Does the recommendation task affect a CARS performance?

Umberto Panniello
Politecnico di Bari
Viale Japigia, 182 – 70026 Bari (Italy)
+390805962765
m.gorgoglione@poliba.it

Michele Gorgoglione
Politecnico di Bari
Viale Japigia, 182 – 70026 Bari (Italy)
+390805962765
u.panniello@poliba.it

ABSTRACT
There is growing interest on improving the performance of recommender systems (RSs) by using context-aware recommender systems (CARS). However, companies may adopt different recommendation tasks, from recommending few items of very high interest to recommending all possibly interesting items. These tasks are influenced by some characteristics of the application and other business conditions. The aim of this research is to experimentally investigate whether a CARS always outperforms a traditional RS or if it happens only in some specific recommendation tasks. To this aim, we have compared the performance of a RS and three approaches to CARS, namely a pre-filtering and two post-filtering methods, by using two recommendation tasks, the first recommending only the top-k items, the second recommending all the items potentially good for the customer. Experiments were performed across several conditions and using three datasets from e-commerce applications.

Categories and Subject Descriptors
H.3 [**Information storage and retrieval**]: H.3.3 Information Search and Retrieval–*Information filtering*

General Terms
Algorithms, Performance, Modeling

Keywords
Recommender System, Context, Recommendation strategy

1. INTRODUCTION
Context-Aware Recommender Systems (CARS) are receiving growing attention by researchers and companies, as they were proved to improve the recommendation performance in several conditions. On the industrial side, companies may deliver recommendations in different ways depending on factors related to the business application. For instance, frequently recommending many items may either increase a customer's interest and knowledge on products or annoy her causing an opposite reaction. On the other hand, few recommendations may cause the company to lose sales opportunities, but also increase a customer's attention. Similarly, recommending only the new products the customer has never seen before may be useful when products are rarely purchased more than once, such as comics books. When repeated purchases are frequent, e.g., in e-grocery, companies find more useful to remind customers to buy products they have already bought in the past. In general, different recommendation strategies may have different effects.

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

Companies adopting Recommender Systems (RS) are interested in increasing the recommendation performance. Including context in the recommendation engine may improve performance under certain circumstances [11, 1, 2]. However, we know from prior research that CARSs often outperform un-contextual RSs, but not in all conditions [11]. As an example, although the F-measure of a CARS is very often greater than that of a RS, Precision and Recall can present opposite figures. Hence, in some conditions (e.g., when not missing an important recommendation is critical) a company should prefer to improve Recall rather than Precision, and the usage of a CARS would be detrimental. Therefore, the issue of whether a CARS outperform a RS depending on the way recommendations are generated and delivered is not trivial.

This research experimentally studies whether the recommendation task affects the comparison between a CARS and a RS performance and which conditions can affect the relative performance. To this aim, we have compared the performance of three approaches to CARS, namely a pre-filtering and two post-filtering methods, by using two recommendation tasks ("top-k", which recommends only a subset of items taken from those most likely to be purchased, and "find-all", which recommends all the items that are potentially good for a customer). The experiments were done across several conditions and using three data sets, corresponding to three different e-commerce applications.

2. PRIOR RESEARCH
Recommender systems were classified based on the goal they are designed for: recommending good items, optimizing utility and predicting ratings [9]. In this paper we focus on the most common one, i.e., to recommend good items to users [6, 9]. There are many business applications pursuing this goal, such as Amazon or Netflix. In these cases, two tasks can be considered [5]. The first aims at recommending the "top-k" items. In this case only a subset of all the recommendable items is shown to customers. The key assumption is that although a large number of items may appeal to the user, she does not have enough resources (e.g., time) to consider them all. Therefore, the main important issue is to focus on the most interesting items and not to present any disliked item [4, 8]. The second task aims at recommending "all good items" and it is common in research and business [6, 5, 9]. In the "find-all" task it is important to recommend all the potential good items and avoid to miss a relevant item [5].

Companies can choose one of the two tasks because of business and technology constraints, such as the communication channel, the space available, the frequency and other customers constraints. For instance, Amazon presents its customers the recommended items on Web pages where many items ("all good items") can be listed. Toysdirect lists only a few (the top-k) recommendations in

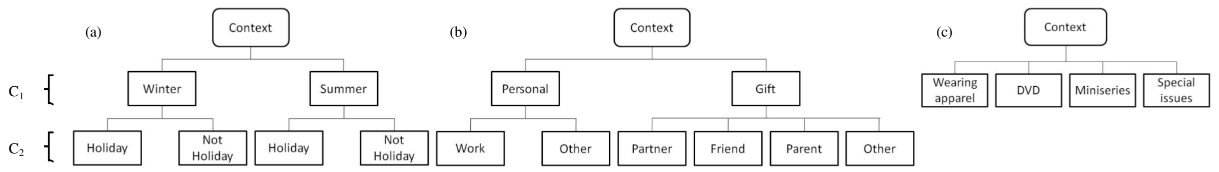

Figure 1. Hierarchical structure of the contextual attributes (a) Time of the day, (b) Intent of purchase and (c) Store.

an e-mailed newsletter. Companies sending recommendations via sms focus only on the top-one product. When recommendations are e-mailed frequently, listing all the good items might annoy customers, depending on the time they want to spend in the early stage of the purchasing process. Since the performance of RSs should be compared by different metrics in different tasks [5], we believe that studying the comparison between CARSs and RSs in different task is important for companies because the adoption of a specific task can affect the decision of using a CARS or a RS.

Other factors may affect the way recommendations are generated, such as the misclassification costs and the overall number of recommendable items. In some cases, the cost of false positives is higher than that of false negatives. For instance, when TV shows are recommended, customers may feel offended by wrong recommendations, but can tolerate the absence of a good show. In this case, the major problem is avoiding wrong recommendations, i.e., minimizing the number of false positives. Companies should adopt a CARS or a RS depending on which one presents a higher Precision, which is defined as number of true positives divided by the sum of true positives and false positives. In other cases, the cost of false positives is lower than that of false negatives. For instance, a Website recommending scientific papers should be concern to not missing any relevant item (i.e., minimizing false negatives). Therefore, a CARS or a RS should be adopted depending on the Recall, which is defined as number of true positives divided by the sum of true positives and false negatives. Recommendations can be generated by picking items from either a small or a great set of products. For instance, Amazon presents both types of recommendations. The "new for you" box includes items taken from the latest releases, while the "recommended for you" presents items from the whole catalogue. The catalogue is much wider than the latest releases list. LeShop.ch recommends items that a customer has never bought before during the navigation, and items that the customer regularly bought in the past at the checkout. The number of recommendable items is much lower in the first case than in the second. This can affect the performance of a CARS compared to that of a RS because it can affect the sparsity of the User-Item matrix and, in turn, the recommendation performance [11, 1, 2].

The usage of contextual information in CARS can be broadly classified into two groups: recommendation via context-driven querying and search [13], recommendation via contextual preference elicitation and estimation [1, 3]. These methods are reviewed in [3] and [2]. In [2] the following three different algorithmic paradigms for incorporating contextual information into the recommendation process are presented: (i) contextual pre-filtering, where context is used to filter out irrelevant ratings before they are used for computing recommendations with standard methods; (ii) contextual post-filtering, where context is used after the standard recommendation methods are applied to the data; (iii) contextual modeling, where context is used inside the algorithms that generate recommendations. We used pre-filtering and post-filtering, as they are the only approaches which have been studied so far [11].

Although many practical examples exist of "top-k" and "find-all" tasks and much literature have tackled the issue of which performance metrics should be used in each of the two recommendation tasks [6, 5], no experimental study have raised the question of whether context should be included in a RS depending on the recommendation task. Part of this research comes from our previous experiments reported in [11]. However, there are several important differences. First, the goal of [11] was to compare two CARS, while this research goal is to argue in which conditions context should be included. Second, [11] only considered the "find-all" task, while this research also considers the "top-k" task. Finally, [11] considered only two data sets.

3. EXPERIMENTAL SETUP

We compared the performance of an un-contextual RS and three CARSs by varying several settings: the recommendation task ("top-k" vs. "find-all"), the number of overall recommendable items (by using three data sets with different numbers), three performance metrics (Precision, Recall and F-measure), the number of items in the recommendations list ("k" in a top-k task), and the granularity of the context (in the CARS). We decided to focus our experiment on the "find all" and "top-k" strategies since they are the most popular in business and research. We selected three datasets coming from three e-commerce applications. The first (DB1) comes from a portal commercially operating in an European country which sells electronic products to approximately 120,000 users and contains about 220,000 purchasing transactions. The second (DB2) consists of various purchasing transactions performed by students on the Amazon website in a controlled environment that also recorded the contextual information of the purchases via a special-purpose browser developed for the project. More information about this dataset and the browser can be found in [10]. The third dataset (DB3) comes from an e-commerce portal which sells comics and comics-related products (e.g., t-shirts, DVDs), including about 50,000 transactions and 5,000 users. We used product categories instead of single items in our study because the e-commerce applications have huge numbers of items (hundreds of thousands). Therefore, if single items were used, the conversion from implicit to explicit ratings would not work due to the low amount of rated data. Items were aggregated according to the Web site catalogues: 14 categories for DB1, 24 for DB2, 136 for DB3.

The datasets were used also in order to perform the comparison by using different contextual variables and granularity. In DB1 we used the time of the year as a contextual variable. Its hierarchical structure is presented in Fig. 1(a). This structure defines two levels of contextual granularity: C_1 (coarser level) and C_2 (finer level). The classification into Summer or Winter and Holiday or Not Holiday was based on the experiences of the CEO of the e-commerce website that we used in our study. The contextual information in DB2 was the intent of a purchase, collected by the above-mentioned special purpose browser. Its hierarchical structure is in Fig. 1(b). In DB3 we used the store (i.e., the section in the Web site where products are bought) as a contextual variable: the product may be bought in the "Wearing apparel", "DVD", "Miniseries" or "Special issues" section (store) of the portal (see Fig. 1(c)). This is a contextual variable because customers' behavior changes when navigating and buying products in different sections of the Web site. We performed t-tests in order to determine if context matters. All the comparisons were significant at 95%. The utilities of items for the customers were measured by the purchasing frequencies, according to the transaction-based RS approach [7]. This measure serves as a proxy for the rating, as it measures how much a customer likes a product. Estimations of unknown utilities were done by using a standard user-based collaborative filtering (CF) method [12]. According to CF, the neighborhood was formed using the cosine similarity. After several experiments its size was set to 80 users.

When comparing the traditional RS and the CARSs, we used one pre-filtering and two post-filtering methods (*Weight* and *Filter*). In the pre-filtering approach the contextual information is used as a label for filtering out those ratings that do not correspond to the specified contextual information [2]. This filtering is done before the main recommendation method is launched on the remaining data that passed the filter. We used the exact pre-filtering method (*EPF*) [2]. For the post-filtering case, according to [2], we first ignored the contextual information in the data and applied a traditional 2D recommendation method, such as CF, on the whole un-contextual dataset. Once the unknown ratings are estimated using the 2D method and the un-contextual recommendations are produced, we "contextualized" these recommendations. Various methods exist for contextualizing the 2D recommendations, as described in [2]. We focused on two approaches, called *Weight* and *Filter*. Detailed information on *EPF*, *Weight* and *Filter* methods is in [11]. We used the same user-based CF method for estimating unknown ratings in all the CARSs.

Finally, we measured Precision, Recall and F-measure [6]. In order to compute Precision and Recall, we set a threshold between relevant and irrelevant items. We assumed that if an item is purchased more than onece, it is relevant (the threshold is set at 1). Therefore, if the system predicts a rating greater or equal to 2, we decide to "recommend" that item, otherwise we do not. Then, we verify if the recommended item was actually purchased in the validation set, and if it is so we consider it as a correct recommendation, otherwise as bad. We did not use MAE and RMSE as they are not applicable to the "top-k" strategy because they are calculated on the whole matrix of predicted ratings.

4. RESULTS

Figure 2 reports the comparison between the two CARSs (described in section 4) and the RS in the "top-k" task, for k=1 (Fig. 2a) and k=4 (Fig. 2b), while Figure 3 reports the same comparison in the "find-all" task. We made experiments in the "top-k" task with k=2 and k=3 as well, but for the sake of brevity we present two of the four cases. Graphs are reported for each data set (DB1, 2 and 3). We decided to show both Precision and Recall, in addition to F-measure, since we expected that the comparison also depends on the specific metric used, and because a company may choose to maximize that metric depending on the business settings. When a "top-k" task is used and the number of items in the recommendation list is very low (Fig. 2a), the un-contextual RS dominates the CARS across each experimental setting in DB1 and DB2 in terms of all the three performance measures, while the *Filter* and *EPF* CARSs dominate the RS in DB3 (Fig. 2a). When k increases to 4 (Fig. 2b), the un-contextual RS always dominates the CARSs in terms of Recall, while the Precision and F-measure of RS and CARSs tend to become similar in DB1 and DB2. In DB3 the *Filter* and *EPF* CARSs dominate the RS. On the contrary, when a "find-all" task is used, the CARSs outperform the RS in terms of Precision and F-measure in all data sets (depending on the CARS), while the RS presents a higher Recall in the three datasets.

The results allow us to answer the issue of whether context should be included in a RS depending on the recommendation task and other conditions. Firstly, the comparison between a RS and a CARS does depend on the recommendation task. Secondly, the main conditions that affect the comparison are the overall number of recommendable items, number of items in the recommendation list and the specific performance that has to be measured. In the "find-all" task the comparison seems to depend on the performance, whereas in the "top-k" task the comparison depends on the overall number of recommendable items (i.e., the data set) and the number of items in the recommendation list (i.e., the value of k). In fact, in the "find-all" task two out of three CARSs (*Filter* and *EPF*) always outperform the RS in terms of F-measure and Precision, while the RS is better in terms of Recall (see Fig. 3). This happens independently of the number of items, i.e. in all three data sets. In the "top-k" task, when the overall number of items is low and only one item is in the recommendation list (DB1 and DB2 in Fig.2a)) the RS is always preferable to the CARSs. When k increases to 4, the F-measure and Precision of the CARSs become greater (or equivalent) to those of the RS, while the Recall of the RS is still greater (DB1 and DB2 in Fig.2b). When the overall number of recommendable items increases (DB3 in Fig.2a and 2b), all the performance measures of the CARSs are greater than those of the RS. The approach to CARS and the context granularity do not affect the comparison significantly.

5. CONCLUSIONS

In this research we experimentally compared a RS to three CARSs by using two different recommendation tasks ("top-k" and "find-all"), three data sets coming from three different e-commerce applications and different contextual variables. The results show that the comparison depends on the recommendation task. In the "find-all" task, CARSs are preferable if the performance has to be measured in terms of F-measure and Precision. In the "top-k" task, RSs are preferable only when the overall number of recommendable items is low and very few items (less than four) may be put in the recommendable list.

Although generalizing these findings to any business application is hard, the findings suggest that companies that aim at improving the recommendation performance by including context in a RS should consider the recommendation task and some characteristics of the business applications, namely the overall number of items, the number of items in the list, the performance.

Although CARSs seem to be often preferable to RSs, in certain particular conditions, using a RS is preferable to using a CARS.

Further research is needed to strengthen the conclusions. A model should be defined in order to associate several business conditions with each recommendation task and performance metrics. Customer satisfaction-related performance metrics should be as well as other CARS algorithms. Other contextual aspects and non e-commerce domains should be considered.

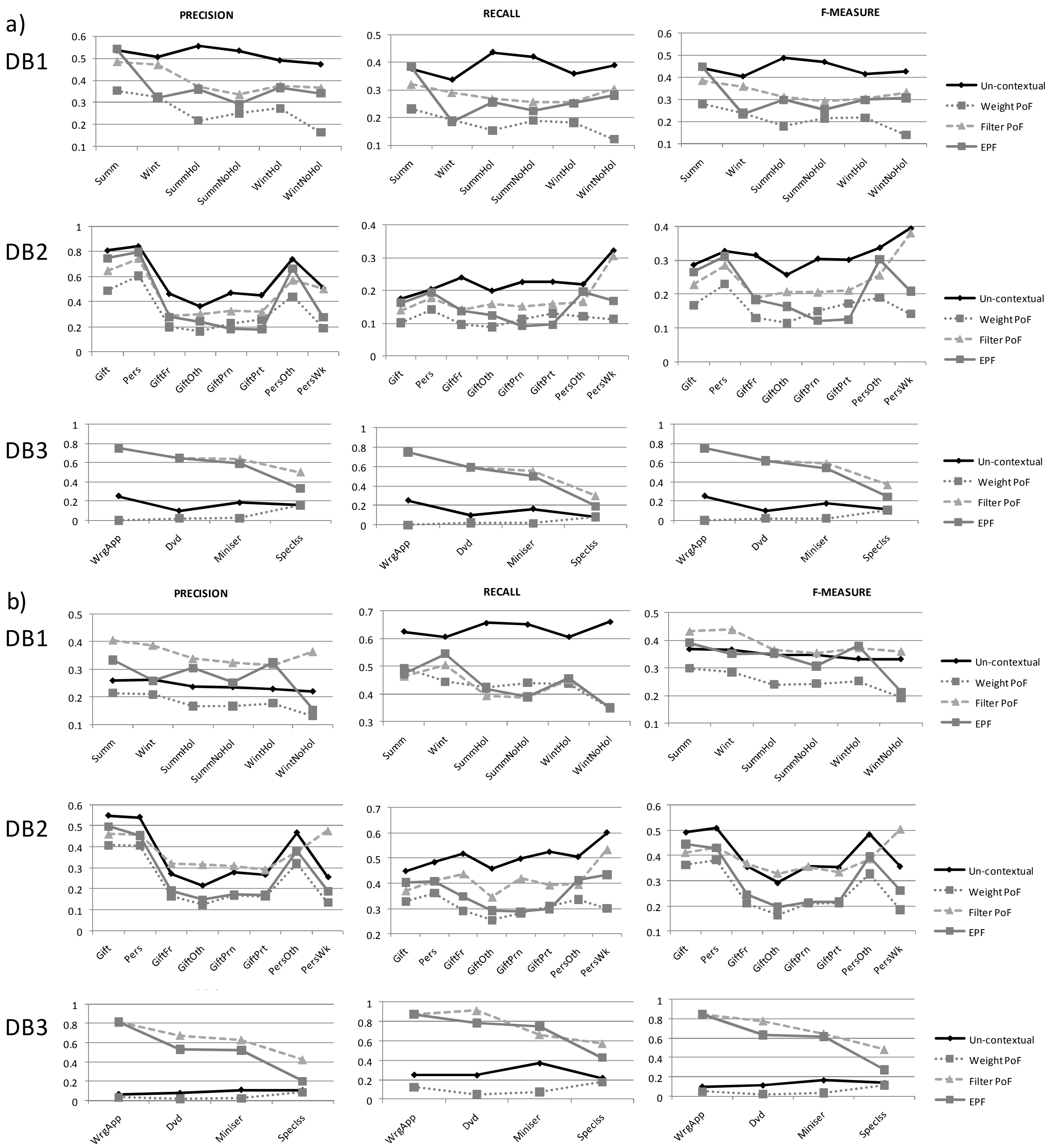

Figure 2. Comparison between RS and CARSs using a) top-1 and b) top-4 recommendation tasks for the three data sets.

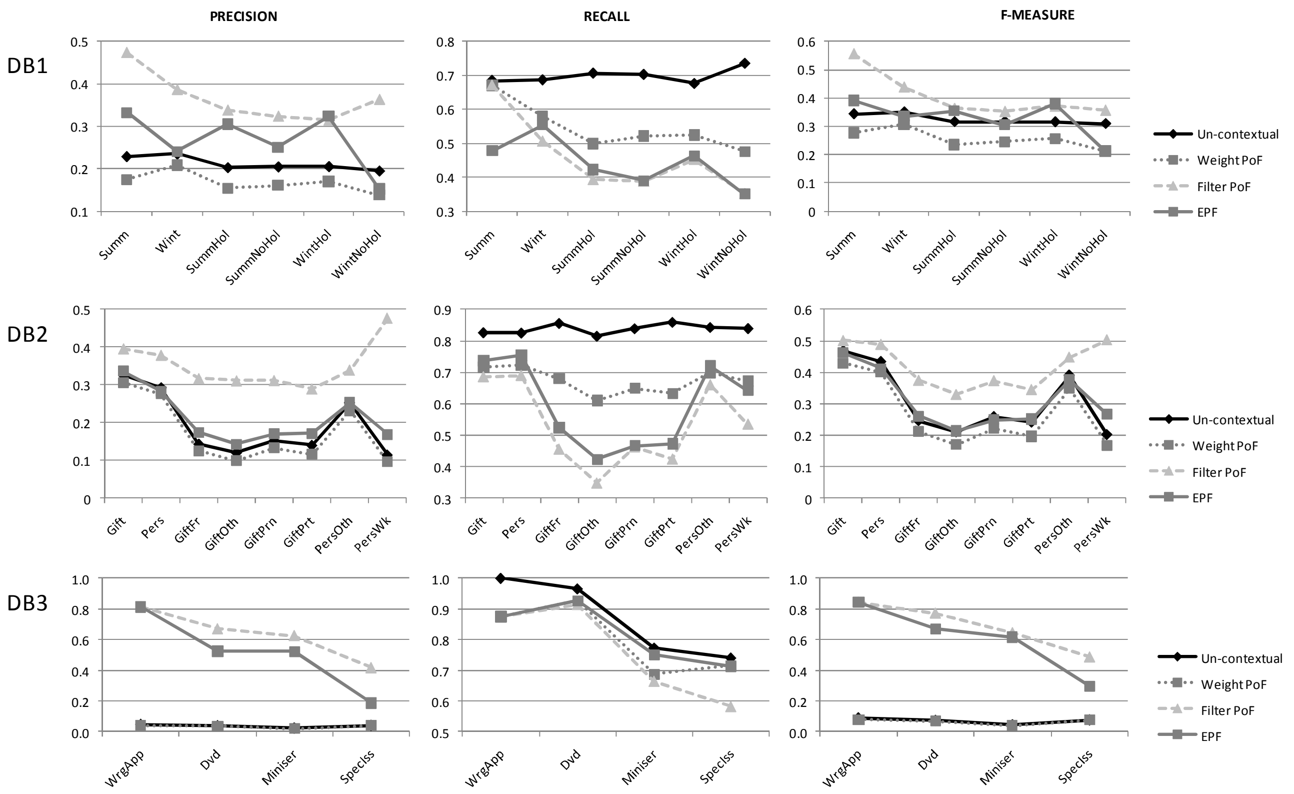

Figure 3. Comparison between RS and CARSs using a "find all good items" recommendation task for the three data sets.

6. 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, 103-145.

[2] Adomavicius G., and Tuzhilin, A. 2008. Context-Aware Recommender Systems. Handbook on Recommender Systems, Springer, 2010 (appeared as a tutorial at *the 2008 ACM Conference on Recommender systems*, 335-336).

[3] Anand, A., and Mobasher, B. 2007. Contextual Recommendation. *From Web to Social Web: Discovering and Deploying User and Content Profiles*, Springer, Berlin.

[4] Deshpande, M., and Karypis, G. 2004. Item-Based Top-N Recommendation Algorithms, *ACM Trans. Information Systems* 22, 1, 143-177.

[5] Gunawardana, A., and Shani, G. 2009. A survey of accuracy evaluation metrics of recommendation tasks, *J. Mach. Learn. Res.* 10 (December 2009), 2935-2962.

[6] Herlocker, J.L., Konstan, J.A., Terveen, L.G., and Riedl, J.T. 2004. Evaluating collaborative filtering recommender systems, *ACM T. Inform. Syst.* 22, 1, 5-53.

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

[8] Linden, G., Smith, B., and York, J., 2003. Amazon.com Recommendations: Item-to-Item Collaborative Filtering, *IEEE Internet Computing*, Jan./Feb. 2003.

[9] McNee, S. M., Riedl, J., and Konstan, J. K. 2006. Making recommendations better: an analytic model for human-recommender interaction, In *CHI '06 Extended Abstracts on Human Factors in Computing Systems.*

[10] Palmisano, C., Tuzhilin, A., and Gorgoglione, M., 2008. Using Context to Improve Predictive Models of Customers in Personalization Applications, *IEEE T. Knowl. Data En.*. 20, 11, 1535-1549.

[11] 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, *Proceedings of RecSys '09*, 265-268.

[12] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J., 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews, *Proceedings of Conf. on Computer Supported Cooperative Work*, 175-186.

[13] van Setten, M., Pokraev, S., and Koolwaaij, J., 2004. Context-aware recommendations in the mobile tourist application COMPASS, *Adaptive Hypermedia*, 235–244.