A Contextual Modeling Approach to Context-Aware Recommender Systems

Umberto Panniello
Polytechnic of Bari
Bari, Italy
u.panniello@poliba.it

Michele Gorgoglione
Polytechnic of Bari
Bari, Italy
m.gorgoglione@poliba.it

ABSTRACT

Methods for generating context-aware recommendations were classified into the pre-filtering, post-filtering and contextual modeling approaches. This paper proposes a novel type of contextual modeling (CM) based on the contextual neighbors approach and introduces four specific contextual neighbors methods. It compares these four types of contextual neighbors techniques to determine the best-performing alternative among them. Then it compares this best-of-breed method with the contextual pre-filtering, post-filtering and un-contextual methods to determine how well the CM approach compares with other context-aware recommendation techniques.

Categories and Subject Descriptors

H.3.3 [Information Systems]: Information Search and Retrieval

General Terms

Recommender system, Context-aware, Algorithms.

Keywords

Recommender systems, pre-filtering, post-filtering, contextual modeling.

1. INTRODUCTION

To incorporate contextual information into recommender systems (RSes), a new subfield, called CARS (Context–Aware Recommender Systems), has recently emerged. There are several approaches to incorporating contextual information into recommender systems that were previously proposed in the literature [2]. In particular, they are categorized into contextual modeling, pre-filtering and post-filtering methods [2]. Although the contextual pre- and post-filtering methods have been previously studied before, e.g. in [9], the contextual modeling methods have been little explored. Among the few attempts, [6] and [11] proposed two approaches to include context in the recommendation engine. In this paper, we study the contextual modeling (CM) methods and propose a specific type of CM that we call *contextual neighbors*. We also propose four specific types of contextual neighbors methods, called Mdl_1, Mdl_2, Mdl_3 and Mdl_4, each of them selecting contextual neighborhoods in its own way. The long-term goal of this research is to identify several different contextual modeling approaches, including our *contextual neighbor* approach, and investigate strength and weak points of each one by comparing these approaches among each other and to other approaches to CARS. Because of space limits, in this paper we only present the results of the first step of this research, i.e. the comparison of the four *contextual neighbors* methods among them, to identify the best performing one. Then we select the best-of-breed *contextual neighbors* method and compare it with the pre-, post-filtering and un-contextual methods across various experimental settings in order to determine how well the CM methods fit into the overall taxonomy of CARS methods. Future papers will present the comparison of the *contextual neighbor* approach to other existing approaches to CM, such as those proposed by [6] and [11].

CARS-2011, October 23, 2011, Chicago, Illinois, USA Copyright is held by the author/owner(s).

2. BACKGROUND ON CARS

Unlike the traditional two-dimensional (2D) recommender systems that deal with two types of entities, Users (e.g., customers) and Items (e.g., products) and try to estimate unknown ratings in the Users × Items matrix of Users and Items, context-aware recommender systems (CARS) also take into account contextual information [1]. In this paper, we follow the representational view to modeling contextual information [3] and assume that the context is defined with a predefined set of observable attributes. For the CARS paradigm, this means that the rating (or utility) function is of the following form [2]:

R: *Users x Items x Context* $\rightarrow$ *Ratings*

where *Context* is a set of contextual variables, each such variable K having a hierarchical structure defined by a set of k atomic variables, i.e., $K = (K_1,\dots, K_q)$ [2]. Further, the values taken by variable K_q define finer (more granular) levels, while K_1 coarser (less granular) levels of contextual knowledge [7]. For example, Figure 1(a, b) presents the hierarchy for the contextual variables "Season" and "Intent of the purchase" respectively that we use in the study presented in Sections 4.

The function R can be of the following two types. In the ratings-based RSes, users rate some of the items that they have seen in the past by specifying how much they liked these items. Alternatively, in the transaction-based RSes, function R defines the utility of an item for a user and is usually specified either as (a) a Boolean variable indicating if the user bought a particular item or not, (b) as the purchasing frequency of an item, or (c) as a click-through rate (CTR) of various Web objects (URLs, ads, etc.) [5]. In this paper we follow the transaction-based approach and measure the utility of product j for user i with the purchasing frequency x_{ij} specifying how often user i purchased product j.

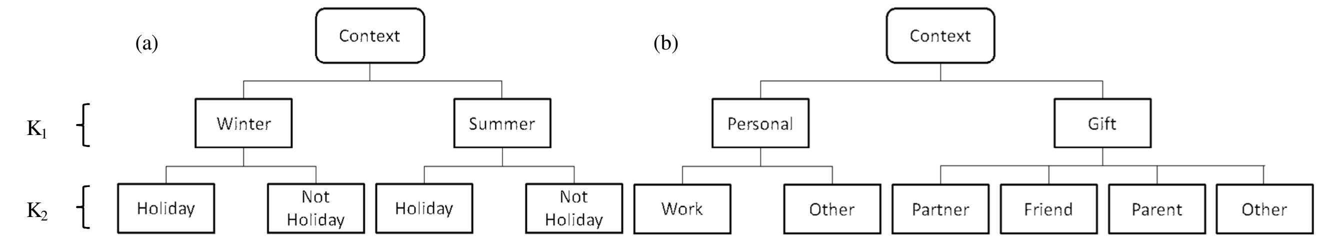

Figure 1 Hierarchical structure of the contextual information for (a) DB1 and (b) DB2 datasets.

As proposed in [2] and shown in Figure 2, this estimation can be done using the following three types of methods, each of them starting with data (on users, items, ratings and contextual information) and resulting in generating contextual recommendations:

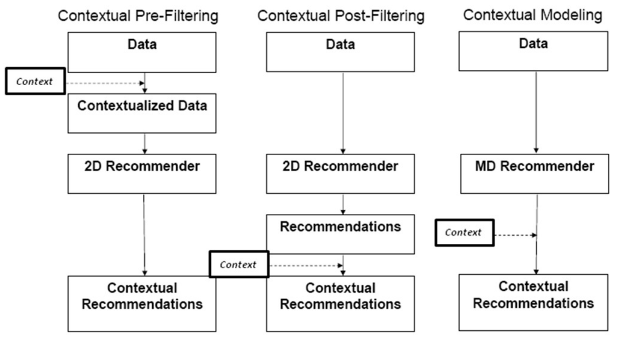

Figure 2 How to use context in the recommendation process.

1. *Contextual pre-filtering* (*PreF*): contextual information is used to filter out irrelevant ratings *before* they are used for computing recommendations using classical (2D) methods.

2. *Contextual post-filtering* (*PoF*): contextual information is used *after* the classical (2D) recommendation methods are applied to the standard (non-contextual) recommendation data.

3. *Contextual modeling* (*CM*): contextual information is used *inside* the recommendation-generating algorithms.

The work presented in [2] helped researchers to understand different aspects of using the contextual information in the recommendation process. However, [2] did not examine which of these methods are more effective for providing contextual recommendations. To address this issue, Panniello et al. [9] proposed certain contextual pre- and post-filtering approaches and compared them among themselves and also with the un-contextual (2D) approach to determine which one is better. More specifically, [9] proposed the *Weight* and *Filter* post-filtering approaches and the exact pre-filtering (EPF) method. Because of space limits we do not present the details which can be found in the original paper. [9] shows that the comparison of the un-contextual and the contextual RSes depends very significantly on the type of the post-filtering method used. The results also suggested that there is a big difference between good and bad post-filtering approaches in terms of performance measures. For example, the performance differences between the Filter and Weight methods range between 37% and 90% for the F-measure across different datasets and varies between 2.5 and 17 for the MAE and 1 and 3.5 for the RMSE metric.

3. CONTEXTUAL MODELING

In this section we present a new CM method called *contextual-neighbors* CM, and see how it compares against the pre- and the post-filtering methods. This approach is based on user-based collaborative filtering and works as follows. First, for each user *i* and context *k*, we define the user profile in context *k*, i.e. the contextual profile *Prof(i, k)*. For example, if contextual variable *k* has two values (e.g., Winter and Summer), then we have two contextual profiles for each user, one for the Winter and the other for the Summer.

Note that these contextual profiles can be defined in many different ways, some of which are presented in [8] [6, 11], and our approach does not depend on any particular choice of a profiling method. However, in the experimental study described in Section 4 we use the following specific contextual profiling technique. As explained in Section 2, we follow the transaction-based approach to RSes and measure the utility r_{ijk} of product *j* for user *i* in context *k* with the purchasing frequency x_{ijk} specifying how often user *i* purchased product *j* in context *k*. Then we use this measure to define contextual profile as $Prof(i, k) = (r_{i1k}, \dots r_{ink})$.

We use these profiles to define similarity among users and also to define and find *N* nearest "neighbors" of user *i* in context *k*, where "neighbors" are determined using contextual profiles *Prof(i', k')* and similarity measures between the profiles. In order to focus on the comparison among CARS, we decided to use a popular CF approach as a common method on the CARS that we compare, despite much research has generated more sophisticated methods, and defined the distance using the cosine similarity in our experiments. The basis for generating different *contextual neighbors* approaches is the way context is used to form the neighborhood. We find *N* pairs *(i', k')* such that the similarity between these profiles is the largest among all the candidate pairs *(i', k')* subject to the following constraints:

- Mdl_1: There are no constraints on the set of *(i', k')* pairs, and we select *N* pairs that are the most similar to *(i, k)*.

- Mdl_2: we select an equal proportion of pairs *(i', k')* corresponding to each context *k* (e.g., if the contextual variable has only two values, Winter and Summer respectively, and the neighborhood size is 80, we select 40 neighbors from Winter and 40 from Summer).

- Mdl_3: we select *N* pairs *(i', k')* that are the most similar to *(i, k)* corresponding to each context *k* at the same level of the context of interest (e.g., if the context of interest is "Winter Holiday" in Fig. 2(a), we select the neighborhood by using only profiles referred to level K_2 of that contextual variable).

- Mdl_4: we select an equal proportion of pairs *(i', k')* corresponding to each context *k* at the same level of the context of interest (e.g., if the context of interest is "Winter Holiday" in Fig. 2(a) and the neighborhood size is 80, we define the neighborhood by using 20 users from the context "Winter Holiday", 20 users from the context "Winter Not Holiday", 20 users from the context "Summer Holiday" and 20 users from the context "Summer Not Holiday").

After selecting the neighbors, we used their contextual profiles to make the rating predictions. Once we introduced the contextual neighbors approach and its four implementations Mdl_1, Mdl_2, Mdl_3 and Mdl_4, we next want to (a) compare them to determine which one is the best among them, and (b) see how it compares against the previously studied pre- and post-filtering methods.

4. EXPERIMENTAL SETUP

In this study, we compared the four types of CM methods, contextual modeling vs. the un-contextual case, and pre- vs. post-filtering vs. contextual modeling recommendations across a wide range of experimental settings. First, we selected two different data sets having contextual information. The first dataset (DB1) comes from an e-commerce website commercially operating in a certain European country which sells electronic products. For this dataset, we selected the time of the year (or Season) as a contextual variable (Fig. 1(a)). The classification into Summer or Winter and Holiday or Not Holiday is based on the experiences of the CEO of the e-commerce website that we used in our study.

The second dataset (DB2) is taken from the study described in [8]. The key contextual information elicited from the students was the intent of a purchase (IntentOfPurchase), and it was hierarchically classified as in Fig. 1(b).

In our study, we recommend product categories instead of individual items because the e-commerce applications that we consider have very large numbers of items (hundreds of thousands or even millions). Therefore, if single items were used, the conversion from implicit to explicit ratings would not work due to the low amount of rated data (e.g., many of the products were not purchased at all). We tried different item aggregation strategies and found that the best results are for 14 categories for DB1 and 24 categories for DB2. In particular, we performed experiments varying the number of categories and we found that each recommender system reached the best performances with these levels of aggregation. For our two datasets, we aggregated items into categories of products according to the classification provided by the Web site product catalogue. When using a context-aware recommender system it is useful to recommend a category instead of a product because users may not know what categories to look for in a specific context (for example, one may would like to receive a recommendation about the category of items for a not familiar context, such as a gift for a child).

The utility of items for the customers were measured by the purchasing frequencies, as described in Section 2 for the transaction-based RSes. Estimations of unknown utilities were done by using a standard user-based collaborative filtering (CF) method [10].

The neighborhood size N was set to $N = 80$ users as follows. We performed an experiment where we varied the neighborhood size, moving from 30 to 200 users, and we computed the F-measure. We performed this experiment for each dataset. In general, the F-measure increased as we increased the number of neighbors. However, these improvement gains stopped when we set the neighborhood size around 80, and the performance decreased when it went over 80 users. Therefore, we set $N = 80$ users as an appropriate neighborhood size for our experiments.

When comparing the pre-, the post-filtering and the contextual modeling methods, we used the two post-filtering approaches (*Weight* and *Filter*), the exact pre-filtering (*EPF*) method and the four contextual modeling methods Mdl_1, Mdl_2, Mdl_3 and Mdl_4 described above. Furthermore, we used the same user-based CF method for estimating unknown ratings in the pre-, the post-filtering and CMs cases to make sure that we compare "apples with apples". Since our aim was to compare different contextual approaches instead of finding the best contextual approach, we used a well known collaborative filtering instead of a newest, but less known, recommendation engine.

Further, we have performed t-tests in order to determine if the chosen contextual variables matter. The results of these tests demonstrated that the contextual variables Season and IntentOfPurchase matter (i.e., result in statistically significant differences in ratings across the values of the contextual variable at 95%). We used Precision, Recall, F-Measure, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) [4] as performance measures in our experiments. To this aim we divided each dataset into the training and the validation sets, the training set containing 2/3 and the validation set 1/3 of the whole dataset. For the DB1 dataset, the first two years were the training set and the third year was the validation set. For the DB2 dataset, we randomly split it in 2/3 for the training set and the remaining 1/3 for the validation set (in this case, it was impossible to make a good temporal split because all the transactions were made within a couple of months).

5. RESULTS

First of all we compared the four contextual modeling approaches among themselves across each experimental setting. In particular, Fig. 3 shows the comparison between the four CM approaches (namely Mdl_1, Mdl_2, Mdl_3 and Mdl_4) for each dataset. Because of space limits, we only show the graphs of F-measure, Recall and RMSE for the two databases. The graphs of Precision and MAE are very similar to those of F-measure and RMSE, respectively. Fig. 3 demonstrates that the performances of the four CM approaches are not remarkably different. The difference between Mdl_1 and Mdl_2 for DB1 is 0.008, 0.13 and 0.02 in terms of F-, MAE and RMSE measures, respectively and for DB2 is 0.09, 0.17, 0.18, respectively, which is not very significant. In comparison, performance differences between various pre- and post-filtering methods, as reported in [9], are much more pronounced in comparison to these differences (as an example, [9] reports that post-filtering Filter PoF method outperformed exact pre-filtering on DB1 by 0.21 and on DB2 by 0.4 in terms of the F-measure). Also, Mdl_1 slightly dominates other *Mdl* methods in some cases (see Fig. 3(b)) and is very close to Mdl_2 in other cases (see Fig. 3(a)). This makes sense because the N neighbors are selected for Mdl_1 in an unconstrained manner, whereas they are selected according to various types of constraints for the other three approaches. Since Mdl_1 (slightly) outperforms other *Mdl* methods, we selected it as the "best-of-breed" and will use it for comparing it with the pre- and the post-filtering methods in the rest of the paper.

We have also compared the *contextual neighbors* methods with the un-contextual approach across various experimental conditions. Table 1 reports all the accuracy gains (in terms of F-measure) across each recommender systems for DB1 and DB2 (negative values mean performance reduction). For example, its first row shows the performance gains (reductions), in terms of F-measure, for the un-contextual RS vis-à-vis the EPF, Filter PoF, Weight PoF and Mdl_1 methods. The matrix in Table 1 is anti-symmetric, as should be the case when two methods are compared in terms of their relative performance. As Table 1 shows, the contextual modeling approaches dominate the un-contextual case across all the levels of context for the F-Measure, Precision, MAE

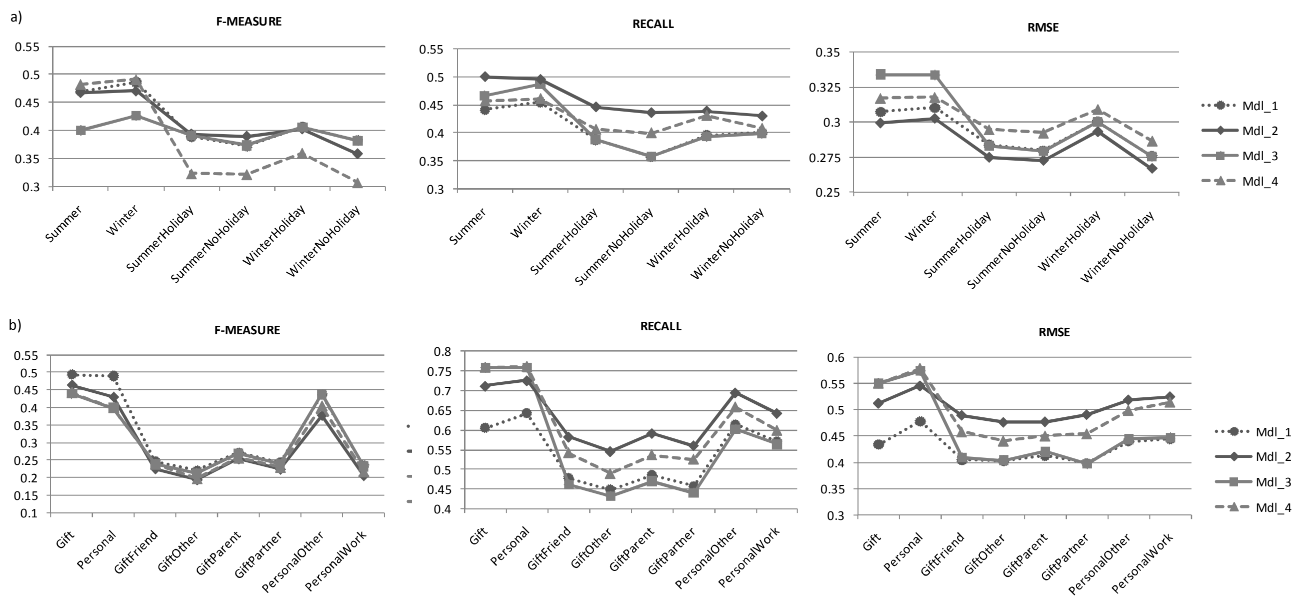

Figure 3 Comparison between the four contextual modeling methods for (a) DB1 and (b) DB2.

and RMSE. For example, if we consider Mdl_1 and the F-measure, for DB1, the difference between contextual and un-contextual models is 22% on average and for DB2 it is 7% on average.Mdl_1 clearly outperform the un-contextual method. The fact that the contextual modeling methods outperform the un-contextual one in almost all of the cases is not surprising because the contextual modeling method uses the same information as the un-contextual one and also includes the contextual variable which brings homogeneity in the data without causing the sparsity effect. We next compare the CM, pre-filtering and post-filtering approaches to determine the best among them.

As explained in Section 2, the performance of the post-filtering methods may significantly depend on the type of the post-filtering approach being used [9]. Therefore, we decided to use two post-filtering methods in our experiments, Weight PoF and Filter PoF, to account for these differences. Fig. 4 presents the comparison results among the two post-filtering methods Weight PoF and Filter PoF, the exact pre-filtering (EPF) and the contextual modeling method Mdl_1 across each contextual level and each dataset (DB1 and DB2). As Fig. 4 (and Table 1) demonstrates, Filter PoF dominates the EPF approach across the considered experimental settings. In particular, the difference between Filter PoF and EPF in terms of F-measure is 19% on average for DB1 and 26% for DB2. In contrast, EPF dominates Weight PoF in our experiments. In particular the difference between EPF and Weight PoF models in terms of F-measure is 29% on average for DB1 and 16% for DB2. In addition, the CM method Mdl_1, dominates the Weight PoF and in some cases the EPF. In particular, the difference between Mdl_1 and Weight PoF models in terms of F-measure is 39% on average for DB1 and 18% for DB2, while the difference between Mdl_1 and EPF is 21% on average for DB1 and 5% for DB2. In contrast, the Filter PoF dominates the modeling method. In particular the difference between Mdl_1 and Filter PoF models in terms of F-Measure is 3% on average for DB1 and 28% for DB2.

These results mean that the performance of the CM approach (as represented by Mdl_1) is very similar to that of the EPF method. This also implies, among other things, that CM is better than the un-contextual case and some of the weaker post-filtering methods, such as Weight PoF. However, like EPF, it is inferior to the best-performing post-filtering methods, such as Filter PoF. However, as argued in [9], finding the best-performing post-filtering methods can be a hard problem. Therefore, the CM approach, as represented in this paper by the Mdl_1, Mdl_2, Mdl_3 and Mdl_4 methods, constitutes a stable, easy to implement and a reasonably well-performing alternative that does not require expensive identification procedures, unlike the post-filtering methods. Therefore, considering our experiment settings it has its niche among the range of various CARS methods, as EPF does.

Table 1 F-measure gains (reductions) across recommender systems for DB1 and DB2.

		Un-contextual	EPF	Filter PoF	Weight PoF	Mdl_1
DB1	Un-contextual	0%	-2%	-20%	27%	-22%
	EPF	2%	0%	-19%	29%	-21%
	Filter PoF	20%	19%	0%	59%	-3%
	Weight PoF	-27%	-29%	-59%	0%	-39%
	Mdl_1	22%	21%	3%	39%	0%
DB2	Un-contextual	0%	-2%	-27%	14%	-7%
	EPF	2%	0%	-26%	16%	-5%
	Filter PoF	27%	26%	0%	57%	28%
	Weight PoF	-14%	-16%	-57%	0%	-18%
	Mdl_1	7%	5%	-28%	18%	0%

6. CONCLUSIONS

In this paper we proposed a new type of CM, that we called *contextual neighbors*, and four specific types of contextual neighbors methods, called Mdl_1, Mdl_2, Mdl_3, Mdl_4, each of them selecting contextual neighborhoods in a different way. We also compared the contextual neighbors methods Mdl_1, Mdl_2, Mdl_3, Mdl_4 to identify the best performing one. Finally, we compare it to other approaches to CARS. This is the first step of a broader research in which we want to compare the relative performance of different contextual modeling approaches, including ours.

Although Mdl_1 slightly outperforms the others, we have shown that there are no relevant performance differences among them.

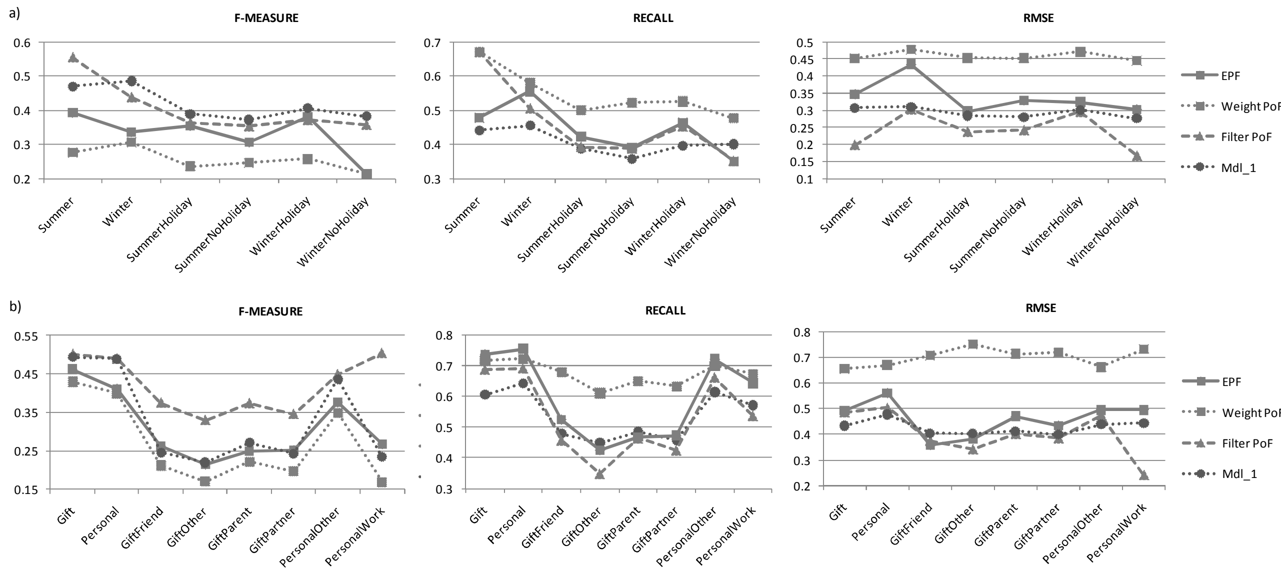

Figure 4 Comparison between EPF, Weight PoF, Filter PoF and Mdl$_1$ for (a) DB1 and (b) DB2.

This result is not surprising because different ways of selecting contextual neighborhood do not fundamentally change recommendation results. We have also compare *Mdl$_1$* with the pre-, post- filtering and un-contextual methods developed in our previous studies across various experimental settings, including two datasets, different levels of item aggregation, different neighborhood sizes, different contextual levels (K_1 and K_2) and several performance measures (Precision, Recall, F-Measure, MAE and RMSE). We have shown that *Mdl$_1$* dominates the traditional un-contextual approach and is comparable to the pre-filtering method (EPF). We have also shown that *Mdl$_1$* dominates some of the less advanced post-filtering methods (such as Weight PoF) but is inferior to the best post-filtering methods (such as Filter PoF). Since identification and selection of the best post-filtering methods is a laborious process (as argued in [9]), this means that contextual neighbors CM methods (such as *Mdl$_1$*) have their prominent place in the spectrum of various CARS recommendation methods: they are easy to implement, reasonably well-performing and do not require expensive identification procedures, unlike the post-filtering methods. The main limit of these results is the fact that we do not compare our contextual modeling approach to the existing ones. The reason is that we only present the first step of the research.

In a future work, we will present the comparison of the *contextual neighbor* to other CM approaches. In addition, we will use other recommendation engines and other representations of the contextual variables different from the straightforward kNN and the hierarchical representation of the context used in this paper. We will use other performance metrics, beyond those accuracy-based, such as the recommendations diversity, in order to better understand the impact of the different contextual approaches on customers behavior. In future research steps we will also measure the effect of different CM approaches on customers' trust and on their actual purchases.

7. REFERENCES

[1] Adomavicius, G., Sankaranarayanan, R.,Sen, S., and Tuzhilin, A. 2005. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM T. Inform. Syst.* 23, 1 (2005), 103-145.

[2] Adomavicius, G., and Tuzhilin, A. 2011. Recommender Systems Handbook, Chapter 7, Springer.

[3] Dourish, P. 2004. What we talk about when we talk about context. *Personal and Ubiquitous Computing* 8, 19-30.

[4] Herlocker, J.L., Konstan, J.A., Terveen, L.G., and Riedl, J.T. 2004. Evaluating collaborative filtering recommender systems. *ACM Transaction on Information System* 22, 5-53.

[5] Huang, Z., Li, X., and Chen, H. 2005. Link prediction approach to collaborative filtering, In *Proc. of the 5th ACM/IEEE-CS joint conference on Digital libraries*, 141-142.

[6] Karatzoglou, A., Amatriain, X., Baltrunas, L., and Oliver, N. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering, In *Proc. of the fourth ACM conference on recommender Systems*, 79-86.

[7] Kwon, O, and Kim, J. 2009. Concept lattices for visualizing and generating user profiles for context-aware service recommendations. *Expert Systems with Applications* 36, 1893-1902.

[8] Palmisano, C., Tuzhilin, A., and Gorgoglione, M. 2008. Using Context to Improve Predictive Models of Customers in Personalization Applications. *IEEE TKDE*, 20(11), 1535-1549.

[9] Panniello, U., Tuzhilin, A., Gorgoglione, M., Palmisano C., and Pedone, A. 2009. Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. In *Proc. of RecSys '09*, 265-268.

[10] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proc. of Conference on Computer Supported Cooperative Work*, 175-186.

[11] Shi, Y., Larson, M., and Hanjalic, A. 2010. Mining mood-specific movie similarity with matrix factorization for context-aware recommendation. In *Proc. of the Workshop on Context-Aware Movie Recommendation*, 34-40.