Group-based Collaborative Filtering Supported by Multiple Users' Feedback to Improve Personalized Ranking

Arthur F. da Costa, Marcelo G. Manzato and Ricardo J. G. B. Campello
Institute of Mathematics and Computer Science
University of São Paulo
São Carlos, SP, Brazil
{fortes,mmanzato,campello}@icmc.usp.br

ABSTRACT

Recommender systems were created to represent user preferences for the purpose of suggesting items to purchase or examine. However, there are several optimizations to be made in these systems mainly with respect to modeling the user profile and remove the noise information. This paper proposes a collaborative filtering approach based on preferences of groups of users to improve the accuracy of recommendation, where the distance among users is computed using multiple types of users' feedback. The advantage of this approach is that relevant items will be suggested based only on the subjects of interest of each group of users. Using this technique, we use a state-of-art collaborative filtering algorithm to generate a personalized ranking of items according to the preferences of an individual within each cluster. The experimental results show that the proposed technique has a higher precision than the traditional models without clustering.

CCS Concepts

• **Information systems** → **Clustering; Personalization;**

Keywords

Recommender Systems; Collaborative Filtering; Data Clustering

1. INTRODUCTION

Recommender Systems (RS) represent a technology that uses statistical techniques and machine learning to make recommendations of items to users based on a history of past activities. RS have become an important research area since the mid-90s, because it supports users to deal with the information overload problem existent on the Web [13].

The recommendation task can be seen as a prediction problem: the system tries to predict the relevance of certain items to a user and then sorts them according to the provided relevance values. The importance of an item is usually represented by a numerical value that reflects the degree of user's interest as provided herein. The result of an RS is usually a set of items ordered in descending order by importance scheduled for a given user [1].

Traditionally, recommender systems employ filtering techniques and machine learning information to generate appropriate recommendations to the user's interests from the representation of his profile. However, other techniques, such as Neural Networks, Bayesian Networks and Association Rules are also used in the filtering process [1]. The most used types of filtering are currently: content-based, responsible for selecting information based on descriptions of items which were rated in the past. Email messages filtered out to the trash messages containing unwanted words is an example of content-based filtering. The second approach is collaborative filtering, which is based on the relationship between people and items. A simple example is the selection of electronic messages based on the relationship between sender and recipient of a message. Finally, the hybrid approach which combines the filtering based on content and collaborative filtering [13].

The collaborative filtering approach tends to calculate its recommendations based on all users' interactions that were given to them [5]. However, this task can increase the computational cost and cause problems in the final result. Once the recommendation will be generated based on the preferences of undistinguished users, the system may recommend items for users who has no interest in those selected content. Another problem is that depending on the size of the database, computational cost is very high in order to process all the information. A possible solution to these problems would be the previous construction of groups of users with similar interests, and further recommendation, similarly to when recommending a content to a friend. For the person who receives the recommendation, it works as a filter or a particular view of a universe of possibilities usually inaccessible. It can also take into consideration the preference of those who are looking for suggestions and not just those who make it. It is also possible to make recommendations based on the opinions of others. As for instance, someone who is not an admirer of the jazz genre may be recommended based on what his friends enjoy, except this style which is unpleasant for him. Still, the recommendation may include explanations of how it was generated to allow recipient to assess.

This work proposes a technique to generate more accurate recommendation of items using groups of users with

WebMedia '16, November 08-11, 2016, Teresina, PI, Brazil

DOI: http://dx.doi.org/10.1145/2976796.2976852

similar preferences. We analyze a variety of users' interaction paradigms in order to obtain richer information about their interests. Such rich information is then used to create groups of similar users. The recommendation list for each user of a particular group is generated using a recommendation algorithm based on collaborative filtering. We evaluated our proposal by comparison with two state-of-the-art collaborative recommenders (User KNN and BPR MF), and we compared different types of recommendations generated from our approach to demonstrate the proposal's generality. The experiments were executed with two real datasets from different domains: the first is MovieLens, which contains data from a movies reviews system; and the second is Last FM, a music website. Our study shows that the proposed group-based approach is able to provide better accuracy than individual recommenders.

This paper is structured as follows: Section 2 addresses the related work; Section 3 describes techniques which are explored in this work; Section 4 presents the proposal in details; Section 5 reports the evaluation executed in the system; finally, Section 6 as devoted the final remarks and future works.

2. RELATED WORK

In this section, we review some work related to our proposal. First, we depict approaches related to different types of interaction in recommender systems, and then, we provide a review of data mining approaches in recommender systems.

2.1 Multiple Interactions Based Approaches

With the increasing number of interactions between users and content, several studies have emerged in order to work with the integration of these interactions, so that more information about the users preferences are gathered by the systems. The recommendation systems can be extended in various ways in order to improve the understanding of users and items, including, for example, new types of interaction in the recommendation process and making the combination of them [13].

The SVD algorithm proposed in [8] uses explicit (ratings) and implicit (viewing history) information from users in a factorization model, called SVD++. Another factorization model, called Factorization Machines (FM) [14], can consider many types of information regarding users, items and/or their interactions. These techniques have the drawback that they process only certain types of interactions, with little capability of extension to other different types. In recent studies, Costa et al. [4] developed an ensemble recommender technique, called Ensemble BPR Learning, to unify different types of feedback from users, processed in different recommendation techniques. While this model is extensible for any types of user's interaction, its learning phase has high computational cost, since it depends on the execution of several recommendation techniques beforehand.

The present work differs from the aforementioned since it explores an alternative to build user profiles, in a preprocessing step, that summarizes the interactions made by users in the system, producing users groups with similar behavior.

2.2 Data Mining Based Approaches

In this study, we have used data mining techniques to cluster users with similar preferences to generate recommendations based on their group. Several researchers have proposed recommender systems for online personalization through data mining to provide recommendation services. This kind of recommendation system is used to predict the user navigation behavior and their preferences using web log data.

Kim et al. [7] present an improved recommendation algorithm for CF. The algorithm uses the K-Means Clustering method to reduce the search space. It then utilizes a graph approach to the best cluster with respect to a given test customer in selecting the neighbors with higher similarities as well as lower similarities. The graph approach allows us to exploit the transitivity of similarity. The algorithm also considers the attributes of each item. The work developed by Wen and Zhou [17] presents an improved collaborative filtering recommendation algorithm based on dynamic item clustering method. A similarity threshold limits the similarity between clusters. By calculating the similarity between the current item and the cluster center, it chooses the greatest similitude cluster, and then it finds the target items' nearest neighbors. Li et al. [9] propose an improved collaborative filtering method based on user ranking and item clustering, in which the users are classified and ranked in multiple item clusters by computing their rating qualities based on the previous rating records, and items are recommended for target users according to their similar users with high-ranks in different item categories. Experiments on real world data sets have demonstrated the effectiveness of their approach.

Our approach is different than their work in the sense that we analyze various paradigms of users' interactions on a particular item, in order to create groups of users with similar preferences and thus, a more accurate personal profile. The advantages of this approach is the ease of extending the template for insertion of other types of interactions; the reduced time and computational processing, considering that the sets of data would be reduced to a single cluster before computing the recommendation. Our contribution, therefore, can be considered the proposal of a recommender system based on multiple feedback types of similar users' groups.

3. RELATED MODELS OVERVIEW

This study involves the pre-processing of data by means of data mining techniques and the recommendation of items through filtering algorithms. The following sections present the main concepts covered in this paper.

3.1 Notation

We use special indexing letters to distinguish users and items: a user is indicated as u and an item is referred as i, j; and r_{ui} is used to refer to either explicit or implicit feedback from a user u to an item i. In the first case, it is an integer provided by the user indicating how much he liked the content; in the second, it is just a boolean indicating whether the user consumed or visited the content or not. The prediction of the system about the preference of user u to item i is represented by $\hat{r}_{ui}$, which is a floating point value guessed by the recommender algorithm. The set of

pairs (u, i) for which r_{ui} is known are represented by the set $K = \{(u, i)|r_{ui} \text{ is known}\}$. Additional sets used in this paper are: $N(u)$ to indicate the set of items for which user u provided an implicit feedback, and $\bar{N}(u)$ to indicate the set of items that are unknown to user u.

3.2 k-medoids Clustering Algorithm

Clustering is the process of grouping a set of objects into clusters so that objects within a cluster are similar to each other but are dissimilar to objects in other clusters [6]. K-means clustering [10] and partitioning around medoids are well known techniques for performing non-hierarchical clustering. K-means clustering iteratively finds the k centroids and assigns every object to the nearest centroid, where the coordinate of each centroid is the mean of the coordinates of the objects in the cluster. Unfortunately, K-means clustering is known to be sensitive to the outliers although it is quite efficient in terms of the computational time. For this reason, K-medoids clustering are sometimes used, where representative objects called medoids are considered instead of centroids. Because it is based on the most centrally located object in a cluster, it is less sensitive to outliers in comparison with the K-means clustering [12].

k-medoids clustering algorithm is a partition-based clustering algorithm based on k-means. It attempts to minimize the distance between points labeled to be in a cluster and a point designated as the center of that cluster [12]. In contrast to the k-means algorithm, k-medoids chooses datapoints as centers (medoids or exemplars) and works with an arbitrary matrix of distances between datapoints. It is more robust to noise and outliers as compared to k-means because it minimizes a sum of pairwise dissimilarities instead of a sum of squared Euclidean distances [12]. A medoid can be defined as the object of a cluster whose average dissimilarity to all the objects in the cluster is minimal, i.e., it is a most centrally located point in the cluster. The implementation of the K-medoids used in this paper is explained as follows:

K-Medoid Clustering Algorithm

1. Initialization: randomly select (without replacement) k of the n data points as the medoids.
2. Associate each data point to the closest medoid. (Using a dissimilarity measure like cosine, Pearson, etc.)
3. For each medoid m
   - For each non-medoid data point o
     - Swap m and o and compute the total cost of the configuration
4. Select the configuration with the lowest cost.
5. Repeat steps 2 to 4 until there is no change in the medoid.

The advantage of this k-medoids implementation is that large datasets can be efficiently classified and its convergence is proved regardless of the dissimilarity measure. Furthermore, it is reliable in theory, simple and fast [12]. In this work, to generate users' groups, we use the k-medoids algorithm to group users with similar preferences. This algorithm was chosen to accept different types of distance and not only metrics, such as Manhattan and Euclidean, which do not work well for recommendation approaches.

3.3 Recommendation Algorithms

We evaluate our proposal with two recommender algorithms: a neighborhood-based [8, 13] and a matrix factorization [15] approaches. The following subsections detail each algorithm.

3.3.1 BPR MF: Recommendation Algorithm

The BPR MF approach [15] consists of providing personalized ranking of items to a user according only to implicit feedback (e.g. navigation, clicks, etc.). An important characteristic of this type of feedback is that we only know the positive observations; the non-observed user-item pairs can be either an actual negative feedback or simply the fact that the user does not know about the item's existence. The authors have proposed a generic method for learning models for personalized ranking, where instead of training the model using only the user-item pairs, they also consider the relative order between a pair of items, according to the user's preferences [15]. It is inferred that if an item i has been viewed by user u and j has not ($i \in N(u)$ and $j \in \bar{N}(u)$), then $i >_u j$, which means that he prefers i over j. Each user u is associated with a user-factors vector $p_u \in \mathbb{R}^f$, and each item i with an item-factors vector $q_i \in \mathbb{R}^f$.

It is important to mention that when i and j are unknown to the user, or equivalently, both are known, then it is impossible to infer any conclusion about their relative importance to the user. To estimate whether a user prefers an item over another, Rendle et al. proposed a Bayesian analysis using the likelihood function for $p(i >_u j|\Theta)$ and the prior probability for the model parameter $p(\Theta)$. The final optimization criterion, BPR-Opt, is defined as:

$$\text{BPR-Opt} := \sum_{(u,i,j)\in D_K} \ln \sigma(\hat{s}_{uij}) - \Lambda_\Theta ||\Theta||^2 \quad , \qquad (1)$$

where $\hat{s}_{uij} := \hat{r}_{ui} - \hat{r}_{uj}$ and $D_K = \{(u, i, j)|i \in N(u)\ \&\ j \in \bar{N}(u)\}$. The symbol Θ represents the parameters of the model, Λ_Θ is a regularization constant, and σ is the logistic function, defined as: $\sigma(x) = 1/(1 + e^{-x})$.

For learning the model, the authors also proposed a variation of the stochastic gradient descent technique, denominated LearnBPR, which randomly samples from D_K to adjust Θ.

3.3.2 User KNN

This recommendation algorithm is the well-known User KNN, whose details can be found in [8, 13]. We adopted this algorithm because of its well-acceptance, and because it can be intuitively extended to include other information. The main goal of the algorithm is to find similar users and predict the best items for them based on their similar items.

In this way, a score is predicted for a unknown user-item pair $\hat{r}_{ui}$ considering the interaction that other users with similar preferences to u have assigned to item i. To find similar users, a measure of similarity p_{uv} is employed be-

tween their vectors. The similarity measure may be based on several similarity measures, such as Pearson correlation coefficient or cosine similarity. The final similarity measure is a retracted coefficient, s_{uv}:

$$s_{uv} = \frac{n_{uv}}{n_{uv} + \lambda_1} p_{uv}, \tag{2}$$

where n_{uv} is the number of items that users u and v have in common, and λ_1 is a regularization constant, set as 100 according to suggestions found in the literature [8].

Using the similarity values obtained, the algorithm identifies the k most similar users of u who evaluated item i, denoted as $S_u^k(i;v)$, and performs a score prediction based on the interactions of the k similar users weighted by their similarity towards u [8]. Then the final score is predicted using the k most similar users through the Equation (3).

$$\hat{r}_{ui} = \frac{\sum_{v \in S^k(i;u)} s_{uv}}{\textit{Number of neighbors}} \quad , \tag{3}$$

4. PROPOSED METHOD

In this paper, we propose a technique capable of generating recommendations based on users groups' preferences. Our approach consists of a pre-processing step, responsible for combining users into groups according different type of interactions in the system with a particular item. In this way, the combination of assigning tags and user history during navigation, for example, can be made to improve the quality of the groups, since we can better represent the behavior of each user. The recommendation list for each user of a particular group is generated using a recommendation algorithm based on collaborative filtering. In this work, we adopted three algorithms: k-medoids, BPR MF and User kNN described in Section 3. The first one is used to generate groups based on the similarity among users, while the second and the third are used to generate recommendations based on interactions of each group.

Figure 1 illustrates all steps involved in our technique. Multiple users' interactions are captured and used to generate user vs. item matrices, where each cell in these matrices is a value containing the relevance feedback of each interaction type. These matrices are then used to compute the distance of users using some dissimilarity measure. After this step, we generated a single distance matrix resulting from the combination of the others. Then, we send this matrix to the clustering module to generate users' groups. Each of these clusters corresponds to users who have similar interests on particular subjects. Finally, based on these groups, we compute particular recommendations to each user in the database, using the navigation history (implicit feedback) of each group. The browsing history of each group was built using all kinds of interactions considered by the algorithm, making explicit interactions in binary form and removing duplicate entries.

Particularly in this paper, we adopted different types of implicit feedback, although our approach accepts explicit information too. As implicit feedback, we considered two types: i) whether a user assigned a tag or not to an item; and ii) his navigation history, inferred from other interactions (if the user has some kind of interaction, then he has viewed the item). As shown in Figure 1, we used the BPR MF algorithm to generate a personalized ranking for each user using a binary matrix (user per item), where the values will be one if user u made some interaction with an item i and zero otherwise. There are three phases in our technique: data representation; finding the nearest neighbor; and generating the lists of recommendations. The following subsections detail each of them.

4.1 Data representation

The algorithm inputs are represented by user × item interactions matrices, e.g if we consider ratings and tags we will have two input matrices. Each cell in the matrix represents the interactions made by users on items. Different methods can be used to represent interactions. Discrete values (such as 1, 2, 3, 4, 5) can be used to represent degrees of user's preferences towards the items; numerical values can be used to characterize the amount of times a user accessed an item; boolean values (such as 0 and 1) to represent whether a user assigned or not a tag on an item. Each cell of each matrix gets its respective type of interaction. If a user has not interacted with the corresponding item, its value in the matrix is 0, otherwise it will be specific for each type of interaction. For example, if the user explicitly rated an item, this value will be the provided rating; if he has only viewed the item, this value will be 1.

4.2 Finding the nearest neighbors

The dissimilarity between a user and other users is acquired based on their interactions. We use Cosine angle and Pearson correlation to calculate how two users are alike, considering all the interactions made by users on all items in the database. These metrics were chosen because: i) they discards the matrix cells that has no interaction, and ii) they are the metrics most commonly used in the area of recommender systems [13, 1]. This is particularly useful because we can not assume that users are similar based on the fact they have not interacted with certain items. For each interaction, we used these metrics to generate a new matrix of $M \times M$ users, which represents the dissimilarity among users. Cosine similarity between two users u and v is represented by Equation (4):

$$d_{(u,v)}^{cosine} = 1 - \frac{uv}{\|u\|_2 \|v\|_2} \quad , \tag{4}$$

where $d_{(u,v)}$ is the distance of each type of interaction between two users, $\|*\|_2$ is the 2-norm of its argument $*$, and $u \cdot v$ is the dot product of u and v. The Pearson correlation is defined by:

$$d_{(u,v)}^{pearson} = 1 - \frac{\sum(u,v)}{\sigma_u \sigma_v}, \tag{5}$$

where Σ is the covariance between two users u and v, and σ is the standard deviation between them.

To combine the distances of each type of interaction in a single distance matrix, we computed a weighted average of the values, as shown in Equation (6).

$$d_{(u,v)}^{final} = \frac{1}{N_f} \sum_{n=1}^{|N_f|} \alpha_n d_n \tag{6}$$

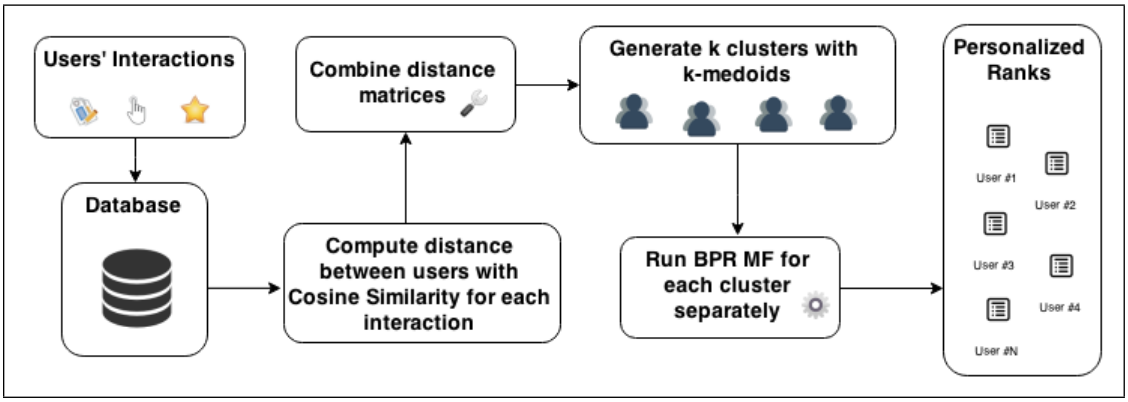


Figure 1: Schematic visualization of the proposed technique.

where N_f is the number of interactions' types and α_* are variables used to weight each interaction type. It is defined as:

$$\alpha = \frac{N_{uv}(N_u + N_v)}{(N_u N_v)}. \quad (7)$$

In the equation above, N_u and N_v denote the number of interactions made by users u and v, respectively, and N_{uv} denotes the number of interactions in common of these users.

After computing the distance matrix, we use k-medoids presented in Subsection 3.2, to generate groups. Here, k data objects are selected randomly as medoids to represent k clusters and all remaining data objects are placed in a cluster having medoid nearest (or most similar) to that data object. After processing all data objects, a new medoid is determined which can represent cluster in a better way and the entire process is repeated. Again all data objects are bound to the clusters based on the new medoids. In each iteration, medoids change their location step by step; in other words, medoids move in each iteration. This process is continued until all medoids stop moving over each iteration. As a result, k clusters are found representing a set of n data objects.

4.3 Generating the lists of recommendations

We have used the state-of-the-art CF-based algorithms to process the interactions of each cluster and generate a list of recommended items for each user in that cluster. At this stage, each user receives personalized recommendations based on his behavior and his neighbors behavior. The algorithm is responsible for assembling a matrix that contains all users and items of a given group k and individual interactions of each user to predict items that he can enjoy, both highlighting the items he visited as the ones he has not. Finally, we concatenate the rankings generated for each user in a single ranking with all users. The proposed technique generates four values for k from training set of samples. In this step, the sample is divided into training and test in order to verify the precision and MAP values generated by this sample; then this procedure is repeated n times, returning the best values for k.

5. EVALUATION

To evaluate our approach we first check which of the dissimilarity metric provided the best accuracy in the proposed model using all the feedback types. Furthermore, we compare the algorithms having the best combination of interactions with CF-based algorithms, namely BPR MF [15] and User KNN [8], presented in Subsection 3.3. In this way, we execute the proposed approach applied to each Group-based BPR MF algorithm (GB-BPR MF) and Group-based User KNN (GB-User KNN), respectively. The algorithms implementation used in our work is available in the MyMediaLite library [3].

5.1 Datasets

The system evaluation was based on two datasets provided by Cantador et al. [3]. Last fm 2k consists of 92,834 user-listened artist relations, 186,479 interaction tags applied by 1,892 users to 17,632 artists. As feedback types, we considered: whether a user tagged an item or not; and the number of times the user has visited a particular item. MovieLens 2k consists of 10 million ratings, 100,000 interactions tags applied to 10,000 users and 72,000 movies. As explicit information, we used the ratings that users assigned to items to calculate the distance matrix, and as implicit information, we considered whether a user tagged an item or not and the navigation history to compute matrices and recommendations.

5.2 Methodology

We adapted the All But One [2] protocol for the construction of the ground truth and 10-fold-cross-validation. Given the data set, we randomly divided it into the same 10 subsets and for each sample we use $n-1$, these subsets of data for training and the rest for testing. The training set t_r was used to test the proposed technique and in the test set T_e we randomly separated an item for each user to create the truth set H. After that, the remaining items form the set of observable O, used to test the algorithm. To assess the outcomes of the systems we use evaluation metric Mean Average Precision (MAP) [16], as follows:

Mean Average Precision computes the precision considering the respective position of items in the ordered list. With this metric, we obtain a single accuracy score value for a set of test users T_e:

$$MAP(T_e) = \frac{1}{|T_e|} \sum_{j=1}^{|T_e|} AveP(R_j, H_j), \quad (8)$$

where R is the set of recommendations that the system computed, given the set of observables O, and the ground truth set H. The average precision (AveP) is given by

$$AveP(R_j, H_j) = \frac{1}{|H_j|} \sum_{r=1}^{|H_j|} [Prec(R_j, r) \times \delta(R_j(r), H_j)], \quad (9)$$

where $Prec(R_j, r)$ is the precision for all recommended items up to rank r and $\delta(R_j(r), H_j) = 1$, iff the predicted item at rank r is a relevant item ($R_j(r) \in H_j$) or zero otherwise.

We executed the clustering algorithm with different k values (between 2 and 30) and used the "knee finding" technique to choice the best number of clusters. For both datasets we deduce experimentally $k = 3$ as best value. For results we use Cosine Similarity, once our results indicate that this similarity is superior than the other measures such as Pearson Correlation, Jaccard measure, Euclidean measure that we tested.

In this work we used MAP@N, where N corresponds to the number of recommendations. We tested for the following values: $1, 3, 5$ and 10 in the ranks returned by the system. For each configuration and measure, the 10-fold values are summarized by using mean and standard deviation. In order to compare the results in statistical form, we apply the two-sided paired t-test with a 95% confidence level [11].

5.3 Results

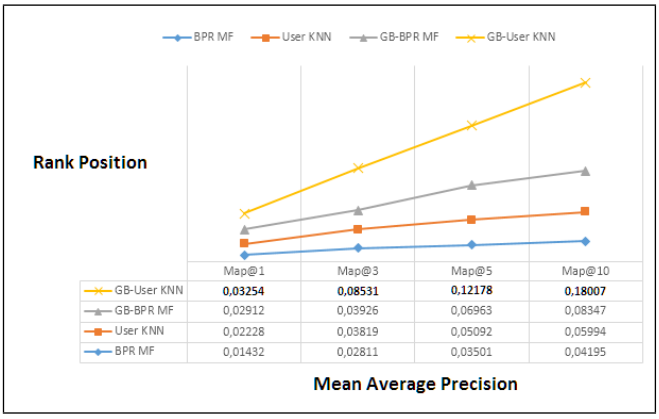
The results of the experiments in each of the datasets are discussed in the following subsections.

5.3.1 *Last fm 2k*

Tables 1 and 2 show the results using only one type of interaction by user (navigation history and tag, respectively) to compute similarity between users. Figure 2 shows the results using the combination of both types of interactions in the proposed methods against the best results in the baselines. The best results are highlighted in bold.

We note that MAP has a tendency for higher values as the number of returned items increases. This can be explained because MAP only considers the relevant items and their positions in the ranking. Thus, as more items are returned, the number of relevant items is also increased.

We observed that regardless of the number of groups or types of interactions, the use of clustering algorithms considerably improves the results of recommendation, proven by the t-student analysis ($p < 0.05$). This is because instead of generating recommendations through all interactions in the base, we use only related interactions with users' interactions, discarding items that the user probably would never use. The combination of the two types of feedback (navigation history and tag) resulted in better outcomes for all top N recommendations. This is because the more information a user has, the better we can relate them to other users with similar tastes and we can thus generate more accurate recommendations for him.

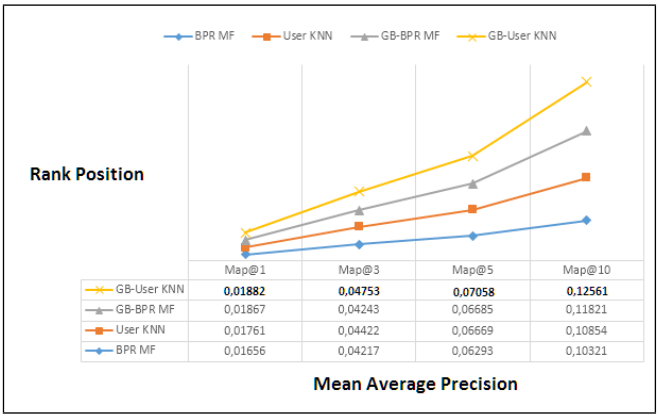


	Map@1	Map@3	Map@5	Map@10
GB-User KNN	**0,03254**	**0,08531**	**0,12178**	**0,18007**
GB-BPR MF	0,02912	0,03926	0,06963	0,08347
User KNN	0,02228	0,03819	0,05092	0,05994
BPR MF	0,01432	0,02811	0,03501	0,04195

Figure 2: Graph and table comparing the MAP with two types of interaction (Last fm 2k).

5.3.2 *MovieLens 2k*

Tables 3 and 4 show the results of this evaulation in the MovieLens 2k dataset, using tags and ratings as interaction types. Figure 3 illustrates the algorithms' performance in Top@N vs. MAP graphs with two types of interactions.



	Map@1	Map@3	Map@5	Map@10
GB-User KNN	**0,01882**	**0,04753**	**0,07058**	**0,12561**
GB-BPR MF	0,01867	0,04243	0,06685	0,11821
User KNN	0,01761	0,04422	0,06669	0,10854
BPR MF	0,01656	0,04217	0,06293	0,10321

Figure 3: Graph and table comparing the MAP with both types of interactions (MovieLens 2k).

The results demonstrate that the proposed technique was able to achieve better accuracy in two different domains (recommendation of artists and movies), proven by the t-student analysis ($p < 0.05$). Most of the results showed improvement in the accuracy of the recommendation, especially when using more than one type of feedback in the process. This emphasizes that we can represent user's behavior better when we have a large number of information about him.

The overall results obtained and described in the paper are small because of the evaluation protocol used in the experiments. The All But One hides one item from each user in the test set and considers it as the ground truth. As we are recommending top N items, the MAP will decrease because the system thinks there are N relevant items, although the protocol has set only the hided item as relevant. In this way, it is important to rely only on the differences among the approaches, and we managed to increase the results of our proposal when compared to the baselines. As shown in the experiments, our approach is flexible and ex-

Table 1: Comparative MAP table using navigation history (Last fm 2k)

		Top 1	Top 3	Top 5	Top 10
BPR MF	MAP	0,01273	0.03342	0.04456	0.06259
	Standard deviation	0.000043	0.000014	0.000024	0.00011
User KNN	MAP	0.02175	**0.04721**	0.06206	0.07745
	Standard deviation	0.000087	0.000034	0.000217	0.000089
GB-BPR MF	MAP	0.01346	0.03065	0.04509	0.06803
	Standard deviation	0.000132	0.000187	0.000042	0.000131
GB-User KNN	MAP	**0.02374**	0.04573	**0.06532**	**0.07908**
	Standard deviation	0.000017	0.000123	0.000133	0.000054

Table 2: Comparative MAP table using Tags (Last fm 2k)

		Top 1	Top 3	Top 5	Top 10
BPR MF	MAP	0.01538	0.04509	0.06631	0.10291
	Standard deviation	0.000127	0.000051	0.000126	0.000012
User KNN	MAP	0.02864	0.08381	**0.11724**	0.15542
	Standard deviation	0.000152	0.000082	0.000012	0.000215
GB-BPR MF	MAP	0.01754	0.04954	0.07165	0.12658
	Standard deviation	0.000041	0.000504	0.000145	0.000097
GB-User KNN	MAP	**0.03154**	**0.09014**	0.11297	**0.17541**
	Standard deviation	0.000026	0.000071	0.000259	0.000195

Table 3: Comparative MAP table using navigation history (MovieLens 2k)

		Top 1	Top 3	Top 5	Top 10
BPR MF	MAP	0.01594	0.04208	0.06298	0.10226
	Standard deviation	0.000043	0.000014	0.000024	0.00011
User KNN	MAP	0.01809	0.04432	0.06206	0.10882
	Standard deviation	0.000087	0.000034	0.000217	0.000089
GB-BPR MF	MAP	0.01346	**0.04503**	0.06307	0.10903
	Standard deviation	0.000132	0.000187	0.000042	0.000131
GB-User KNN	MAP	**0.01915**	0.04398	**0.06612**	**0.11054**
	Standard deviation	0.000017	0.000123	0.000133	0.000054

Table 4: Comparative MAP table using ratings (MovieLens 2k)

		Top 1	Top 3	Top 5	Top 10
BPR MF	MAP	0.00214	0.00681	0.01071	0.01842
	Standard deviation	0.000043	0.000014	0.000024	0.00011
User KNN	MAP	**0.00233**	0.00775	0.01147	0.02075
	Standard deviation	0.000087	0.000034	0.000217	0.000089
GB-BPR MF	MAP	0.00119	0.00678	**0.01164**	0.01893
	Standard deviation	0.000132	0.000187	0.000042	0.000131
GB-User KNN	MAP	0.00209	**0.00807**	0.01143	**0.02106**
	Standard deviation	0.000017	0.000123	0.000133	0.000054

tensible to different algorithms and domains, although some of these configurations result in marginal improvements over the baselines (in particular KNN-based algorithms).

6. FINAL REMARKS

It is very crucial for a recommender system to have the capability of making accurate prediction by analyzing and retrieving customer's preferences. Although collaborative filtering is widely used in recommender systems, some efforts to overcome its drawbacks have to be made to improve the prediction quality. Selecting the proper neighbors plays an important role to improving the prediction quality.

We have proposed a technique that yields better recommendation accuracy using users' groups, generated from the similarity between them. It considers different type of interactions of each user for customer's preferences and low similarities as well. The experimental results showed that our algorithm provides the better prediction accuracy than baselines in two datasets. The main advantages of our approach are extensibility and flexibility, once it enables developers to use different recommender and clustering algorithms, dissimilarity metrics and different types of feedback.

As future work, we plan to evaluate our approach with additional datasets from other domains in order to check

the accuracy with different information types. Furthermore, we plan to consider community detection in graphs to select better users' groups to make the recommendation.

7. ACKNOWLEDGMENTS

We would like to acknowledge CAPES and CNPq for the financial support.

8. REFERENCES

[1] M. B.-R. F. T. A. Adomavicius, G. Context-aware recommender systems. *AI Magazine*, pages 67–80, 2011.

[2] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 43–52, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[3] I. Cantador, P. Brusilovsky, and T. Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.

[4] A. da Costa Fortes and M. G. Manzato. Ensemble learning in recommender systems: Combining multiple user interactions for ranking personalization. In *Proceedings of the 20th Brazilian Symposium on Multimedia and the Web*, WebMedia '14, pages 47–54, New York, NY, USA, 2014. ACM.

[5] A. da Silveira Dias, L. K. Wives, and V. Roesler. Enhancing the accuracy of ratings predictions of video recommender system by segments of interest. In *19th Brazilian Symposium on Multimedia and the Web, WebMedia '13, Salvador, Brazil, November 5-8, 2013*, pages 241–248, 2013.

[6] J. Han, M. Kamber, and A. K. H. Tung. Spatial clustering methods in data mining: A survey. In H. J. Miller and J. Han, editors, *Geographic Data Mining and Knowledge Discovery, Research Monographs in GIS*. Taylor and Francis, 2001.

[7] T.-H. Kim, Y.-S. Ryu, S.-I. Park, and S.-B. Yang. An improved recommendation algorithm in collaborative filtering. In *Proceedings of the Third International Conference on E-Commerce and Web Technologies*, EC-WEB '02, pages 254–261, London, UK, UK, 2002. Springer-Verlag.

[8] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(1):1–24, Jan. 2010.

[9] W. Li and W. He. An improved collaborative filtering approach based on user ranking and item clustering. In M. Pathan, G. Wei, and G. Fortino, editors, *Internet and Distributed Computing Systems*, volume 8223 of *Lecture Notes in Computer Science*, pages 134–144. Springer, 2013.

[10] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[11] T. M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[12] J.-S. L. Park, Hae-Sang and C.-H. Jun. A k-means-like algorithm for k-medoids clustering and its performance. *Proceedings of ICCIE (2006)*, pages 102–117, 2006.

[13] F. R., L. R., and B. S. Introduction to recommender systems handbook. In *Recommender Systems Handbook*, pages 1–35. Springer US, 2011.

[14] S. Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, May 2012.

[15] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *CoRR*, abs/1205.2618, 2012.

[16] E. M. Voorhees and D. K. Harman. *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press, 2005.

[17] J. Wen and W. Zhou. An improved item-based collaborative filtering algorithm based on clustering method. In M. Pathan, G. Wei, and G. Fortino, editors, *Journal of Computational Information Systems*, volume 8 of *Lecture Notes in Computer Science*, pages 571–578. Springer Berlin Heidelberg, 2012.