Best Usage Context Prediction for Music Tracks

Linas Baltrunas
Free University of Bozen-Bolzano,
Piazza Domenicani 3,
Bolzano, Italy
lbaltrunas@unibz.it

Marius Kaminskas
Free University of Bozen-Bolzano,
Piazza Domenicani 3,
Bolzano, Italy
mkaminskas@unibz.it

Francesco Ricci
Free University of Bozen-Bolzano,
Piazza Domenicani 3,
Bolzano, Italy
fricci@unibz.it

Lior Rokach
Ben-Gurion University of the Negev,
P.O.B. 653,
Beer-Sheva, Israel
liorrk@bgu.ac.il

Bracha Shapira
Ben-Gurion University of the Negev,
P.O.B. 653,
Beer-Sheva, Israel
bshapira@bgu.ac.il

Karl-Heinz Luke
Deutsche Telekom AG, Laboratories,
Ernst-Reuter-Platz 7, D-10587
Berlin, Germany
Karl-Heinz.Lueke@telekom.de

ABSTRACT

Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user. Often, better recommendations can be generated if the context of the recommendation is known, e.g., in a music RS, the user mood or activity. However, to adapt the recommendations to the context the dependency of the user preferences from the contextual conditions must be modeled. This requires explicit user evaluations/ratings for items in alternative contexts. In this work we investigate a novel approach for collecting and using contextually dependent ratings in recommender systems. We introduce the concept of "best context", i.e., the contextual conditions most suited for a particular item to be recommended. We designed an interface for collecting such data for music tracks. The collected data was then used to evaluate the quality of several "best context" prediction methods based on user-to-user collaborative filtering. The results, in opposition to what we expected, show that the notion of best context is user dependent. Moreover, among the approaches we tried, the best performing one uses a k-nearest neighbors classifier where the user-to- user similarity measures the agreement of two users in assigning the best context to items.

1. INTRODUCTION

Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user. Recommendations are computed by exploiting historical data about the users' preferences for items [2]. Assuming that the user's preferences does not change quickly, previously recorded observations can help in predicting future behavior. This assumption is valid only to some extent. In fact, the user's preferences can be relatively stable, but they can also be influenced by many additional and varying factors. For example, musical preferences may be influenced by the user's activity or mood.

These additional factors are commonly referred to as context and they provide additional information useful for building better predictive models [4]. Context was generally defined as *"any information that can be used to characterize the situation of an entity"* [6]. In a RS the entity that must be characterized is the user while is considering a recommendation, and the relevancy of the contextual information depends on the nature of the recommended items. For instance, in a travel planning application, the travel companion or the weather in the travel destination are known to influence the user preferences and decisions.

Context-aware recommender systems (CARSs) are becoming a wide research area and they are gaining more and more attention [4]. Various approaches have been used to incorporate contextual information into recommender systems, improving performance measures, such as: mean absolute error [5], or recall [3], or prediction accuracy [8].

Despite these promising results, CARSs are still not widely used in real-world settings. One of the reasons for this is the lack of context-dependent items' ratings data. Such data is difficult to obtain because it requires a substantial user effort, since the user must provide items' evaluations (ratings) in several different contextual conditions. So for instance, in the music domain, the user should rate a music track when she is happy or sad, when it is day or night, and so on so forth. This is clearly expensive, error prone, and these evaluations cannot be reliably assessed using sensors or other implicit indicators. In fact, to our best knowledge, nobody has substantially improved the accuracy of context- free CF predictions using implicit indications (e.g., the time spent listening a music track). This problem makes bootstrapping a context-aware system particularly difficult.

Context-dependent rating acquisition is not only expensive, sometimes it is even impossible, as users most probably have not experienced the same item in many different conditions, hence they can provide only a small part of all the

CARS-2010, September 26, 2010, Barcelona, Spain.

possible ratings. An approach to tame this problem consists of asking users to rate items in hypothetical contextual conditions, i.e., to evaluate items imagining to be in a target context [7]. For instance, asking the user to evaluate a particular music track while imagining to be sad. This approach can potentially provide all the possible in-context ratings, however, it requires to the user a high cognitive effort. Moreover, it was shown that users rates differently in real and supposed contexts [7]. Still, this type of solution could be used to acquire ratings in the contextual conditions that are easier to imagine, especially if the user was really exposed to that context (e.g., she was sad and was searching in his library a music for that situation).

In this work we present a novel approach for collecting and using context-dependent ratings in RSs. In our approach for each item and user pair we acquire a small number of subjective user's evaluations: if certain contextual conditions are appropriate for consuming the item in. We call these evaluations the "best context" for the item. Going back to the previous example, instead of asking to rate a particular music track in several target contexts (e.g., happy or sad) we ask the user to state under what contextual conditions she likes to listen to that music track. This is surely easier for the user, as it does not require her to rate the items in several different conditions and, as we have done, easier to implement and collect. This results in an association between a user, an item, and some contextual conditions. Note, that in our approach we still use a standard rating matrix, to measure how much a user generally likes an item, but in addition to that we ask to the user to indicate what is the "best context" for that rated item.

Such data can be still used to solve the main problem of CARS: what items to recommend when the user is experiencing a particular context, for instance when the user is sad or is traveling by car on a fast motorway. In fact, we propose to solve this by a process consisting of two steps: first, predicting for each item the best context, and then searching for the items whose predicted "best context" is exactly, or similar, to the target context of the recommendation query. The first step can be done off-line, and requires to predict for each item and user combination not a single rating, as in classical Collaborative Filtering (CF), but a context description, which is in our case a Boolean vector indicating whether a set of contextual conditions are true or false. Only the second step should be performed on-line and can be easily computed as a k-nearest neighbor query over the best context descriptions of the items. We observe that the best context for an item should be predicted because it is unrealistic to assume that a user has already provided this evaluation for all the items in the catalogue. A sample of these evaluations, on a subset of the items, can be used to predict the best context for the remaining items.

We also observe that our approach deals gracefully with uncovered contextual condition. In fact, if there is no item whose best context is similar to the target context then the system can rely on a default approach, i.e., a non context-dependent one. This solves a major limitation of previous CARS approaches that cannot detect if the target context matters or not for a particular recommendation.

In order to face this new type of context-dependent recommendations and evaluate the feasibility of our approach, we have designed an interface for collecting best context evaluations of users for items, in particular, for music tracks. For that purpose, we have identified the important contextual dimensions for the task. Then we asked to a set of user to assign the best context for a set of items. The collected data was then used in an off line experiment aimed at determining the quality of several predictors of the "best context" evaluation of the user for an item. Some innovative predictors have been designed, using the user-to-user collaborative filtering approach, but basing the user-to-user similarity on the way users evaluate the best context for items.

We note that to our best knowledge the problem of predicting the "best context" for an item has not been investigated so far in the literature, and the whole approach to make context-dependent recommendations based on this prediction is innovative as well. We also observe that since the "best context" is composed of several user evaluations the best context prediction problem is similar to that of predicting k multi-criteria ratings in a multi-criteria RS[1]. The main differences are that in our case there is no overall rating to predict and in multi-criteria RSs the reasoning process terminates with the prediction of the overall rating while we are using the "best context" to predict what items to recommend in contexts that may differ from the best one.

The rest of this paper is structured as follows. In section 2 we describe the process of "best context" data acquisition, in section 3 we give a formal definition of "best context" and explain how it can be used for item recommendation. Section 4 lists the different approaches for best context prediction that we have implemented. Section 5 contains the results of the comparison of the different approaches and, finally, in section 6 we draw the conclusions of our experiments and we indicate some future work.

2. DATA ACQUISITION

The rating and context elicitation was carried out as an assignment in a mandatory course on electronic commerce during 2010. The users were undergraduate students in the department of information system engineering in Ben-Gurion University. The acquisition was carried out in the faculty undergraduate computer laboratory. Before participating in the experiment, the students have been instructed during 10 minutes regarding the process and the acquisition tool.

The 82 subjects participating in the experiment constituted about 90% of the course enrollment. Each of the participants rated 100 music tracks obtained from the musicload.de site. This was done with the aid of a front-end software that was developed for this purpose. The users used a 1-5 scale to rate a song after listening to a representative and short sample of the track (see Figure 1). Once a user rated the song, she also indicated for which contextual conditions the song best fits according to her subjective opinion. We used five different contextual dimensions: Activity ('Work', 'Party', 'Relaxation', and 'Sport'), Weather ('Sunny', 'Rainy', 'Cold'), Time of Day ('Morning', 'Noon', 'Evening'), the 'Valence' Mood ('Happy','Sad') and the 'Arousal' Mood ('Calm', 'Energetic'). This means to let the user to express a preference over the contextual conditions that she considers more suited for a particular music item. For instance, a user may state that she likes listening "Born in USA, by B. Springsteen" when she is 'Energetic' or doing 'Sport' activity.

Seven out of the 100 songs were repeated in the list in the attempt to validate the consistency of the ratings provided

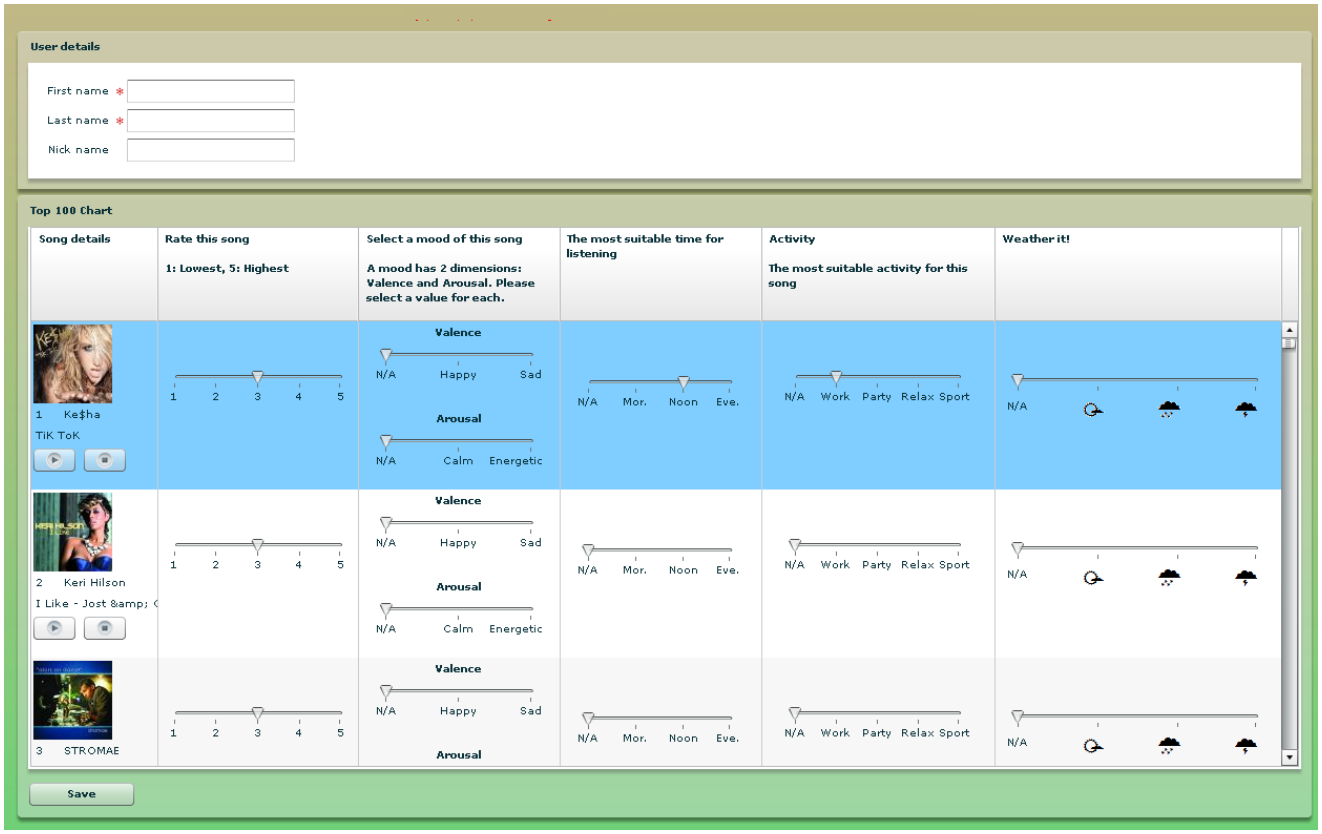

Figure 1: GUI for rating and best context input.

by each user. Eight out of 82 users who have demonstrated large inconsistency values (with negative Pearson correlation) have been removed from the database suspecting that they arbitrarily rated the items.

Then we transformed each contextual dimension in a set of Boolean dimensions, one for each distinct value of the original dimension. For instance, the Activity dimension produces four Boolean contextual dimensions corresponding to the activities: work, party, relax, sport (see Figure 1). In total 14 Boolean contextual dimensions were created.

3. BEST CONTEXT PREDICTION

Now we describe how the prediction of the best context for an item can be performed and how this is ultimately linked to context-aware item recommendations, that is, to recommend music tracks for a user when she is in a given context. In order to achieve this goal we first formulate the following task: based on user ratings and best context assigned to some music items, predict the best context that a user would assign to a particular item. This is also the best context for recommending that item to a given user.

We represent the best context that the user u assigns to the item i as the vector:

$$BC_{ui} = (BC^1_{ui}, BC^2_{ui}, ..., BC^k_{ui})$$

where k is the number of contextual dimensions and BC^1_{ui}, ..., BC^k_{ui} are their Boolean values. $BC^j_{ui} = 1$ means that the user u believes that j (for instance when his mood is "sad") is a good context for consuming item i.

Let us say that the best context vector BC_{ui} for a user u and an item i is unknown. We conjecture that we can compute a prediction, $\hat{BC}_{ui}$, by analyzing the best contexts that other users have assigned to i. That prediction can be personalized, by considering only "similar" users (neighbors) best context assignments, or non-personalized, considering all available assignments. In case of personalized predictions, each user can obtain different predictions, because, in general, every user has a specific set of neighbors. In case of a non-personalized prediction every user obtains the same result for a given item.

Being able to predict the best context for an item, we can then perform the task of recommending items for a user u in a particular context C in the following way. Given the user u, compute the prediction for the best context values $\hat{BC}_{u1}, \ldots, \hat{BC}_{um}$ for all the items in the dataset, where m is the total number of items. Then compute the similarity (based on, e.g., Hamming distance) between the current context C and the predicted best contexts $\hat{BC}_{u1}, ..., \hat{BC}_{um}$, and finally recommend the items whose best predicted context has the highest similarity to the current context. This recommendation could be extended by taking into account also a user taste for the item (i.e., the classical prediction of a recommender system). For that purpose, one can use a combination of predictors; one traditional and one based on our new approach. We observe that it is out of the scope of this short paper to fully illustrate this problem and to evaluate alternative approaches; we focus here and best context prediction.

4. PREDICTION METHODS

4.1 Averaging Best Contexts

In the baseline method, the best context for an item is predicted by averaging its best contexts vectors across all the users. This approach is non-personalized since, for each item, its prediction is the same for every user. The predicted vector $\hat{BC}_{ui}$ for the user $u \in U$ and item $i \in I$ is computed as the centroid of the best contexts assigned to i by all the users:

$$AvgBC_i = avg\{BC_{u_1 i}, ..., BC_{u_n i}\}$$

where $U = \{u_1, ..., u_n\}$ is the set of all the users that assigned best context values to the item $i = 1, \ldots, m$. This prediction assigns a probability value for each context dimension. The resulting average context vector $AvgBC_i = (AvgBC_i^1, AvgBC_i^2, ..., AvgBC_i^k)$ with real value entries is then transformed to a binary vector by assigning $\hat{BC}_{ui}^j$ to 0 or (1) if $AvgBC_i^j < 0.5 (>= 0.5)$.

4.2 Ratings Based Similarity

The second prediction method is personalized, and estimates the vector $\hat{BC}_{ui}$ using the k-nearest neighbors of u. In order to compute the k-nearest neighbors we represent each user by its rating vector $R_u = (r_{u1}, ..., r_{um})$, where r_{uj} is the rating of user u for item j, and m is the total number of items. We then use the cosine of the angle between two users' rating vectors as the user-to-user similarity measure.

The best context BC_{ui} is then computed for each context dimension separately using a standard user-based collaborative filtering method [2]:

$$\hat{BC}_{ui}^j = \overline{BC}_u^j + K \sum_{v \in U(u)} Sim(u, v)(BC_{vi}^j - \overline{BC}_v^j) \qquad (1)$$

where $\overline{BC}_u^j$ is the average value of the j^{th} context dimension evaluations of u, $U(u)$ is the set of top-k neighbors of u, $Sim(u, v)$ is the similarity of u and v, and K is a normalization factor equal to $\frac{1}{\sum_{v \in U(u)} Sim(u,v)}$. The predicted values $\hat{BC}_{ui}^j$ are then transformed into 0/1 as described above.

4.3 Best Context Vector based Similarity

The two personalized prediction methods, which are described in this subsection and in the following one, exploit the prediction formula 1 as well, but they use the best context vectors to compute the user-to-user similarity. In other words, for the user-to-user similarity computation instead of using the vector of user's ratings we consider the vectors $BC_u = (BC_{u1}, ..., BC_{um})$, where BC_{uj} is the best context vector that the user u assigned to the item j, and m is the total number of items.

So, we note that instead of having a single value for each user-item combination, which is the case in a classical CF prediction, we have here a Boolean vector, and for that reason we must define an appropriate similarity function between users (see [1] for similar definitions).

In the first approach we first compute the normalized Hamming distance between the corresponding components (items) of the vectors BC_u and BC_v:

$$d_i(BC_{ui}, BC_{vi}) = Hamming(BC_{ui}, BC_{vi})$$

This Hamming distance is equal to the number of contextual dimensions that differ in the two vectors BC_{ui} and BC_{vi} divided by their length, i.e., the number of contextual dimensions. Then, we compute the distance between two users as the quadratic mean of all the distances, along all the items:

$$d(u, v) = \sqrt{\frac{\sum_i d_i(BC_{ui}, BC_{vi})^2}{m}}$$

where m is the number of items that both users u and v have assigned best context to. Finally, the computed distance is transformed into a similarity:

$$Sim(u, v) = 1 - d(u, v)$$

Having computed the k-nearest neighbors of the active user, we estimate the best context, $\hat{BC}_{ui}$, by computing the predictions for each context dimension separately as in the previous approach (see formula 1).

4.4 Best Context based Similarity

In this third personalized best context prediction method we consider for each context dimension a specific user-to-user similarity, in contrast with the previous method where the full best context vector is used for all the best context dimensions (predictions).

Given a best context vector $BC_{ui} = (BC_{ui}^1, \ldots, BC_{ui}^k)$, where k is the number of context dimensions, then in order to predict the value BC_{ui}^j we use only the values BC_{vl}^j given by the other users v on the other items l, but only for the same contextual dimension j. In other words, we assume that the prediction for a best context dimension is not influenced by the other context dimensions.

Therefore, we form k matrices $BC^1, ..., BC^k$, each matrix representing a context dimension. For example, the entry in position (u, i) of matrix BC^1 is BC_{ui}^1, and represents the value that user u specified for item i for the 1^{st} contextual condition. We predict the context value BC_{ui}^j for user u and item i by applying again the formula 1, but in this case $Sim(u, v)$ is the Pearson correlation between the user vectors $BC_u^j = (BC_{u1}^j, \ldots, BC_{um}^j)$ and $BC_v^j = (BC_{v1}^j, \ldots, BC_{vm}^j)$.

5. EXPERIMENTAL RESULTS

The experiments were carried out in order to compare the performance of the different best context prediction methods illustrated above. For every method an off-line evaluation with 5-fold cross-validation was performed in order to compare the prediction against the true best context assigned by the users. Since the context is represented as a binary vector, to evaluate the prediction quality we measured the normalized Hamming distance between the predicted and the actual context vector. We used the data collected in the on line survey described in section 2. This comprises a set of 74 users that expressed their ratings and best context evaluations on 100 music tracks along 14 contextual dimensions.

The results show that using the non-personalized approach (sec. 4.1) the predicted best context for an item has an average normalized Hamming distance of 0.255 from the true best context vector (see Figure 2), i.e., on average 75% of the context dimensions are predicted correctly.

All the personalized methods are more effective than the

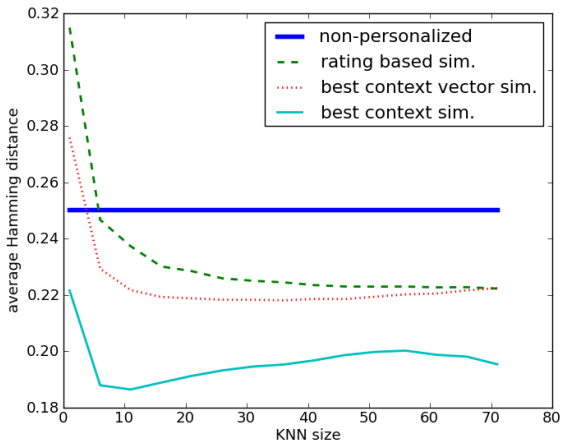

Figure 2: Comparison of different user-to-user based best context prediction approaches.

non-personalized one. The method that uses the rating-based user-to-user similarity (section 4.2) is more effective than the non-personalized one when $k > 5$. The first method that uses the best-context vectors to establish the user-to-user similarity (section 4.3) is also more effective than the non-personalized approach for $k > 3$ (see Figure 2), but it is also better than the personalized approach that uses the standard (cosine-based) similarity computed on the ratings for items.

The last method, which predicts each best context dimension using a specific user-to-user similarity (section 4.4), is clearly the most effective. The best prediction accuracy is achieved for $k = 11$ (average Hamming distance around 0.18) improving the non-personalized approach by 25%.

6. CONCLUSIONS

In this short paper we have presented and discussed a new problem for context-dependent recommender system: predicting the best context for consuming an item. We have argued that this is a common and important problem, and that the solution of this problem can also provide efficient methods for solving the more classical problem of identifying the best items to recommend to a user in a particular target context.

We have proposed three personalized best context prediction methods that are based on the classical user-to-user collaborative filtering approach. The differences rely in the user-to-user similarity computation. In the particular dataset that we used these personalized approaches have shown better performances than the non-personalized one. The first relies on the classical, ratings-based, user-to-user similarity to generate the user neighborhood and to build the best context prediction. The second and the third exploit the similarity of the users' behavior in assigning the best context for the items. We have proved that these two methods are more effective than that based on ratings. In fact the best approach consists of generating independently a prediction for each contextual dimension, considering how similar are the users in assigning that particular contextual dimension as a best context for items. In other words if a user agrees with the active user in assigning music tracks as appropriate for a particular contextual dimension, e.g., "sad", then this user becomes a neighbor and his evaluations for that contextual dimensions are used in the predictions for the active user.

There are a number of open issues that still have to be explored. First of all we want to integrate the best context prediction with the classical rating prediction performed by collaborative filtering to build a hybrid approach. Then we would like to experimentally evaluate, in live users experiments, if the context management functionality here proposed can generate more relevant recommendations and how it compares with a classical context-aware approach that exploit a data set of ratings acquired in several contextual conditions.

7. ACKNOWLEDGEMENTS

This work has been supported in part by Deutsche Telekom contract RECOM. We would like to thank Aviram Dayan, for his support i building the GUI used to acquire the best context data.

8. REFERENCES

[1] G. Adomavicius and Y. Kwon. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems*, 22(3):48–55, 2007.

[2] G. Adomavicius and A. Tuzhilin. 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, 2005.

[3] G. Adomavicius, R. Sankaranarayanan, S. Sen, A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23(1):103–145, 2005.

[4] G. Adomavicius, A. Tuzhilin. Context-aware recommender systems. In F. Ricci et al. (eds) *Recommender Systems Handbook*, 243–279, Springer, 2010.

[5] L. Baltrunas, F. Ricci Context-Dependent Items Generation in Collaborative Filtering *Workshop on Context-Aware Recommender Systems (CARS-2009) New York, NY, USA - October 25, 2009.*

[6] A.K. Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[7] C. Ono, Y. Takishima, Y. Motomura, H. Asoh. Context-Aware Preference Model Based on a Study of Difference between Real and Supposed Situation Data. *User Modeling, Adaptation, and Personalization*, 102–113, 2009.

[8] C. Palmisano, A. Tuzhilin, M. Gorgoglione. User profiling with hierarchical context: An e-retailer case study. *Modeling and Using Context, 6th International and Interdisciplinary Conference, CONTEXT 2007*, 369–383, 2007.