Context-Aware Recommender Systems

Gediminas Adomavicius, Bamshad Mobasher,
Francesco Ricci, and Alex Tuzhilin

■ *Context-aware recommender systems (CARS) generate more relevant recommendations by adapting them to the specific contextual situation of the user. This article explores how contextual information can be used to create intelligent and useful recommender systems. It provides an overview of the multifaceted notion of context, discusses several approaches for incorporating contextual information in the recommendation process, and illustrates the usage of such approaches in several application areas where different types of contexts are exploited. The article concludes by discussing the challenges and future research directions for context-aware recommender systems.*

Traditional recommender systems, such as those based on content-based and collaborative filtering, tend to use fairly simple user models. For example, user-based collaborative filtering generally models the user as a vector of item ratings. As additional observations are made about users' preferences, the user models are extended, and the full collection of user preferences is used to generate recommendations or make predictions. This approach, therefore, ignores the notion of "situated actions" (Suchman 1987), the fact that users interact with the system within a particular "context" and that preferences for items within one context may be different from those in another context.

In many application domains, a context-independent representation may lose predictive power because potentially useful information from multiple contexts is aggregated. For example, when a user is buying books, the preferences the user expresses in one context, such as "books for my children," may be of no predictive value when the user seeks recommendations in a different context, such as "work-related books." The ideal context-aware recommendation system would, therefore, be able reliably to label each user action with an appropriate context and effectively tailor the system output to the user in that given context.

The concept of "context" has been studied extensively in several areas of computing and other disciplines. For example, Bazire and Brezillon (2005) examine and compare some 150 different definitions of context from a number of different fields and conclude that the multifaceted nature of the concept makes it difficult to find a unifying definition: "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?" Although Bazire and Brezillon (2005) do not settle on a specific definition, the questions they raise do permeate virtually all domains in which context awareness is necessary or desired, including recommender systems.

Observing the various uses of context, Dourish (2004) has distinguished between two different views of context: the *representational* view and the *interactional* view. The representational view makes four key assumptions: context is a form of information, it is delineable, it is stable, and it is independent from the underlying activity. In this view, context can be described using a set of observable attributes that are known a priori. Furthermore, the structure of these contextual attributes does not change over time. The interactional view of context, according to Dourish, takes a different stance on the key assumptions made by the representational view. In the interactional view, the scope of contextual features is defined dynamically, and it is occasioned rather than static. Rather than assuming that context acts as a set of conditions under which an activity occurs, this view assumes a cyclical relationship between context and activity, where the activity gives rise to context and the context influences activity.

Our goal in this article is not the exploration of the notion of context in general, but rather the examination of how context can be defined and used in recommender systems in order to create more intelligent and useful recommendations. We begin by presenting our own approach to defining context in recommender systems and discuss different paradigms of incorporating it into the recommendation process. We also highlight several applications where context-aware recommender systems have been used effectively, from travel guides to information search and music recommendation. Finally, we identify several areas where challenges in integrating context into recommender systems remain and point to some future research directions.

Context in Recommender Systems

In our classification of context in recommender systems, we follow many previous approaches by assuming the existence of certain *contextual factors*, such as time, location, and the purchasing purpose, that identify the context in which recom-

mendations are provided. We assume that each of these contextual factors can have a structure; the Time factor, for example, can be defined in terms of seconds, minutes, hours, days, months, and years. The classification of context that we propose in this article is based on the following two aspects of contextual factors: what a recommender system may know about these contextual factors, and how contextual factors change over time. We will explain each of these two dimensions in greater detail later in this article.

What a Recommender System Knows About Contextual Factors

A recommender system can have different types of knowledge, which may include the exact list of all the relevant factors, their structure, and their values, about the contextual factors. Depending on what exactly the system knows (that is, what is being observed), we can classify the knowledge of a recommender system about the contextual factors into three categories: fully observable, partially observable, and unobservable.

Fully observable: The contextual factors relevant to the application, as well as their structure and their values at the time when recommendations are made, are known *explicitly*. For example, in case of recommending a purchase of a certain product, such as a shirt, the recommender system may know that only the Time, PurchasingPurpose, and ShoppingCompanion factors matter in this application. Further, the recommender system may know the structure of all these three contextual factors, such as having categories of weekday, weekend, and holiday for Time. Further, the recommender system may also know the values of the contextual factors at the recommendation time (for example, when this purchase is made, with whom, and for whom).

Partially observable: Only some of the information about the contextual factors, as described above, is known explicitly. For example, the recommender system may know all the contextual factors, such as Time, PurchasingPurpose, and ShoppingCompanion, but not their structure. Note that there can possibly be different levels of "partial observability." In this article we do not differentiate between them and group various cases of partially observable knowledge into this general category.

Unobservable: No information about contextual factors is explicitly available to the recommender system, and it makes recommendations by utilizing only the *latent* knowledge of context in an implicit manner. For example, the recommender system may build a latent predictive model, such as hierarchical linear or hidden Markov models, to estimate unknown ratings, where unobservable context is modeled using latent variables.

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

Figure 1. Contextual Information Dimensions.

How Contextual Factors Change over Time

Depending on whether contextual factors change over time or not, we have the following two categories: static and dynamic.

Static: The relevant contextual factors and their structure remains the same (*stable*) over time. For example, in case of recommending a purchase of a certain product, such as a shirt, we can include the contextual factors of Time, PurchasingPurpose, ShoppingCompanion and only them during the entire lifespan of the purchasing recommendation application.

Dynamic: This is the case when the contextual factors change in some way. For example, the recommender system (or the system designer) may realize over time that the ShoppingCompanion factor is no longer relevant for purchasing recommendations and may decide to drop it. Furthermore, the structure of some of the contextual factors can change over time (for example, new categories can be added to the PurchasingPurpose contextual factor over time).

These two dimensions give rise to the 3 x 2 diagram presented in figure 1. The two extremes in figure 1 are the cases of the static, fully observable contextual factors and the dynamic, unobservable contextual factors. The former case pertains to the situation when all the contextual factors are known and do not change over time in any way. This means that the system knows *everything* about the context. This corresponds to the case called by Dourish (2004) the *representational view*. The other extreme is the case of the dynamic unobservable contextual factors when *nothing* is known about the context. Between these two extremes lie four intermediate cases, such as dynamic fully or partially observable contextual factors and static unobservable or partially observable factors.

One of these cases is defined by the static unobservable contextual information, when the structure of one or several unobserved latent contextual factors does not change over time. Since this latent structure is stable, it can be modeled with latent variables, and the unobserved contextual information can be learned using some of the machine-learning methods, such as matrix factorization (Koren 2008), probabilistic latent semantic analysis (PLSA), or hierarchical linear models (HLMs). Regarding the dynamic observable or partially observable cases in figure 1, they are related to the interactional view of the context. As an example of the dynamic observable case, Mahmood, Ricci, and Venturini (2010) present a recommender system that adapts the dialogue to the interaction context. This is modeled by a set of dynamic contextual factors representing, for instance, whether the user provided certain information or acted on the recommendations (for example, put an item into the shopping cart). In a dynamic way, step by step, the system adapts the interaction considering a selection of these factors depending on the state of the interaction. An example of the static partially observable contextual case is presented in Palmisano, Tuzhilin, and Gorgoglione (2008), where the contextual information is defined using a Bayesian Network (BN). In this BN, the observed context corresponds to the external layers of the network and the unobserved latent one to the middle layers of the BN.

Further, some of the unobserved parameters of the BN were learned from the data using standard machine-learning methods (Palmisano, Tuzhilin, and Gorgoglione 2008).

Dynamic context may change based on passive observations or explicit user feedback. For example, the latter is the case in conversational recommender systems. In standard conversational systems, user feedback is used to iteratively refine the user profile (or the initial user query) resulting in more appropriate recommendations. In a context-aware conversational system (for example, Baltrunas et al. [2011]), the user feedback may also be used to iteratively modify contextual factors and not just user profiles. For example, in the course of conversation with a user, a restaurant recommender may determine that the user is on a romantic date. This observation, in turn, may result in filtering out restaurants that tend to be noisy or without an adequate wine selection. Conversational recommender systems have been an active area of research. The iterative refinement of context in such systems will introduce a new set of research questions that will require further investigation.

Most of the prior work on CARS follows the static, fully observable (or representational) view of context (Palmisano, Tuzhilin, and Gorgoglione 2008). Some notable exceptions, in addition to those already mentioned above, include work by Jin, Zhou, and Mobasher (2005) and Anand and Mobasher (2007), which examined approaches to modeling context based on the interactional view, primarily in the realm of web personalization. In much of the subsequent discussion we will focus on the representational view of context-aware recommendation.

One of the main challenges of the representational view of context is the well-known domain engineering problem typically associated with knowledge-based systems. By their nature, representational context-aware systems are domain specific, and the set of contextual variables that are relevant for the computational task (for example, generating recommendations) must be specified as part of the design of the system. It is often difficult to determine a priori what all relevant contextual factors are. For example, in a restaurant recommendation domain, it may be difficult to determine whether the user's attire would be relevant to his or her choice of restaurant. Another challenge of representational systems is that the information related to relevant contextual factors must be determined as part of the data collection, and such historical contextual information is often not available when designing the system. For example, in movie recommendation domains, it is rarely the case that the users would have the capability (or desire) to rate the same item multiple times under different contexts.

The approaches that assume unobservable or dynamic context also have their own challenges. These challenges are primarily related to determining the efficacy of models (such as latent factors) that may characterize different contexts. In addition, in such context-aware systems, contextual information must be extracted from users' ongoing activities, and changes in contexts must be recorded. Without an explicit model or representation of context, it is often difficult or impractical to determine what characterizes a specific context in a user's activity. In many applications, the characterization of contextual cues may have to be accomplished based on domain-specific heuristics. This is even a bigger challenge if the context is assumed to be dynamic. In that case, the system must be able to determine when to "switch" to a different underlying context model.

Next we describe how context can be incorporated into the recommendation process, focusing mainly on the static, fully observable view of context, as it constitutes a predominant paradigm in CARS so far, as mentioned earlier.

Representing and Modeling Context

Traditionally, the recommendation problem has been viewed as a prediction problem in which, given a user profile and a target item, the recommender system's task is to predict that user's rating for that item, reflecting the degree of user's preference for that item. Specifically, a recommender system tries to estimate a rating function:

$$R : Users \times Items \rightarrow Ratings \qquad (1)$$

that maps user-item pairs to an ordered set of rating values. Note that R can be viewed as a general-purpose *utility* (or preference) measure for user-item pairs; however, due to the popularity of rating-based utility models in recommender systems research, we use rating-based examples in this article. The view of recommendation as a prediction problem comes from the fact that R is a partial function: the ratings for all user-item pairs are not known and, therefore, must be inferred. For the subset of user-item pairs for which the rating values are known, those rating values are obtained explicitly from users (for example, movie ratings) or implicitly (for example, based on observations of user behavior, such as purchase transactions). Once an initial set of ratings have been specified, a recommender system tries to estimate the rating values for user-item pairs that have not yet been rated by the users. We call such traditional recommender systems *two-dimensional* (2D) since they consider only the Users and Items dimensions as input in the recommendation process. Note that the representation of users and items in recommender systems is not limited by their IDs (as done by many collaborative filtering algorithms) but can take into account com-

prehensive user profiles and item content features.

In contrast to the traditional model, context-aware recommender systems try to incorporate or utilize additional evidence (beyond information about users and items) to estimate user preferences on unseen items. When such contextual evidence can be incorporated as part of the input to the recommender system, the rating function can be viewed as "multidimensional":

$$R : Users \times Items \times Contexts \rightarrow Ratings \qquad (2)$$

where *Contexts* represents a set of factors that further delineate the conditions under which the user-item pair is assigned a particular rating. The underlying assumption of this extended model is that user preferences for items are not only a function of items themselves, but also a function of the context in which items are being considered. It should be noted that the above mapping is not unique to the static, fully observable view of context, but, for instance — even in the unobservable case — the ratings may depend on a set of latent factors derived from observations of user activity and used implicitly as part of the estimation of user preferences (as mentioned previously).

In the static, fully observable view, however, which is the primary focus of this article, *Contexts* represents a set of explicit variables that model contextual factors in the underlying domain (for example, time, location, surroundings, device, occasion, and so on). Regardless of the context representation, context-aware recommenders must be able to obtain contextual information that corresponds to user's activity (for example, making a purchase or rating an item). On the one hand, this contextual information is potentially needed as part of the learning and modeling process (such as discovering rules, segmenting users, or building regression models). On the other hand, for a given target user and a target item, the system must be able to identify the values of specific contextual variables as part of the user's ongoing interaction with the system.

The contextual information can be obtained in many different ways. It can be obtained explicitly or implicitly. Explicit contextual information may be obtained from users themselves or from sensors designed to measure specific physical or environmental information. For instance, the application may ask an individual looking for a restaurant recommendation to specify whether he or she is on a romantic date or out with coworkers for a business dinner. If the restaurant recommender is on a mobile device, additional contextual information can be obtained through GPS or other sensors about the location, time, and weather conditions. In some cases, however, contextual information must be derived or inferred from other observed data. For example, an e-commerce system may

attempt, using previously learned models of user behavior, to distinguish between the context in which a user is buying a gift for a spouse and the context of purchasing a work-related book. Approaches to implicitly infer contextual information typically require building predictive models from historical data (Palmisano, Tuzhilin, and Gorgoglione 2008).

Paradigms for Using Contextual Information

Traditional recommender systems are built based on the partial knowledge of user preferences, that is, user preferences for some (often limited) set of items, and the input data for traditional recommender systems is typically based on the records of the form *<user, item, rating>*. In contrast, context-aware recommender systems are built based on the knowledge of partial contextual user preferences and typically deal with data records of the form *<user, item, context, rating>*, where each specific record includes not only how much a given user liked a specific item, but also the contextual information in which the item was consumed by this user (for example, context = Saturday). Furthermore, unlike the traditional recommendation process that does not take context into account, the information about the current context *c* can be used in various stages of the recommendation process, leading to several different approaches to context-aware recommender systems.

In particular, from the algorithmic perspective, while all the context-aware recommendation approaches would work with the data of the form $U \times I \times C \times R$, where C is an additional contextual dimension, and produce a list of contextual recommendations $i_1, i_2, i_3 \ldots$ for each user *u*, the context-aware recommendation process can take one of the following three forms, based on how the contextual information is used, as shown in figure 2 (Adomavicius and Tuzhilin 2011): contextual prefiltering, contextual postfiltering, or contextual modeling.

Contextual prefiltering (or contextualization of recommendation input). In this paradigm information about the current context *c* is used for selecting only the relevant set of data, and ratings are predicted using any traditional 2D recommender system on the selected data.

Contextual postfiltering (or contextualization of recommendation output). In this paradigm contextual information is initially ignored, and the ratings are predicted using any traditional 2D recommender system on the entire data. Then, the resulting set of recommendations is adjusted (contextualized) for each user using the contextual information.

Contextual modeling (or contextualization of rec-

(a) Contextual Prefiltering

(b) Contextual Postfiltering

(c) Contextual Modeling

| Data |
| $U \times I \times C \times R$ |

c

| Contextualized Data |
| $U \times I \times R$ |

| 2D Recomender |
| $U \times I \to R$ |

u

| Contextual Recommendations |
| $i_1, i_2, i_3...$ |

| Data |
| $U \times I \times C \times R$ |

| 2D Recomender |
| $U \times I \to R$ |

u

| Recommendations |
| $i_1, i_2, i_3...$ |

c

| Contextual Recommendations |
| $i_1, i_2, i_3...$ |

| Data |
| $U \times I \times C \times R$ |

| MD Recomender |
| $U \times I \times C \to R$ |

u

c

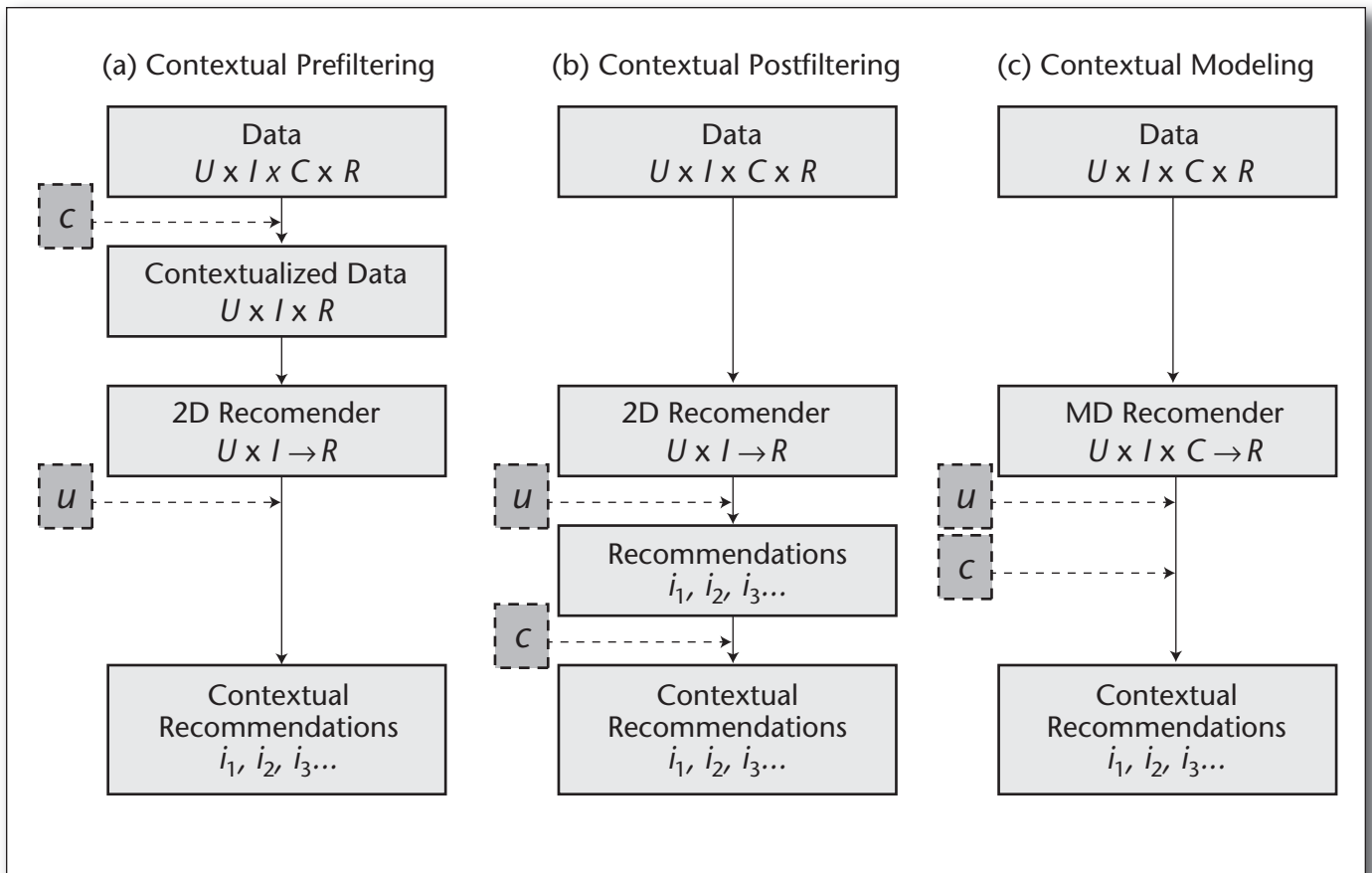| Contextual Recommendations |
| $i_1, i_2, i_3...$ |

Figure 2. Paradigms for Incorporating Context in Recommender Systems.

ommendation function). In this paradigm contextual information is used directly in the modeling technique as part of the rating estimation.

We note that adapting a recommendation to context is complementary to the adaptation to the user preferences (personalization). Personalization can be implemented with the same general approaches mentioned above. Note that, in principle, a recommender system can be context aware without being personalized and vice versa, although most CARS perform both kind of adaptations.

Contextual Prefiltering

As shown in figure 2a, the contextual prefiltering approach uses contextual information to select the most relevant 2D (*Users* x *Items*) data for generating recommendations. One major advantage of this approach is that it allows deployment of any of the numerous traditional recommendation techniques previously proposed in the literature (Adomavicius and Tuzhilin 2005). In particular, when using this approach, context *c* essentially serves as a query (or

a filter) for selecting relevant rating data. An example of a contextual data filter for a movie recommender system would be: if a person wants to see a movie on Saturday, only the Saturday rating data is used to recommend movies. Note that this example represents an *exact* prefilter because the data was filtered using exactly the specified context.

However, the exact context sometimes can be too narrow. Consider, for example, the context of watching a movie with a girlfriend in a movie theater on Saturday or, more formally, *c* = (*Girlfriend, Theater, Saturday*). Using this exact context as a data-filtering query may be problematic, because certain aspects of the overly specific context may not be significant, and the exact context may not have enough data for accurate rating prediction. To address this issue, Adomavicius et al. (2005) introduced *generalized* prefiltering that generalizes data filtering context $c = (c_1, ..., c_k)$ with less specific context $c' = (c'_1, ..., c'_k)$ such that $c_i \to c'_i$ for every $i = 1, ..., k$ in the corresponding context hierarchy (for example, *Saturday* → *Weekend*). Then, *c'* is used

instead of *c* to obtain contextualized ratings data. Note that the prefiltering approach is related to the task of building local models in machine learning and data mining (Alpaydin 2004). Rather than building the global rating estimation model utilizing all the available ratings, the prefiltering approach builds a local rating estimation model that uses only the ratings pertaining to the user-specified criteria in which a recommendation is made (for example, Saturday).

It is important to know if a local model generated by the prefiltering approach outperforms the global model of the traditional 2D technique, where all the information associated with the contextual dimensions is simply ignored. For example, it is possible that it is better to use the contextual prefiltering to recommend movies to see in the movie theaters on weekends, but use the traditional 2D technique for movies to see at home on VCRs. The trade-off of having more relevant data for calculating an unknown rating based only on the ratings with the same or similar context versus having fewer such data points used in this calculation belonging to a particular segment (that is, the sparsity effect) explains why the prefiltering recommendation method can outperform traditional 2D recommendation techniques in some contexts and underperform on others. Which of these two trends dominates for a particular context may depend on the application domain and on the specifics of the available data. Based on this observation, Adomavicius et al. (2005) propose to combine a number of contextual prefilters with the traditional 2D technique (that is, as a default filter, where no filtering is done).

Baltrunas and Ricci (2009) take a somewhat different approach to contextual prefiltering in proposing and evaluating the benefits of the item-splitting technique, where each item is split into several fictitious items based on the different contexts in which these items can be consumed. Similarly to the item-splitting idea, Baltrunas and Amatriain (2009) introduce the idea of microprofiling, which splits the user profile into several (possibly overlapping) subprofiles, each representing the given user in a particular context. The predictions are done using these contextual microprofiles instead of a single user model.

Contextual Postfiltering

As shown in figure 2b, the contextual postfiltering approach ignores context information in the input data when generating recommendations, that is, when generating the ranked list of all candidate items from which any number of top-*N* recommendations can be made. Instead, the contextual postfiltering approach uses contextual information to adjust the obtained recommendation list for each user. The recommendation list adjustments can be made by: (1) filtering out recommendations that are irrelevant in a given context, or (2) adjusting the ranking of recommendations in the list. For example, in a movie recommendation application, if a person wants to see a movie on a weekend, and on weekends he or she only watches comedies, the system can filter out all noncomedies from the recommended list.

As with many recommendation techniques, the contextual postfiltering approaches can be classified into heuristic and model-based techniques. Heuristic postfiltering approaches focus on finding common item characteristics (attributes) for a given user in a given context (for example, preferred actors to watch in a given context) and then use these attributes to adjust the recommendations. In contrast, model-based postfiltering approaches can build predictive models that calculate the probability with which the user chooses a certain type of item in a given context (for example, likelihood of choosing movies of a certain genre in a given context) and then use this probability to adjust the recommendations.

Panniello et al. (2009) provide an experimental comparison of the exact (that is, nongeneralized) prefiltering method versus two different postfiltering methods, called Weight and Filter, using several real-world e-commerce data sets. The Weight postfiltering method reorders the recommended items by weighting the predicted rating with the probability of relevance in that specific context, and the Filter postfiltering method filters out recommended items that have small probability of relevance in the specific context. Interestingly, the empirical results show that the Weight postfiltering method dominates the exact prefiltering, which in turn dominates the Filter method, thus, indicating that the best approach to use (pre- or postfiltering) really depends on a given application.

As was the case with contextual prefiltering, a major advantage of the contextual postfiltering approach is that it allows using any traditional recommendation technique previously proposed in the literature (Adomavicius and Tuzhilin 2005). Also, similarly to the contextual prefiltering approaches, incorporating context-generalization techniques into postfiltering methods constitutes an interesting issue for future research.

Contextual Modeling

As shown in figure 2c, the contextual modeling approach uses contextual information directly in the recommendation function as an explicit predictor of a user's rating for an item and, thus, gives rise to truly multidimensional recommendation functions representing either predictive models (such as decision trees, regressions, and so on) or heuristic calculations that incorporate contextual

information in addition to the user and item data. A number of recommendation algorithms based on a variety of heuristics as well as predictive modeling techniques have been developed over the last 10 to 15 years (Adomavicius and Tuzhilin 2005), and some of these techniques can be extended from the 2D to the multidimensional recommendation settings.

One example of recent developments in 2D recommendation algorithms is the rise of matrix factorization approaches, made popular by the recent Netflix Prize competition (Koren 2008). Karatzoglou et al. (2010) follow a similar approach for context-aware recommender systems by introducing a tensor factorization method that models the *Users* x *Items* x *Contexts* space as an *n*-dimensional tensor, the factorization of which provides a compact model for computing context-aware recommendations.

Some other 2D recommendation methods that can be directly extended to the multidimensional case include the technique proposed by Ansari, Essegaier, and Kohli (2000), which combines the information about users and items into a single hierarchical regression-based Bayesian preference model that uses Markov Chain Monte Carlo (MCMC) techniques to estimate its parameters. This approach can be directly extended to include the contextual information in a natural way by adding contextual variables and their interaction effects directly into the hierarchical model, as described in Adomavicius and Tuzhilin (2011).

In addition to possible extensions of existing 2D recommendation techniques to multiple dimensions, there have also been some new techniques developed specifically for context-aware recommender systems based on the contextual modeling paradigm. For example, Oku et al. (2006) propose to incorporate additional contextual dimensions directly into recommendation space and use machine-learning techniques to provide recommendations in a restaurant recommender system. In particular, they use the support vector machine (SVM) classification method, which views the set of liked items and the set of disliked items of a user in various contexts as two sets of vectors in an *n*-dimensional space and constructs a separating hyperplane in this space, which maximizes the separation between the two data sets. The resulting hyperplane represents a classifier for future recommendation decisions. Furthermore, Oku et al. (2006) empirically show that context-aware SVM significantly outperforms noncontextual SVM-based recommendation algorithm in terms of predictive accuracy and user's satisfaction with recommendations.

Key CARS Applications

In this section we will survey some representative CARS applications. This list obviously cannot be complete; new applications are continuously proposed, as CARS is becoming an increasingly popular topic in recommender systems research. We will mostly discuss mobile applications, especially in the travel and tourism domains, since a number of CARS applications are dealing with this scenario (Ricci 2011). As mentioned earlier, context is a broad concept; therefore, in addition to referring to the aforementioned paradigms for generating context-aware recommendations (prefiltering, postfiltering, and modeling), for better positioning of the applications we will classify them according to the exploited contextual information. We consider four broad types of context that are used by the reviewed applications (extended from Fling [2009]): physical context, social context, interaction media context, and modal context.

Physical context: representing the time, position, and activity of the user, but also the weather, light, and temperature when the recommendation is supposed to be used.

Social context: representing the presence and role of other people (either using or not using the application) around the user, and whether the user is alone or in a group when using the application.

Interaction media context: describing the device used to access the system (for example, a mobile phone or a kiosk) as well as the type of media that are browsed and personalized. The latter can be ordinary text, music, images, movies, or queries made to the recommender system.

Modal context: representing the current state of mind of the user, the user's goals, mood, experience, and cognitive capabilities.

In the rest of this section we will describe several groups of applications, focusing on particular functionalities (information search, travel guides), or recommendation methodologies (proactive, distributed, and conversational), or item types (music).

Information Search

Church et al. (2007) propose a novel interface to support multidimensional, context-sensitive mobile information search. The system integrates the user's *physical* context (temporal and location data) with preference information derived from the queries of mobile searchers with similar interests (*social*), and presents graphically, with icons on a map display, a view of the evolving search activities of this community of users (*media*). This system adopts a *postfiltering* model to support users in browsing through community search experiences, manipulating the searches of others, learning from these searches, and initiating their own. The main idea is that, instead of recommending information

explicitly requested by the user, the system becomes proactive by presenting information, in this case, by presenting searches performed by other users in similar contexts. The ultimate objective is to support a more exploratory, context-driven approach to information search.

A similar postfiltering approach is taken by Lee and Park (2007); however, it is emphasizing the importance of the current user search context (*modal*), rather then the history of previous similar search sessions. Their justification is that, on the mobile web, news services must focus on the distribution of current news content rather than on past or related news articles. It is therefore important, when personalizing news, to consider the recency and importance of news articles, which can only be assessed by relating them to the user context.

Travel Guides

Several applications of CARS deal with suggesting places of interests. Cena et al. (2006) present a tourist guide that uses context for intelligent content adaptation. Their system (UbiquiTO) is a tourist guide integrating different forms of context-related adaptation: to the media device type, to the user characteristics and preferences, to the physical context of the interaction (location and time). UbiquiTO uses a modeling approach (rule-based) to adapt the content of the provided recommendation, such as the amount / type of information / features associated with each recommendation. Also, the system developed by Park, Hong, and Cho (2007) is aimed at personalizing the selection of restaurants and uses the modeling approach (Bayesian network) for expressing the probabilistic influences of the user's contextual state and preferences on the restaurant attribute values. The user's physical contextual situation is automatically detected, including the season (for example, spring), the time of the day (for example, breakfast), the position, the weather (for example, sunny), and the temperature (for example, warm). A restaurant's recommendation score is computed as the weighted sum of the conditional probabilities of the restaurant's attribute values. More recently, Baltrunas, Ricci, and Ludwig (2011) and Baltrunas et al. (2011) have also explored a modeling approach (based on mutual information) to assess and estimate the relevance of various contextual information (physical, social, and modal) for place-of-interest (POI) recommendations by relying on subjective users' evaluations of the impact of contextual conditions on the likelihood to select a POI. The recommendation algorithm is based on a factor model, extended with parameters explicitly modeling the impact of the selected contextual conditions on the predicted rating. A snapshot from the user interface of the mobile system

implementing these techniques is shown in figure 3. The user also can decide what contextual factors the system must take into account. Moreover, some recommendations are marked with a special icon to indicate that for these the context played a major role in promoting them to the top-rank positions. An explanation, based on the contextual factor, for example, "this castle is recommended because it is not crowded today," is also provided.

Proactive and Distributed Approaches

Recent developments in wireless communication technologies are transforming many kinds of appliances (for example, electric household appliances) into computerized and networked machines. If the surrounding environment can use sensors to recognize us and our activities, then novel proactive and distributed services will be possible ("ubiquitous computing").

In this line of research, Sae-Ueng et al. (2008) focus on personalized shopping assistance by using personal behavior log analysis on ubiquitous shop space. The authors classify the possible user behaviors (*physical* context) in five classes: "standing," "viewing," "touching," "carrying," and "fitting," where the last behavior refers to a consumer using a mirror to match a certain object with his appearance. RFID sensors and a digital camera are used to detect these behaviors. The system presents recommendations using three channels: a display next to the product, a robot interacting with the user, and with music and sounds illustrating products (*modal* context). The system offers an augmented reality environment. The proposed recommendation approach is *prefiltering*; it first checks the position of the user, then it accumulates data on the consumer's actions, discovers commonalities among the items that were interesting for the user, and finally recommends personalized item information to the user, based on his behavior and current situation.

Also, the MoMa-system (Bulander et al. 2005) offers proactive recommendations using a *postfiltering* approach for matching order specifications with offers. When creating an order, the client application will automatically fill in the appropriate *physical* context and profile parameters, for example, "location" and "weather," so that, for example, the facility should not be too far away from the current location of the user and beer gardens should not be recommended if it is raining. On the other side, advertisers' suppliers put offers into the MoMa-system. These offers are also formulated according to the catalogue. When the system detects a pair of context matching order and offer, the end user is notified, in the preferred manner (for example, SMS, email). At this point, the user can decide whether to contact the advertiser to accept the offer.
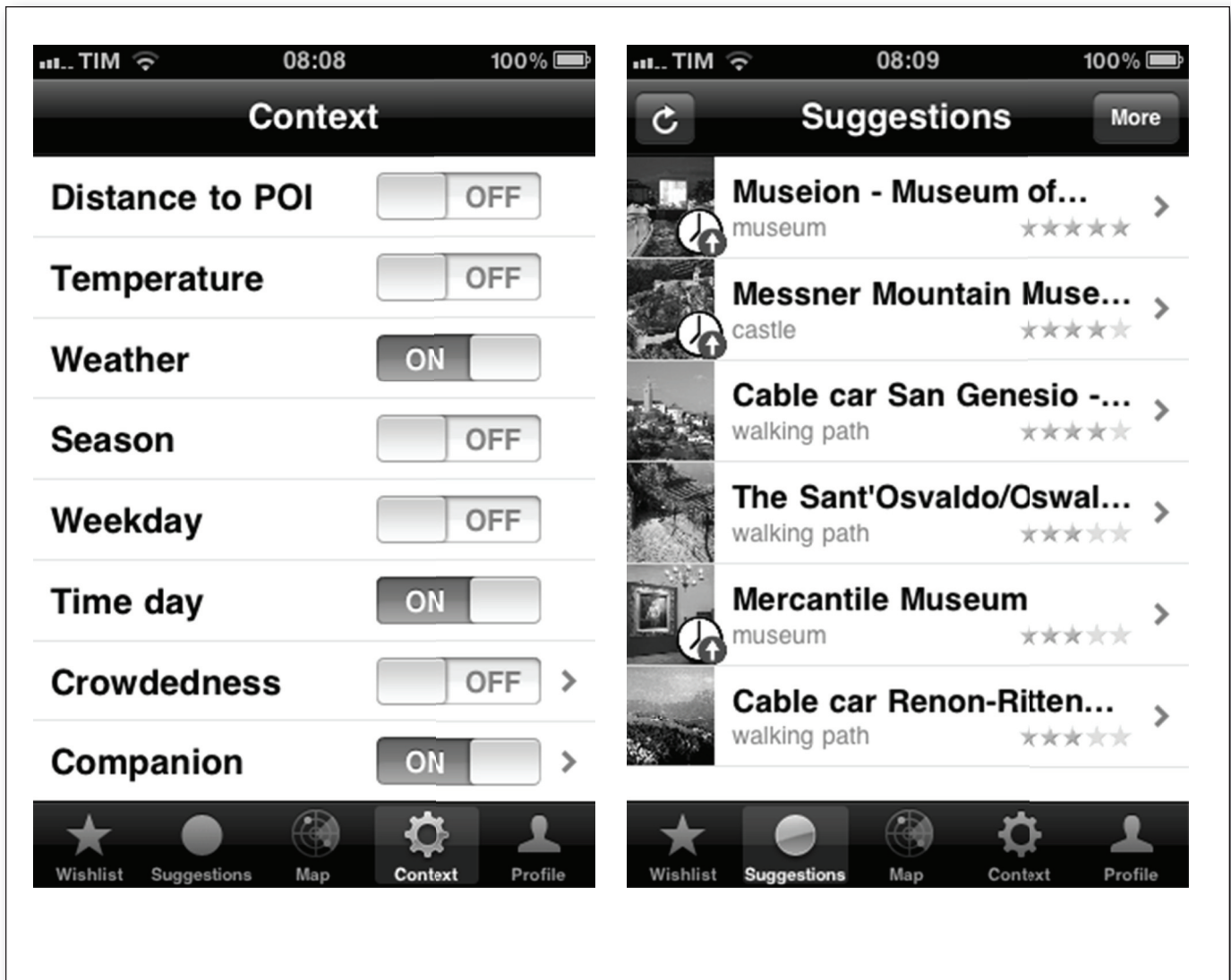
Figure 3. GUIs for Context Specification and Suggestions.

Finally, Schifanella et al. (2008) develop Mob-Hinter, a context-dependent *distributed model*, where a user device can directly connect to other mobile devices that are in *physical* proximity through ad hoc connections, hence relying on a very limited portion of the users' *community* and just on a subset of all available data (*prefiltering*). The relationships between users are modeled with a similarity graph. MobHinter allows a mobile device to identify the affinity network neighbors from random ad hoc communications. The collected information is then used to incrementally refine locally calculated predictions, with no need of interacting with a remote server or accessing the Internet. Recommendations are then computed using the information (such as ratings) available from the discovered neighbors.

Conversational/Adaptive Approaches

In conversational applications, the recommender is using the interaction state and the data collected by observing the human/computer interactions, in order to personalize the interaction workflow, for example, recommending alternative web pages or actions (*interaction media* context). The goal is to support more effective interactions, that is, increasing the probability to end up in relevant pages or products.

In this line of research, Smyth and Cotter (2004) proposed a solution for minimizing the time the

user spends in navigating to content of a mobile portal, given his or her current navigational context. They developed a *modeling* solution reducing the click-distance of the content items, which a given user is likely to be interested in, by "promoting" these items to higher positions within the portal menu structure. They used a probabilistic model of user navigation to predict the likelihood that some menu option will be selected, given that the user is currently in a particular menu, based on his or her past navigation history. Hence, when a user arrives at a menu page, instead of displaying the default options, the system computes the options that are most likely to be accessed by the user from that position (context).

Another example of adaptation of the *interaction media* context is offered by Mahmood, Ricci, and Venturini (2010). The authors describe the effects and advantages of a novel recommendation methodology based on Markov decision processes and reinforcement learning that allows conversational systems to autonomously improve an initial interaction strategy in order to learn a more effective and efficient one. They applied and tested their approach within an online travel recommender system. They showed that the learned interaction strategy adapts its actions to the served users and, more importantly, that it is able to assist online users in acquiring their goals more efficiently than the initial strategy. This approach can be used by a system designer to understand the limitations of an existing interaction design and guide the designer in the adoption of a new one that is capable to improve customer relationships, web site usage, and conversion rate.

Music Recommender Systems

The majority of available music recommender systems are based on collaborative filtering and do not take into consideration contextual conditions that may influence the user's information or entertainment needs. Recently there has been an emerging interest in contextual, or situational, music recommendation. These are examples of *media* context adaptation.

One example of contextual music recommenders is offered by Lifetrak (Reddy and Mascia 2006). It is a system that generates a playlist from the user's music library considering the current user's *physical* context. Lifetrak considers: location, time of day, day of week, noise or traffic level, temperature, and weather. The context is obtained using sensors on the user mobile device and RSS feeds (for weather and traffic conditions). The songs in the user's library are supposed to be tagged by the user with tags representing the values of possible contextual conditions that suit the song. Given the library of tagged songs, the system determines the current context of the user and, using a *postfiltering*

approach, ranks all the songs in the library based on their match with the current context.

Another notable example is illustrated in Park, Yoo, and Cho (2006). The context model used in this work is again *physical*, and includes temperature, humidity, noise, illuminance, current weather, weather forecast, season, and time of day. Interestingly, here the authors use a *modeling* approach to link two types of context, physical and modal. A Bayesian network is used to infer the emotional state of the user (depressed, content, exuberant, or anxious/frantic), that is, the *modal* context. Then, in order to compute the recommendations and to relate music to the emotional state classes, the users must explicitly express their preferences for each music attribute (for example, genre) in all possible context dimensions. Given this information and the values of current context parameters, the system can infer the current emotional state of the user and, using a simple *postfiltering* approach, select the relevant tracks.

In Jones et al. (2008), the authors, instead of adapting the music to the context, use the music to influence the *physical* context, that is, exploit the practice of listening to music while being mobile to provide navigation cues. Their system adapts volume and spatial balance of played music to lead users to destinations. The authors describe a pocket PC prototype used in a small field study with 10 users. Eighty-six percent of the users successfully completed the navigation task. This work shows another potential use of music in context-aware services.

The user's mood is a contextual factor that has been frequently used in music recommendations, that is, how the user feels in the active/passive and sad/happy dimensions. The emotional context is also appealing because it can be used to establish a bridge between music items and items from other different domains, and perform *cross-media* recommendations (Berkovsky, Kuflik, and Ricci 2007). One example concerns text and is a system recommending music while the user is reading web documents (Cai et al. 2007). In order to match music with the content of a web page, both items are represented as text documents, where a music track is described by its lyrics and online reviews. Then music and web pages are mapped to a common scale of emotions using a generative model, assuming that the frequencies of certain terms depend on the emotions associated to the text. Hence, after having computed the emotion distributions from the term frequencies of two resources (with respect to 40 selected emotions), the similarity of a web page and a music page is computed by comparing their distributions (*postfiltering*).

Another example deals with the *physical* context specified by a place of interest (POI), it uses a *postfiltering* approach, and the recommender selects

music that fits a POI context (Kaminskas and Ricci 2009). The motivation for such functionality is to implement more engaging tourist guide applications with adapted soundtracks as well as for recommending music to users on the move, depending on their location. The authors have used a set of emotional tags attached by a user population to both music and POIs. Moreover, they considered a set of similarity measures for tagged resources to establish a match between music tracks and POIs. The outcome of a live experiment, where the users were asked to assess the music selected by the system for a set of POIs, showed that there is a strong overlap between the users' selections and the best matching music that is recommended by the system for a POI.

Challenges and Future Research Directions

CARS is a relatively new research area having many underexplored topics and open research problems, some of them having been discussed in this article. For example, the very topic of "what is context in recommender systems" is still pretty much open to further studies. We discussed this multifaceted topic in this article and proposed a classification of contextual information in figure 1. We also pointed out that most of the work on CARS has focused on the representational view of the context and the alternative (nonrepresentational) methods have been underexplored. Therefore, it is important for the CARS community to go beyond the representational approach and study various alternatives corresponding to the remaining five cells of figure 1.

Further, the proposed classification constitutes only an initial and general high-level framework for characterizing the multifaceted topic of contextual information in recommender systems. Therefore, the CARS community needs to explore possible refinements or alternative approaches to classifying contextual information in order to deepen our understanding of various ways of incorporating context into recommender systems.

Although the representational view of context has been studied for some time now, there are still many unresolved issues and research challenges pertaining to this view, some of which were described in Adomavicius and Tuzhilin (2011). In the remainder of this section, we will focus on three topics that we think are of critical importance to achieve further advances in CARS. The first topic deals with a better understanding and further exploration of the prefiltering, postfiltering, and contextual modeling paradigms that were either not addressed or partially explored in the current literature. In particular, although some initial examples of prefiltering, postfiltering, and con-

textual modeling methods were mentioned in the paper, it is important to identify and study various *alternative* methods for all the three paradigms in a comprehensive manner.

Another important research direction is the comparison of the three CARS paradigms in order to identify strengths and weaknesses of each paradigm and to determine which one is better than the others and in which circumstances. An initial work on comparing these paradigms is presented in Panniello et al. (2009), where a few specific prefiltering and postfiltering algorithms were selected and compared with each other across various experimental conditions. Again, this constitutes only the initial work in this direction, and much more work is required to gain strong insights into comparing the three CARS paradigms.

Finally, one does not have to use the prefiltering, postfiltering, and contextual modeling paradigms in isolation and can combine the three approaches in different ways. For example, one can combine several algorithms of the same paradigm, such as combining information from several different contextual prefilters, as was done in Adomavicius et al. (2005). The rationale for having a number of different prefilters is based on the fact that, as mentioned earlier, typically there can be multiple different (and potentially relevant) generalizations of the same specific context. For example, context $c =$ (*Girlfriend, Theater, Saturday*) can be generalized to $c_1 =$ (*Friend, AnyPlace, Saturday*), $c_2 =$ (*NotAlone, Theater, AnyTime*), and a number of other contexts. Following this idea, Adomavicius et al. (2005) use prefilters based on the number of possible contexts for each rating and then combine recommendations resulting from each contextual prefilter by choosing the best-performing prefilter, using an "ensemble" of prefilters, or deploying various other combination techniques. Another interesting possibility stems from an observation that complex contextual information can be split into several components, and the utility of each piece of contextual information may be different depending on whether it is used in the prefiltering, postfiltering, or modeling stage. For example, time information (weekday versus weekend) may be most useful to prefilter relevant data, but weather information (sunny versus rainy) may be the most appropriate to use as a postfilter. Determining the utility of different contextual information with respect to different paradigms of context-aware recommender systems constitutes yet another promising direction for future research.

This is a partial list of some of the challenges and future research directions that were used only for illustrative purposes in this section. However, it is indicative of a wealth of open research problems that make CARS an interesting and productive area of future research.

Acknowledgments

References

Adomavicius, G., and Tuzhilin, A. 2011. Context-Aware Recommender Systems. In *Recommender Systems Handbook*, ed. F. Ricci, L. Rokach, B. Shapira, and P. Kantor, 217–256. Berlin: Springer Verlag.

Adomavicius, G., and Tuzhilin, A. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17(6): 734–749.

Adomavicius, G.; Sankaranarayanan, R.; Sen, S.; and Tuzhilin, A. 2005. Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach. *ACM Transactions on Information Systems* 23(1): 103–145.

Alpaydin, E. 2004. *Introduction to Machine Learning*. Cambridge, MA: The MIT Press.

Anand, S. S., and Mobasher, B. 2007. Contextual Recommendation. In *From Web to Social Web: Discovering and Deploying User and Content Profiles,* Lecture Notes in Artificial Intelligence, Volume 4737, 142–160. Berlin: Springer-Verlag.

Ansari, A.; Essegaier, S.; and Kohli, R. 2000. Internet Recommendation Systems. *Journal of Marketing Research* 37(3): 363–375.

Baltrunas, L., and Amatriain, X. 2009. Towards Time-Dependant Recommendation Based on Implicit Feedback. Paper presented at the 2009 Workshop on Context-Aware Recommender Systems. New York, 25 October.

Baltrunas, L., and Ricci, F. 2009. Context-Based Splitting of Item Ratings in Collaborative Filtering. In *Proceedings of the 2009 ACM Conference on Recommender Systems*, 245–248. New York: Association for Computing Machinery.

Baltrunas, L.; Ludwig, B.; Peer, S.; and Ricci, F. 2011. Context-Aware Places of Interest Recommendations for Mobile Users. In *Proceedings of the 14th International Conference on Human-Computer Interaction,* 531–540. Berlin: Springer.

Baltrunas, L.; Ricci, F.; and Ludwig, B. 2011. Context Relevance Assessment for Recommender Systems. In *Proceedings of the 2011 International Conference on Intelligent User Interfaces*, 287–290. New York: Association for Computing Machinery.

Bazire, M., and Brezillon, P. 2005. Understanding Context Before Using It. In *Proceedings of the 5th International Conference on Modeling and Using Context*, Lecture Notes in Artificial Intelligence, ed. A. Dey, B. Kokinov, D. Leake, and R. Turner, 113–192. Berlin: Springer.

Berkovsky, S.; Kuflik, T.; and Ricci, F. 2007. Cross-Domain Mediation in Collaborative Filtering. In *Proceedings of 11th International Conference on User Modeling*, Lecture Notes in Computer Science 4511, ed. C. Conati, K. McCoy, and G. Paliouras, 355–359. Berlin: Springer.

Bulander, R.; Decker, M.; Schiefer, G.; and Kolmel, B. 2005. Comparison of Different Approaches for Mobile Advertising. In *Proceedings of the Second IEEE International Workshop on Mobile Commerce and Services* 174–182. Los Alamitos, CA: IEEE Computer Society.

Cai, R.; Zhang, C.; Wang, C.; Zhang, L.; and Ma, W.-Y. 2007. Musicsense: Contextual Music Recommendation Using Emotional Allocation Modeling. In *Proceedings of the 15th International Conference on Multimedia*, 553–556. New York: Association for Computing Machinery

Cena, F.; Console, L.; Gena, C.; Goy, A.; Levi, G.; Modeo, S.; and Torre, I. 2006. Integrating Heterogeneous Adaptation Techniques to Build a Flexible and Usable Mobile Tourist Guide. *AI Communications* 19(4): 369–384.

Church, K.; Smyth, B.; Cotter, P.; and Bradley, K. 2007. Mobile Information Access: A Study of Emerging Search Behavior on the Mobile Internet. *ACM Transactions on the Web* 1(1): 4.

Dourish, P. 2004. What We Talk About When We Talk About Context. *Personal and Ubiquitous Computing* 8(1): 19–30.

Fling, B. 2009. *Mobile Design and Development*. New York: O'Reilly Media.

Jin, X.; Zhou, Y.; and Mobasher, B. 2005. Task-Oriented Web User Modeling for Recommendation. In *Proceedings of the 10th International Conference on User Modeling*, Lecture Notes in Computer Science 3538. Berlin: Springer.

Jones, M.; Jones, S.; Bradley, G.; Warren, N.; Bainbridge, D.; and Holmes, G. 2008. Ontrack: Dynamically Adapting Music Playback to Support Navigation. *Personal and Ubiquitous Computing* 12(7): 513–525.

Kaminskas, M., and Ricci, F. 2009. Matching Places of Interest with Music. Paper presented at the 2009 Workshop on Exploring Musical Information Spaces, Kanoni, Corfu, Greece, 1–2 October.

Karatzoglou, A.; Amatriain, X.; Baltrunas, L.; and Oliver, N. 2010. Multiverse Recommendation: N-Dimensional Tensor Factorization for Context-Aware Collaborative Filtering. In *Proceedings of the 2010 ACM Conference on Recommender Systems*, 79–86. New York: Association for Computing Machinery.

Koren, Y. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 426–434. New York: Association for Computing Machinery

Lee, H., and Park, S. J. 2007. Moners: A News Recommender for the Mobile Web. *Expert Systems with Applications* 32(1): 143–150.

Mahmood, T.; Ricci, F.; and Venturini, A. 2010. Improving Recommendation Effectiveness by Adapting the Dialogue Strategy in Online Travel Planning. *International Journal of Information Technology and Tourism* 11(4): 285–302.

Oku, K.; Nakajima, S.; Miyazaki, J.; and Uemura, S. 2006. Context-Aware SVM for Context-Dependent Information Recommendation. In *Proceedings of the 7th International Conference on Mobile Data Management*. New York: IEEE Computer Society.

Palmisano, C.; Tuzhilin, A.; and Gorgoglione, M. 2008. Using Context to Improve Predictive Modeling of Customers in Personalization Applications. *IEEE Transactions on Knowledge and Data Engineering* 20(11): 1535–1549.

Panniello, U.; Tuzhilin, A.; Gorgoglione, M.; Palmisano, C.; and Pedone, A. 2009. Experimental Comparison of Pre- Versus Post-Filtering Approaches in Context-Aware Recommender Systems. In *Proceedings of the 2009 ACM Recommender Systems Conference*, 265–268. New York: Association for Computing Machinery.
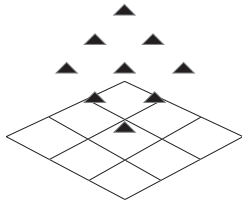
2012 AAAI Spring Symposium Call for Participation

AAAI presents the 2012 Spring Symposium Series, to be held Monday – Wednesday, March 26–28, 2012, at Stanford University. Submissions for the symposia are due on October 7, 2011. Notification of acceptance will be given by November 4, 2011. Material to be included in the technical reports of the symposium must be received by January 20, 2012. The complete Call for Participation is available at www.aaai.org/Symposia/Spring/sss12.php. Please contact AAAI at sss12@aaai.org with any questions.

Park, M.-H.; Hong, J.-H.; and Cho, S.-B. 2007. Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices. In *Proceedings of the 4th International Conference on Ubiquitous Intelligence and Computing*, 1130–1139. Berlin: Springer.

Park, H.-S.; Yoo, J.-O.; and Cho, S.-B. 2006. A Context-Aware Music Recommendation System Using Fuzzy Bayesian Networks with Utility Theory. In *Fuzzy Systems and Knowledge Discovery*. Berlin/Heidelberg: Springer, 970–979.

Reddy, S., and Mascia, J. 2006. Lifetrak: Music in Tune with Your Life. In *Proceedings of the 1st ACM International Workshop on Human-Centered Multimedia*, 25–34. New York: Association for Computing Machinery.

Ricci, F. 2011. Mobile Recommender Systems. *International Journal of Information Technology and Tourism* 12(3): 205–231.

Sae-Ueng, S.; Pinyapong, S.; Ogino, A.; and Kato, T. 2008. Personalized Shopping Assistance Service at Ubiquitous Shop Space. In *22nd International Conference on Advanced Information Networking and Applications—Workshops, 2008*, 838–843. Los Alamitos, CA: IEEE Computer Society.

Schifanella, R.; Panisson, A.; Gena, C.; and Ruffo, G. 2008. MobHinter: Epidemic Collaborative Filtering and Self-Organization in Mobile Ad-Hoc Networks. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, 27–34. New York: Association for Computing Machinery.

Smyth, B., and Cotter, P. 2004. MP[3]—Mobile Portals, Profiles and Personalization. In *Web Dynamics—Adapting to Change in Content, Size, Topology and Use*, 411–434. Berlin: Springer.

Suchman, L. 1987. *Plans and Situated Actions*. Cambridge, UK: Cambridge University Press.

Gediminas Adomavicius is an associate professor of information and decision sciences at the University of Minnesota. He received a Ph.D. in computer science from New York University. His research has been published in leading computer science and information systems journals, including *IEEE Transactions on Knowledge and Data Engineering, ACM Transactions on Information Systems, Data Mining and Knowledge Discovery, Information Systems Research, Management Information Systems Quarterly, INFORMS Journal on Computing,* and *Communications of the ACM.* He currently serves on the editorial boards of *Information Systems Research* and *INFORMS Journal on Computing.*

Bamshad Mobasher is a professor of computer science and the director of the Center for Web Intelligence at the School of Computing of DePaul University in Chicago. He has published extensively in the areas of web data mining, web personalization, recommender systems, and predictive user modeling. He has also served in leadership positions for numerous related conferences and workshops, including as cochair and steering committee member of the ACM International Conference on Recommender Systems. Mobasher has served as an associate editor for the *ACM Transactions on the Web* and on the editorial boards of several other prominent computing journals, including *User Modeling and User-Adapted Interaction,* and the *Journal of Web Semantics.*

Francesco Ricci is an associate professor of computer science at the Free University of Bozen-Bolzano, Italy. His current research interests include recommender systems, intelligent interfaces, mobile systems, machine learning, case-based reasoning, and the applications of information communication technologies to tourism. He has published more than 100 academic papers on these topics. He is on the editorial board of the *Journal of Information Technology and Tourism* and he is president of the steering committee of the ACM Conference on Recommender Systems (RecSys). He served on the program committees of several conferences, including as a program cochair of the ACM Conference on Recommender Systems, the International Conference on Case-Based Reasoning (ICCBR), and the International Conference on Information and Communication Technologies in Tourism (ENTER).

Alexander Tuzhilin is a professor of information systems and the NEC faculty fellow at the Stern School of Business, NYU. He earned his Ph.D. in computer science from the Courant Institute of Mathematical Sciences, NYU. His current research interests include data mining, recommender systems, and personalization. Tuzhilin has produced more than 100 research publications on these and other topics. He has served on the organizing and program committees of numerous computer science and information science conferences, including as a program cochair of the Third IEEE International Conference on Data Mining (ICDM) and as a conference cochair of the Third ACM Conference on Recommender Systems (RecSys). He has also served on the editorial boards of the *IEEE Transactions on Knowledge and Data Engineering,* the *ACM Transactions on Management Information Systems,* the *INFORMS Journal on Computing* (as an area editor), the *Data Mining and Knowledge Discovery Journal,* the *Electronic Commerce Research Journal,* and the *Journal of the Association of Information Systems.*