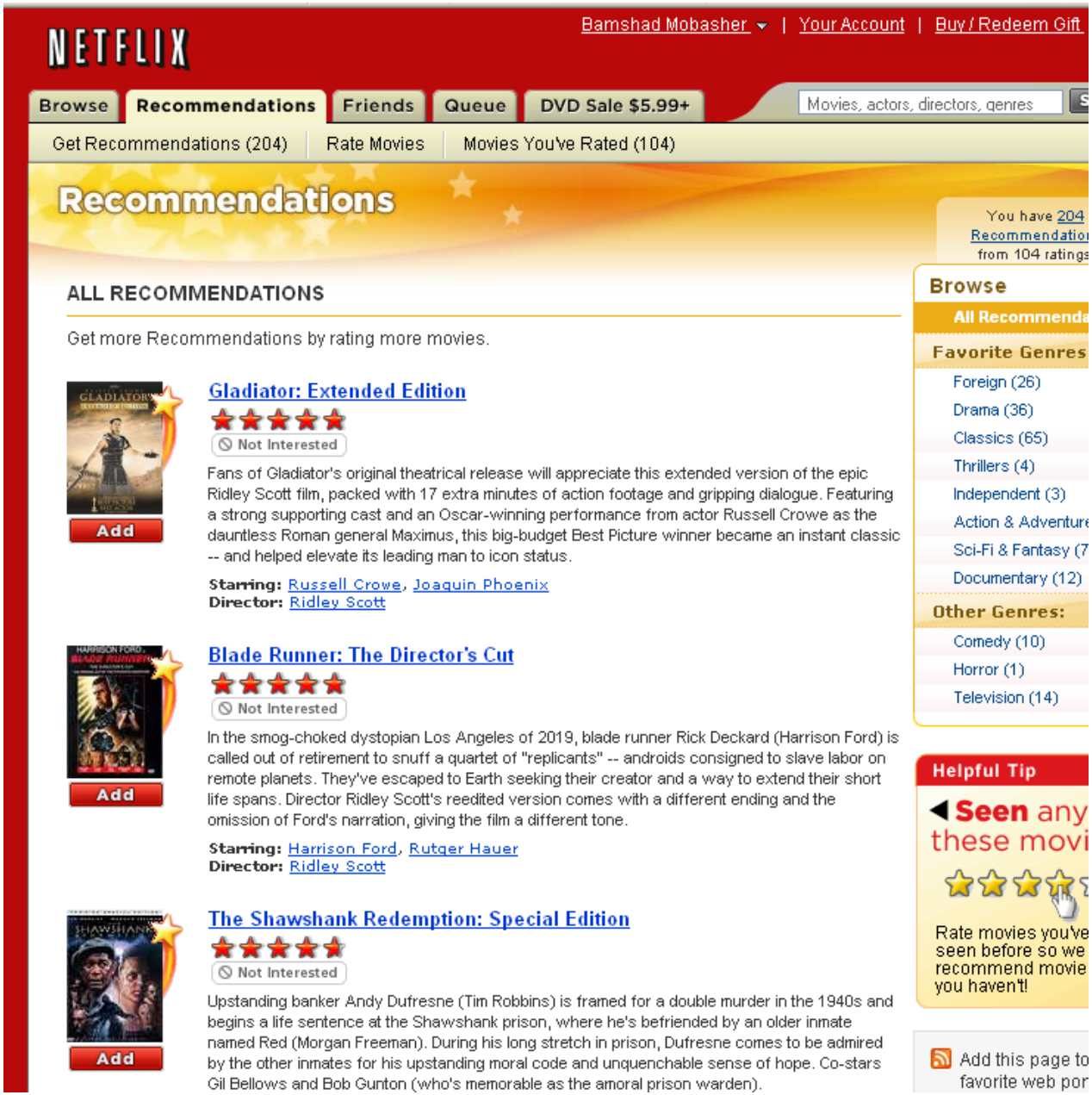
Item-Based Collaborative Filtering

	Item1	Item 2	Item 3	Item 4	Item 5	**Item 6**
Alice	5	2	3	3		**?**
User 1	2		4		4	1
User 2	2	1	3		1	2
User 3	4	2	3	2		1
User 4	3	3	2		3	1
User 5		3		2	2	2
User 6	5	3		1	3	2
User 7		5		1	5	1
Item similarity	0.76	0.79	0.60	0.71	0.75	

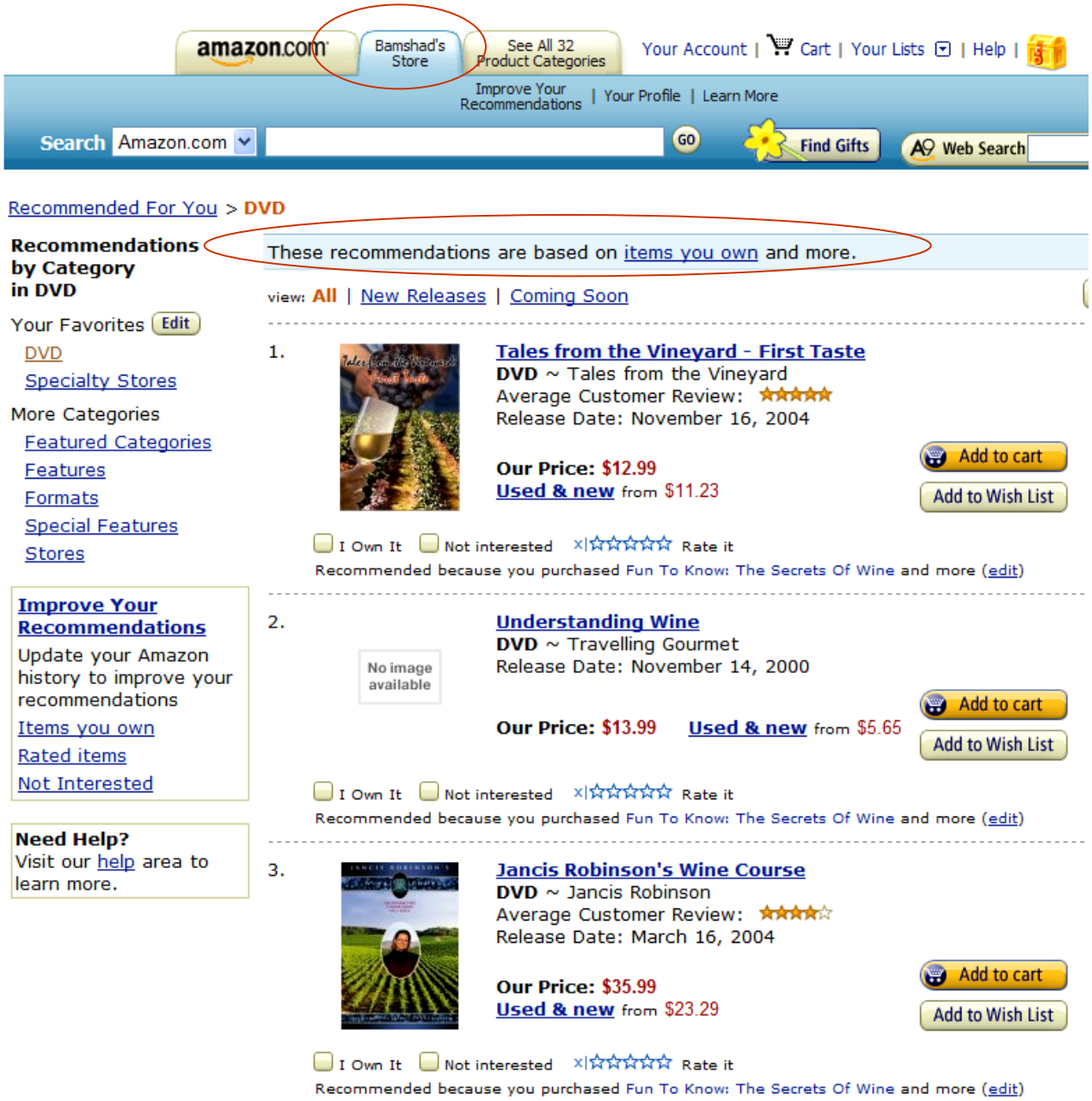
Prediction

Best match

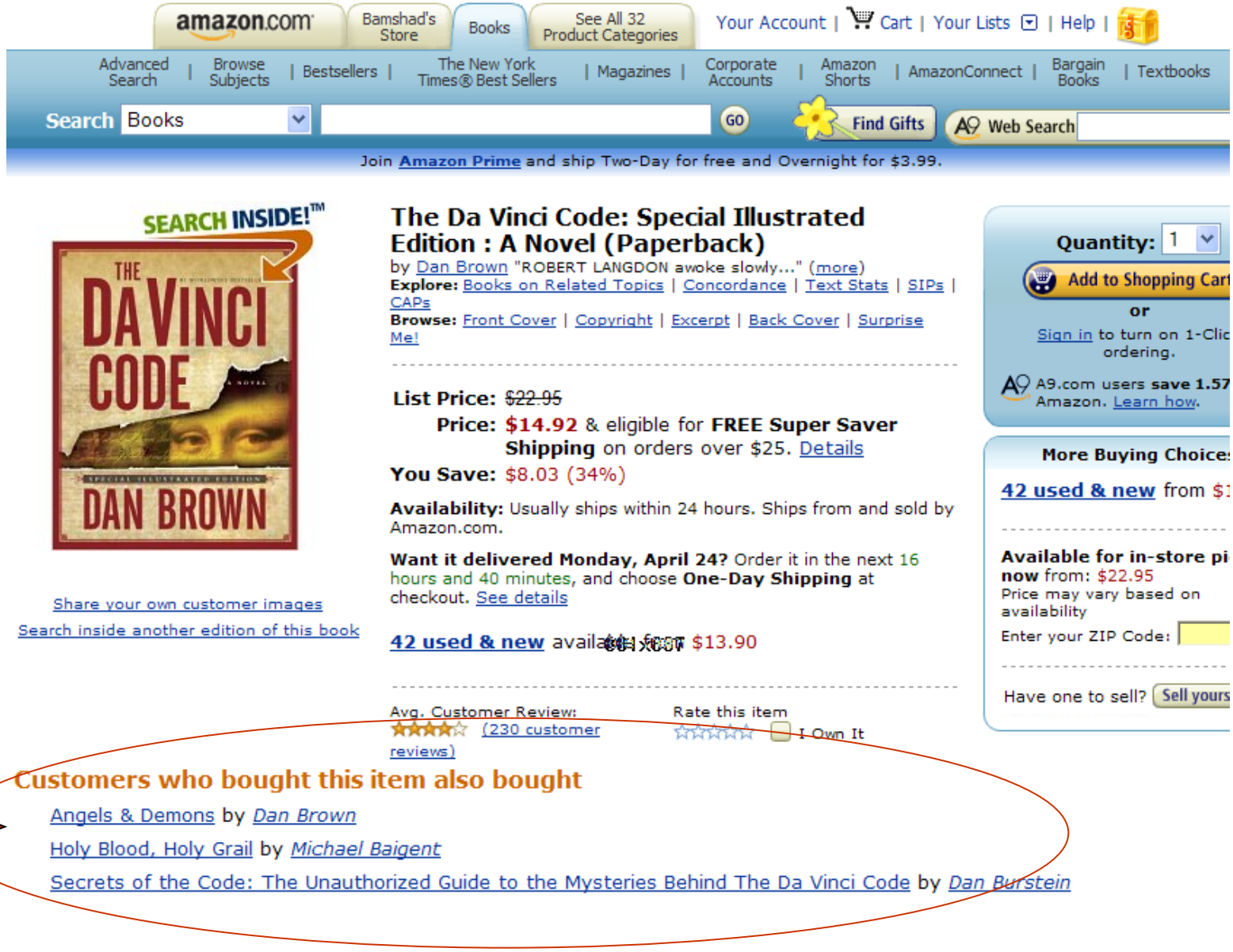
Collaborative Recommender Systems



Collaborative Recommender Systems



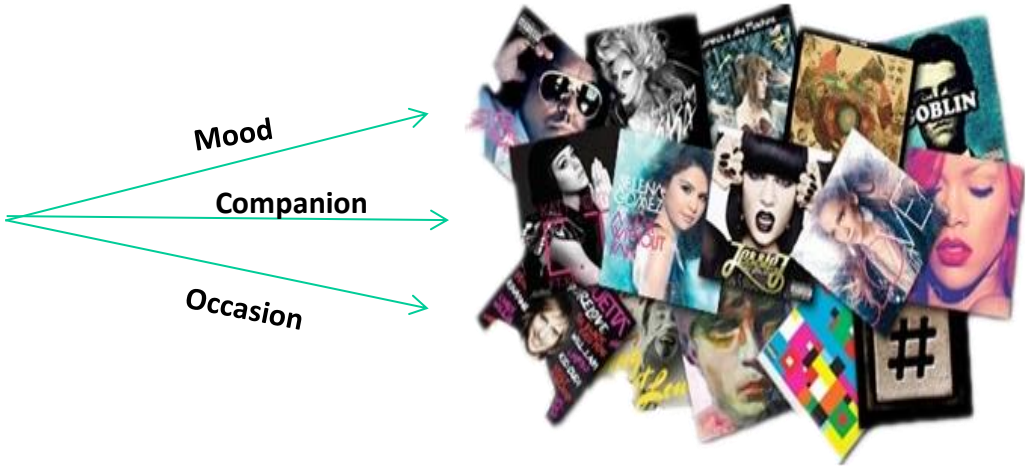
Collaborative Recommender Systems



Context In Recommender Systems

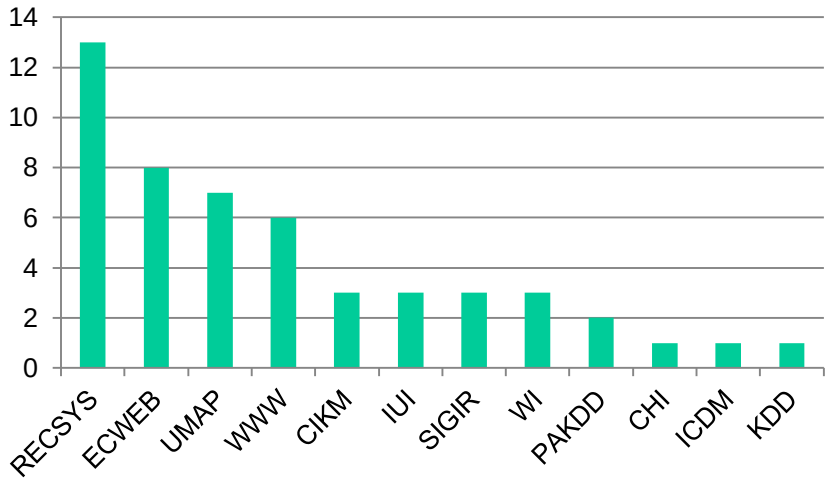
Context-Aware RS (CARS)

- **Traditional RS:** Users × Items → Ratings
- **Contextual RS**: Users × Items × Contexts→ Ratings



- **There is always a context**
  - Recommendations are not “usable” apart from context

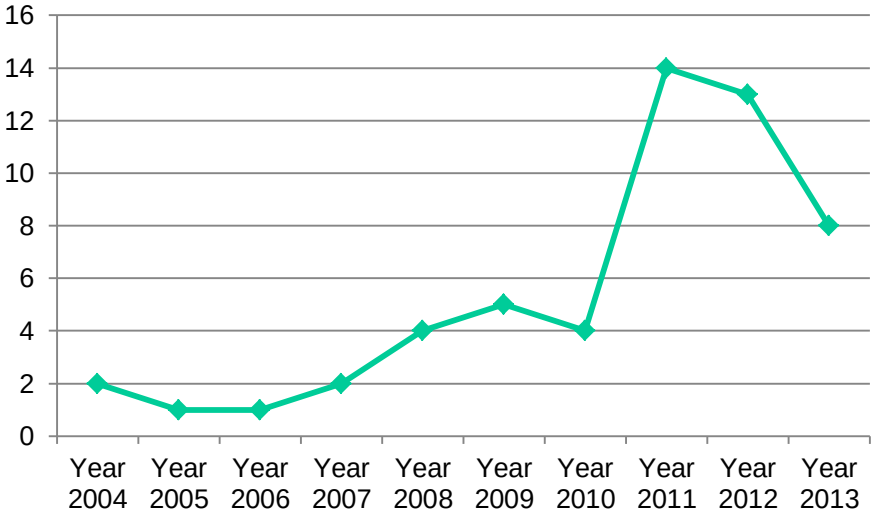
Statistics on CARS Related Papers



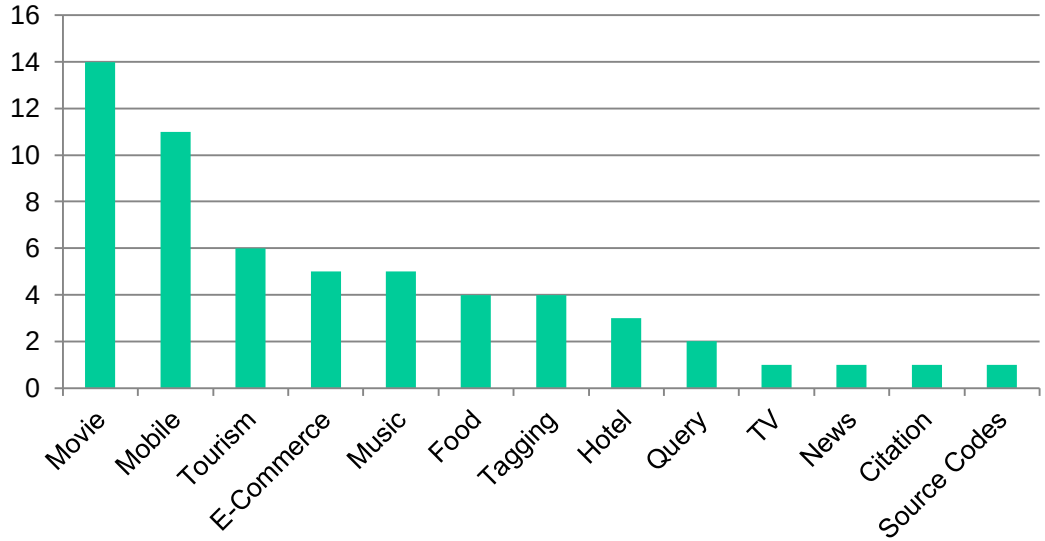
← By Conference

By Year →

2013 only includes:
UMAP, WWW, ECWEB

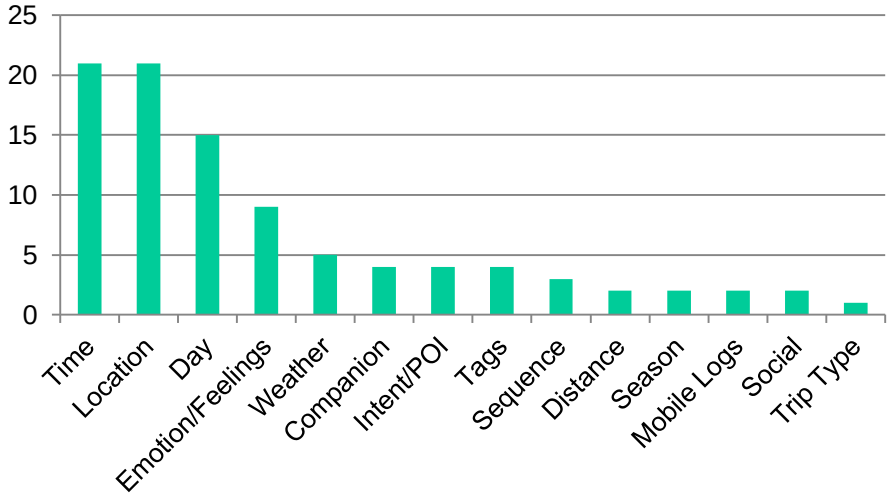


Statistics on CARS Related Papers

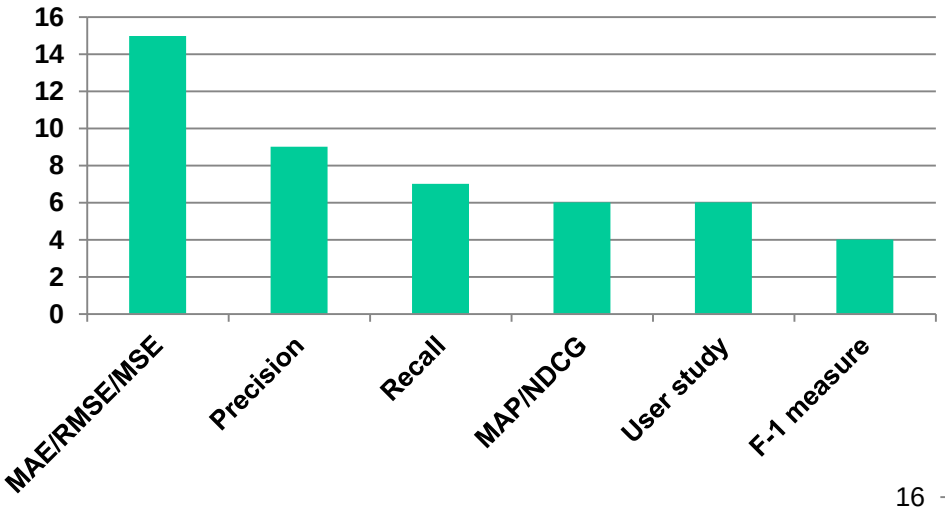


← By Application Domain

By Context Type →

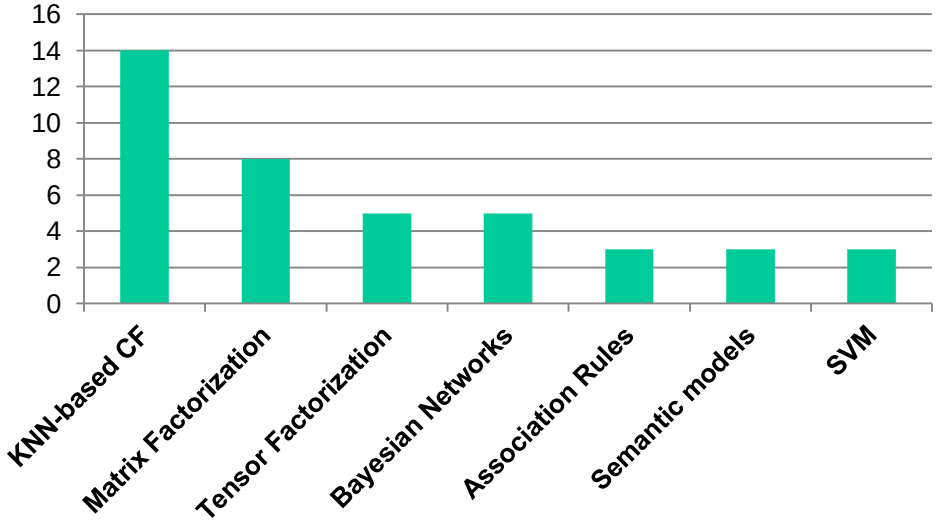


Statistics on CARS Related Papers



← By Evaluation Metric

By Algorithm →

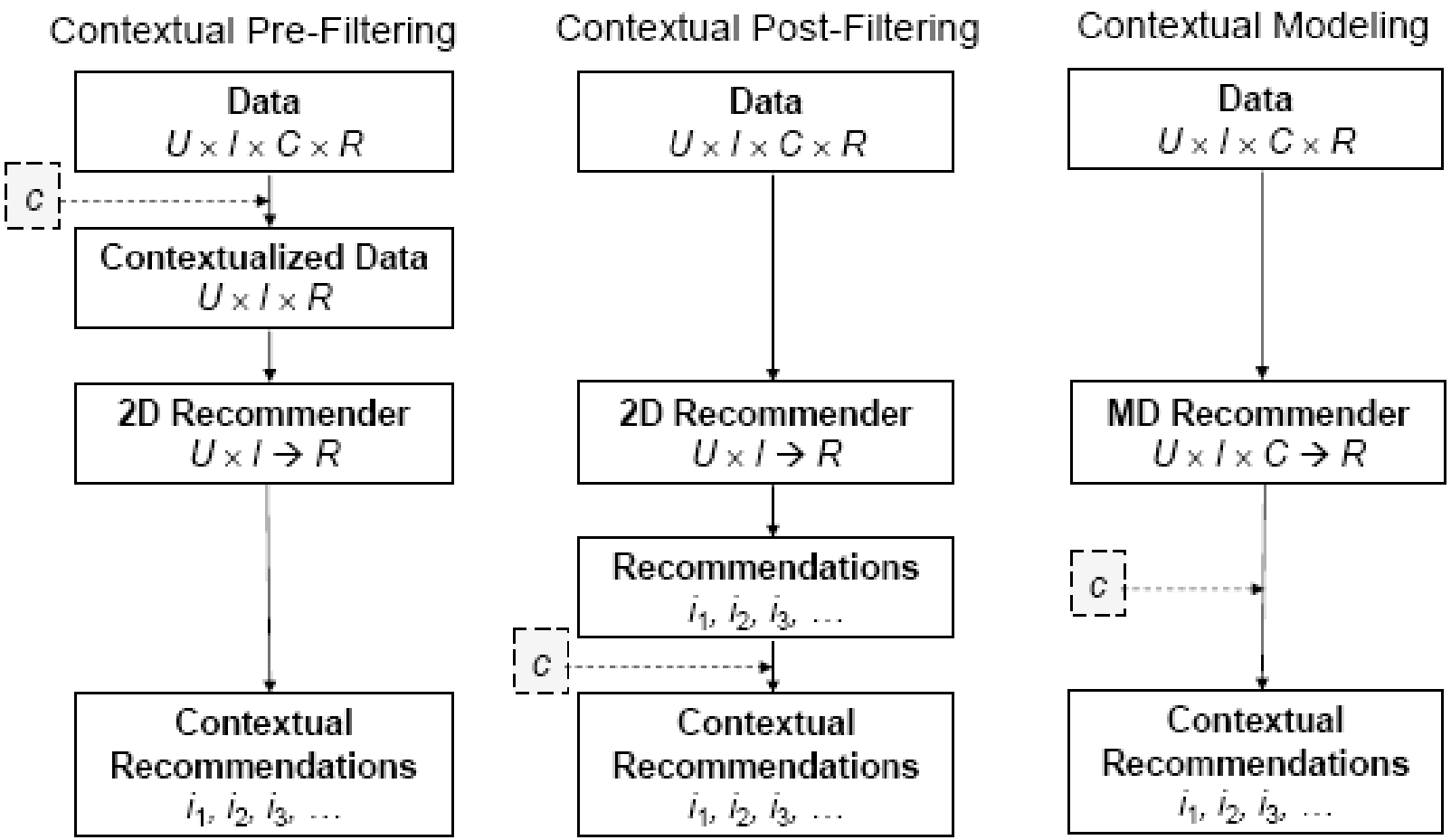


CARS Architectural Models

- **Three types of Architecture for using context in recommendation (Adomavicius, Tuzhilin, 2008)**
  - Contextual Pre-filtering
    - Context information used to select relevant portions of data
  - Contextual Post-filtering
    - Contextual information is used to filter/constrain/re-rank final set of recommendations
  - Contextual Modeling
    - Context information is used directly as part of learning preference models
- **Variants and combinations of these are possible**
- **Originally introduced based on the representational view**
  - Though these architectures are also generally applicable in the interactional view

CARS Architectural Models

From Adomavicius, Tuzhilin, 2008

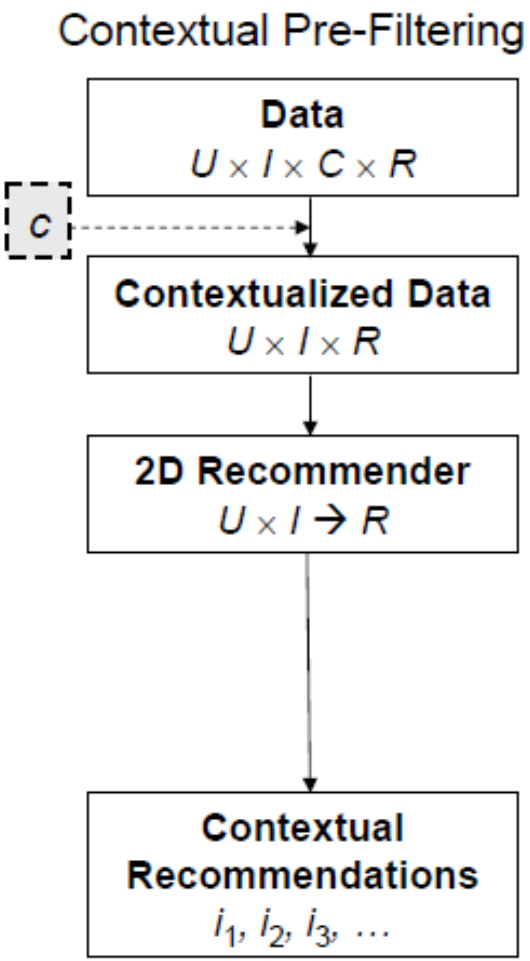


Contextual Pre-Filtering

- *Pre-Filtering*: using contextual information to select the most relevant data for generating recommendations
  - Context *c* serves as a *query* to select relevant ratings data Data(User, Item, Rating, Context), i.e.,
  - SELECT User, Item, Rating

    FROM Data

    WHERE Context = *c*
- *Example:* if a person wants to see a movie on Saturday, *only* the Saturday rating data is used to recommend movies

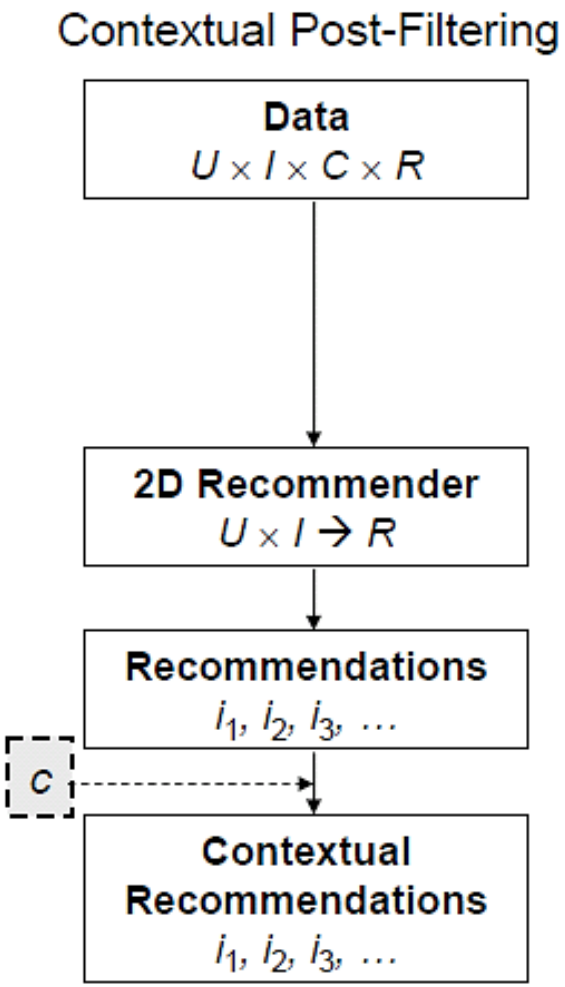


Contextual Pre-Filtering Challenges

- Context Over-Specification
  - Using an exact context may be too narrow:
    - Watching a movie with a girlfriend in a movie theater on Saturday
  - Certain aspects of the overly specific context may not be significant (e.g., Saturday vs. weekend)
  - Sparsity problem: overly specified context may not have enough training examples for accurate prediction
- Pre-Filter Generalization
  - Different Approaches
  - "Roll up" to higher level concepts in context hierarchies
    - E.g., Saturday → weekend, or movie theater → any location
  - Use latent factors models or dimensionality reduction approaches (Matrix factorization, LDA, etc.)

Contextual Post-Filtering

- Ignore context in the data selection and modeling phases, but filter or (re-rank) recommendations based on contextual information
- *Example: Context → Watching a movie with family*
  - Suppose the user generally watches comedies and dramas when going to theater with her family.
  - First, generate recommendations using standard recommender.
  - Then filter out action movies from the recommendation list.

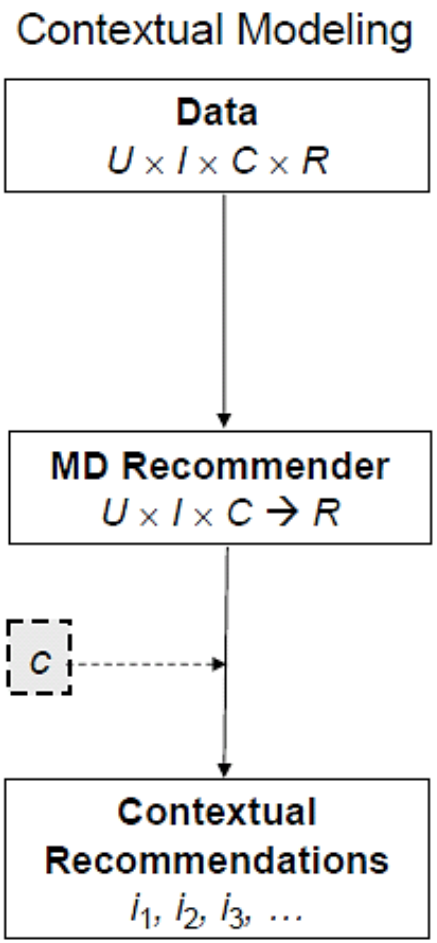


Contextual Post-Filtering

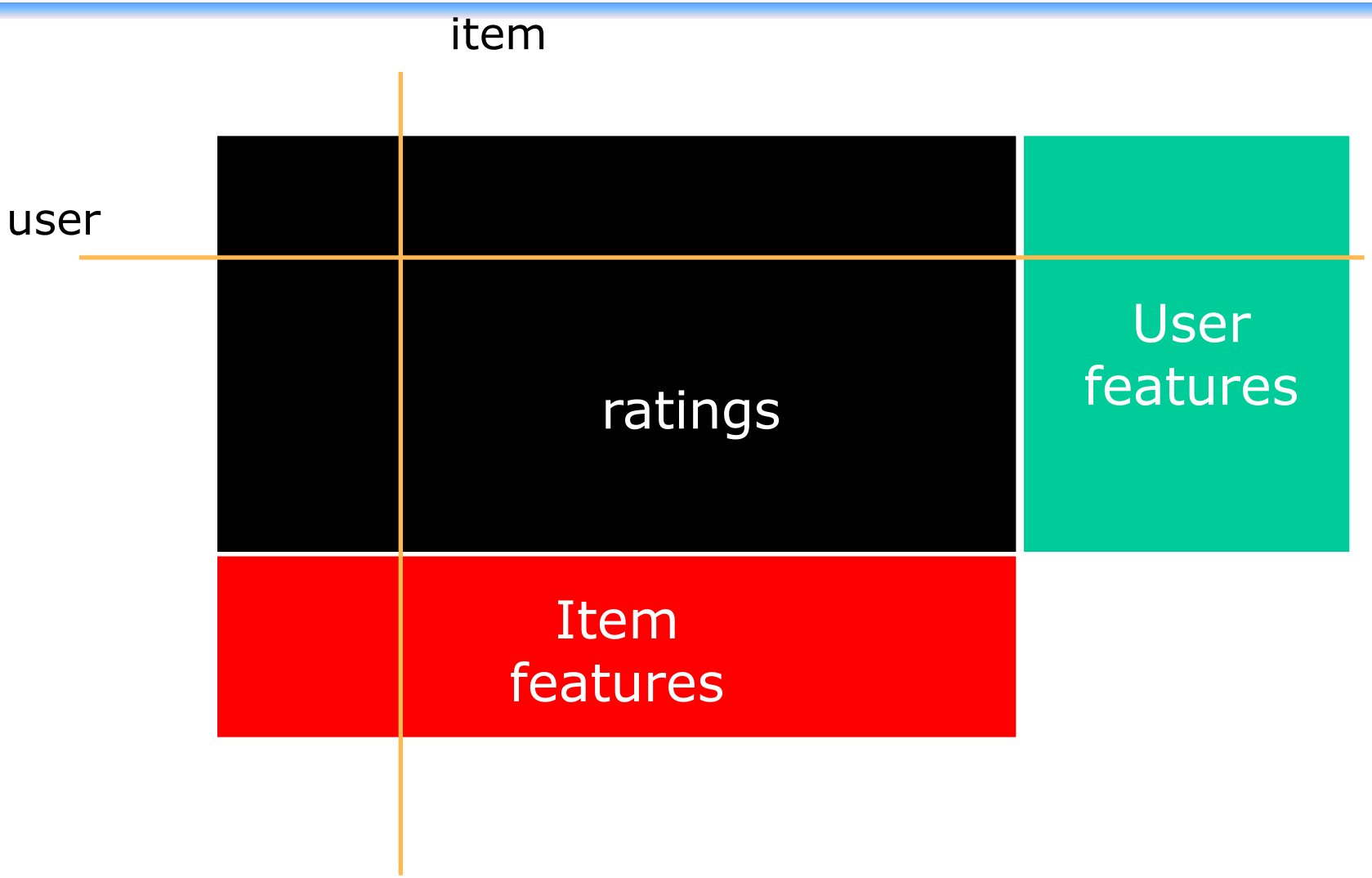
- Contextual Post-Filtering is generally heuristic in nature
  - Basic Idea: Treat the context as an additional constraint
  - Many different approaches are possible
- Example: Filtering Based on Context Similarity
  - Can be represented as a set of features commonly associated with the specified context
  - Adjust the recommendation list by favoring those items that have more of the relevant features
  - Similarity-based approach (but the space of features may be different than the one describing the items)
- Example: Filtering Based on Social/Collaborative Context Representation
  - Mine social features (e.g., annotations, tags, tweets, reviews, etc.) associated with the item and users in a given context *C*
  - Promote items with frequently occurring social features from *C*

Contextual Modeling

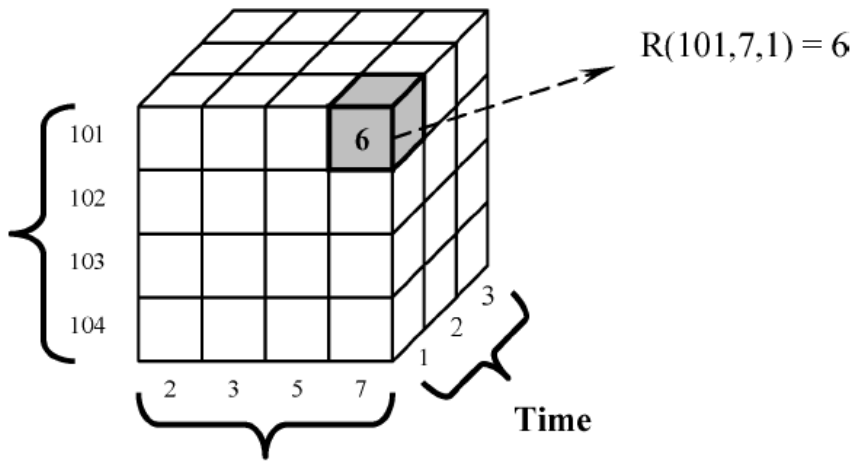
- Using contextual information *directly* in the modeling learning phase
  - Multi-dimensional recommendation models
- Contextual variables are added as dimensions $D_1, \ldots, D_n$ in the feature space in addition to the *Users* and *Items* dimensions
  - $R: U \times I \times D_1 \times \ldots \times D_n \rightarrow Rating$
- Example: Dimensions for movie recommendation application
  - ***User***
  - ***Movie***
  - ***Time*** (weekday, weekend)
  - ***Company*** (alone, partner, family, etc.)
  - ***Place*** (movie theater, at home)



Two Dimensional Model



Multidimensional Model



Id	Name	Age
101	John	25
102	Bob	18
103	Alice	27
104	Mary	21

Item

Id	Name	Cost
2	AB17	250.00
3	AB23	299.95
5	XY70	150.00
7	ZZ55	115.50

Id	Name
1	Weekday
2	Weekend
3	Holiday

[Adomavicius et al., 2005]

Contextual Modeling

- Many different approaches in recent years, both in representational and interactional frameworks
  - Extensions of standard collaborative filtering
    - *CF after Item / user splitting pre-filters*
    - *Differential Context Relaxation*
  - Heuristic distance-based approaches
    - Extend items-item, user-user similarities to contextual dimensions
    - Requires, possibly domain specific, similarity/distance metrics for various contextual dimensions
  - Approaches based on matrix/tensor factorization
    - Model the data as a tensor and apply higher-order factorization techniques (HoSVD, PARAFAC, HyPLSA, etc) to model context in a latent space
    - *Context-Aware Matrix Factorization*
  - *Probabilistic latent variable context models*

Highlighted Approach:
Item / User Splitting

Context-Aware Splitting Approaches

- **Generally based on contextual pre-filtering**
  - May be combined with contextual modeling techniques
- **Goal: produce a 2D data set that incorporates context information associated with preference scores**
  - Advantage: can use a variety of well-known traditional recommendation algorithms in the modeling phase
  - Disadvantages:
    - Determining the variables based on which to split
    - May lead to too much sparsity
- **There are three approaches to splitting:**
  - Item Splitting (Baltrunas et al., 2009, RecSys)
  - User Splitting (Baltrunas et al., 2009, CARS)
  - UI Splitting (Zheng et al., 2013)

Item Splitting and User Splitting

- **Item Splitting**
  - Assumption: the nature of an item, from the user's point of view, may change in different contextual conditions (values of contextual variables)
  - Hence we may consider the item as multiple items – one for each contextual condition
- **User splitting**
  - It may be useful to consider one user as multiple users, if he or she demonstrates significantly different preferences in different contexts
- **Good deal of recent work on these approaches:**
  - L. Baltrunas and F. Ricci. Context-based splitting of item ratings in collaborative filtering. RecSys 2009
  - L. Baltrunas and X. Amatriain. Towards time-dependent recommendation based on implicit feedback. RecSys 2009 Workshop on CARS
  - A. Said, E. W. De Luca, and S. Albayrak. Inferring contextual user profiles – improving recommender performance. RecSys 2011 Workshop on CARS

Example: Item Splitting

User	Movie	Rating	Time	Location	Companion
U1	M1	3	Weekend	Home	Friend
U1	M1	5	Weekend	Theater	Spouse
U1	M1	?	Weekday	Home	Family

Assume Location (Home vs. Theater) is the best split condition

User	Item	Rating
U1	M11	3
U1	M12	5
U1	M11	?

M11: M1 seen at home; M12 = M1 seen not at home

Example: User Splitting

User	Movie	Rating	Time	Location	Companion
U1	M1	3	Weekend	Home	Friend
U1	M1	5	Weekend	Theater	Alone
U1	M1	?	Weekday	Home	Family

Assume Companion (Family vs. Non-Family) is the best split condition

User	Item	Rating
U12	M1	3
U12	M1	5
U11	M1	?

U11: U1 saw the movie with family; U12 = U1 saw the movie alone or with a friend

User-Item (UI) Splitting

- **New approach combining User and Item splitting**
  - The process is simply an application of item splitting followed by user splitting on the resulting output
  - Y. Zheng, B. Mobasher, R. Burke. Splitting approaches for Context-aware Recommendation. (To appear)

- **Using the same conditions as previous example:**

Item Splitting

User	Item	Rating
U1	M11	3
U1	M12	5
U1	M11	?

User Splitting

User	Item	Rating
U12	M1	3
U12	M1	5
U11	M1	?

UI Splitting

User	Item	Rating
U12	M11	3
U12	M12	5
U11	M11	?

Determining the Best Conditions for Splitting

- **Impurity Criteria**
  - Statistical criteria to evaluate whether items are being rated significantly differently under an alternative contextual condition
  - e.g. the location → home vs. not-home

- **Commonly used criteria**
  - t_{mean} (t-test)
  - t_{prop} (z-test)
  - t_{chi} (chi-square test)
  - t_{IG} (Information gain)

- **Thresholds:**
  - P-value is used to judge significance

Splitting Criteria

- **Example: t_{mean}**
  - Uses the two-sample *t* test and computes how significantly different are the means of the ratings in the two rating subsets, when the split *c* (context condition) is used
  - $$t_{mean} = \left| \frac{\mu_{i_c} - \mu_{i_{\bar{c}}}}{\sqrt{s_{i_c}/n_{i_c} + s_{i_{\bar{c}}}/n_{i_{\bar{c}}}}} \right|$$
  - *S* is the rating variance, and *n* is the number of ratings in the given contextual condition, *c* and $\bar{c}$ denote alternative conditions
  - The bigger the *t* value of the test is, the more likely the difference of the means in the two partitions is significant
  - This process is iterated over all contextual conditions

Splitting Criteria Example

- ***t**mean* ; condition: time= weekend and not weekend

$$t_{mean} = \left| \frac{\mu_{i_c} - \mu_{i_{\bar{c}}}}{\sqrt{s_{i_c}/n_{i_c} + s_{i_{\bar{c}}}/n_{i_{\bar{c}}}}} \right|$$

User	Item	Rating	Time	Location	Companion
U1	T1	3	Weekend	Home	Sister
U1	T1	5	Weekend	Cinema	Girlfriend
U2	T1	4	Weekday	Home	Family
U3	T1	2	Weekday	Home	Sister

- **mean1=4 , mean2 =3 , s1 =1 , s2 =1 , n1= 2, n2=2**
- **Impurity criteria *t*mean = (4-3)/1 = 1**
- **P-value of t-test used to determine significance (0.05 as threshold)**

Other Splitting Criteria

- ***t_{prop}***
  - uses the two proportion z test and determines whether there is a significant difference between the proportions of high and low ratings when a contextual condition is used
  - Usually rating = 3 is used as the threshold to divide proportions if the rating scale is 1-5.

$$t_{prop} = \frac{p_{i_c} - p_{i_{\bar{c}}}}{\sqrt{p(1-p)(1/n_{i_c} + 1/n_{i_{\bar{c}}})}}$$

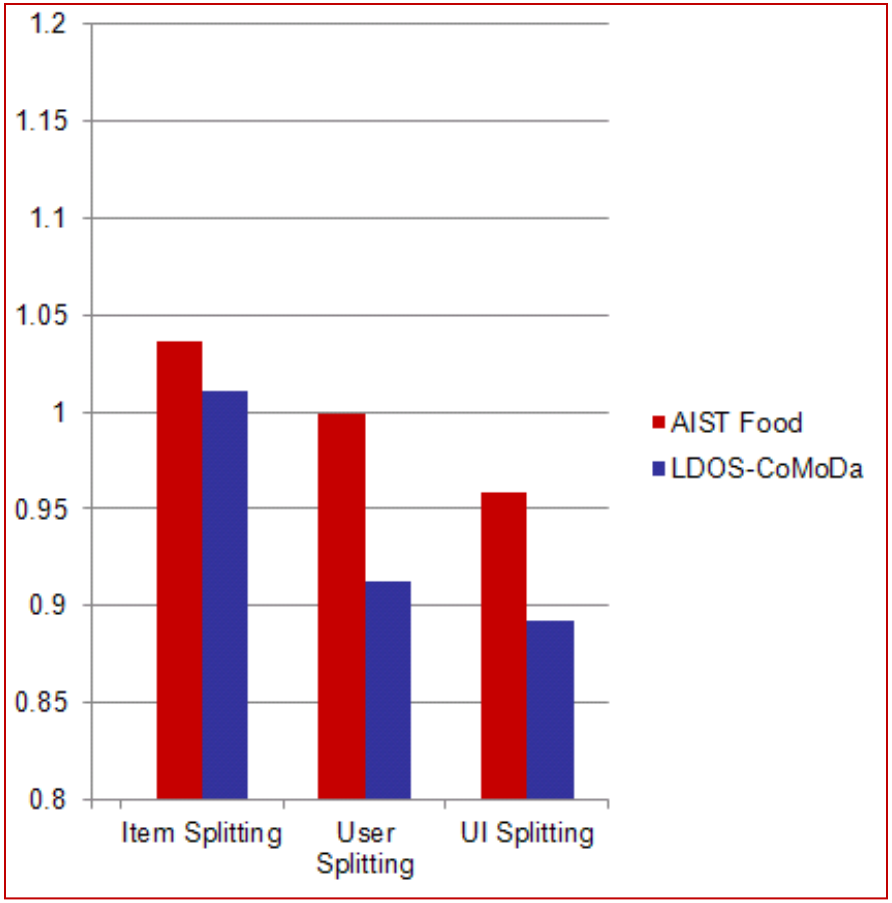
where $p = (p_{i_c} n_{i_c} + p_{i_{\bar{c}}} n_{i_{\bar{c}}})/(n_{i_c} + n_{i_{\bar{c}}})$, $p_{i_c}(p_{i_{\bar{c}}})$ is the proportion of high ratings in $i_c(i_{\bar{c}})$, and $n_{i_c}(n_{i_{\bar{c}}})$ is the number of ratings in $i_c(n_{i_{\bar{c}}})$.

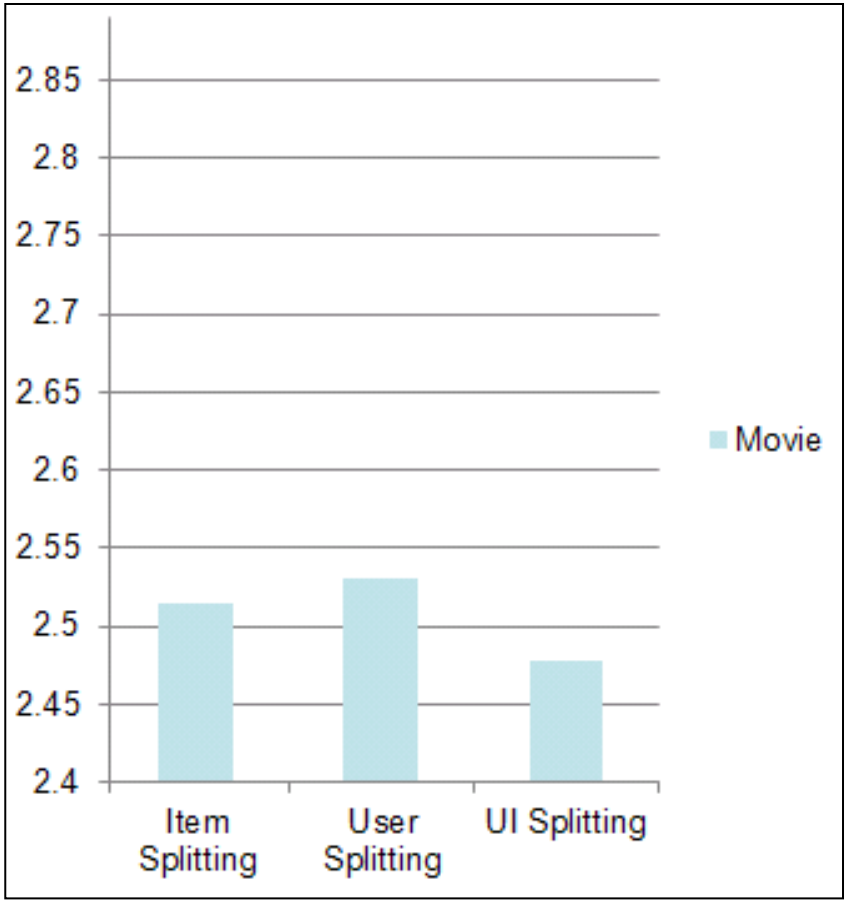
- ***t_{chi}*** **→ similar as t_{prop}, but use chi square test instead of z test**
- ***t_{IG}*** **→ uses high/low proportion, but measures information gain**

Example Results

	Food Data	Movie Data
Ratings	6360	1010
Users	212	69
Items	20	176
Contexts	Real hunger (full/normal/hungry) Virtual hunger	Time (weekend, weekday) Location (home, cinema) Companions (friends, alone, etc)
Contexts-linked Features	User gender food genre, food style, food stuff	User gender, year of the movie
Density	Dense in contexts	Sparse in contexts

Example Results (RMSE)





Results based on splitting followed by Matrix Factorization (discussed next)

Highlighted Approach:

Contextual Modeling using Matrix Factorization

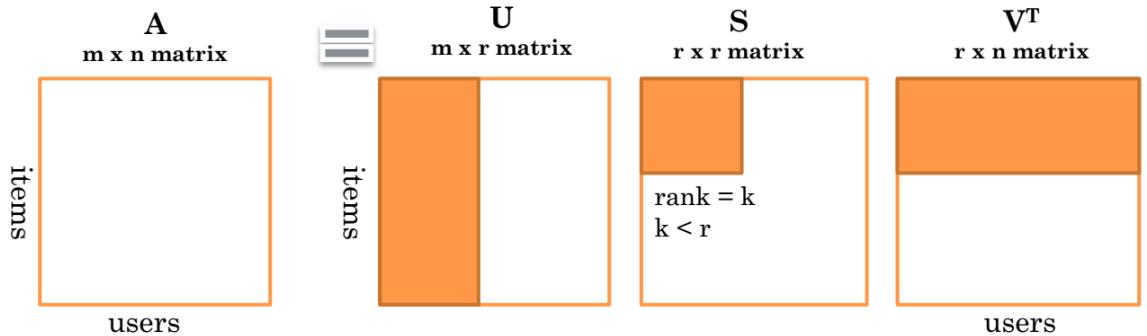
Contextual Modeling via Factorization

- Recent approaches to contextual modeling attempt to fit the data using various regression models
  - Prominent example: Tensor Factorization (TF)
- Tensor Factorization
  - Extends the two-dimensional matrix factorization problem into an multi-dimensional version of the same problem
  - Multi-dimensional matrix is factored into lower-dimensional representation, where the user, the item and each contextual dimension are represented with a lower dimensional feature vector
- Problem: TF can introduce a huge number of model parameters that must be learned using the training data
  - the number of model parameters grow exponentially with the number of contextual factors
  - Simpler models, with less parameters, can often perform as well
  - One Solution: Context-Aware Matrix Factorization

Matrix Factorization - Summary

- **From SVD to MF**

$$A = U \times S \times V^T$$

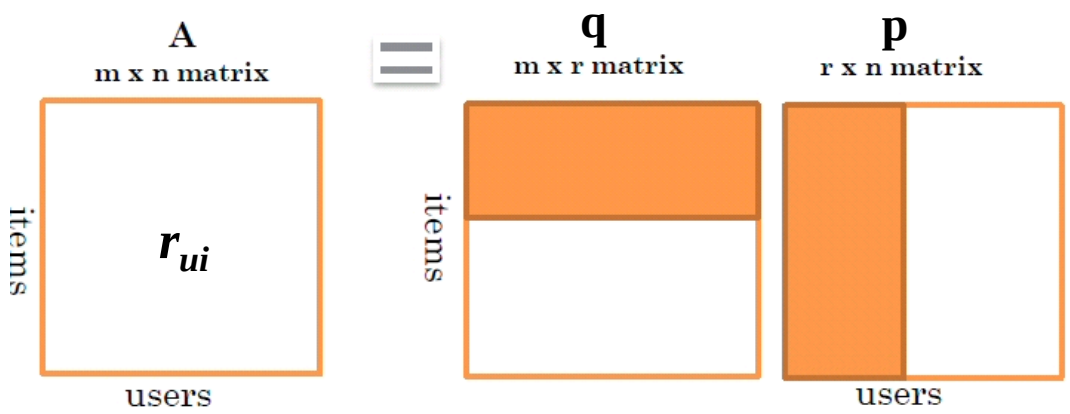


$$A_k = U_k \times S_k \times V_k^T$$

- In SVD, *A* is a rating matrix which can be decomposed into *U*, *S*, V^T
- Standard SVD requires filling out empty entries by zeros which adds noise and may distort the data

Matrix Factorization - Summary

- In MF, A is decomposed into two factor matrices



- $r_{ui} = q_i^{\top} p_u$
- p_u is the user-factor vector, and q_i is the item-factor vector
- Typically, user and item vectors are initialized and then learned from the non-zero entries in the rating matrix
- The optimization is usually performed using Stochastic Gradient Descent (SGD) or Alternating Least Squares (ALS)

Matrix Factorization - Summary

- **Learning the factor vectors**
  - Minimize errors on known ratings

$$\min_{q^*.p^*} \sum_{(u,i)\in k} (r_{ui} - x_{ui})^2$$

Predicted rating by user *u* on item *i*

Actual rating by user *u* on item *i*

Minimizing Cost Function
(Least Squares Problem)

  - However, this is typically done after standardization by removing the global mean from the rating data:

$$\min_{q^*.p^*} \sum_{(u,i)\in k} (r_{ui} - \mu - q_i^T p_u)^2$$

Matrix Factorization - Summary

- **Learning the factor vectors**
  - Adding user and item bias to the factor model

$$\min_{q^*.p^*} \sum_{(u,i)\in k} (r_{ui} - \mu - b_i - b_u - q_i^T p_u)^2$$

  - Regularization is used to prevent over-fitting:

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

r_{ui} : actual rating of user u on item i
μ : training rating average
b_u : user u user bias
b_i : item i item bias
q_i : latent factor vector of item i
p_u : later factor vector of user u
λ : regularization Parameters

Matrix Factorization - Summary

- **Standard MF vs. BiasMF**
  - Standard MF: Predict(u, i) = $q_i^{\mathsf{T}} p_u$
  - BiasMF: Predict(u, i) = $q_i^{\mathsf{T}} p_u + \mu + b_u + b_i$
  - b_u and b_i are user bias and item bias respectively, μ is the global mean rating

- **The reason why to add bias**
  - Standard MF tries to capture the interactions between users and items that produce the different rating values
  - However, much of the observed variation in rating values is due to effects associated with either user or items, known as biases, independent of any interactions.
  - Adding user and item biases can help explain the rating values better than only relying on the interaction of the form $q_i^{\mathsf{T}} p_u$.

Basic Steps in MF-Based Models

- Construct User-Item Matrix (sparse data structure)
- Define factorization model - Cost function
- Take out global mean
- Decide what parameters in the model
  - Item or user bias, preference factor, time dependent factors, context, etc.
- Minimizing cost function - model fitting
  - Stochastic gradient descent
  - Alternating least squares
- Assemble the predictions
- Evaluate predictions (RMSE, MAE etc..)
- Continue to tune the model

Context-Aware Matrix Factorization

- **Recall the difference between MF and BiasMF:**
  - Standard MF: Predict(u, i) = $q_i^{\mathsf{T}} p_u$
  - BiasMF: Predict(u, i) = $q_i^{\mathsf{T}} p_u + \mu + b_u + b_i$
  - b_u and b_i are user bias and item bias respectively, μ is the global mean rating

- **Context-Aware MF**
  - CAMF replaces the simple item bias, b_i, by the interaction between item and contextual conditions
    - Predicted rating will now be also a function of contextual conditions, c_1, …, c_k giving rise to a particular context
    - The item bias is modeled by how that item is rated in different contexts
    - i.e., sum of biases for the given item across all contextual conditions, c_1, …, c_k
    - Different levels of granularity in aggregating item biases can lead to different variants of CAMF

Context-Aware MF (CAMF)

(Baltrunas, Ludwig, Ricci, RecSys 2011)

- There are three models derived based on the interaction of context and items

- **CAMF-C assumes that each contextual condition has a global influence on the ratings - independently from the item** — Global

- **CAMF-CI introduces one parameter per each contextual condition and item pair** — Item

- **CAMF-CC introduces one model parameter for each contextual condition and item category (music genre).** — Genre

Slide from F. Ricci
UMAP 2012

Context-Aware MF (CAMF)

(Baltrunas, Ludwig, Ricci, RecSys 2011)

$$\hat{r}_{uic_1 \ldots c_k} = \vec{v}_u \cdot \vec{q}_i + \bar{\imath} + b_u + \sum_{j=1}^{k} B_{ijc_j}$$

- v_u and q_i are d dimensional real valued vectors representing the user u and the item i
- $\bar{\imath}$ is the average of the item i ratings
- b_u is a baseline parameter for user u
- CAMF-C: only one parameter for each context condition,

$$\text{i.e., } B_{ijc_j} = B_{fjc_j} \text{ for every item } i \text{ and } f.$$

- CAMF-CI: In B_{ijc_j}, i denotes item, j is the jth context dimension, and c_j is the condition in this dimension
- CAMF-CC: it considers the item category; that is, if item i and f fall into the same category, then $B_{ijc_j} = B_{fjc_j}$

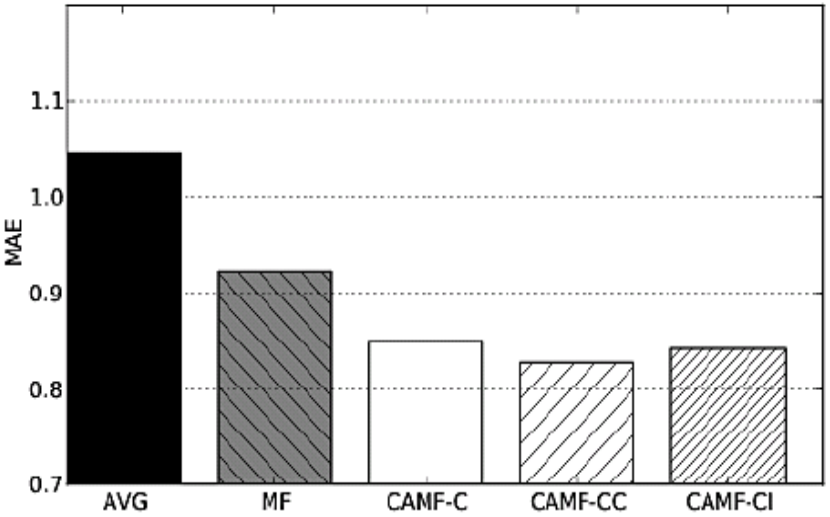
CAMF Example

User	Item	Rating	Time	Location	Companion
U1	T1	3	Weekend	Home	Alone
U1	T2	5	Weekend	Cinema	Friend
U2	T2	4	Weekday	Home	Family
U3	T1	2	Weekday	Home	Alone

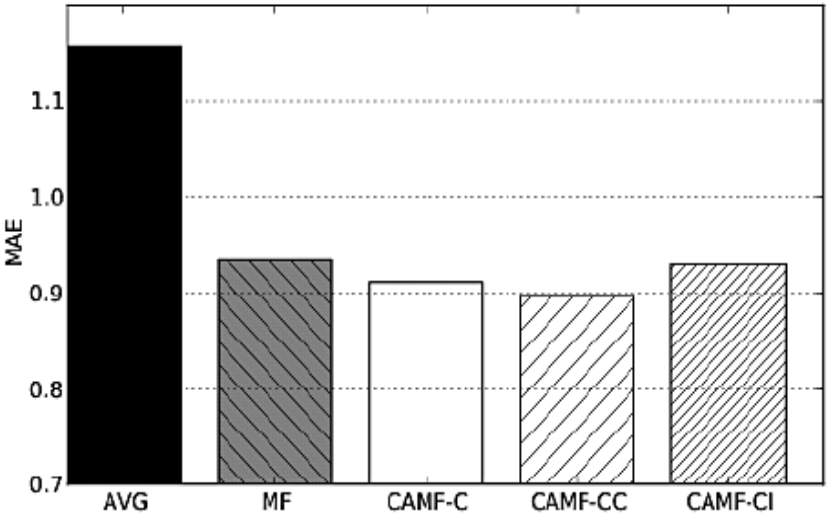
- CAMF-C
  - only one parameter for each context condition, e.g. time as a context dimension has two conditions: weekend and weekday.
  - the item bias is equal to the bias in each context condition
  - e.g., the interaction <T1, weekend> and <T2, weekend> is the same bias value
- CAMF-CI
  - the finest-grained interaction modeling
  - <T1, weekend> and <T2, weekend> may be different values and the value is learned in the matrix factorization process
- CAMF-CC
  - considers the item category
  - if items *i* and *f* fall into the same category, then $B_{ijc_j} = B_{fjc_j}$

CAMF – Example Results

(Baltrunas, Ludwig, Ricci, RecSys 2011)



(a) Tourism



(b) Music

Optimization in CAMF

- **Similar to the BiasMF Optimization based on SGD**

$$\hat{r}_{uic_1\ldots c_k} = \vec{v}_u \cdot \vec{q}_i + \bar{\imath} + b_u + \sum_{j=1}^{k} B_{ijc_j}$$

- **Optimization:**

$$\min_{v_*,q_*,b_*} \sum_{r\in R} \left[\left(r_{uic_1\ldots c_k} - v_u q_i^\top - \bar{\imath} - b_u - \sum_{j=1}^{k} b_{ic_j} \right)^2 + \lambda(b_u^2 + \|v_u\|^2 + \|q_i\|^2 + \sum_{j=1}^{k} b_{ic_j}^2) \right]$$

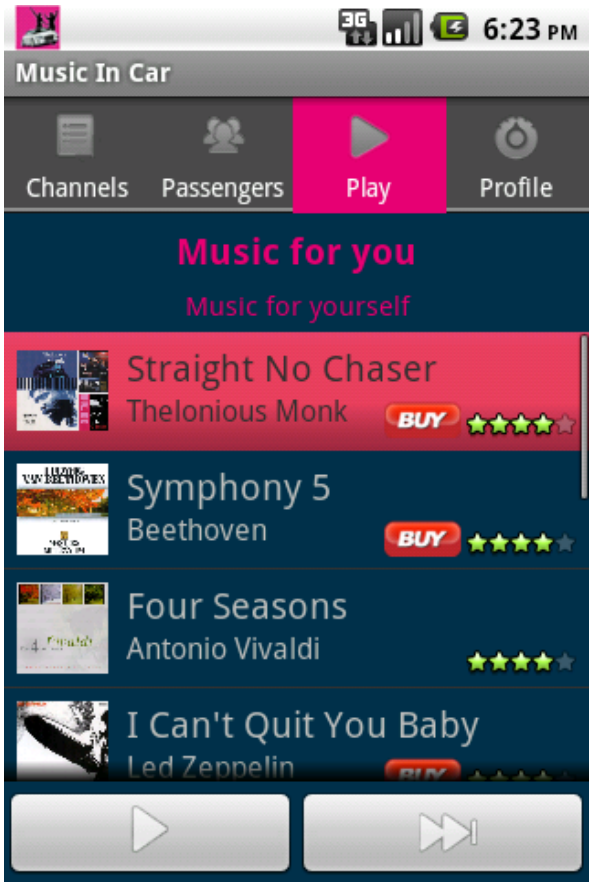
- **Parameters updates as follows:**
  - $b_u \leftarrow b_u + \gamma_{b_u}(err - \lambda b_u)$
  - $b_{ic_j} \leftarrow b_{ic_j} + \gamma_{b_{ic_j}}(err - \lambda b_{ic_j}), \forall c_j \neq 0, j = 1, \ldots, k$
  - $v_u \leftarrow v_u + \gamma_{v_u}(err \cdot q_i - \lambda v_u)$
  - $q_i \leftarrow q_i + \gamma_{q_i}(err \cdot v_u - \lambda q_i)$

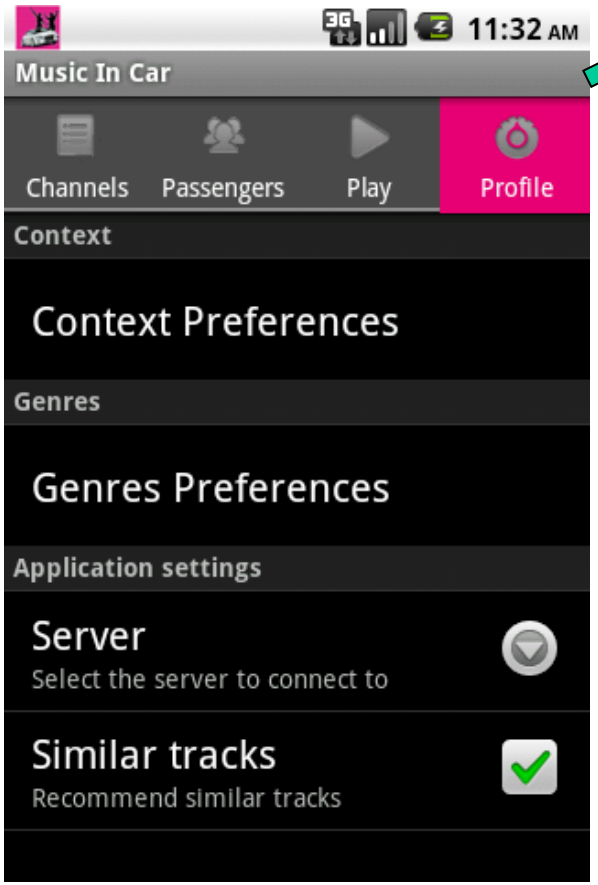
Example Implementation - InCarMusic

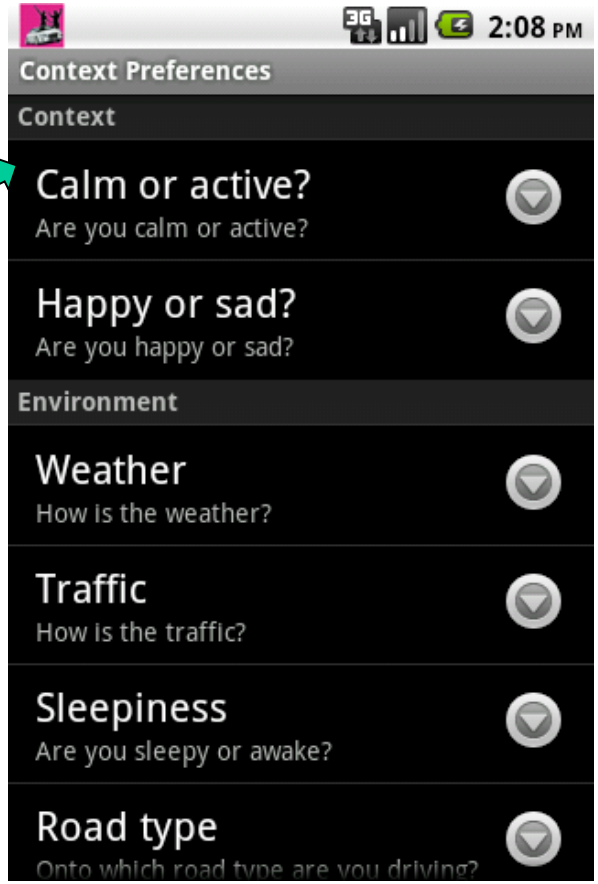
- Detecting relevant contextual factors – based on user survey (expected utility)
- Acquiring ratings in context
- Generating rating predictions with context-aware matrix factorization

Slide from F. Ricci
UMAP 2012

Android Application







[Baltrunas et al., 2011]

Methodological Approach

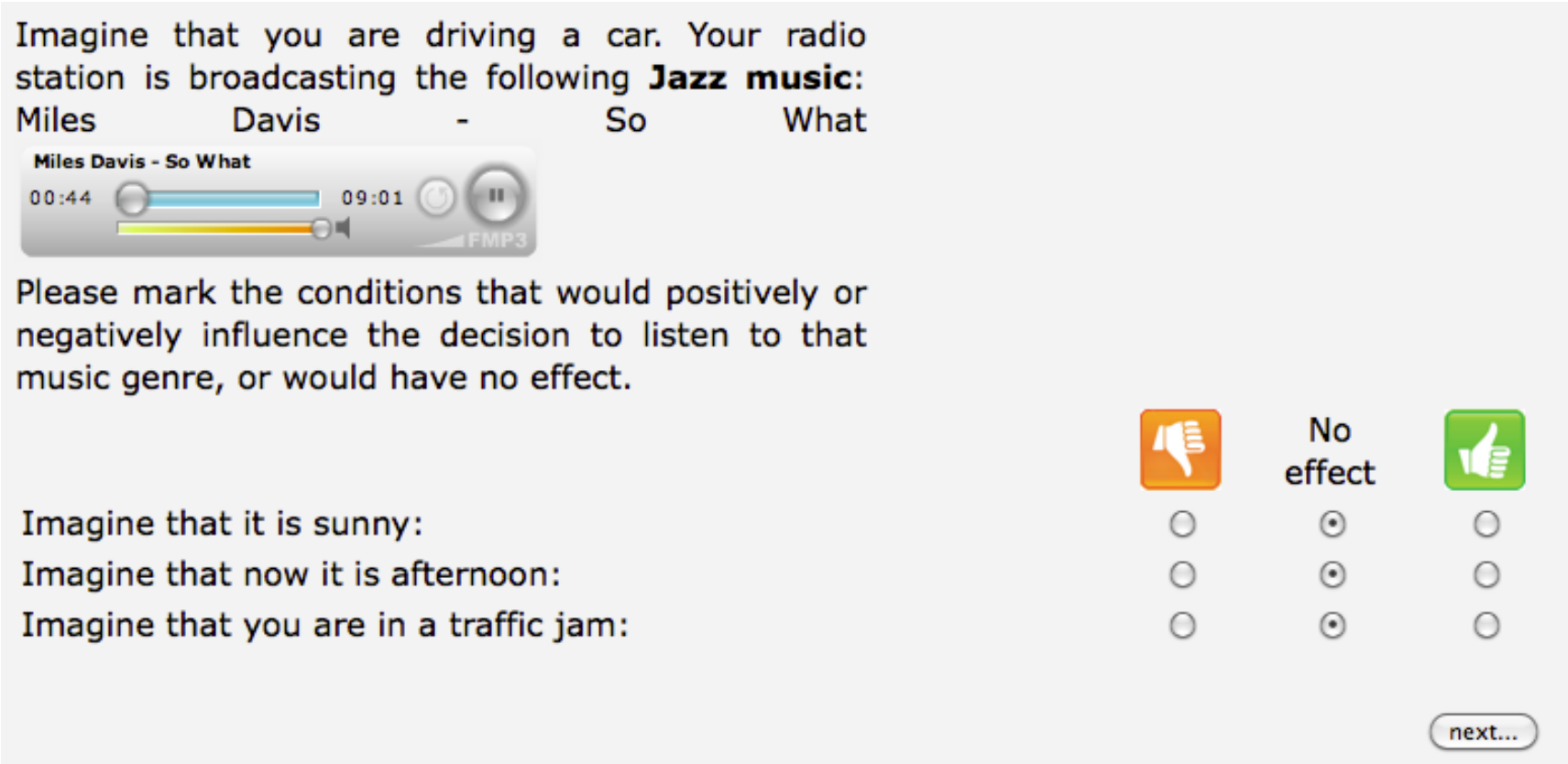
1. **Identifying potentially relevant contextual factors**
   - *Heuristics, consumer behavior literature*
2. **Ranking contextual factors**
   - *Based on subjective evaluations (what if scenario)*
3. **Measuring the dependency of the ratings from the contextual conditions and the users**
   - *Users rate items in imagined contexts*
4. **Modeling the rating dependency from context**
   - *Extended matrix factorization model*
5. **Learning the prediction model**
   - *Stochastic gradient descent*
6. **Delivering context-aware rating predictions and item recommendation**

Contextual Factors

- driving style (DS): relaxed driving, sport driving
- road type(RT): city, highway, serpentine
- landscape (L): coast line, country side, mountains/hills, urban
- sleepiness (S): awake, sleepy
- traffic conditions (TC): free road, many cars, traffic jam
- mood (M): active, happy, lazy, sad
- weather (W): cloudy, snowing, sunny, rainy
- natural phenomena (NP): day time, morning, night, afternoon

Slide from F. Ricci, UMAP 2012

Determine Context Relevance



- **Web based application**
- **2436 evaluations from 59 users**

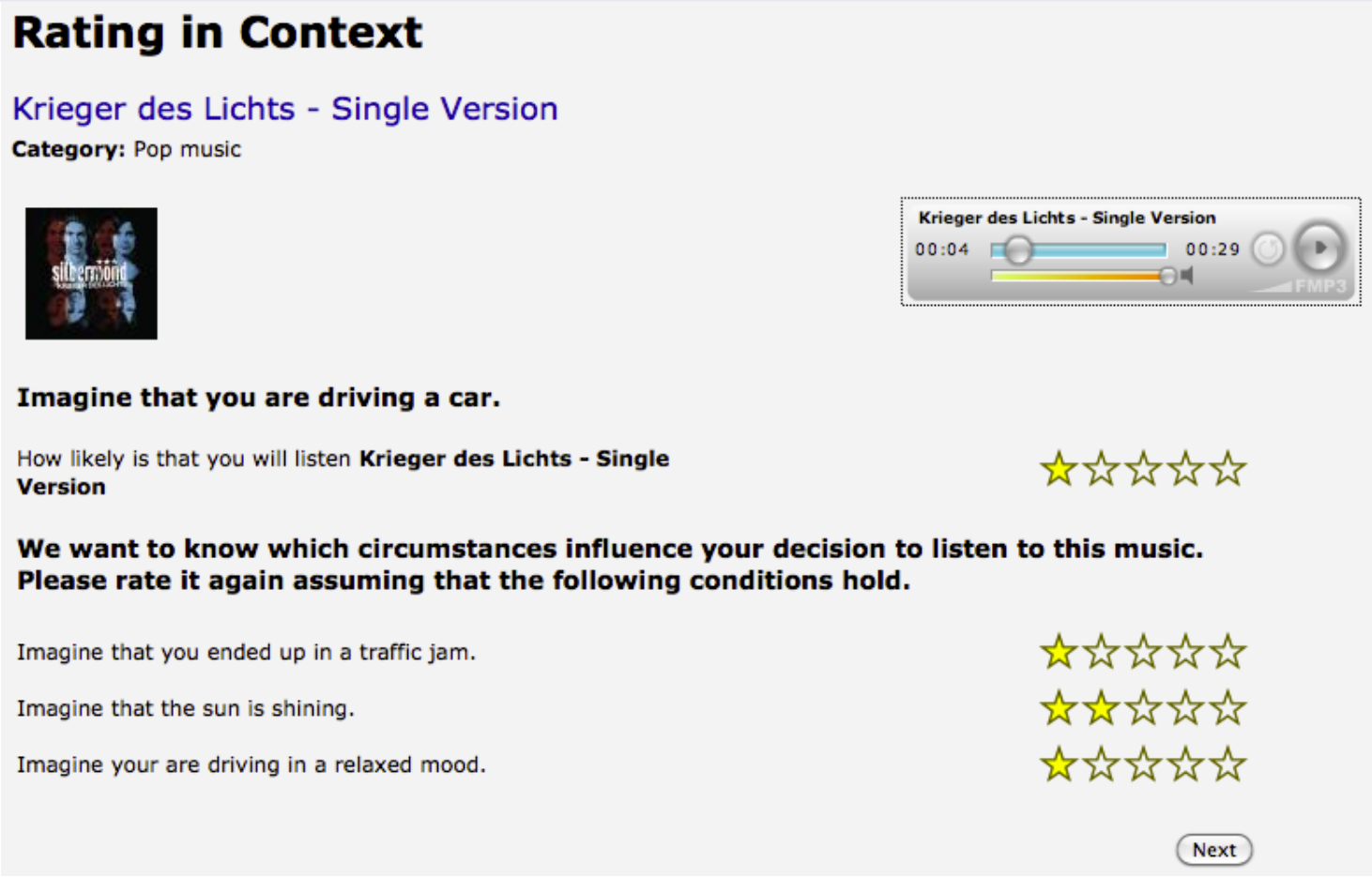
Slide from F. Ricci, UMAP 2012

User Study Results

Blues	MI	Classical	MI	Country	MI	Disco	MI	Hip Hop	MI
driving style	0.32	driving style	0.77	sleepiness	0.47	mood	0.18	traffic conditions	0.19
road type	0.22	sleepiness	0.21	driving style	0.36	weather	0.17	mood	0.15
sleepiness	0.14	weather	0.09	weather	0.19	sleepiness	0.15	sleepiness	0.11
traffic conditions	0.12	natural phenomena	0.09	mood	0.13	traffic conditions	0.13	natural phenomena	0.11
natural phenomena	0.11	mood	0.09	landscape	0.11	driving style	0.10	weather	0.07
landscape	0.11	landscape	0.06	road type	0.11	road type	0.06	landscape	0.05
weather	0.09	road type	0.02	traffic conditions	0.10	natural phenomena	0.05	driving style	0.05
mood	0.06	traffic conditions	0.02	natural phenomena	0.04	landscape	0.05	road type	0.01

- **Normalized Mutual Information of the contextual condition on the Influence variable (1/0/-1)**
- **The higher the MI the larger the influence**

In Context Ratings



- Contextual conditions are sampled with probability proportional to the MI of the contextual factor and music genre

Slide from F. Ricci, UMAP 2012

Influence on the Average Rating

Condition	ratings	no-context MCN	context MCY	Influence
- *Driving style*				
relaxed driving	167	2.38	2.27	↓
sport driving	165	2.46	2.34	↓
- *Landscape*				
coast line	119	2.42	2.48	↑
country side	118	2.31	2.03	↓
mountains/hills	132	2.53	2.34	↓
urban	113	2.45	2.14	↓

In the No-Context condition users are evaluating rating in the default context
The default context is the context where consuming the items makes sense – best context

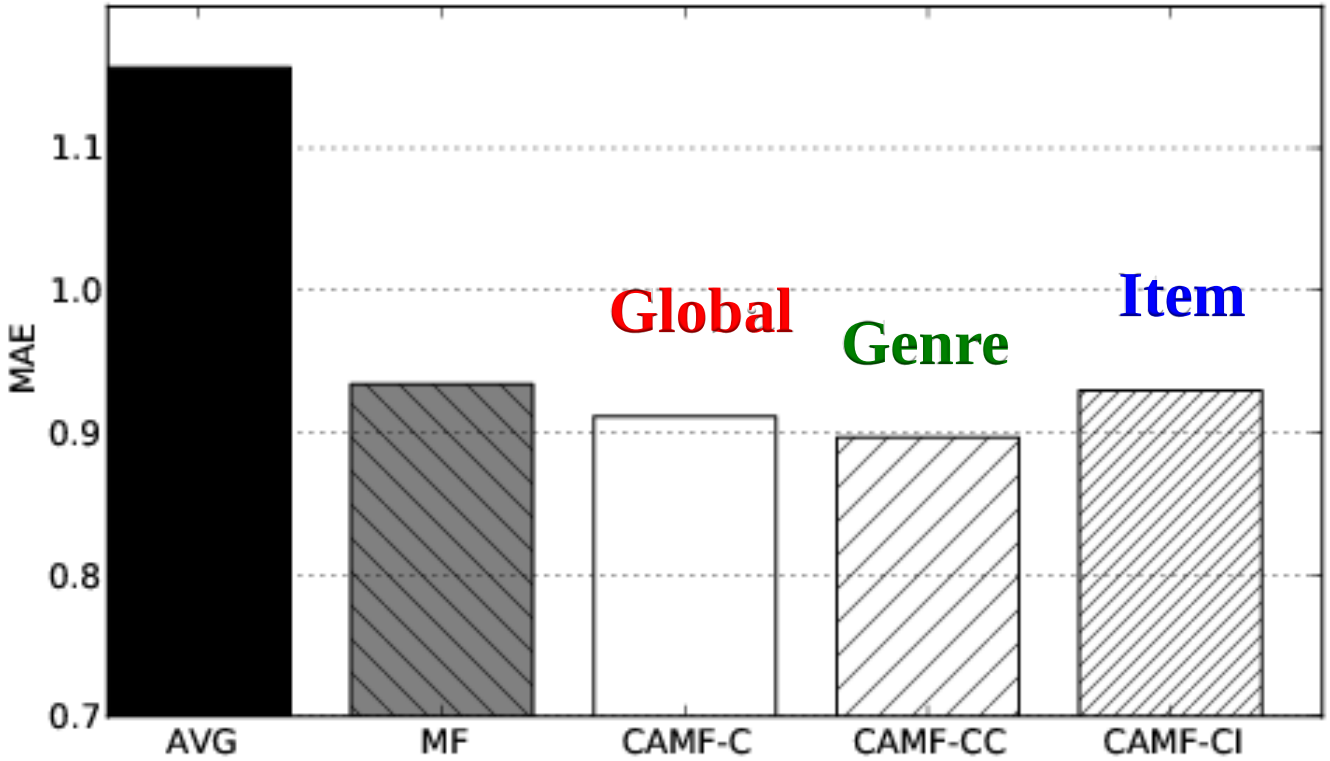
Predictive Model

$$\hat{r}_{uic_1 \ldots c_k} = \boldsymbol{v}_u \cdot \boldsymbol{q}_i + \bar{\imath} + b_u + \sum_{j=1}^{k} b_{g_i j c_j}$$

- ***v_u*** **and** ***q_i*** **are d dimensional real valued vectors representing the user *u* and the item *i***
- $\bar{\imath}$ **is the average of the item *i* ratings**
- ***b_u*** **is a baseline parameter for user u**
- ***b_{gjc}*** **is the baseline of the contextual condition *c_j* (factor *j*) and genre *g_i* of item *i***
  - *assume that context influences uniformly all the tracks with a given genre*
- **If a contextual factor is unknown, i.e., c_j = 0, then the corresponding baseline b_{gjc} is set to 0.**

Predicting Expected Utility in Context

[Baltrunas et al., 2011]



Item average

InCarMusic Approach Limitations

- **Framework is personalized**
  - Using the extended MF model
- **But, it requires users' ratings in different contextual conditions to be trained**
- **Alternative Approach**
  - Use relationships between items and the user's context using other information sources
    - Semantic information associated with items and contexts
    - Social cues relevant to items and users (e.g., tags)
  - We'll discuss some examples later

Some Additional References for MF Based Approaches

- Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model. KDD 2008
- Yehuda Koren, Collaborative filtering with temporal dynamics. KDD 2009
- A. Karatzoglou , X. Amatriain , L. Baltrunas , N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. RecSys 2010
- L. Baltrunas, B. Ludwig, and F. Ricci. Matrix factorization techniques for context aware recommendation. RecSys 2011
- L. Baltrunas, M. Kaminskas, B. Ludwig, O. Moling, F. Ricci, A. Aydin, K.-H. Luke, and R. Schwaiger. InCarMusic: Context-aware music recommendations in a car. ECWeb 2011
- Rendle, Gantner, Freudenthaler, Schmidt-Thieme: Fast context-aware recommendations with factorization machines. SIGIR 2011

Highlighted Approach: Differential Context Modeling

Differential Context Modeling

- **Basic assumption of many Context-aware RS**
  - It is better to use preferences within the same context to make predictions relevant to that context.
- **Problem: sparsity**
  - Especially when multiple contextual variables define a contextual condition

User	Movie	Time	Location	Companion	Rating
U1	*Titanic*	Weekend	Home	Family	4
U2	*Titanic*	Weekday	Home	Family	5
U3	*Titanic*	Weekday	Cinema	Friend	4
U1	*Titanic*	Weekday	Home	Friend	?

- Are there rating profiles in the context <Weekday, Home, Friend>?

Possible Solutions

User	Movie	Time	Location	Companion	Rating
U1	*Titanic*	Weekend	Home	Family	4
U2	*Titanic*	Weekday	Home	Family	5
U3	*Titanic*	Weekday	Cinema	Friend	4
U1	*Titanic*	Weekday	Home	Friend	?

- Context Matching ➔ only the exact context <Weekday, Home, Friend>?
- Context Selection ➔ use only the most relevant contexts
- Context Relaxation ➔ relax set of constraints (dimensions) defining the context, e.g. use only time
- Context Weighting ➔ use all dimensions, but weight them according to co-occurrence relationships among contexts

Differential Context Modeling

- **There are two parts in DCM**
  - "Differential" Part
    - Separate one algorithm into different functional components;
    - Apply differential context constraints to each component;
    - Maximize the global contextual effects by algorithm components;
  - "Modeling" Part
    - It can be performed by context relaxation or context weighting
    - Differential Context Relaxation (DCR)
    - Differential Context Weighting (DCW)

DCR – Algorithm Decomposition

- **Example: User-based Collaborative Filtering (UBCF)**

	Pirates of the Caribbean 4	Kung Fu Panda 2	Harry Potter 6	Harry Potter 7
U1	4	4	1	2
U2	3	4	2	1
U3	2	2	4	4
U4	4	4	1	?

Standard Process in UBCF:

1). Find neighbors based on user-user similarity
2). Aggregate neighbors' contributions
3). Make final predictions

DCR – Algorithm Decomposition

- **User-based Collaborative Filtering Predictions:**

1.Neighbor Selection

2.Neighbor contribution

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u \in N} (r_{u,i} - \bar{r}_u) \times sim(a,u)}{\sum_{u \in N} sim(a,u)}$$

3.User baseline

4.User Similarity

Standard UBCF → all components are assumed to range over the same set of contextual conditions

UBCF with DCR → find the appropriate contextual condition for each component

DCR – Context Relaxation Example

User	Movie	Time	Location	Companion	Rating
U1	*Titanic*	Weekend	Home	Family	4
U2	*Titanic*	Weekday	Home	Family	5
U3	*Titanic*	Weekday	Cinema	Friend	4
U1	*Titanic*	Weekday	Home	Friend	?

- **Context Relaxation:**
  - Use {Time, Location, Companion} ➔ 0 records matched!
  - Use {Time, Location} ➔ 1 record matched!
  - Use {Time} ➔ 2 records matched!
- In DCR, we choose appropriate context relaxation for each component

DCR – Context Relaxation Example

1.Neighbor Selection

2.Neighbor contribution

$$P_{a,i,c} = \bar{r}_{a,C_3} + \frac{\sum_{u \in N_{C_1}} (r_{u,i,C_2} - \bar{r}_{u,C_2}) \times sim_c(a,u,C_4)}{\sum_{u \in N_{C_1}} sim_c(a,u,C_4)}$$

3.User baseline

4.User Similarity

- c is the original context, e.g. <Weekday, Home, Friend>
- C1, C2, C3, C4 are the relaxed contexts
  - They could be the full c or partial constraints from c
- The selection is modeled by a binary vector.
  - E.g. <1, 0, 0> denotes we just selected the first context dimension
- The optimal relaxations are found through optimization

DCR – Drawbacks

1.Neighbor Selection

2.Neighbor contribution

$$P_{a,i,c} = \bar{r}_{a,C_3} + \frac{\sum_{u \in N_{C_1}} (r_{u,i,C_2} - \bar{r}_{u,C_2}) \times sim_c(a,u,C_4)}{\sum_{u \in N_{C_1}} sim_c(a,u,C_4)}$$

3.User baseline

4.User Similarity

- Context relaxation may still be too strict when data is too sparse
- Components are dependent
  - E.g., neighbor contribution is dependent on neighbor selection
  - C1: Location = Cinema, is not guaranteed, but, neighbor has ratings under contexts C2: Time = Weekend
- Need a finer-grained solution ➔ Differential Context Weighting

Differential Context Weighting

User	Movie	Time	Location	Companion	Rating
U1	*Titanic*	Weekend	Home	Friend	4
U2	*Titanic*	Weekday	Home	Friend	5
U3	*Titanic*	Weekday	Cinema	Family	4
U1	*Titanic*	Weekday	Home	Family	?

- **Goal: Use all dimensions, but weight them based on the similarity of contexts**
  - Assumption: the more similar two contexts are, the more similar the ratings will be in those contexts
  - Similarity can be measured by Weighted Jaccard similarity $J(c,d,\sigma) = \frac{\sum_{f \in c \cap d} \sigma_f}{\sum_{f \in c \cup d} \sigma_f}$
  - Example:
    - c and d are two contexts (two red regions in the Table)
    - σ is the weighting vector <w1, w2, w3> for three dimensions.
    - Assume they are equal weights, w1 = w2 = w3 = 1
    - J(c, d, σ) = # of matched dimensions / # of all dimensions = 2/3

Differential Context Weighting

1.Neighbor Selection

2.Neighbor contribution

$$P_{a,i,\sigma} = \bar{\rho}(a,\sigma_3,\epsilon_3) + \frac{\sum_{u \in N_{a,\sigma_1,\epsilon_1}} (\rho(u,i,\sigma_2,\epsilon_2) - \bar{\rho}(u,\sigma_2,\epsilon_2)) \times sim_w(a,u,\sigma_4,\epsilon_4)}{\sum_{u \in N_{a,\sigma_1,\epsilon_1}} sim_w(a,u,\sigma_4,\epsilon_4)}$$

3.User baseline

4.User Similarity

"Differential" part ➔ Components are all the same

"Context Weighting" part:

σ is the weighting vector, and ϵ is a threshold for the similarity of contexts

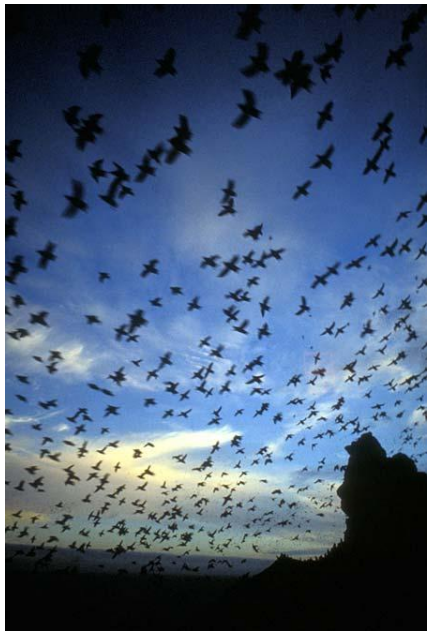
i.e., only records with similar enough contexts (≥ ϵ) can be included in the calculations

Need to find optimal weighting vectors

Particle Swarm Optimization (PSO)

- **PSO is derived from swarm intelligence**
  - achieve a goal by collaborative work via a swarm

Fish



Birds

Bees

Particle Swarm Optimization (PSO)

Swarm = a group of birds
Particle = each bird ≈ *each run in algorithm*
Vector = bird's position in the space ≈ *Vectors for DCR/DCW*
Goal = the location of pizza ≈ *RMSE*

So, how to find goal by swam?

1.Looking for the pizza
Assume a machine can tell the distance

2.Each iteration is an attempt or move

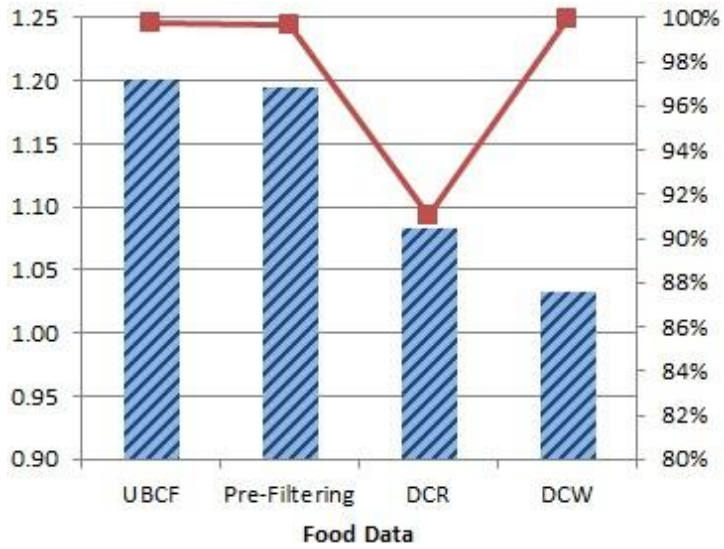
3.Cognitive learning from particle itself
Am I closer to the pizza comparing with my "best" locations in previous steps?

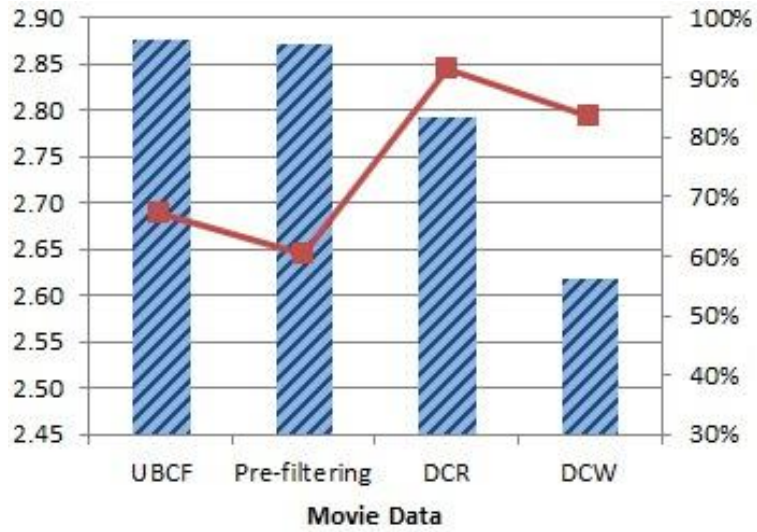
4.Social Learning from the swarm
Hey, my distance is 1 mile. It is the closest! Follow me!! Then other birds move towards here

DCR – Feature selection – Modeled by binary vectors – Binary PSO
DCW – Feature weighting – Modeled by real-number vectors – PSO

How it works? DCR and Binary PSO example:
Assume there are 3 components and 4 contextual dimensions
Thus there are 3 binary vectors for each component respectively
We integrate the vectors into a single one, the vector size is 3*4 = 12
This single vector is the particle's position vector in PSO process

Predictive Performance





Blue bars are RMSE values, ***Red lines*** are coverage curves

Findings:

1) DCW works better than DCR and two baselines
2) t-test shows DCW works better significantly in movie data
3) but DCR was not significant over two baselines
4) DCW can further alleviate sparsity of contexts
5) DCW offers better coverage over baselines!

Performance of PSO Optimizer

	Food Data		Movie Data	
	Iteration	Running Time	Iteration	Running Time
DCR via BPSO	11	66.9	18	4.9
DCW via PSO	13	248.4	66	23.2

- Running time in seconds
- Factors influencing the running performances:
  - More particles, quicker convergence but probably more cost
  - # of contextual variables: more contexts, slower
  - Density of the data set: denser, more calculations
- Typically DCW costs more than DCR, because it uses all contextual dimensions and the calculation for similarity of contexts
  - especially time consuming for dense data, like the Food data

DCR in Item-based CF (IBCF)

- **Algorithm Decomposition and Context Relaxation (DCR)**

$$P_{a,i} = \frac{\sum_{j \in N_i} r_{a,j} \times sim(i,j)}{\sum_{j \in N_i} sim(i,j)} \quad (1)$$

1 2 3

$$sim(i,j) = \frac{\sum_{u \in N} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in N} (r_{u,i} - \bar{r}_u)^2 \sum_{u \in N} (r_{u,j} - \bar{r}_u)^2}} \quad (2)$$

Put IBCF on DCR

$$P_{a,i,c} = \frac{\sum_{j \in N_{C_1}} r_{a,j,C_2} \times sim_c(i,j,C_3)}{\sum_{j \in N_{C_1}} sim_c(i,j,C_3)} \quad (5)$$

$$sim_c(i,j,C_3) = \frac{\sum_{u \in N} (r_{u,i,C_3} - \bar{r}_{u,C_3})(r_{u,j,C_3} - \bar{r}_{u,C_3})}{\sqrt{\sum_{u \in N} (r_{u,i,C_3} - \bar{r}_{u,C_3})^2 \sum_{u \in N} (r_{u,j,C_3} - \bar{r}_{u,C_3})^2}} \quad (6)$$

DCW in Item-Based CF (IBCF)

- **Algorithm Decomposition and Context Weighting (DCW)**

Put IBCF on DCW

$$P_{a,i,\sigma} = \frac{\sum_{j \in N_{i,\sigma_1,\epsilon_1}} r_{a,j,\sigma_2,\epsilon_2} \times sim_w(i,j,\sigma_3,\epsilon_3)}{\sum_{j \in N_{i,\sigma_1,\epsilon_1}} sim_w(i,j,\sigma_3,\epsilon_3)}$$

σ is the weighting vector, and ϵ is a threshold for the similarity of contexts, i.e., only records that are similar enough (≥ ϵ) can be included in the predictions

Predictive Performances (RMSE)

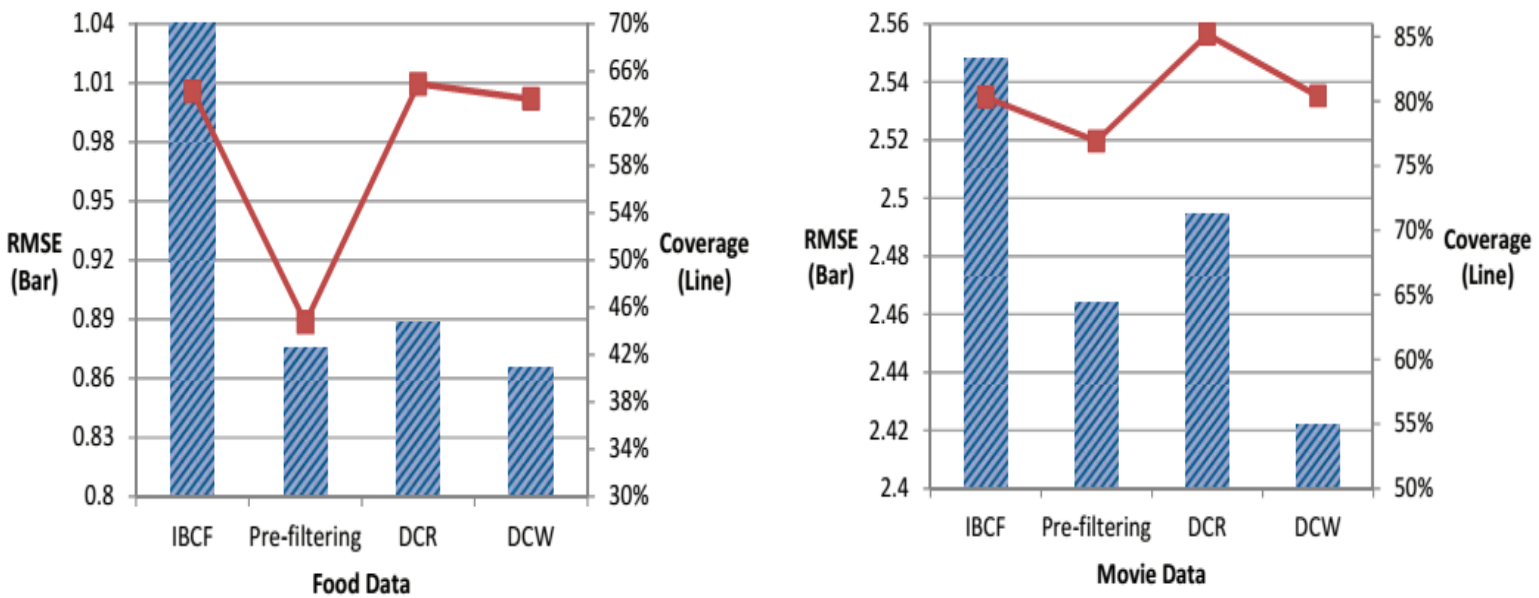


Figure 1: Application of DCR and DCW to IBCF

DCW works the best, where pre-filtering is better than DCR but very low coverage!

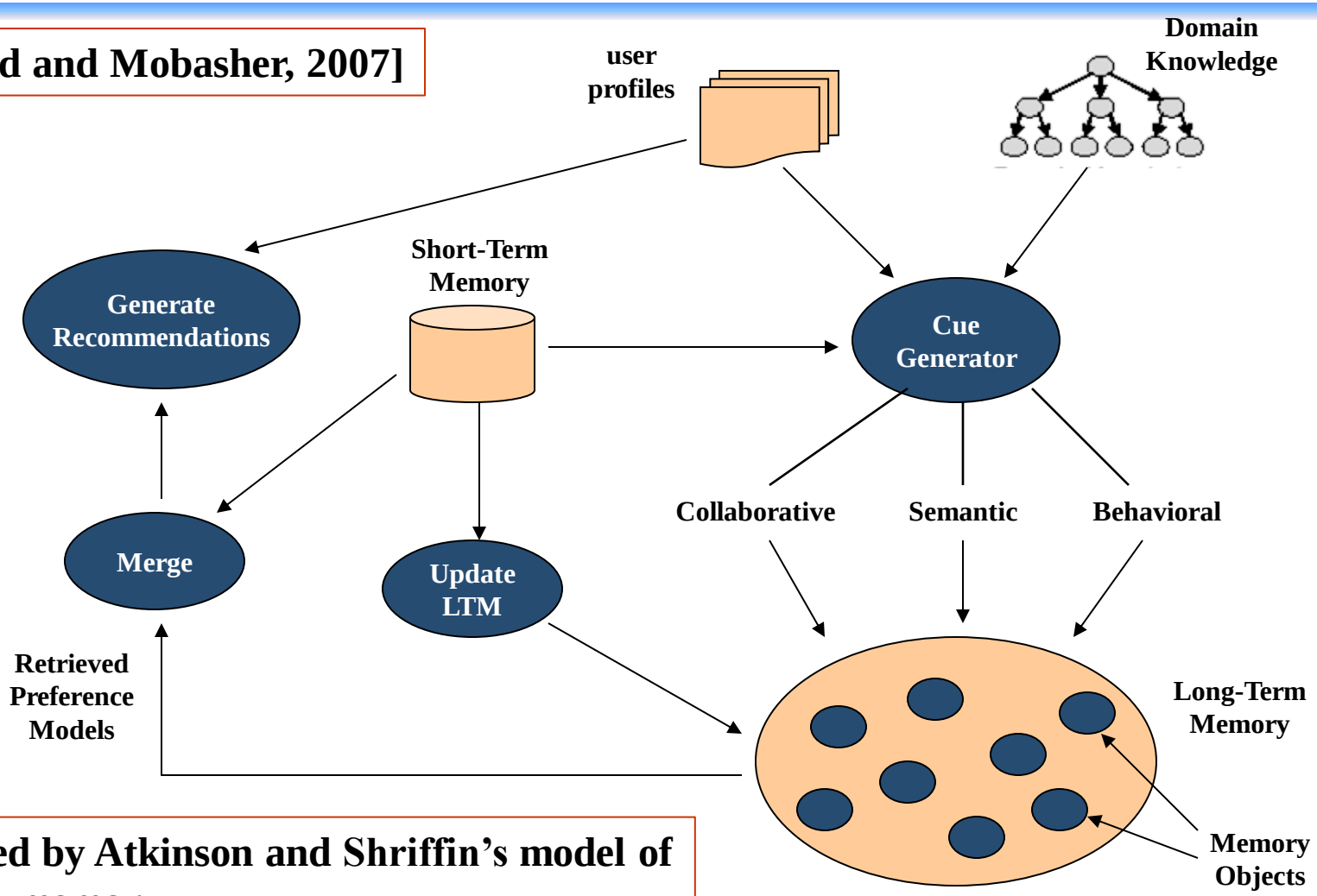
Differential Context Modeling

- **Some relevant work**
  - Y. Zheng, R. Burke, B. Mobasher. "Differential Context Relaxation for Context-aware Travel Recommendation". In EC-WEB, 2012 [DCR]
  - Y. Zheng, R. Burke, B. Mobasher. "Optimal Feature Selection for Context-Aware Recommendation using Differential Relaxation". In ACM RecSys Workshop on CARS, 2012 [DCR + Optimizer]
  - Y. Zheng, R. Burke, B. Mobasher. "Recommendation with Differential Context Weighting". In UMAP, 2013 [DCW]
  - Y. Zheng, R. Burke, B. Mobasher. “Differential Context Modeling in Collaborative Filtering". In SOCRS-2013, DePaul University, Chicago, IL, May 31, 2013 [DCM]
- **Future Work**
  - Try other similarity of contexts instead of the simple Jaccard
  - Introduce semantics into the similarity of contexts to further alleviate the sparsity of contexts, e.g., Rome is closer to Florence than Paris
  - Parallel PSO or put PSO on MapReduce to speed up optimizer
  - Additional recommendation algorithms on DCR or DCW

An Interactional Architecture for Context-Aware Recommendation

An Interactional Model for Contextual Recommendation

[Anand and Mobasher, 2007]



Inspired by Atkinson and Shriffin's model of human memory

Contextual Recommendation Generation

- **Explicit or implicit preferences for items from the active interaction are stored in the STM**
- **Contextual cues are derived from this data and used to retrieve relevant preference models from LTM**
  - Relevant = belong to the same context as the active interaction.
- **Merged with STM preferences and used to predict preferences for unseen items**
- **New Observations used to update preference models in LTM**
- **Lots of variations:**
  - LTM objects can be organized based on ontological or semantic relationships
  - LTM preference models may be aggregate objects based on similarities among users
  - Identifying relevant LTM objects can be done in a variety ways (typically using appropriate similarity functions or probabilistic approaches)

Retrieving Preference Models Using Contextual Cues

- **Task of retrieving memory objects from LTM can be viewed as estimating:**

$$\Pr(L_i \mid STM) = \frac{1}{\Pr(STM)} \sum_j \Pr(L_i \mid CC_j) \bullet \Pr(STM \mid CC_j) \bullet \Pr(CC_j)$$

Where L_i are memory objects stored in LTM and CC_j are contextual cues generated from STM

- **The calculation $\Pr(STM|CC_j)$ would be highly dependent on the particular type of cue being used.**
  - $\Pr(STM|CC_j)$ may be estimated based on collaborative, semantic, or behavioral observations
  - E.g., $\Pr(STM|CC_j)$ *could be* a weight associated with a concept (such as the fraction of positive ratings in STM associated with items in a given category

Type of Contextual Cues

- **Collaborative Cues**
  - represent items as vectors over user ratings
  - Memory objects from LTM with preference models that have a similarity greater than a particular threshold are retrieved and used in the recommendation generation
- **Semantic Cues**
  - Retrieve LTM preference models based on semantic similarity with user preference model from the active interaction.
  - Assume the existence of an item knowledge base (or textual feature space for documents) and use item semantics to compute similarity between items.
- **Behavioral Cues**
  - Can use various implicit metrics for user preferences.
  - Similarity between these metrics computed for the active interaction and LTM preference models are used as the basis for retrieving objects from LTM.
  - Another approach is to extract latent factors that drive user choice, for example, impact values associated with item attributes extracted from an ontology, or factors derived from user actions representing various tasks or topics.

Characteristics of the Framework

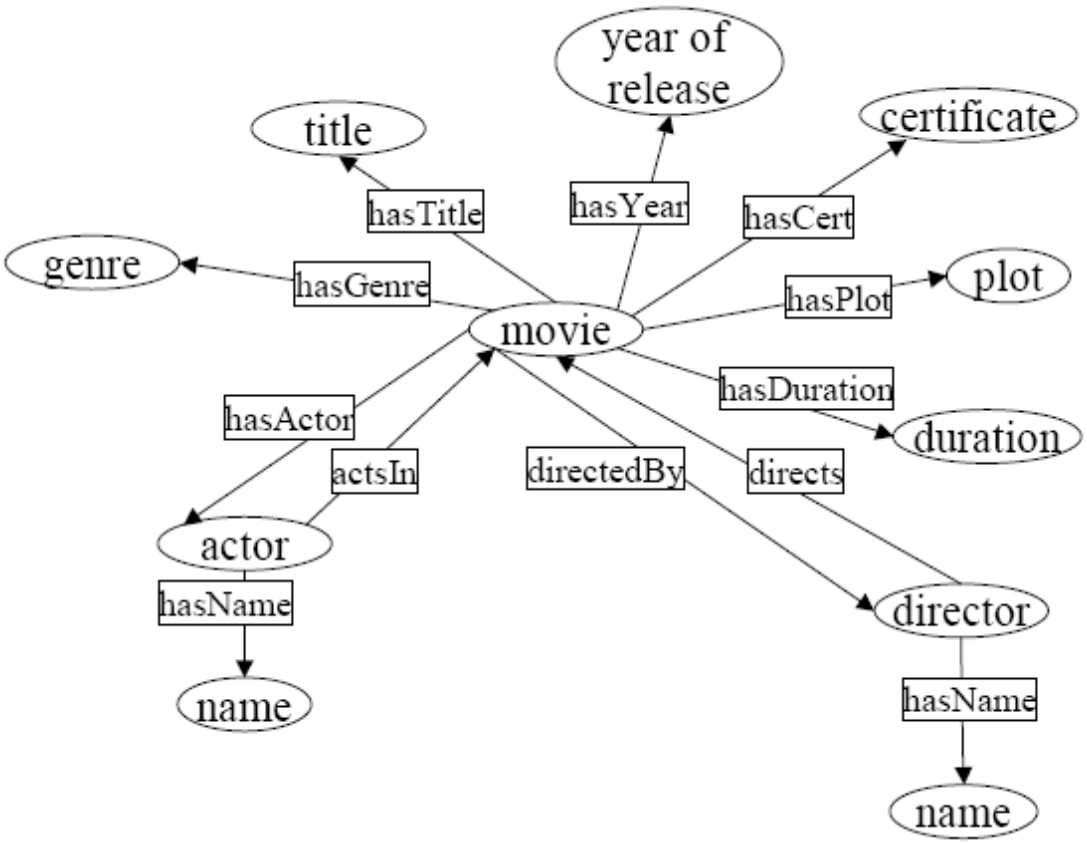
- **Different, but not in contradiction to the three architectural models for contextual filtering**
- **The Framework Emphasizes**
  - The distinction between local, transient preference models in STM and the long-term established models in LTM
  - The importance of user's interaction with the system in deriving contextual cues
  - The mutually reinforcing relationship between user activity and the context model
    - This, in turn, emphasizes the dynamic nature of context
- **Does Not Emphasize**
  - Explicit knowledge-based representation of contextual attributes
  - A rigid formulation of contextual modeling approaches
    - Very general framework and many implementations are possible (we will look at several next)

Example Implementations: Contextual Collaborative Models

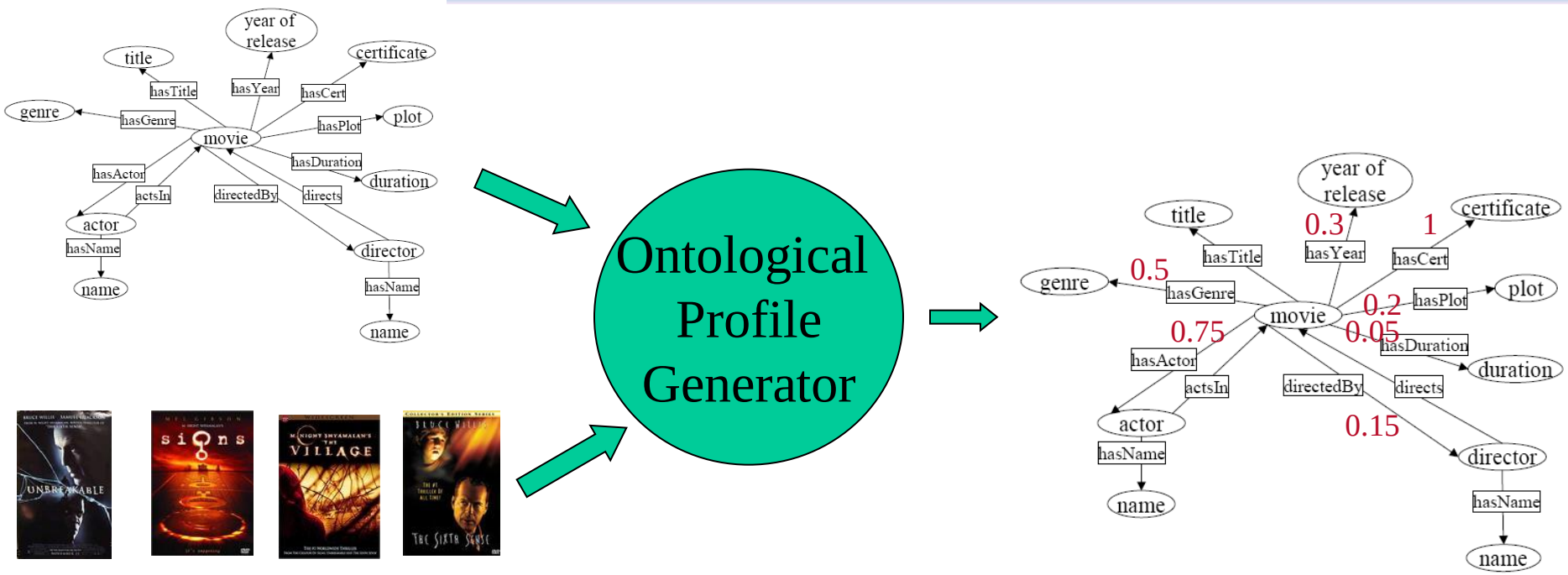
- **Inclusive Memory Model**
  - Uses all ratings in the LTM and STM of u_a to define neighborhood
- **Temporal Memory Model**
  - Uses ratings from STM and the last k ratings from LTM
- **Contextual Memory Model**
  - Uses ratings from STM and those ratings from LTM rated within the same context as current context

Example Implementation: Item Knowledge Bases and Context

- **Given user behavioral data and an item knowledge bases**
- **Discover different user behaviors that can be associated with different user interaction contexts**



A High Level View



- **One visitor may have multiple such profiles**
  - If they are distinct enough, they would represent a different context for the user visit
  - Clustering of these profiles using identified 27 distinct clusters (contexts) within 15,000 user visits

Measuring Impact

- **Defined based on an observed (f(x)) and an expected distribution (g(x)) of instances of the concept**
  - The greater the divergence (Kullback-Leibler) between these distributions, the greater the impact

- **Assumes g(x) to be uniform**
  - All instances on the concept are equally likely to be viewed by the user

$$imp_u = 1 - \frac{H(f(x))}{\log s}$$

  - s is the number of unique instances of the concept and H(f(x)) is the entropy of f(x)

Measuring Impact (II)

- **impl assumes g(x) is the likelihood of instances of the concept being viewed within a random sample**
  - Simulated
    - using the item knowledge base, assuming each item has an equal probability of being selected
    - Popularity of the item across all users (takes temporal factors into account)

$$imp_I = \frac{\sum_{x \in D_l} f(x) \log \frac{f(x)}{\hat{f}(x)}}{-\log(min\hat{f}(x))}$$

Some Evaluation Results

Algorithm	Precision	Recall	F1
RandomNeighbour	80.4%	1.7%	0.033
Traditional CF	80.45%	8.22%	0.149
ContextualRecommender	83.8%	10.38%	0.184
ContextualSemanticRecommender	84.3%	10.81%	0.191

Next: Focus on Several Other Implementations of the Interactional Model

Example Implementation: Concepts as Context Models

Sieg, Mobasher, Burke, 2007 *

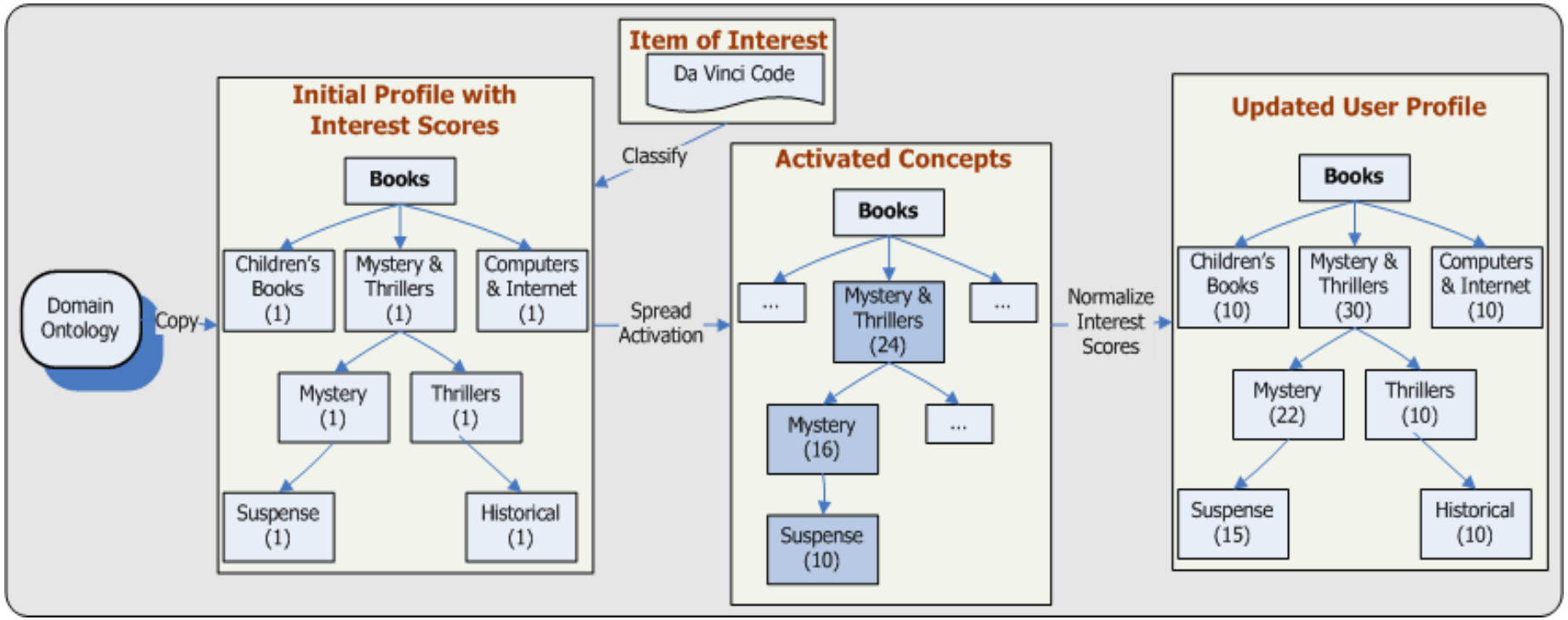
- **Ontological user profile is an instance of the reference ontology**
  - E.g., Amazon's Book Taxonomy
  - Each concept is annotated with an interest score
- **System maintains and updates the ontological profiles based on the user behavior and ongoing interaction**
  - Interactional / Representational hybrid
  - Representational part due to existence of a pre-specified ontology
- **Profile Normalization**
  - Relative importance of concepts in the profile reflect the changing interests and varied information contexts of the user

* For more information visit: *http://www.ahusieg.com/?page_id=15*

Updating User Context by Spreading Activation

- **Interest score**
  - Indicates the importance of a concept for the user
  - Gets incremented or decremented based on the user's behavior over many interactions
- **Spreading Activation**
  - Ontological User Profile is treated as the semantic network
  - Interest scores updated based on activation values
  - Initial set of concepts is assigned an initial activation value based on similarity to user's short-term interests
  - Activate other concepts based on a set of weighted relations
    - Relationship between adjacent concepts is determined based on the degree of overlap
  - Obtain a set of concepts and their respective activations

Profile Updating Illustrated



Augmenting Collaborative Filtering with the Context Model

- **Collaborative Filtering with Ontological Profiles**
  - User similarities are computed based on their interest scores across ontology concepts, instead of their ratings on individual items
    - This also helps broaden the recommendations and alleviate typical problems with CF: “cold start,” “diversity,” “serendipity”
  - Additional filtering is performed by selecting only neighbors that have significant interest in the concept of the “target item”
    - This helps in identifying the relevant “information access context” and improves accuracy

Ontology-Based Collaborative Recommendation

- **Semantic Neighborhood Generation**
  - Compare the ontological user profiles for each user to form semantic neighborhoods
  - Euclidean Distance

$$\text{distance}_{u,v} = \sqrt{\sum_{j \in C} \left(IS(C_{j,u}) - IS(C_{j,v})\right)^2}$$

    - C - set of all concepts in the reference ontology
    - $IS(C_{j,u})$ – interest score for concept C_j for target user u
    - $IS(C_{j,v})$ – interest score for concept C_j for target user v

  - Normalize the distance
  - Calculate similarity based on the inverse of the normalized distance

Ontology-Based Collaborative Recommendation

- **Prediction Computation**
  - Compute the prediction for an item *i* for target user *u*
    - Select most similar *k* neighbors
    - Concept-based filtering on the neighbors
  - Variation of Resnick's standard prediction formula

$$p_{u,i} = \bar{r}_u + \frac{\sum_{v \in V} sim_{u,v} * (r_{v,i} - \bar{r}_v)}{\sum_{v \in V} sim_{u,v}}$$

In fact a function of user, item and concept

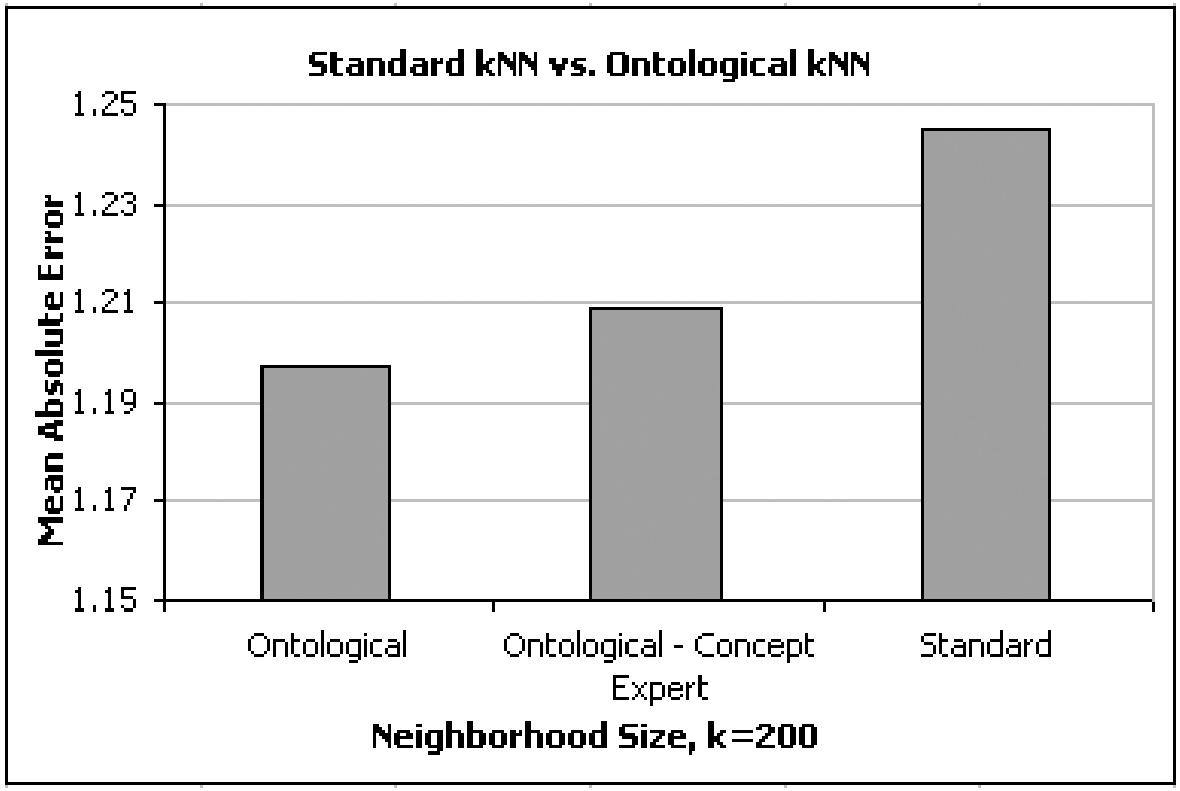
    - We use concept-based mean ratings for the target user and specific neighbors
    - *V* – set of *k* similar users

Experimental Setting

- **Reference Ontology**
  - Amazon's Book Taxonomy
    - ISBN – unique identifier for each book
    - Category, title, URL, and editorial reviews
    - 4,093 concepts and 75,646 distinct books
- **Evaluation using the book ratings collected by Ziegler**
  - 4-week crawl from the BookCrossing community
  - 72,582 book ratings belonging to users with 20 or more ratings
  - Training data utilized for spreading activation
  - Test data used for predicting ratings
  - K-fold cross validation, $k = 5$

Experimental Results

- **Mean Absolute Error, k=200**
  - ANOVA significance test with 99% confidence interval, *p-Value < 0.01 (6.9E-11)*



Experimental Results

- **Recommendation Diversity**
  - Personalization
    - uniqueness of different users' recommendation lists based on inter-list distance
    - q_{ij} - number of common items in the top N recommendations for two given users i and j

$$d_{ij}(N) = 1 - \frac{q_{ij}(N)}{N}$$

  - Surprisal
    - unexpectedness of a recommended item relative to its overall popularity
    - i – given item in a user's recommendation list
    - $frequency_i$ – number of overall positive ratings for i divided by the total number of users

$$I_i = \log_2\left(\frac{1}{frequency_i}\right)$$

Experimental Results

- **Recommendation Diversity**
  - Improved Personalization
  - Improved Surprisal

Algorithm	Personalization, d(20)	Surprisal/Novelty, I(20)
Standard kNN	0.922	6.544
Ontological kNN	0.975	7.286
ANOVA p-value	1.9417E-276	4.9221E-181

For more information related to this work visit:
http://www.ahusieg.com/?page_id=15

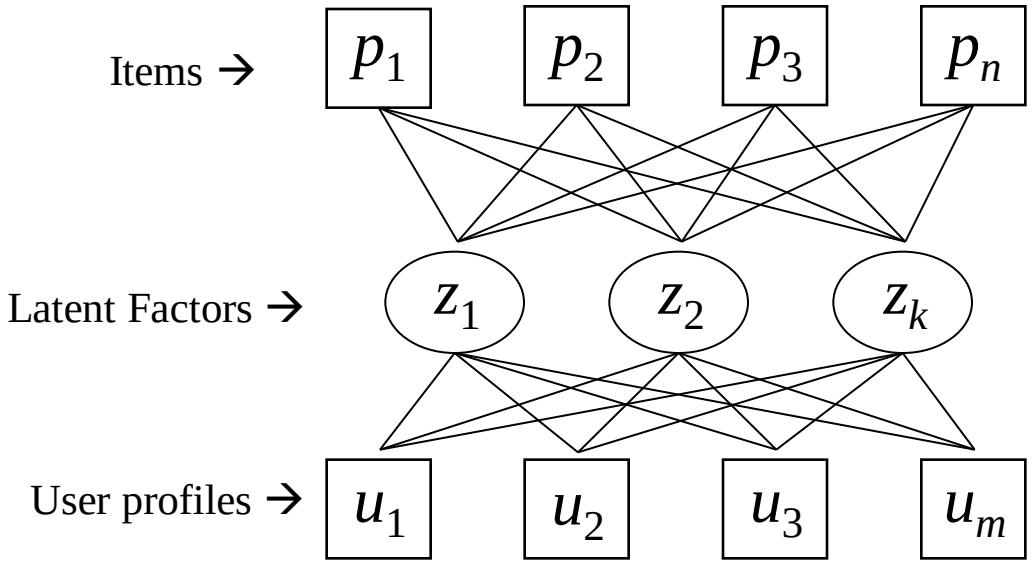
Example Implementation: Latent Variable Context Models

Jin, Zhou, Mobasher, 2005

- **Generative approach to modeling user context**
  - Basic assumption:
    - users' interactions involve a relatively small set of contextual states that can "explain" users' behavior
    - useful when dealing with applications involving user's performing informational or functional tasks
  - Contexts correspond to tasks and are derived as latent factors in the observational data collected in the short-term memory.
  - Probabilistic Latent Semantic Analysis (PLSA) can be used to automatically learn and characterize these tasks, as well as the relationships between the tasks and items or users
  - Algorithm based on Bayesian updating to discover individual user's task transition patterns and generating ***task-level user models*.**

Latent Variable Models

- **Assume the existence of a set of latent (unobserved) variables (or factors) which "explain" the underlying relationships between two sets of observed variables.**



Advantage of PLSA:

Probabilistically determine the association between each latent factor and items, or between each factor and users.

In navigational data, the latent factors correspond to distinguishable patterns usually associated with performing certain informational or functional tasks. Context = Task!

PLSA Model – Behavioral Observations

- **represented as a matrix (UP_{mxn})**
- **each entry UP_{ij} corresponds to a weight of item *j* within a user interaction *i*. The weight can be binary, or based on the various implicit or explicit measures of interest.**

	p1	p2	p3	p4	p5	. . .
User 1	1	0	0	1	1	. . .
User 2	0	1	1	0	1	. . .
…	…	…	…	…	…	…

Note: similar models can be built using other types of observation data, e.g., <users, query terms>, <pages, keywords>, etc.

PLSA Model

- **Consider each single observation (u_i, p_j) as a generative process:**
  - 1. select a user profile u_i with $\Pr(u_i)$
  - 2. select a latent factor z_k associated with u_i with $\Pr(z_k \mid u_i)$
  - 3. given the factor z_k, pick an item p_j with $\Pr(p_j|z_k)$
  - each observation is represented as

$$\Pr(u_i, p_j) = \sum_{z_k} \Pr(u_i) \bullet \Pr(z_k \mid u_i) \bullet \Pr(p_j \mid z_k)$$

  - using Bayes' rule, we can also obtain:

$$\Pr(u_i, p_j) = \sum_{z_k} \Pr(z_k) \bullet \Pr(u_i \mid z_k) \bullet \Pr(p_j \mid z_k)$$

Model Fitting

- **Maximizing the likelihood of user observations**

$$L(U,P) = \sum_{u_i, p_j} UP_{ij} \bullet \log \Pr(u_i, p_j) = \sum_{u_i, p_j} UP_{ij} \bullet \log \sum_{z_k} \Pr(z_k) \Pr(u_i \mid z_k) \Pr(p_j \mid z_k)$$

- **Estimating parameters using Expectation-Maximization (EM) algorithm**

- **Output of the EM algorithm:**
  - **$\Pr(u_i \mid z_k)$, $\Pr(p_j \mid z_k)$, $\Pr(z_k)$**

- **Using Bayes' rule, we can also obtain:**
  - **$\Pr(z_k \mid u_i)$, $\Pr(z_k \mid p_j)$**

Context-Based Patterns

- **Identify user segments**
  - For each context z_k, find top users with the highest $\Pr(u|z_k)$ as a user segment.
  - Applications: collaborative recommendation; market segmentation
- **Identify characteristic items or users w.r.t. each task**
  - Characteristic items: $\{p_{ch} : \Pr(p_{ch}|z_k)*\Pr(z_k|p_{ch}) \geq \alpha\}$
  - Characteristic user profiles: $\{u_{ch} : \Pr(u_{ch}|z_k)*\Pr(z_k|u_{ch}) \geq \theta\}$
  - Applications: task/context based search; user or item classification
- **Identify a given user's context**
  - For each user (interaction) u, find tasks with the highest $\Pr(z|u)$
  - Allows for performing higher-level behavioral analysis (based on discovered tasks or contexts)

Methodological Note

- **Two Web navigational data sets**
  - CTI: about 21,000 sessions, 700 pages, 30 tasks (defined), user history length set to 4+, 20,000+ features
  - Realty Data: about 5,000 sessions, 300 properties, user history length set to 4+, 8000+ features
- **Experimental Methodology:**
  - Measure the accuracy of our recommendation system, compare it to a standard recommendation system based on first-order Markov model
  - Use “hit ratio” to measure recommendation accuracy
  - Hit ratio:
    - Given a test session, use the first k items to generate a top-N recommendation set.
    - If this set contains the $k+1^{th}$ item of the test session, we consider it a hit.
    - HitRatio = totalHits / totalSessions

      (averaged over 10 runs in 10-fold cross-validation)

Examples of Inferred Tasks

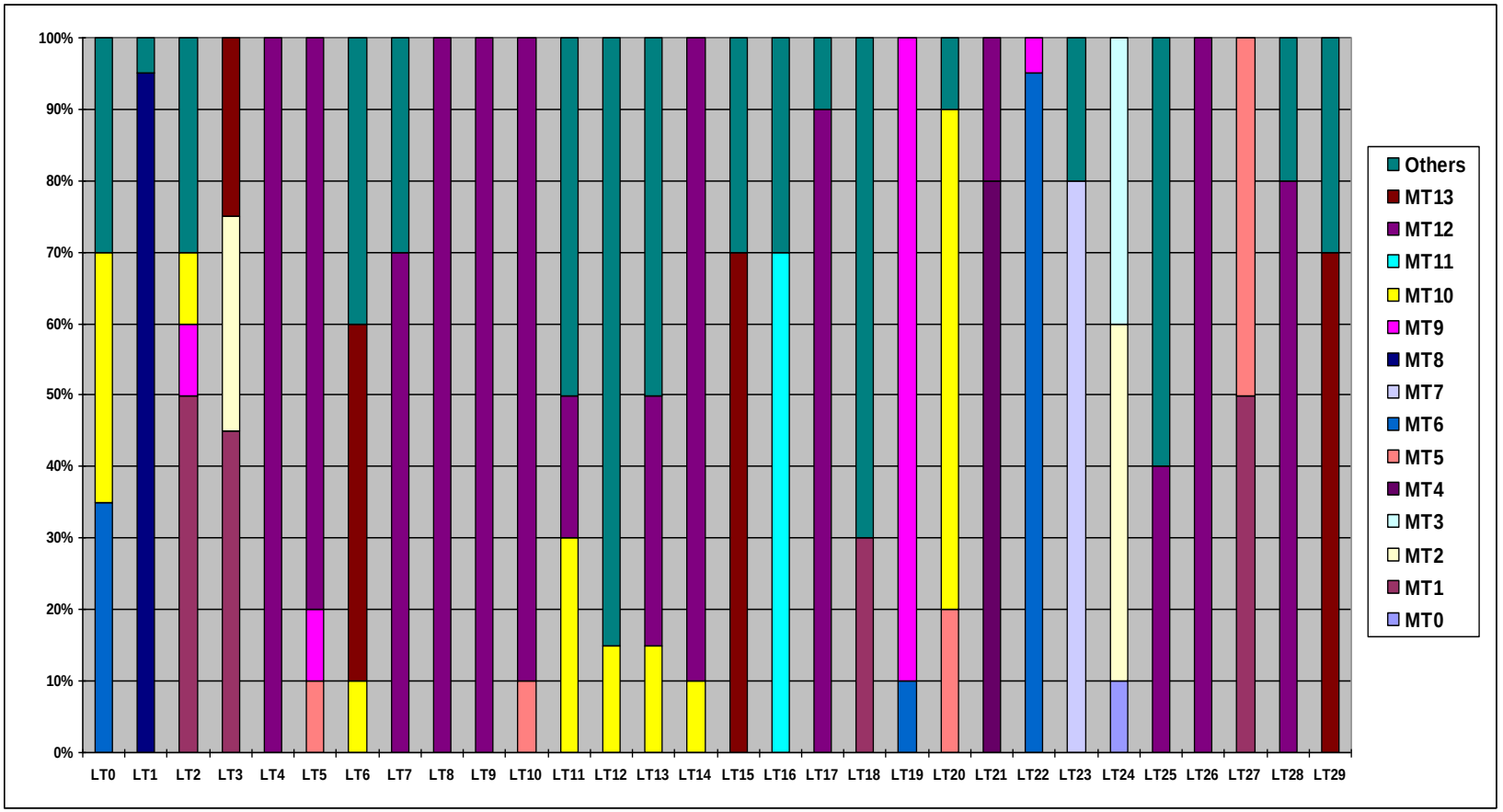
	A real user session (page listed in the order of being visited)
1	Admission main page
2	Welcome information – Chinese version
3	Admission info for international students
4	Admission - requirements
5	Admission – mail request
6	Admission – orientation info
7	Admission – F1 visa and I20 info
8	Application – status check
9	Online application - start
10	Online application – step 1
11	Online application – step 2
12	Online application - finish
13	Department main page

Top tasks given this user – Pr(task | user)	
Task 10	**0.4527**
Task 21	**0.3994**
Task 3	0.0489
Task 26	0.0458

PageName
Department main page
Admission requirements
Admission main page
Admission costs
Programs
Online application – step 1
…
Admission – international students

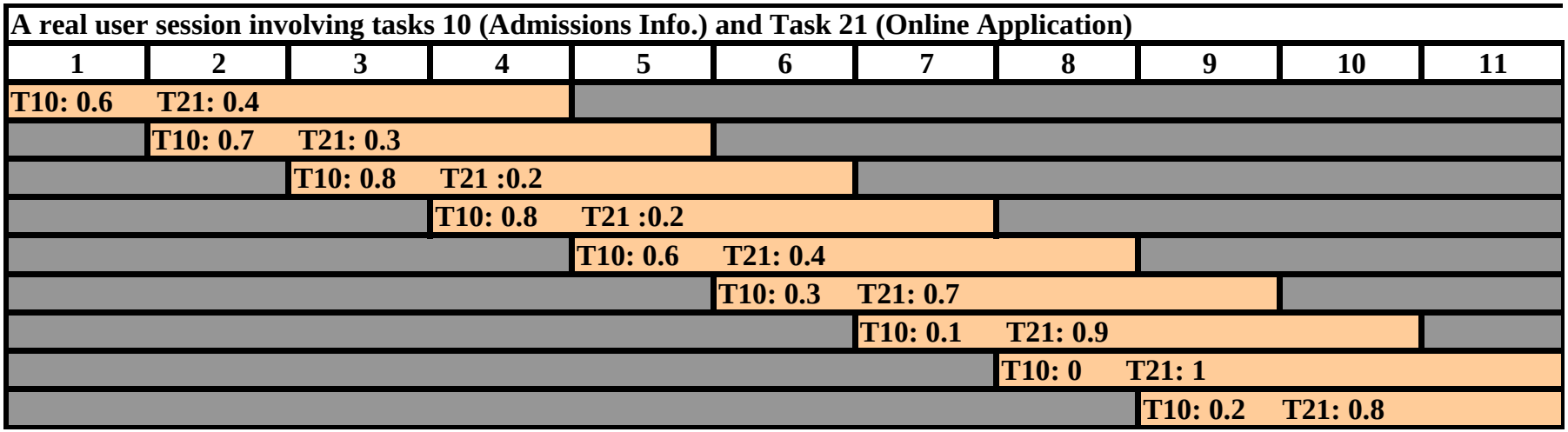
PageName
Online application – start
Online application – step1
Online application – step2
Online application - finish
Online application - submit
…
Department main page

Distribution of Learned Tasks



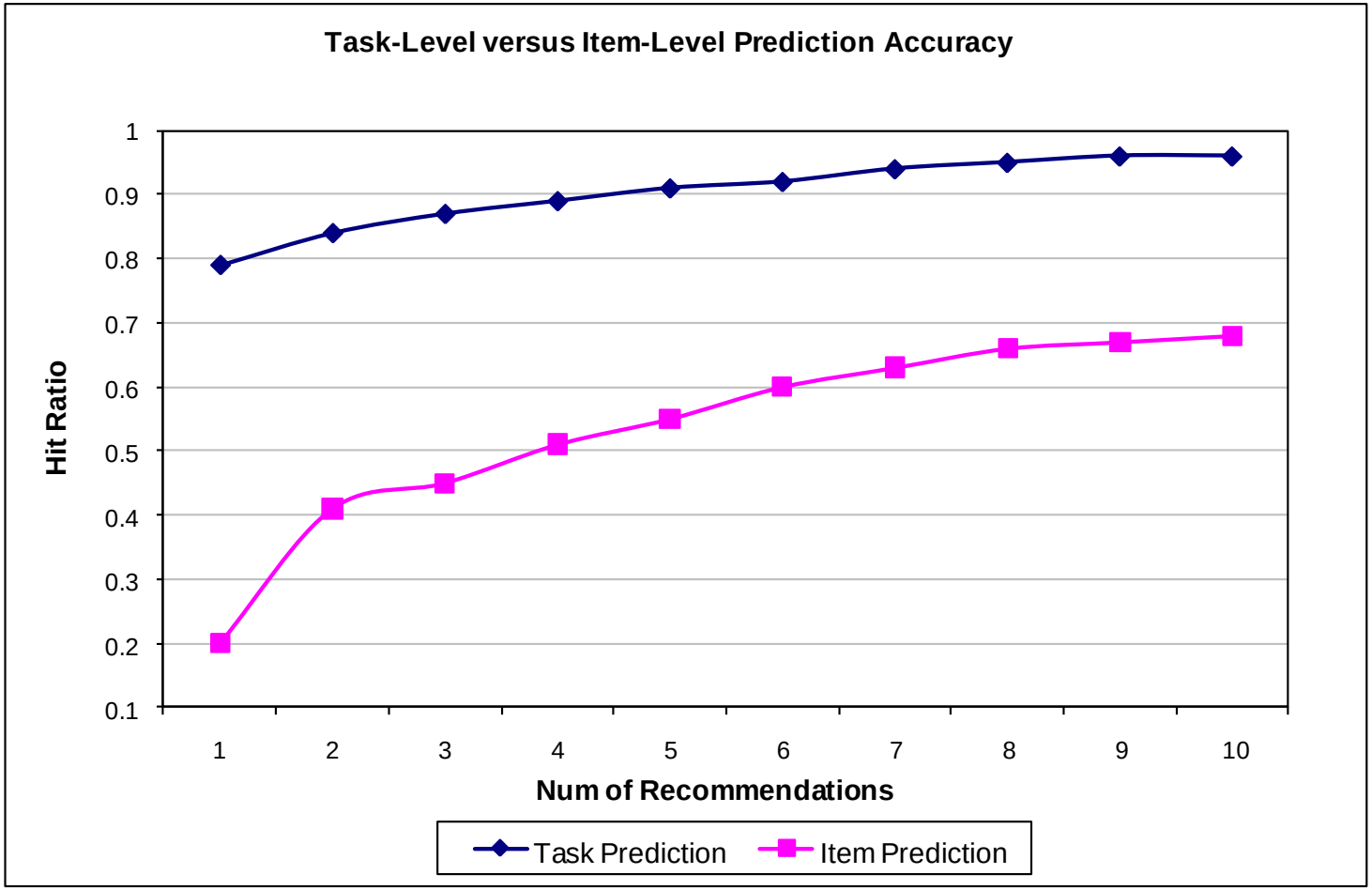
MT0 – MT13 were actual tasks, commonly performed by users on the Web site, selected manually by domain experts

Task-Level User Tracking?



Sliding window, W (with $|W| = 4$) moves from the beginning to the end of this user session. Top 2 tasks and the corresponding values for Pr(task | W) are shown.

Task/Context Prediction



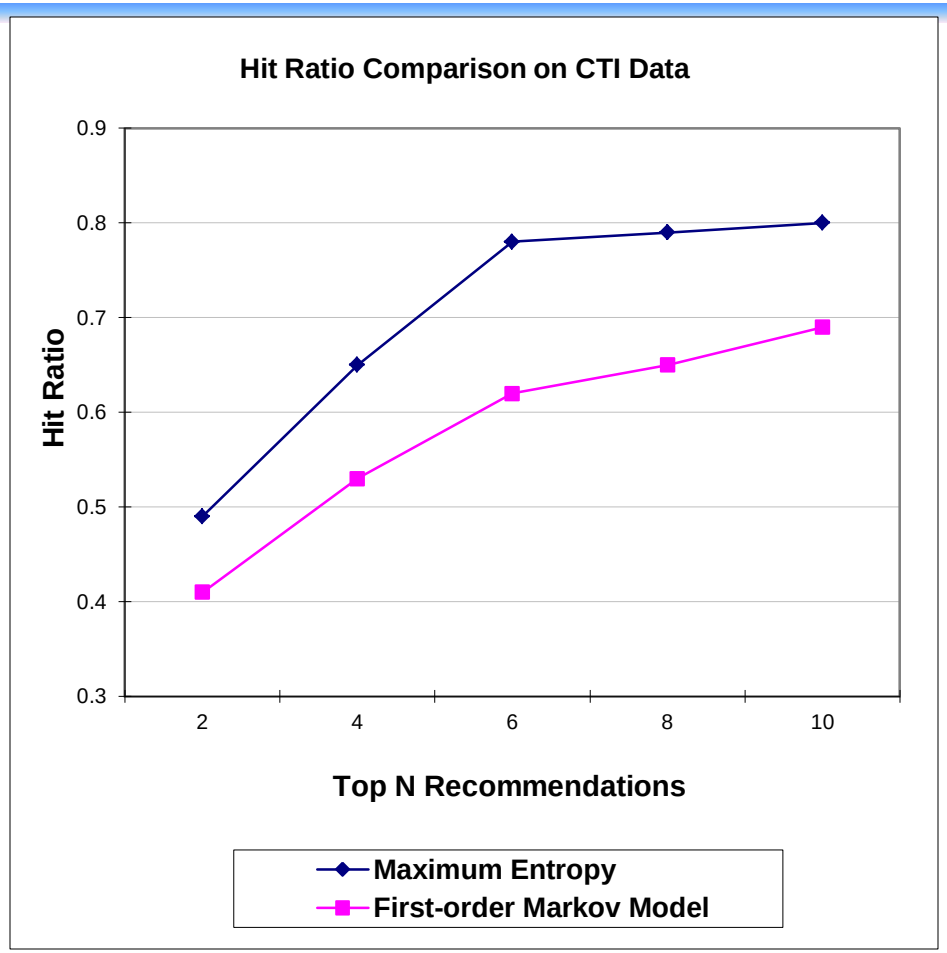
Contextual Modeling: The Maximum Entropy Framework

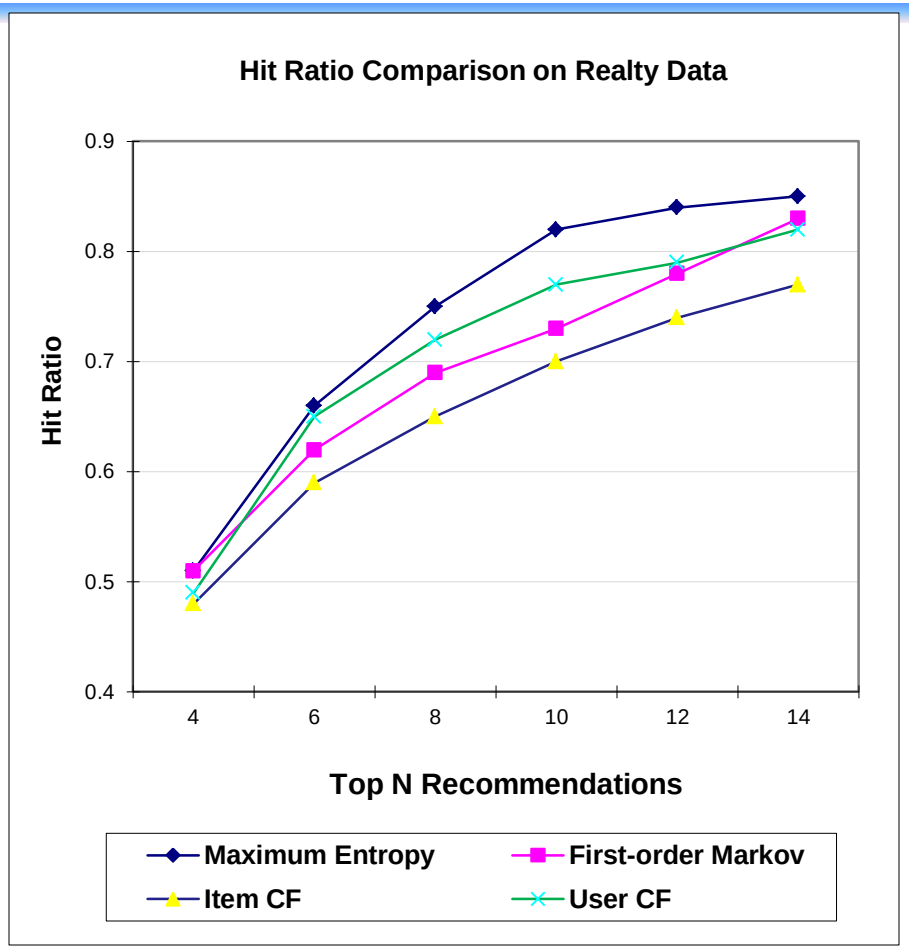
- **A statistical model widely used in language learning, text mining (Rosenfeld 1994, Berger et al. 1996).**
  - Estimate probability distribution from the data
  - Labeled training data used to derive a set of constraints for the model that characterize class-specific expectations in the distribution
  - Constraints represented as expected values of “features” which are real-valued functions of examples
  - Goal: find a probability distribution which satisfies all the constraints imposed on the data while maintaining maximum entropy
- **Advantage:**
  - integration of multiple sources of knowledge or multiple constraints without subjective assumptions or intervention.
  - In this case we use ME to integrate learned contextual information into the preference modeling process.

Using the Maximum Entropy Model

- **The basic Maximum Entropy framework:**
  - First, Identify a set of feature functions that will be useful for the desired task (e.g., prediction or classification)
  - Then, for each feature:
    - Measure its expected value over the training data
    - Take this expected value to be a constraint for the model distribution
- **In our model**
  - Define two sets of features
    - 1. features based on item-level transitions in users interactions
    - 2. features based on task-level transitions in user interactions
  - Max. Ent. Framework uses both sets of features to compute $\Pr(p_d \mid H(u))$

Experimental Results





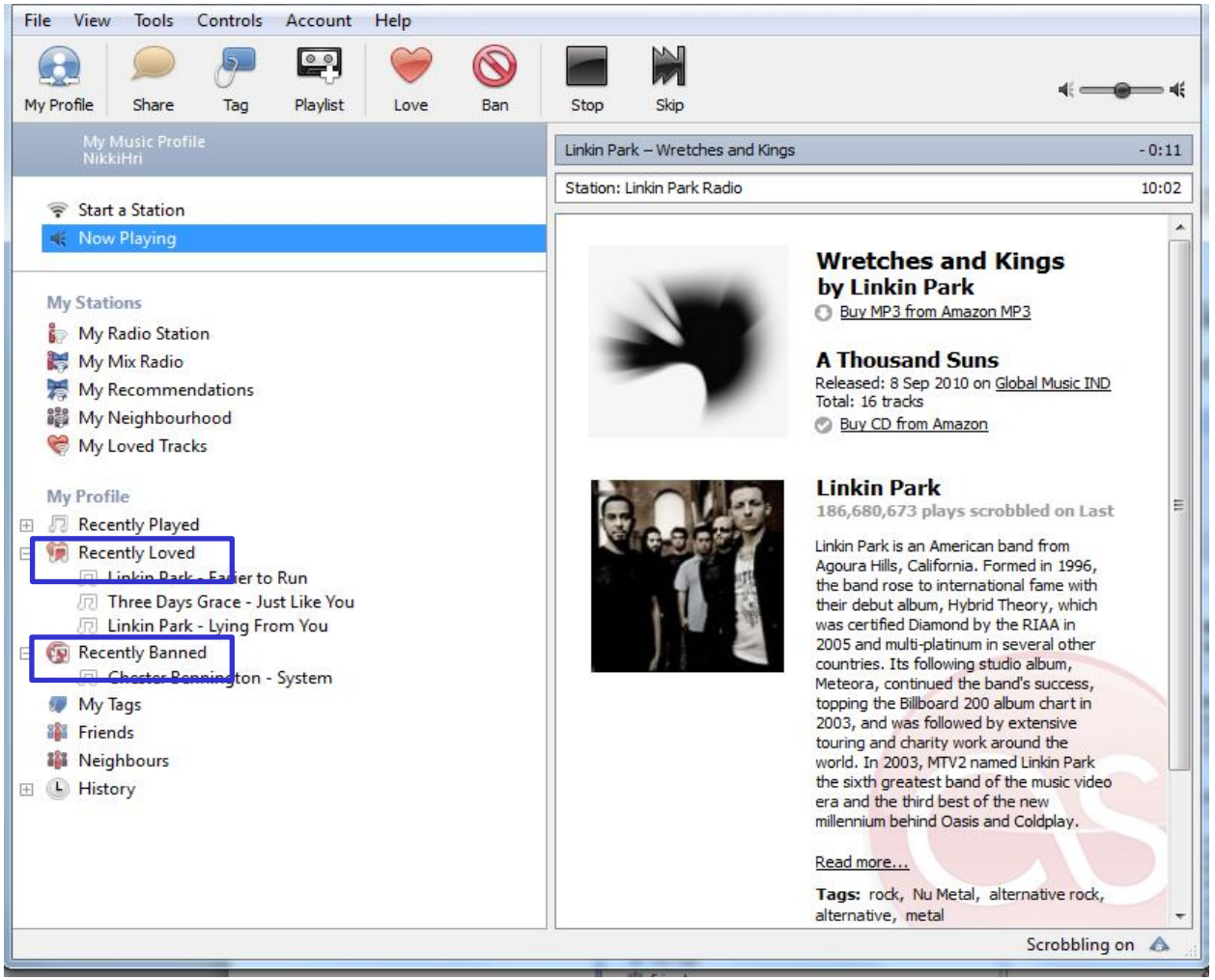
Example Implementation: Inferring Latent Contexts From Social Annotation

Hariri, Mobasher, Burke, RecSys 2012

- **Context-Aware Music Recommendation Based on Latent Topic Sequential Patterns**
- **In the domain of music recommendation:**
  - Context is usually not fully observable
  - Contextual information is dynamic and should be inferred from users' interactions with the system such as:
    - Liking/disliking playing songs
    - Adding/skipping songs
    - Creating different stations by selecting different track seeds or artists
- **Different applications:**
  - Song recommendation
  - Playlist generation
  - Playlist recommendation

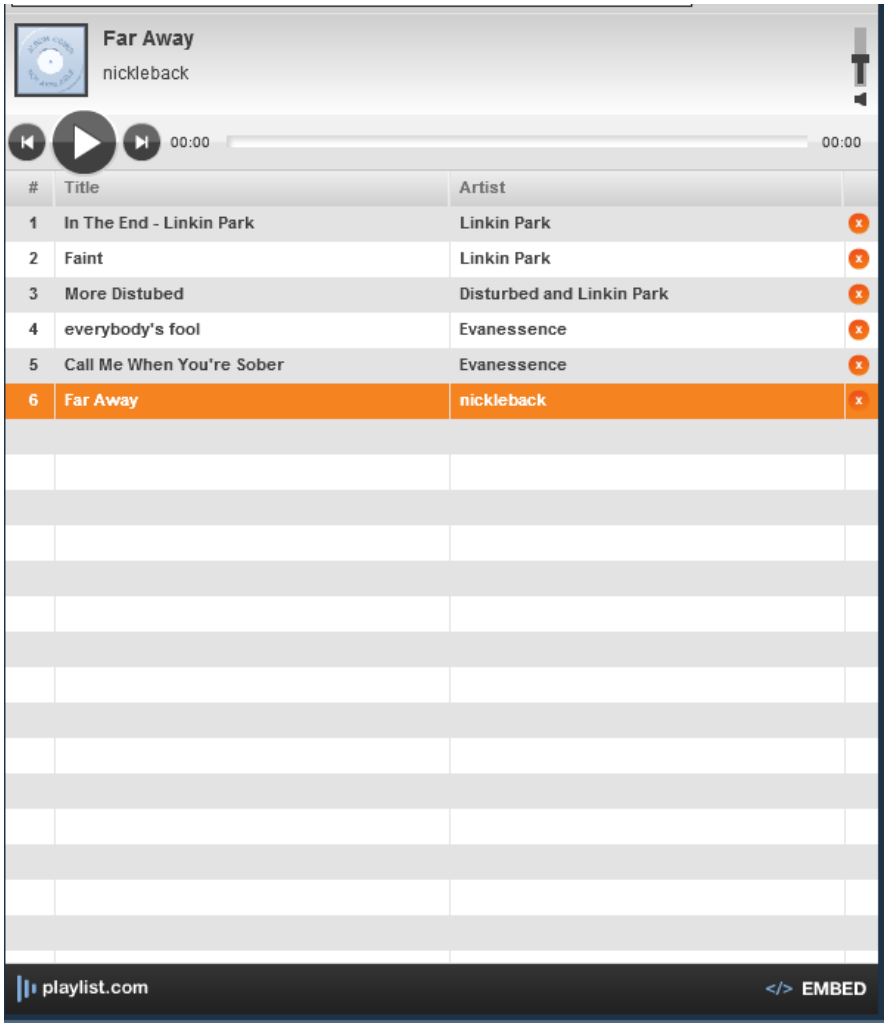
Song Recommendation

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



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



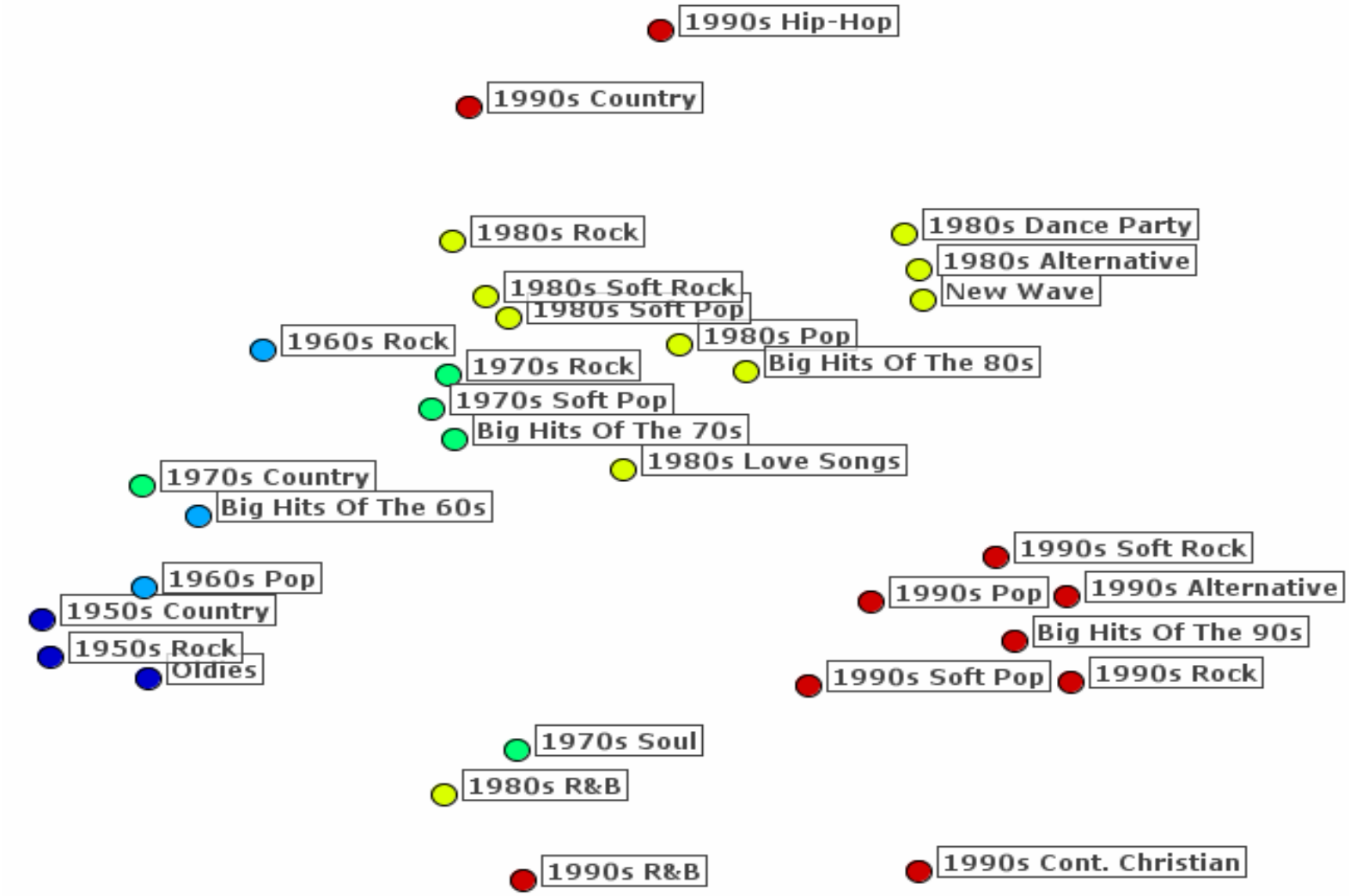
Topic Modeling For Song Context Representation

- **LDA topic modeling approach was used to map user's interaction sequence to a sequence of latent topics which capture more general trends in user's interests**
- **The latent topics are generated from the top most frequent tags associated with songs, obtained from social tagging Web sites such as last.fm.**
  - Tags may characterize song features, user’s situation, mood, etc.
  - 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, the set of dominant topics are selected that have probabilities higher than a specific threshold value

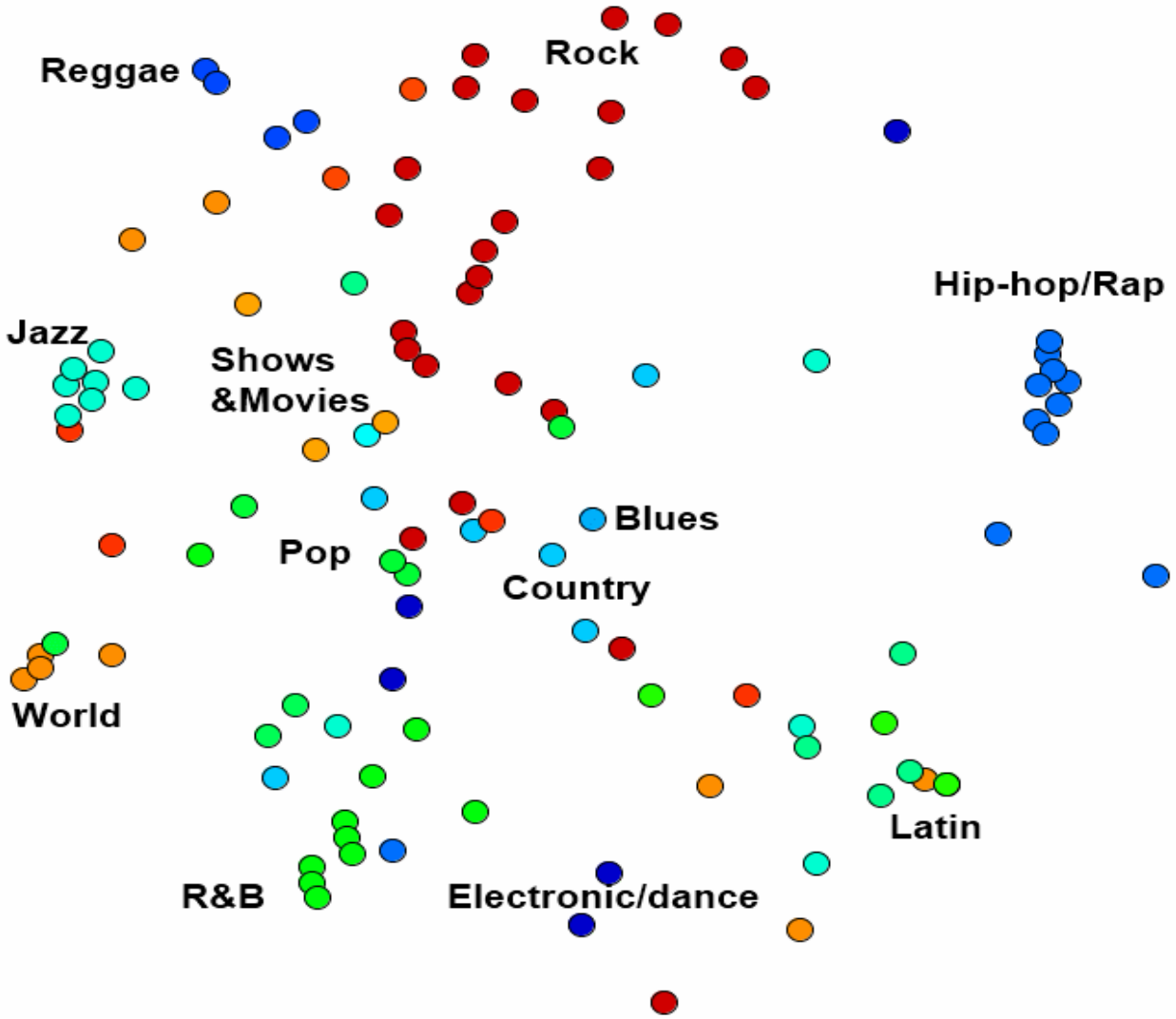
Top Most Frequent Tags for a Sample of Topics

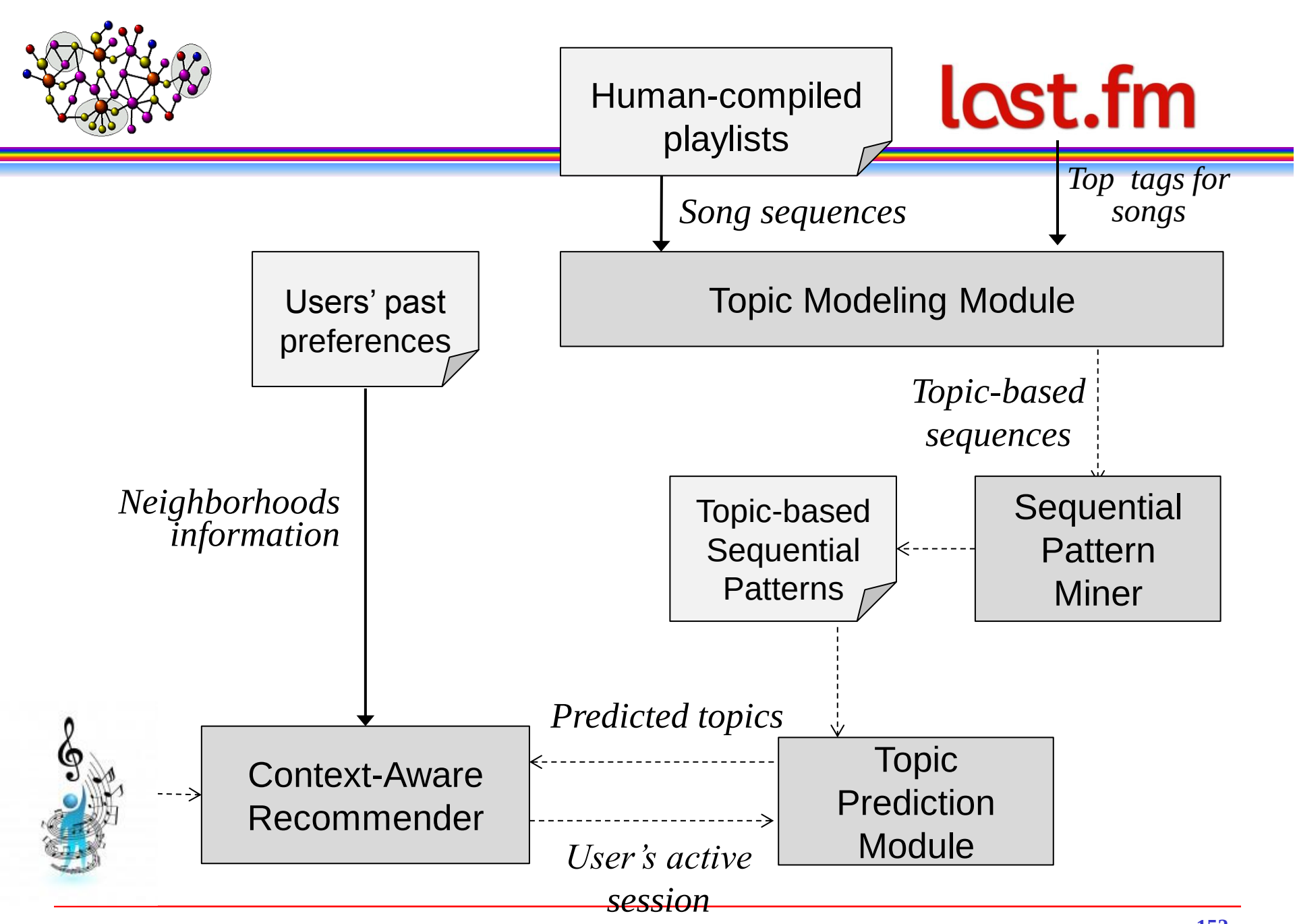
Topic#1	Topic#2	Topic#3	Topic#4	Topic#5	Topic#6	Topic#7
ambient	latin	death	60s	chill	beautiful	electronic
instrumental	world	thrash	oldies	downtempo	sad	electronica
soundtrack	streamable	black	roll	chillout	mellow	house
classical	spanish	heavy	50s	christmas	melancholy	techno
beautiful	para	doom	rockabilly	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

Music Era Visualization by Topic-based Co-Occurrence Analysis



Music Genre Visualization by Topic-based Co-Occurrence Analysis




Human-compiled playlists
last.fm
Song sequences
Top tags for songs
Users' past preferences
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

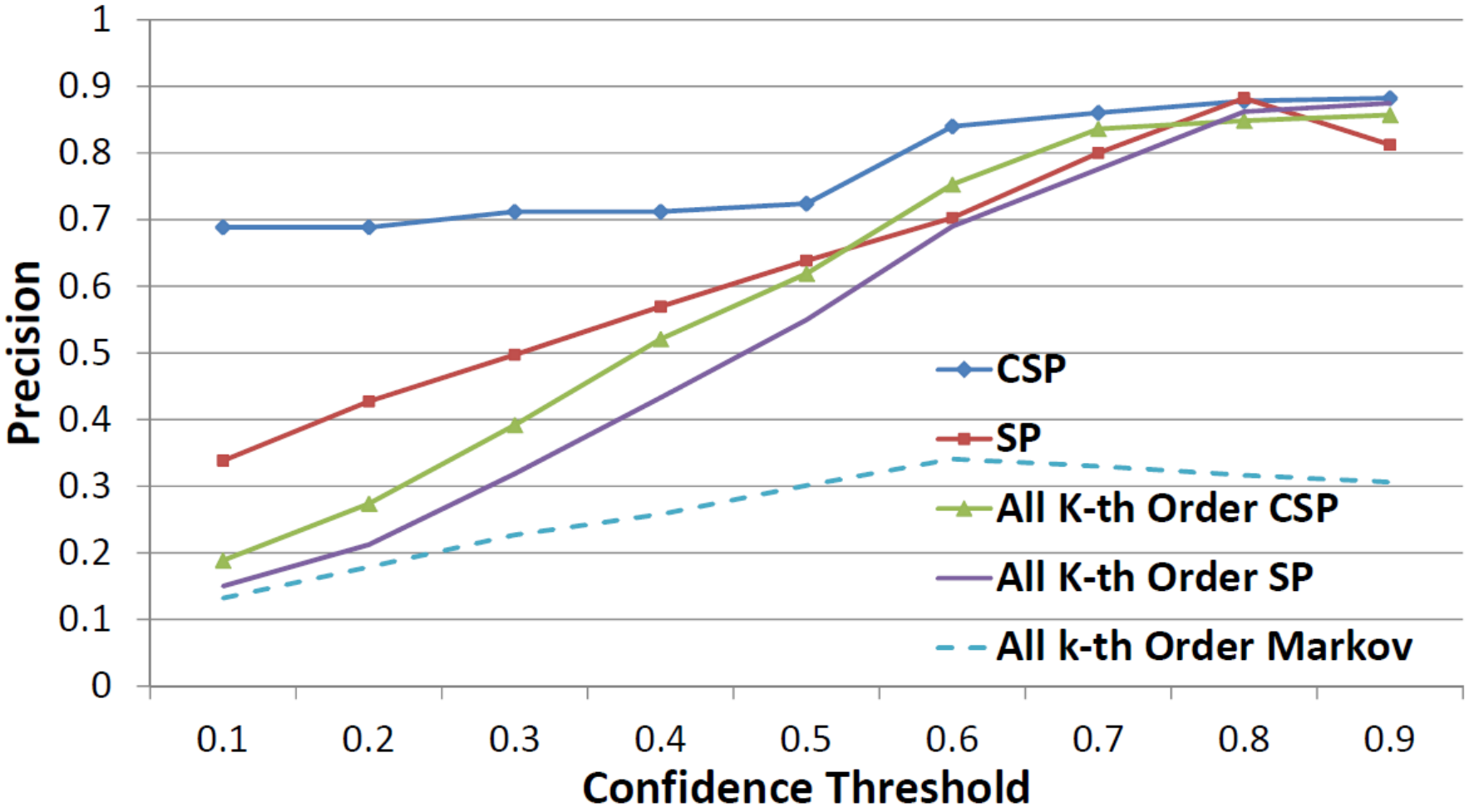
Sequential Pattern Mining and Topic Prediction

- Used to capture typical changes in the contextual states over time
- Using a training set consisting of human-compiled playlists, sequential patterns are mined over the set of latent topics where each pattern represents a frequent sequence of transitions between topics/contexts
- Given a user's current interaction as the sequence of last w songs, the discovered patterns are used to predict the context for the next song
- The predicted context is then used to post-filter and re-rank the recommendations produced based on the whole history of the user's preferences

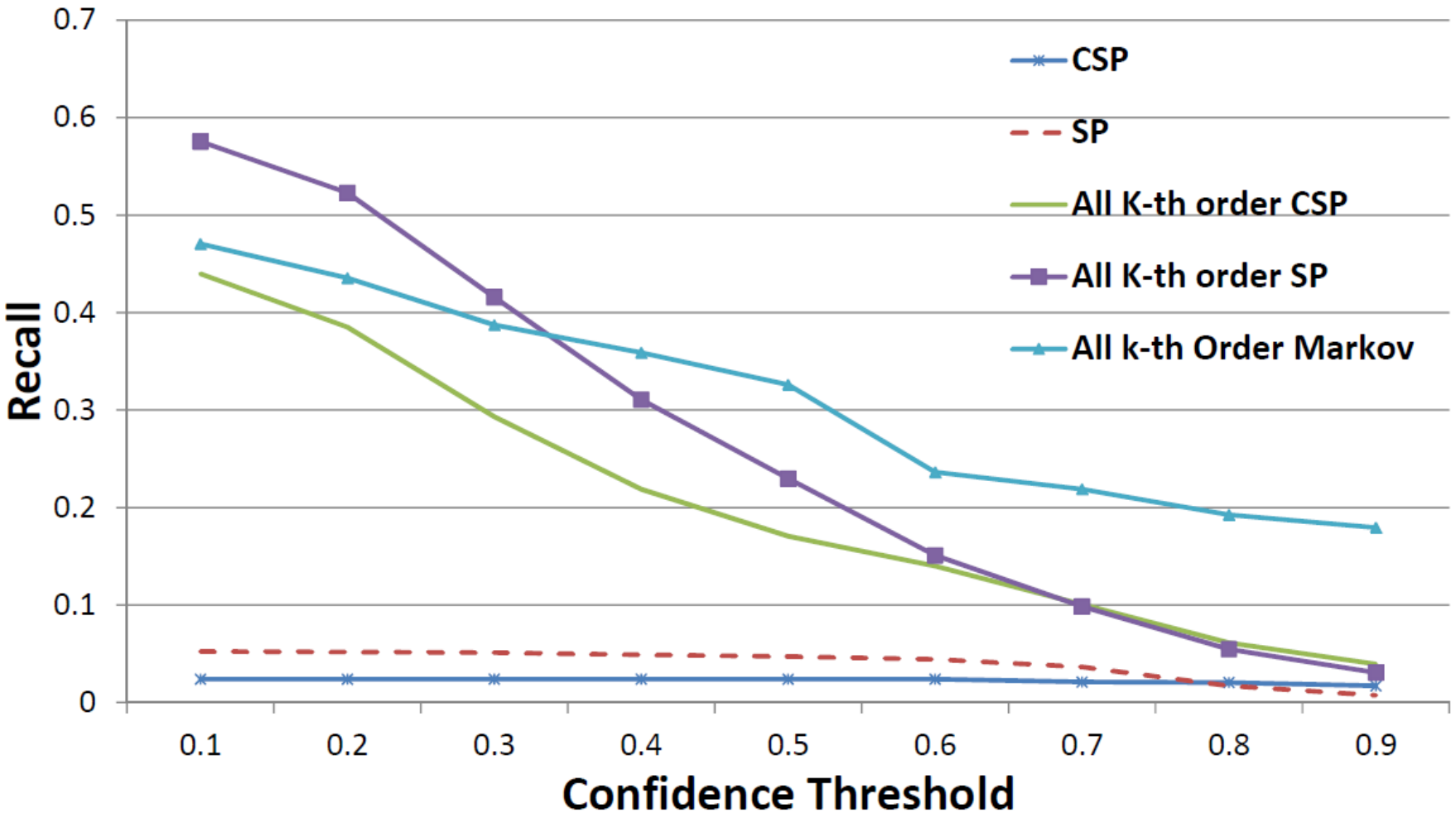
Evaluation

- **Dataset :**
  - 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
  - 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

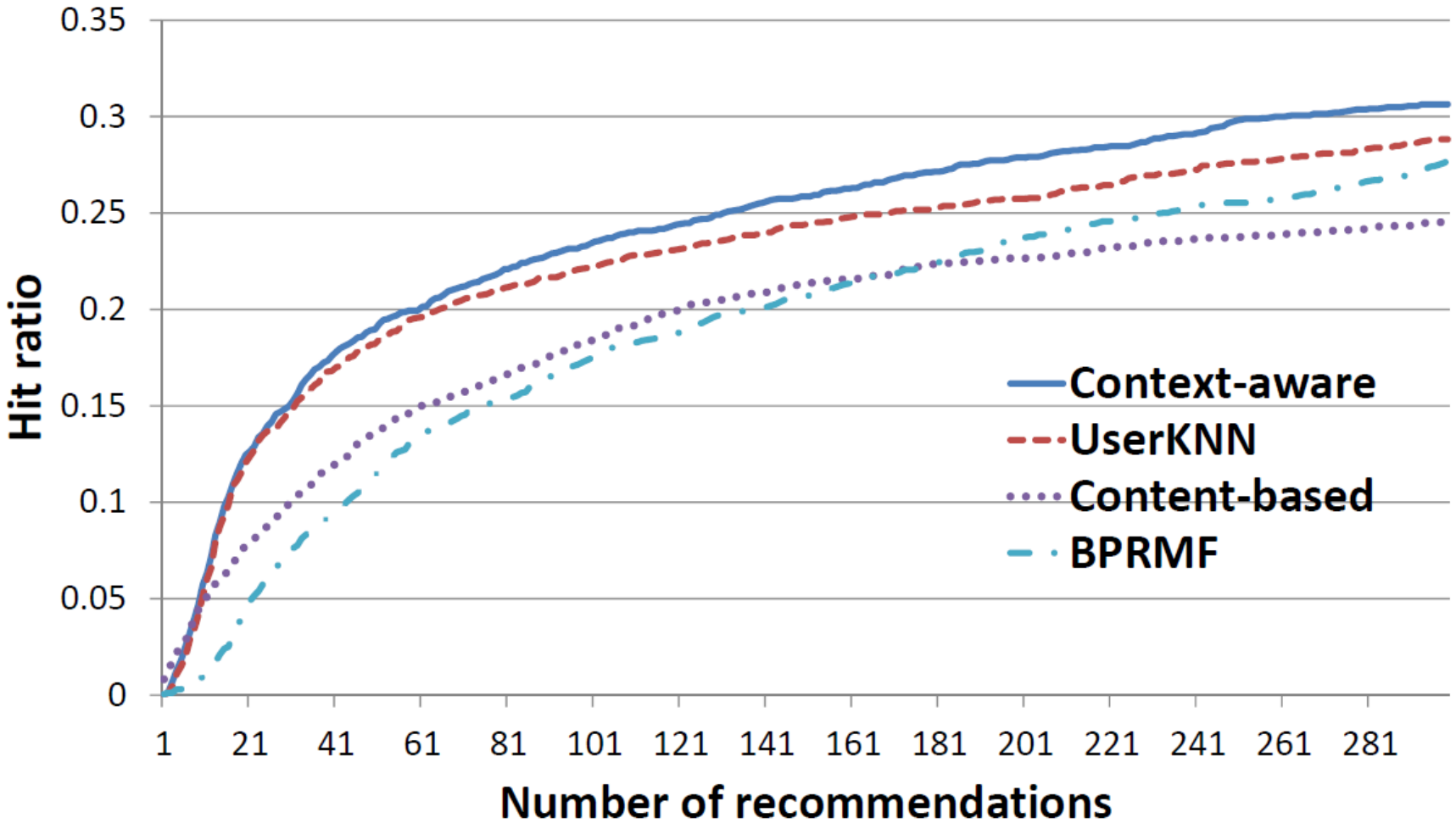
Topic Prediction Precision



Topic Prediction Recall



Song Recommendation Performance



Example Implementation: Query-Driven Context-Aware Recommendation

Hariri, Mobasher, Burke, 2013

- **In some applications, context is represented as a subset of an item feature space**
  - Contextual information is not pre-specified
  - Contextual information can be acquired
    - *explicitly* acquired by directly asking users or
    - *implicitly* based on user behavior or user environment
- **Example Scenario: music recommender system**
  - user can specify his/her current interest in a specific genre of music by providing that information in the form of a query (context)
  - both the queried context and the established user profile are used to recommend songs that best match user's short-term situational needs as well as his or her long-term interests.

Example Playlist

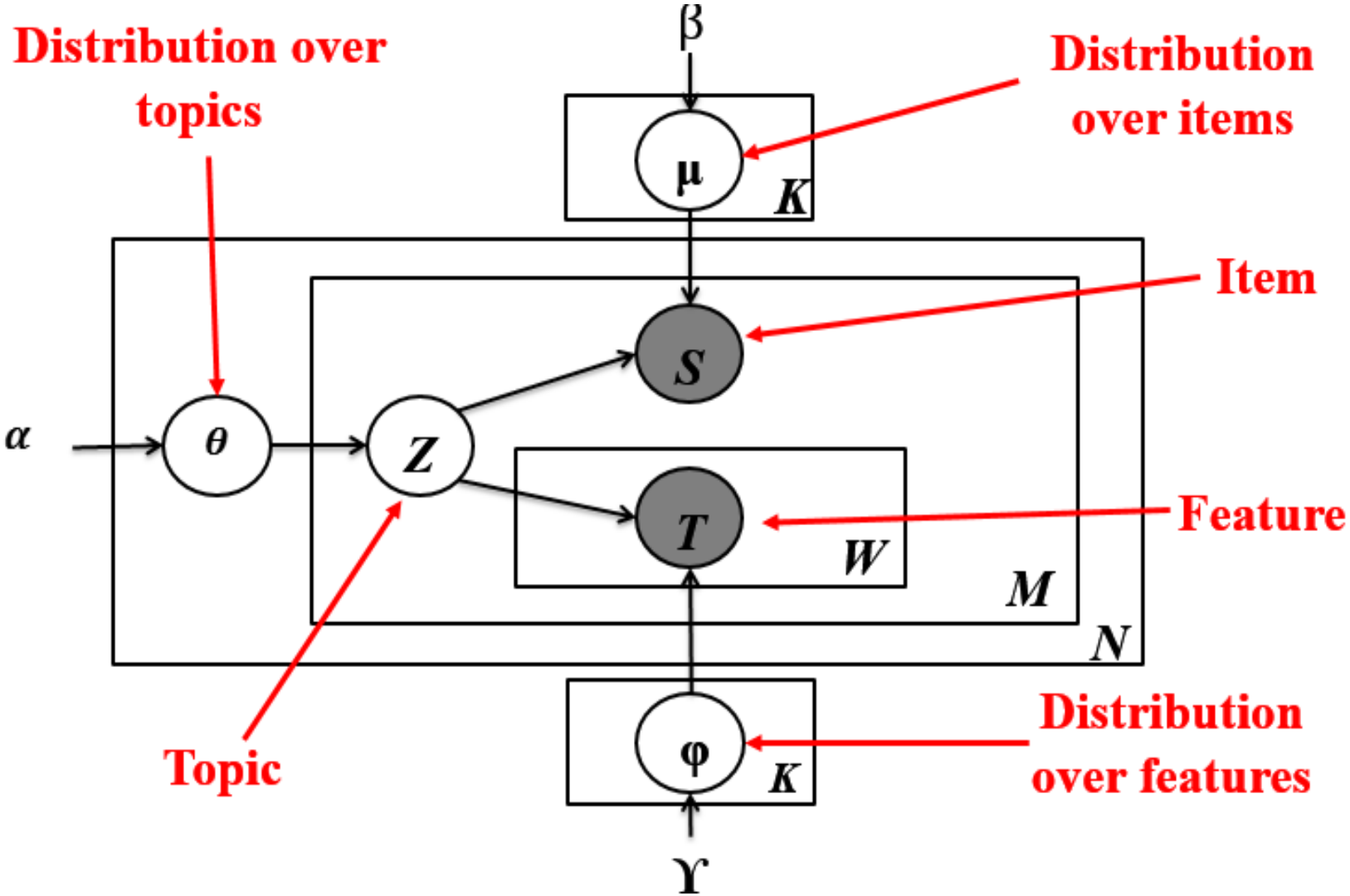
Artist	Song	Popular Tags
Luce	Good day	rock, 2000s, happy, playful, feel good music, music from the oc
Ben Howard	Keep your head up	acoustic, British, folk, indie, singer-songwriter, 2000s, alternative rock
Nada Surf	Always love	alternative, indie rock, 2000s, happy
Vega 4	Life is beautiful	alternative, 2000s, indie rock, Grey's anatomy, happy
Imagine Dragon	On top of the world	alternative, indie rock, 10s, happy, cheerful chill easy listening
State Radio	Right me up	reggae-rock, chill, indie, happiness, fun, mellow, summer

If the user specifies "80s" as the context, ideal recommendations should match the user's previous interests (happy and rock music) as well as the queried context (80s music).

A Probabilistic Model

- **A unified probabilistic model for contextual recommendation**
  - model integrates user profiles, item descriptions, and contextual information.
  - assumes that the contextual information can be represented using the same feature space underlying the representation of items
- **Our model is an extension of the Latent Dirichlet Allocation (LDA).**
  - Each user profile is modeled as a mixture of the latent topics. These topics capture the feature associations as well as the item co-occurrence relations in user profiles.

Graphical representation of the context-aware recommendation model



Context-aware recommendation model

- **Model defines a generative process for user profiles. The process for generating user p is as follows:**

1. Choose $\theta_p \sim Dir(\alpha)$.
2. Choose $\phi_k \sim Dir(\gamma)$, for each topic k
3. Choose $\mu_k \sim Dir(\beta)$, for each topic k
4. For each of the M items, s_i, in the user's profile:
   (a) Choose $z_i \sim Multinomial(\theta_p)$
   (b) Choose $s_i \sim Multinomial(\mu_{z_i})$
   (c) For each of the W features, t_j, associated with s_i
      i. Choose $t_j \sim Multinomial(\phi_{z_i})$

Inference

- **Similar to LDA models exact inference is not possible in the model**
- **We use variational message passing (VMP) for approximate inference and parameter estimation**
  - VMP carries out variational inference using local computations and message passing on the graphical model.
  - We implemented our model using Infer.Net library which provides implementation for different inference algorithms including VMP

Contextual Recommendation Algorithm

- **The uniform contextual modeling recommendation framework enables us to systematically optimize the recommendations for the given context**
  - Given a user's profile P, and context Q, the model computes the recommendation score for each item I as $p(I \mid P, Q)$ and ranks items based on these scores.
- **The high dimensionality of this feature space can be a problem for many of the popular context-aware recommendation methods**
  - One of the advantages of our recommendation method is that the associations between features are captured by the latent factors, and therefore, are integral part of the trained model.

Datasets

- **Art of the mix dataset containing more than 28,963 user compiled playlists.**
  - The dataset contains 218,261 songs.
  - Top tags were retrieved for 73,045 songs in the dataset.
  - 8,769 playlists, each containing more than 10 songs that have tags. The selected playlists contain 86,262 unique songs.
- **CiteULike dataset**
  - This dataset consists of the set of articles posted by each user, and the tags that the user used to post it
  - Data pruned to contain only users with more than 4 postings and articles with frequency of at least 4
  - The number of users and articles after pruning: 11,533 and 67,335, respectively

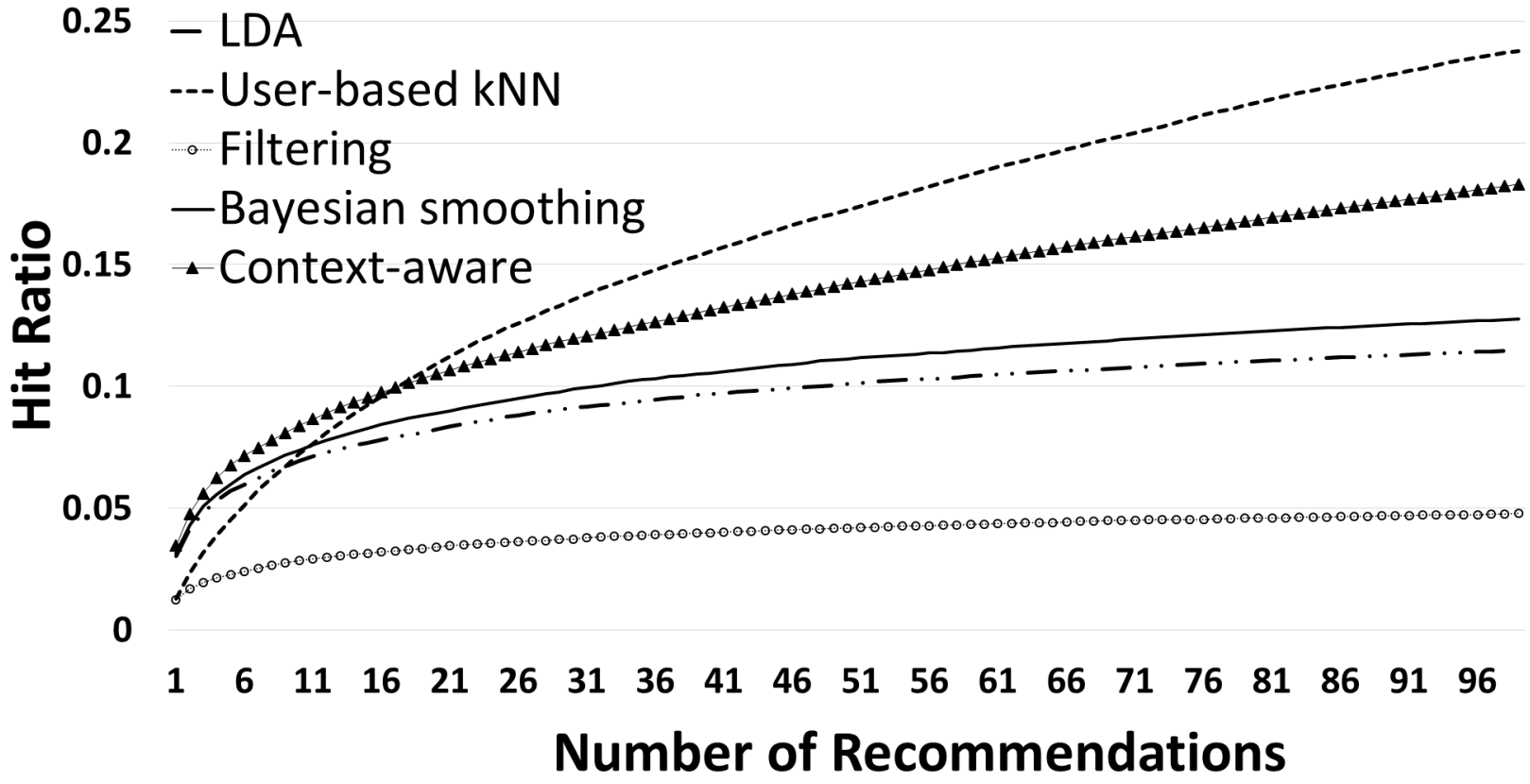
Sample Topics for the Playlists Dataset

Topic#1	Topic#2	Topic#3	Topic#4	Topic#5	Topic#6
Electronic	Alternative	Soul	Rock	Metal	Rock
Electronica	Indie	Rnb	Comedy	Rock	Britpop
Alternative	Rock	Pop	Alternative	Industrial	British
Experimental	Shoegaze	Dance	Funny	Hardcore	Indie
Ambient	Upbeat	Electronica	Punk	Alternative	Jazz
Indie	Nostalgia	Rap	Pop	Metalcore	Swing
Rock	Amazing	Funk	Indie	Dance	Alternative
Chillout	Pop	Chillout	Silly	German	Oldies
Psychedelic	Punk	Jazz	Quirky	Pop	Beatles

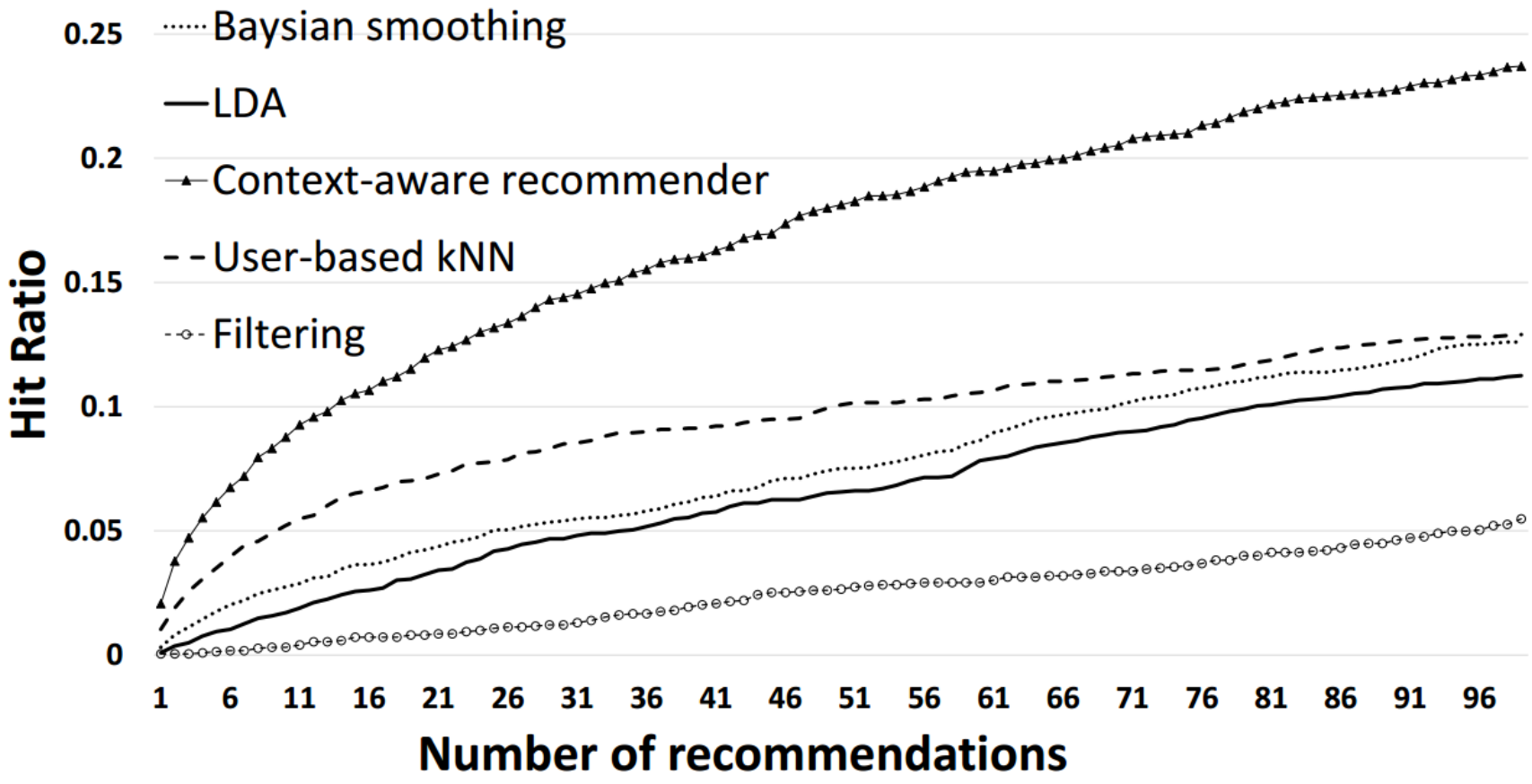
Experiments

- **5-fold cross validation experiment**
  - In each run 80% of the songs in each playlist were selected for training the model and the remaining 20% were selected for testing and were removed from the users profiles.
  - Given the users queries, competing algorithms provide a ranked list of songs.
- **Need user profiles, songs descriptions, and user queries**
  - Simulating the queries

Hit ratio for Different Number of Recommendations (CiteULike Dataset)



Hit ratio for Different Number of Recommendations (Playlists Dataset)



Conclusions

Conclusions

- **Incorporating context in recommendation generation *can* improve the effectiveness of recommender systems**
- **What does it take?**
  - **In representational models:** careful selection of relevant contextual attributes for the specific domain (the classic knowledge engineering task) & effective (but ad hoc) ways of dealing with the qualification problem
  - **In Interactional Models:** effective methods for extraction of contextual cues from user behavior & ways of coping with domains that don't lend themselves to user interactions
- **Work on Interactional Models Suggests:**
  - observable behavior is "conditioned" on the underlying context
  - The context can be inferred (and predicted) effectively in certain kinds of applications
  - The integration of semantic knowledge and user activity can be particularly effective in contextual user modeling

Still many unanswered questions ….

?