Harnessing the Power of "Favorites" Lists for Recommendation Systems

Maryam Khezrzadeh
Dept. of Comp. Sci.
University of Victoria
PO Box 3055, STN CSC
BC, Canada V8W 3P6
maryamk@cs.uvic.ca

Alex Thomo
Dept. of Comp. Sci.
University of Victoria
PO Box 3055, STN CSC
BC, Canada V8W 3P6
thomo@cs.uvic.ca

William W. Wadge
Dept. of Comp. Sci.
University of Victoria
PO Box 3055, STN CSC
BC, Canada V8W 3P6
wwadge@uvic.ca

ABSTRACT

We propose a novel collaborative recommendation approach to take advantage of the information available in user-created lists. Our approach assumes associations among any two items appearing in a list together. We calculate sum of Bayesian ratings (SBR) of all lists containing an item pair as the strength of item-item associations in that pair. SBR takes into consideration not only the number of lists the items have co-appeared in, but also the quality of the lists. We collected a data set of user ratings for books along with Listmania lists on Amazon.com using Amazon Web Services (AWS). Our method shows superior performance to existing user-based and item-based collaborative filtering approaches according to the resulted MAE, coverage and F-measure.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms, Performance, Experimentation

Keywords

Recommender Systems, Collaborative Filtering, Association Analysis, Bayesian Rating

1. INTRODUCTION

Over the years, various approaches to build recommendation systems have been developed which utilize demographic, content or historical information. The collaborative Filtering (CF) method (*cf.* [6]), which is one of the most successful approaches to develop recommendation systems, helps people make choices based on opinions of other like-minded people. However, this classic CF approach, which is also known as user-based method, has some serious scalability and quality challenges associated with it [1]. These challenges have led to the design of a similar item-based scheme which utilizes item-item similarities rather than user-user similarities. The item-based collaborative filtering recommendation algorithms are claimed to address these two challenges simultaneously leading to the design of more accurate and more scalable recommender systems [7]. However, both user-based nad item-based CF suffer from common problems, such as sparsity and non-transitive item associations. To alleviate these problems, some alternative model-based recommender algorithms are explored which mainly aim at generating denser user-item interaction matrix or seeking alternative ways to drive item-item or user-user relationships. Dimensionality reduction techniques [8] and Latent Semantic Analysis [3] are two examples of such algorithms (refer to [5] for more background and a full list of references).

Nowadays, lots of online e-commerce and entertaining recommender systems, enable the users to create and manage lists of their favorite items. Many online music services like Last.fm, iLike and Pandora allow the users to create custom radio stations and playlists from any of the audio tracks in their music library. Amazon, the biggest online retailer in the US, recently has offered Listmania list creation service where users can put together a list of related books, DVDs, music, etc. MovieLens, Netflix, Youtube and many more are other examples of websites offering such services.

Unlike user profiles, which are currently used in many recommendation services, user-created favorites lists usually have a unique theme, topic and taste. The effort a user puts into creating a list is of great importance to the task of recommendation. We can consider each favorite list to represent a collection of highly related items. These relationships are approved by two groups of human experts: the user who creates the list and the viewers who vote for the list. A reviewer, signifies a list as helpful by voting *yes* to the list and voting *no* in case the list is not appealing as a related set of items. Incorporating the opinion of voters has significant benefits towards distinguishing strong item associations versus weak, limited and unreasonable item-item relationships.

There is no known research conducted to take advantage of the valuable information provided by custom-created favorites lists. This paper offers a novel, efficient and flexible way of extracting item-item relationships out of thees lists to enhance the quality of recommendations generated for users in such systems. We call this new approach, the Collective

RecSys'09 , October 22–25, 2009, New York, USA.

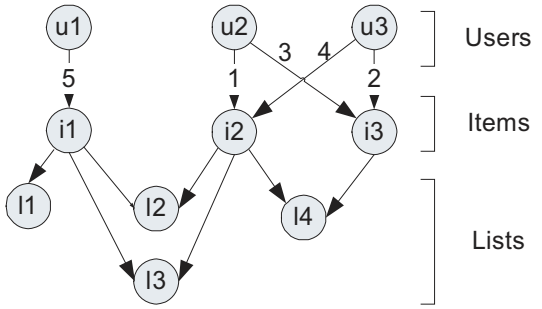

Figure 1: The proposed list-base model comprising user-item and item-item relationships

Intelligence ReCommendation (CIRC), since it draws upon the intelligence of the users creating cohesive collections as well as the opinion of the lists' viewers. CIRC operates on a preference model which comprises both user-items and item-item relationships, and infers a weighted item-item graph from the lists containing those items.

2. OUR MODEL

Figure 1 illustrates the proposed model consisting of `User`, `Item`, and `List` layers. We denote the set of `User` nodes by $U = \{u_1, \ldots, u_p\}$, the set of `Item` nodes by $I = \{i_1, \ldots, i_q\}$, and the set of `List` nodes by $L = \{l_1, \ldots, l_r\}$.

The number of yes/no votes for each `List` is assumed to be known. Therefore we have a set $V = \{(l_1, y_1, n_1), \ldots, (l_r, y_r, n_r)\}$, where a triple (l, y, n) says that y and n are the numbers of the "yes" and "no" votes, respectively, given to l.

An edge connecting a `User` and an `Item` node is a triple (u, i, r_{ui}), where r_{ui} is the original rating that u has given i in a 5-point scale range. The set of all items a user u has already rated, is called the user's *basket*. An edge connecting an `Item` node and a `List` node is a pair (i, l) and it indicates that l contains i. U, I, L along with (u, i, r_{ui}) and (i, l) form our knowledge base $\mathcal{K}$. Finally, we generate the set

$$\mathcal{A}_I = \{(i_a, i_b, w_{ab}) : i_a, i_b \in I \text{ and } w_{ab} \in \mathbb{R}^+\}$$

from item-list edges and list-vote triples. The weight w_{ab} is calculated to reflect the strength of the association between items i_a and i_b. The precise formula for calculating w_{ab} will be given in the next section. Note that set $\mathcal{A}_I$ can also be considered as a weighted item-item graph representation.

3. CIRC

In this section we introduce our recommendation method based on the model we described in the previous section. Initially, we present the weight calculation for the triples in set $\mathcal{A}_I$. The algorithm used to produce recommendations is described in the next subsection.

3.1 Weight Calculation

The frequency of co-occurrence of items is a good indicator of possible associations between items [5], however, it's not just the frequency of co-occurrence that counts, but it's also the quality of co-occurrence. In other words, a pair of items which appears in a very desirable list is as significant as - and sometimes more important than- a pair appearing in many lists. Intuitively, if a user finds a useful and well put together list of items (e.g. a good playlist), he'd rather rate the list than creating a new list containing the same items to express his interest. This way, items that are really related will get more and higher votes and probably they will not appear together in many other lists.

We define Sum of Bayesian Ratings (SBR) as the weight describing the degree of association between two items. SBR takes into consideration not only the frequency of co-occurrence of the two items, but the number of votes that lists have received by anonymous users as well.

Sum of Bayesian Ratings as w_{ab}. Sum of Bayesian ratings of all lists l in L containing both i_a and i_b:

$$w_{ab} = \sum_{(i_a,l)\in\mathcal{A} \text{ and } (i_b,l)\in\mathcal{A}} BR(l)$$

where Bayesian Rating $BR(l)$ for a list l is defined as:

$$BR(l) = \frac{C \times M + R_l \times N_l}{M + N_l}$$

and C, M, R and N are defined in the following table.

Data Type	Number
C	the average rating of all the lists in $\mathcal{K}$
M	the average number of votes given to the lists in $\mathcal{K}$
R_l	the rating of list l
N_l	the total number of votes given to list l

The rating of a list, R_l, is computed as the percentage of yes votes out of total votes given to l. The more desirable the list containing an item pair is, the more confident we are that these two items are related to each other. However, the desirability of lists can not be solely computed based on the percentage of yes votes they receive. If there are only few votes, then these votes should count less than when there are many votes. In other words, the more votes a list has, the higher the weight of these votes.

3.2 Producing Recommendations

Given a target user, we recommend items to users based on the model we described in Section 2. In our algorithm we have a parameter *minW* which denotes the minimum accepted value of w_{ab} for the items i_a and i_b to be considered associated. This threshold can be flexibly configured in CIRC for a specific application. Our algorithm is as follows.

ALGORITHM 1. RECOMMENDITEMS

Input: *A target user u, minW, a knowledge base $\mathcal{K}$, and a set of item associations $\mathcal{A}_I$.*

Output: *A list of (i, p_{ui}) item-prediction pairs for user u.*

Method:

1. *For each item i in the u's basket do:*
   (a) *Retrieve all neighbor items j of i in $\mathcal{A}_I$ such that $w_{ij} \geq minW$.*
   (b) *Let N_i be the set of these neighbors.*
   (c) *For j in N_i do:*
      i. p_{uj} = PREDICTRATING*(u,j)*.
      ii. *Add (j, p_{uj}) to the* RESULT.
2. *return* RESULT.

Function PredictRating generates the predicted rating of item i for user u. This function computes the weighted average of ratings that the user has given to the items that are in association with i according to $\mathcal{A}_I$:

$$\frac{\sum_{(i,j,w_{ij})\in\mathcal{A}_I} r_{uj} \cdot w_{ij}}{\sum_{(i,j,w_{ij})\in\mathcal{A}_I} w_{ij}}.$$

4. EXPERIMENTAL EVALUATIONS

This section presents the experimental results on validating the ability of CIRC to produce high quality recommendations. We first describe the experimental settings and then discuss the results.

4.1 Dataset

To evaluate the performance of CIRC we formed our dataset using data gathered from Amazon.com through Amazon Web Services (AWS). In the following table

we show the type of data gathered and some statistics about this data.

Data Type	Number
Item (Book)	405,238
User	530,160
List	58,618
User-Item-Rating	1,188,435
(l, i)	1,056,932

A pair (l, i), where l is a list and i is an item, indicates that i is in l. Note that Since we focus on book recommendations, we remove all non-book items from the dataset.

4.2 Method and evaluation metrics

We consider five sets (categories) of users based on the number of items they have rated. We randomly selected a 5% of the users in each subset as target users (or sample users). For each target user we use 80% of the items that he/she has rated as *input set* whereas the rest of items as *examination set*. The reason we chose the majority of items in user's basket as input set is the fact that it better corresponds to the deployment of the system in practice where all the ratings are used to produce recommendations and yet it reserves reasonable number of items in examination set to facilitate the evaluation process. Table 1 reports statistics about the sample target users. The final results reported for each category are averages over all users of that category.

Category	# of ratings (R)	Population	Sample size
C1	$R < 5$	497,664	24,884
C2	$5 \leq R < 10$	16,383	820
C3	$10 \leq R < 50$	9,834	492
C4	$50 \leq R < 500$	1,442	73
C5	$R \geq 500$	28	2

Table 1: Averaged Statistics about the sample target users

Since CIRC recommends to the user the items s/he might like, we only consider *positive ratings* in the input set to produce recommendation. We call a rating on item p given by user u to be positive if $r_{u,p} \geq 4$. For users who had at least one positive rating in the input set and one in the examination set, we generated recommendations. A recommended item p to the user u is a successful recommendation if p appears in the examination set for user u and $r_{u,p} \geq 4$. For every target user, we recommend as many items as the number of items in user's examination set. To evaluate the performance of CIRC we use the common metrics used to evaluate recommendation algorithms [2, 4]: *Mean Absolute Error*, *F1* and *Coverage*.

4.3 Results and discussions

This section discusses experimental results on recommendations produced by CIRC. For comparison purposes, we have also included the results of both item-based and user-based collaborative filtering recommendation systems which employ the Pearson nearest neighbor algorithm. For these two collaborative filtering algorithms we use 80% of the data set as training set and 20% as test set. We have also implemented and evaluated CIRC using frequency of co-occurance as weight, but to to space constraints we do not report the result here. The interested reader may refer to the long version of this paper [5].

As it was mentioned before, CIRC has a parameter *minW* which can be flexibly configured. To tune this parameter we conducted an experiment which measures the quality of recommendations produced by CIRC for different values of *minW*. The results of this experiment which is reported in [5] due to space constraints, shows that interesting pairs are preserved by using *SBR* even when the threshold is high. This means that *SBR* is a good indicator of existing associations between items and successfully separates the interesting items from the items the user might not like. We set *minW* equal to 0.5 to get the best possible results CIRC produces in the rest of the experiments that follows.

4.3.1 CIRC vs. item-based and user-based collaborative filtering

We categorize CIRC as a model-based recommendation approach which utilizes the information provided by users in a new way to derive the item-item relationships. We believe that in applications where extra data from user-created lists is available, CIRC can outperform commonly used user-based and item-based approaches. Figures 2 - 4 report the results of comparing the performance of CIRC with that of user-based and item-based recommendation approaches. We report F-measure as an indicator of system accuracy in finding items the user will like, while MAE can be interpreted as the confidence an algorithm has in recommendations it makes and coverage measures the usability of the system to the users by measuring the percentage of the the items in a user's basket for which a prediction was provided. This computation is based on the leave-one-out method.

F-measure. It can be observed from the charts in Figure 2 that *CICR* provides more accurate recommendations according to the reported F-measures. The distinction between our approach and two benchmark CF algorithms that we have implemented becomes even more apparent for categories `C3`, `C4` and `C5` where users have rated more items. We believe this happens partly because as the users rate more and more number of items there is a risk of mixed tastes in their generated neighborhood. Therefore, lots of recommended items are of no or lower interest to the user. CIRC on the other hand, remains on track by just looking at quality item-item associations and is not confused by a misleading neighborhood.

MAE. Figure 3 reports the quality of predictions made

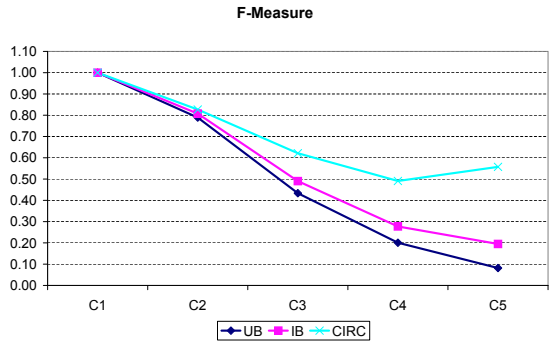

Figure 2: Comparison of classification accuracy of CIRC and user-based and item-based algorithms

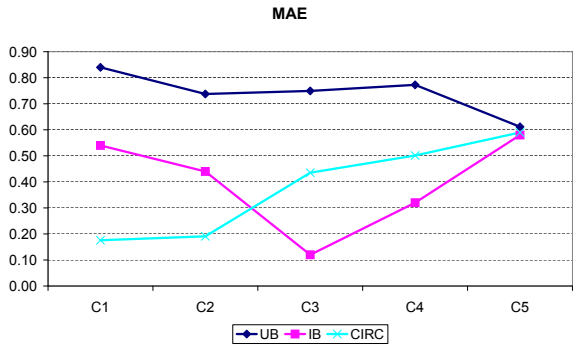

Figure 3: Comparison of prediction accuracy of CIRC and user-based and item-based algorithms

by each of three algorithms under examination. While user-based algorithm always bears the worst prediction quality of all three, item-based approach shows an interesting behavior and even outperforms CIRC predictions in categories `C3` and `C4`. Starting from category `C1` with lowest number of rated items for each user, the quality of item-based predictions increases as the sparsity of data set decreases. However, this increase does not last and it stops at `C3` which we believe is the best case dataset for item-based algorithm.

Coverage. Figure 4 illustrates the usability of these three recommender algorithms in terms of the coverage they provide for different user categories. Without any exception CIRC outperforms item-based and user-based approaches.

4.4 Discussion

From the experimental evaluations of CIRC along with two benchmark algorithms, we observe that CIRC is almost always the winner except for the yielded prediction accuracy of categories `C3` and `C4` in which item-based recommender performs better. We believe in domains where list data is available, CIRC can serve as a complimentary method to other successful CF recommendation algorithms. It is important to note that all measures used to evaluate this work suffer from the underlying biases as is suggested by Herlocker et al [2].However, to alleviate this bias in our evaluations, we only used the portion of the recommendations which also appear in the target user's examination set, otherwise, a recommender with high *true recall* and *true precision* may yield low values on these measures because it recommends lots of *un-rated* relevant items to the user. The comparisons, however, are absolute and show the prominence of our approach over benchmark algorithms.

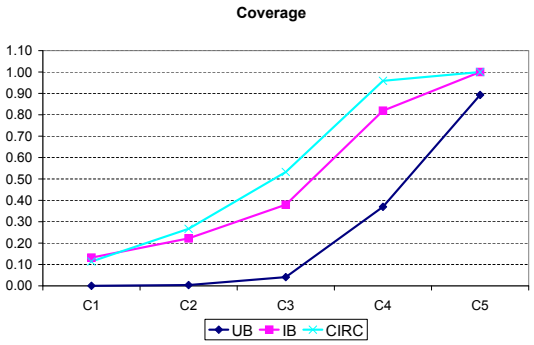

Figure 4: Comparison of coverage of CIRC and user-based and item-based algorithms

5. CONCLUSIONS

In this paper we presented a new model based on valuable information in user-created lists which is usually overlooked. We described and experimentally evaluated CIRC- a new algorithm based on the proposed model- and showed that CIRC outperforms commonly used user-based and items-based CF methods while it is scalable and flexible at the same time. All these results are very promising and suggest that our work has successfully discovered a new potential to extract item-item associations.

6. REFERENCES

[1] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM trans. info. syst.*, 22(1):143, 2004.

[2] J. L. Herlocker, L. G. T. Joseph A. Konstan, and J. T. Ridel. Evaluating collaborative filtering recommender systems. *ACM trans. info. syst.*, 22(1):5, 2004.

[3] T. Hofman. Latent semantic models for collaborative filtering. *ACM trans. info. syst.*, 22(1):89, 2004.

[4] Z. Huang, D. Zeng, and H. Chen. A comparative study of recommendation algorithms in e-commerce applications. *IEEE Intelligent Systems*, 22(5):68–78, 2007.

[5] M. Khezrzadeh, A. Thomo, and B. Wadge. Harnessing the power of favorites lists for recommendation systems. Available at `http://webhome.csc.uvic.ca/ maryamk/Paper2.pdf`.

[6] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herloker, L. R. Gordon, and J. Riedl. Grouplens: applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[7] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001.

[8] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems-a case study. In *ACM WebKDD 2000 Web Mining for E-Commerce Workshop*, 2000.