Context-Aware Mobile Recommendation By A Novel Post-Filtering Approach

Yong Zheng
Department of Information Technology and Management
School of Applied Technology
Illinois Institute of Technology
Chicago, IL, 60616, USA

Abstract

Recommender system has been demonstrated as a successful solution to assist decision makings. Context-awareness becomes necessity in recommendations, especially in mobile computing, since a user's decision may vary from contexts to contexts. Context-aware recommender systems, therefore, emerged to adapt the personalizations to different contextual situations. Context filtering is one of the popular ways to develop the context-aware recommendation models. Contextual pre-filtering techniques have been well developed, but the post-filtering methods are still under investigated. In this paper, we propose a simple but effective post-filtering recommendation approach. We demonstrate the effectiveness of this algorithm in comparison with other context-aware recommendation approaches based on the real-world rating data from mobile applications. Our experimental results reveal that the proposed algorithm is the best post-filtering approach, and it is even able to outperform the popular pre-filtering and contextual modeling recommendation models.

Introduction

Recommender systems (RSs) provide personalized suggestions of products to the end-users in a variety of settings. It has been successfully applied to several domains, such as e-commerce (e.g., Amazon), online streaming (e.g., Netflix), social media (e.g., Facebook), and so on. Recommendation models are built to learn from the user preferences and make predictions on the items a user may like. The user preferences are usually stored and represented in a rating matrix $U \times I$, where the entries in the matrix are known ratings by particular users for given items. Then the process in the traditional recommendation algorithms can be depicted as a *rating prediction* task which infers the likely values of unknown cells in this matrix. i.e., *R*: *Users* $\times$ *Items* $\rightarrow$ *Ratings*. Or, it could be a *top-N recommendation* task, where the system will recommend the top-N items to a user according to his or her preference history.

The importance of context-awareness has been recognized in many areas, e.g., ubiquitous computing and information retrieval. In RS, we believe that a user's tastes may vary from contexts to contexts, e.g., a user may choose a different movie if he or she is going to watch it with *kids* rather than with the *partner*. A user may choose a different destination if he or she is going to travel *in winter* rather than *in summer*. These variables (Zheng 2015), such as time, location or companion, become the crucial factors to affect a user's preferences. The scenario of mobile application is one of the examples in which contexts may play an important role. User may present distinct usage patterns in different contexts, such as time (e.g., weekend and weekday), locations (at home or office) and activities (sitting or driving).

Context-aware recommender systems (CARS) emerged to produce item recommendations by additionally taking contexts into consideration. It turns the prediction task into a multidimensional rating function – *R*: *Users* $\times$ *Items* $\times$ *Contexts* $\rightarrow$ *Ratings* (Adomavicius et al. 2011). CARS are usually developed by following three strategies: pre-filtering, post-filtering and contextual modeling (Adomavicius et al. 2011). As the name would suggest, pre-filtering techniques use the contextual information to remove irrelevant rating profiles from consideration, and then apply the traditional recommendation algorithms only with profiles contain ratings in matched contexts. Post-filtering techniques produce predictions by the traditional way, and then adjust the predicted ratings or re-rank the list of the recommendations. By contrast, the contextual modeling approaches will directly use context information as parts of the predictive models to produce the item recommendations.

The contextual modeling approaches are more powerful in making the predictions, but also more complicated and they usually leave the difficulty to understand the contextual effects in the model, By contrast, the contextual filtering methods, including pre-filtering and post-filtering, are more straightforward and easy to be interpreted. The pre-filtering algorithms have been well developed, but there are limited efforts on the post-filtering techniques. In this paper, we propose a simple but effective post-filtering recommendation algorithm, and demonstrate its effective by comparing the state-of-the-art context-aware recommendation algorithms.

Related Work

In this section, we introduce the existing context-aware recommendation models, especially the ones in the category of the post-filtering methods. To better understand the CARS, we introduce the terminologies in this domain as follows.

In Table 1, there are one user U_1, one movie T_1, and three

Table 1: Contextual Ratings on Movies

User	Item	Rating	Time	Location	Companion
U_1	T_1	3	weekend	home	alone
U_1	T_1	5	weekend	cinema	partner
U_1	T_1	?	weekday	home	family

context dimensions – Time (weekend or weekday), Location (at home or cinema) and Companion (alone, partner, family). In the following discussion, we use *context dimension* to denote the contextual variable, e.g. "Location". The term *context condition* refers to a specific value in a dimension, e.g. "home" and "cinema" are two contextual conditions in "Location". The *contexts* or *context situation* is, therefore, a set of contextual conditions, e.g. {*weekend, home, family*}.

Contextual pre-filtering will use the context information to filter out irrelevant rating profiles and apply the traditional recommendation algorithms to produce the item recommendations. For example, to suggest a list of the movies for a user to watch *at weekend with family*, we need to use the existing ratings that were placed in exactly the same or similar context situations. The popular pre-filtering techniques include splitting-based methods (Zheng, Burke, and Mobasher 2014) and semantic pre-filtering (Codina, Ricci, and Ceccaroni 2015). Contextual modeling (Baltrunas, Ludwig, and Ricci 2011; Zheng, Mobasher, and Burke 2014) is the most complicated strategy, while contexts are directly incorporated into the predictive models. Context-aware matrix factorization (Baltrunas, Ludwig, and Ricci 2011) is one of these techniques that try to learn the rating deviations in different contexts. They may work well but it is difficult to interpret the model or understand the contextual effects.

By contrast, the contextual post-filtering methods are still under investigation. The idea behind post-filtering is straightforward – we produce the predicted ratings or the list of top-N items without considering contexts by the traditional recommendation algorithms. Then, we can remove the items that are irrelevant to the contexts, or re-rank the list of the items, or adjust the predicted ratings. Panniello, et al. (Panniello et al. 2009) proposed the first post-filtering method which can be described by Equation 1.

$$\widehat{R}(u,t,c) = \begin{cases} \widehat{R}(u,t) & Pr(u,t,c) \geq p \\ 0 & Pr(u,t,c) < p \end{cases} \quad (1)$$

$\widehat{R}(u,t,c)$ refers to the predicted rating for the user u on item t within context situation c, while $\widehat{R}(u,t)$ is the predicted rating without considering contexts by a traditional recommendation algorithm. They additional calculate a probability, $Pr(u,t,c)$, with which the user will choose a certain type of item in a given context. This probability is computed as the number of neighbors (i.e., users similar to u) who purchased or consumed the same item t in contexts c divided by the number of the total number of neighbors. We set $\widehat{R}(u,t,c)$ as zero if this probability is smaller than a threshold p, to indicate that the item t is not qualified to be recommended. Otherwise, the model will use $\widehat{R}(u,t)$ to represent $\widehat{R}(u,t,c)$. We name this method as post-filtering based on user neighborhood and denote it by "PoF_Ngbr". The notion of user neighborhood comes from the neighborhood-based collaborative filtering, where the neighborhood can be identified by measuring user-user similarities that can be calculated by the cosine similarity or Pearson correlations. This method is only valid for evaluating the top-N recommendations, since they mark $\widehat{R}(u,t,c)$ as zero if item t is not appropriate to be recommended.

Ramirez, et al. (Ramirez-Garcia and Garca-Valdez 2014) made the second attempt and they proposed a post-filtering method by adjusting the predicted ratings. We refer this method as "PoF_Adj". The prediction can be described by Equation 2. $\overline{R}(t,c)$ denotes the average value of the ratings that are placed on the item t within context c. The predicted contextual rating, therefore, is composed by $\widehat{R}(u,t)$ and $\overline{R}(t,c)$. They set a ratio β (0$<\beta<$1) to control the contributions of each part.

$$\widehat{R}(u,t,c) = \beta \times \widehat{R}(u,t) + (1-\beta) \times \overline{R}(t,c) \quad (2)$$

These two existing approaches utilize a traditional recommendation algorithm to produce the predicted rating without considering contexts (i.e., $\widehat{R}(u,t)$), then try to contextualize this predicted rating by removing irrelevant items associated with the contexts (e.g., PoF_Ngbr) or adjusting the predicted ratings (e.g., PoF_Adj). Apparently, the key challenge in the post-filtering methods is how to contextualize the predicted ratings or recommendations that are produced without considering contexts.

Methodologies

We describe our basic solution first, and then discuss the methods to improve the proposed model in this section.

Post-Filtering Based on Rating Deviations

To contextualize the predicted rating without considering contexts, we need to figure out how to fuse contexts into the post-filtering process. The PoF_Ngbr model removes an item from the recommendation list if the user does not like the item in that context. Whether the user likes or dislikes the item is inferred from the knowledge about how the neighbors of this user like the same item in that context. By contrast, the PoF_Adj model assumes the user taste in context c depends on not only how the user likes the item without considering contexts, but also how appropriate the item is to be purchased or consumed in c.

Inspired by these two approaches, we decide to add the rating deviations between a contextual rating and the rating without contexts to estimate whether the user likes a specific item in context c. It can be described by Equation 3 and 4.

$$\widehat{R}(u,t,c) = \begin{cases} \widehat{R}(u,t) & Dev(u,t,c) > 0 \\ 0 & Dev(u,t,c) \leq 0 \end{cases} \quad (3)$$

$$Dev(u,t,c) = \frac{\sum_{a \epsilon N}(R(a,t,c) - R(a,t)) \times sim(a,u)}{\sum_{a \epsilon N} sim(a,u)} \quad (4)$$

$Dev(u,t,c)$ is used to estimate the rating deviation of u's rating on item t with and without considering context c. It tells that it is appropriate to recommend the item t to u in context c if the deviation is positive. Otherwise, we set the predicted rating as zero to remove this item from the recommendation list. To compute $Dev(u,t,c)$, we utilize Equation 4. N denotes the top-K nearest neighbors for user u who rated item t in our knowledge base, and user a is a user neighbor in set N. The function $sim(a,u)$ is used as a weight to aggregate the contributions by these neighbors. This similarity can be calculated by the popular user-user similarity metrics, e.g., cosine similarity. $R(a,t)$ denotes neighbor a's rating on item t, while $R(a,t,c)$ tells a's rating on item t in contexts c.

The model by Equation 3 is similar to PoF_Ngbr as shown in Equation 1, where we set the predicted rating value as zero to remove an inappropriate item from the recommendation list. From another perspective, we can also use a similar method in the PoF_Adj model to adjust the predicted ratings. It can be shown in Equation 5.

$$\widehat{R}(u,t,c) = \widehat{R}(u,t) + Dev(u,t,c) \tag{5}$$

Due to the fact that the value of $Dev(u,t,c)$ could be a positive or negative one, it implies that we should apply a bonus or penalty to the user u's rating on item t if the context information c is taken into account.

In short, by introducing the context rating deviations, these two basic models are able to contextualize the predicted ratings without considering contexts – either filtering out irrelevant items or adjusting the predicted ratings.

Improvements

The problem of rating sparsity in the context-aware data is a well-known challenge in CARS, especially for the approaches that use contexts as filters. It refers to the situation that users actually did not place multiple ratings on the items in different contextual situations. This problem can be alleviated by the contextual modeling approaches, but it becomes more serious in the pre-filtering and post-filtering methods. Take our proposed model for example, in Equation 4, there are probably a limited number of (or even no) user neighbors who rated item t in the same context c. It results in unreliable computations for $Dev(u,t,c)$, which further deceases the accuracy of our post-filtering models.

One solution is to estimate $R(a,t,c)$ based on rating profiles with similar contexts to c, rather than finding an exact matching by using c. It is because the user neighbor a did not rate item t in the same context c, but he or she may rate the item t in other context situations which are similar to c. Therefore, we can estimate $R(a,t,c)$ by Equation 6.

$$\widehat{R}(a,t,c) = \frac{\sum_{c* \epsilon S} R(a,t,c*) \times sim(c,c*)}{\sum_{c* \epsilon S} sim(c,c*)} \tag{6}$$

First of all, we find the top-K nearest neighbor set N, where each neighbor a rated item t in our data. We extract all the set of contextual situations, S, for the pair of user a and item t. $c*$ is used to represent a context situation in set S. We aggregate the ratings based on $R(a,t,c*)$, and weigh it by the similarity between the contexts c and $c*$. Finally, we use this estimated rating to replace the $R(a,t,c)$ in Equation 4.

The remaining challenge is estimating the similarity between two contexts. Codina, et al. (Codina, Ricci, and Ceccaroni 2015) computed the context similarity by learning the context representations. They build a user-context and an item-context rating matrix based on the original context-aware data set, so that each context condition can be represented by a vector. They compute the context similarity between two contexts by aggregating the cosine similarity based on each vector representations of context conditions. This could be a simple pre-processing stage to obtain the similarity between two contexts. Afterwards, we can utilize the results to estimate $R(a,t,c)$ by Equation 6.

Experimental Results & Discussions

It is well-known that it is difficult to find the context-aware rating data sets, not to mention that we are willing to evaluate the proposed models in the area of mobile computing. The existing context-aware data sets are either small or sparse, since most of them were collected from surveys. In our paper, we adopt two data sets as follows:

- The *Frappe* data (Baltrunas et al. 2015) comes from the mobile usage in the app named as Frappe which is a context-aware app discovery tool that will recommend the right mobile apps for the right moment in smart phones. There are three context dimensions: the time of the day, day of the week and location. This data captures the frequencies of an app used by each user within 2 months. We apply a log transformation on the frequency and view them as the ratings (scale 0 to 4.46). There are 87,580 ratings given by 957 users on 4,082 mobile apps in this data.
- The *South Tyrol Suggests (STS)* data (Braunhofer et al. 2013) was collected from a mobile app which provides context-aware suggestions for attractions, events and restaurants for the tourism in South Tyrol, Italy. There are 14 contextual dimensions, such as budget, companion, daytime, mood, season, weather, etc. There are 2,354 ratings (scale 1 to 5) given by 325 users on 249 items.

We choose matrix factorization (MF) as the traditional recommendation algorithm in the pre-filtering and post-filtering models, due to its ease and popularity. We select UISplitting as the pre-filtering technique, since it is proved as the best performing pre-filtering model (Zheng, Burke, and Mobasher 2014). In terms of the post-filtering techniques, we select PoF_Ngbr (Panniello et al. 2009) and PoF_Adj (Ramirez-Garcia and Garca-Valdez 2014) as the baselines. In addition, we add context-aware matrix factorization (CAMF) (Baltrunas, Ludwig, and Ricci 2011) which is a popular contextual modeling approach based on MF.

In this paper, we propose two basic approaches as shown by Equation 3 and 5. Due to limited space, we only include the model in Equation 5 in this paper, where the two models have similar performance and the one by Equation 5 is slightly better. We refer this basic approach as PoF_Dev. Additionally, we incorporate similarity of contexts to improve the model by Equation 6, while we refer this improved model as PoF_Dev+.

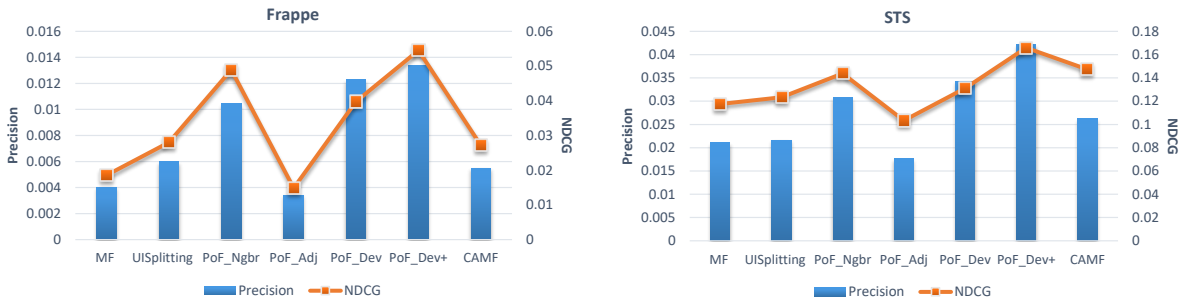

Figure 1: Experimental Results

We apply a 5-fold cross evaluation on these data sets, and use precision and normalized discounted cumulative gain (NDCG) as the metrics to evaluate these algorithms based on the top-10 item recommendations. The experimental results are presented by Figure 1, while the bars describe the results in precision, and the curves depict NDCG. The results by MF tell the performance by the matrix factorization technique without considering contexts.

We can observe that most of the context-aware recommendation models can beat the MF approach in these two data sets, expect the PoF_Adj. Among the four post-filtering approaches, we can find out that our proposed methods, PoF_Dev and PoF_Dev+, can outperform other post-filtering models in precision, but PoF_Dev does not present advantages over the PoF_Ngbr in NDCG. PoF_Dev+ becomes the best performing post-filtering technique in terms of the results in both precision and NDCG. It improves the precision and NDCG over the PoF_Dev model by 8% and 4% in the Frappe data, 2.5% and 3% in the STS data, respectively. In comparison with the selected pre-filtering (i.e., UISplitting) and contextual modeling (i.e., CAMF) approaches, both PoF_Dev and PoF_Dev+ outperform these selected baselines in these two data sets.

We also evaluate these approaches on other data sets rather than the two selected data sets in the mobile domains. We can summarize the results as follows: PoF_Adj can outperform MF in other data sets, but it is still the worst post-filtering model. PoF_Dev+ becomes the best performing post-filtering technique due to the fact that we incorporate the similarity of contexts to better estimate the user's rating on the items in specific contexts. CAMF can beat PoF_Dev+ in some other context-aware data sets.

Conclusions and Future Work

In this paper, we propose simple but effective post-filtering models by introducing the contextual rating deviations. We further discuss the improved model by incorporating the similarity of contexts to alleviate the sparsity problem. The results based on the two mobile data sets demonstrate the effectiveness of our proposed models. PoF_Dev+ is revealed as the best performing post-filtering model. In our future work, we will explore more ways to better estimate or learn the similarity of contexts (Zheng, Mobasher, and Burke 2015) which will be helpful to further improve the proposed PoF_Dev+ model.

References

Adomavicius, G.; Mobasher, B.; Ricci, F.; and Tuzhilin, A. 2011. Context-aware recommender systems. *AI Magazine* 32(3):67–80.

Baltrunas, L.; Church, K.; Karatzoglou, A.; and Oliver, N. 2015. Frappe: Understanding the usage and perception of mobile app recommendations in-the-wild. *CoRR* abs/1505.03014.

Baltrunas, L.; Ludwig, B.; and Ricci, F. 2011. Matrix factorization techniques for context aware recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, 301–304. ACM.

Braunhofer, M.; Elahi, M.; Ricci, F.; and Schievenin, T. 2013. Context-aware points of interest suggestion with dynamic weather data management. In *Information and Communication Technologies in Tourism 2014*. Springer.

Codina, V.; Ricci, F.; and Ceccaroni, L. 2015. Distributional semantic pre-filtering in context-aware recommender systems. *User Modeling and User-Adapted Interaction*.

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 *Proceedings of the third ACM conference on Recommender systems*, 265–268. ACM.

Ramirez-Garcia, X., and Garca-Valdez, M. 2014. Post-filtering for a restaurant context-aware recommender system. In *Recent Advances on Hybrid Approaches for Designing Intelligent Systems*, volume 547, 695–707. Springer.

Zheng, Y.; Burke, R.; and Mobasher, B. 2014. Splitting approaches for context-aware recommendation: An empirical study. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, 274–279. ACM.

Zheng, Y.; Mobasher, B.; and Burke, R. 2014. CSLIM: Contextual SLIM recommendation algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems*, 301–304. ACM.

Zheng, Y.; Mobasher, B.; and Burke, R. 2015. Integrating context similarity with sparse linear recommendation model. In *User Modeling, Adaptation, and Personalization*. Springer Berlin Heidelberg. 370–376.

Zheng, Y. 2015. A revisit to the identification of contexts in recommender systems. In *Proceedings of the ACM Conference on Intelligent User Interfaces Companion*, 133–136.