

# PeerChooser: Visual Interactive Recommendation

John O'Donovan and Barry Smyth  
 School of Computer Science and Informatics  
 University College Dublin,  
 Ireland.  
 {firstname.lastname}@ucd.ie

Brynjar Gretarsson, Svetlin Bostandjiev,  
 Tobias Höllerer  
 Department of Computer Science  
 University of California, Santa Barbara.  
 {brynjar, svetlin, holl}@cs.ucsb.edu

## ABSTRACT

Collaborative filtering (CF) has been successfully deployed over the years to compute predictions on items based on a user's correlation with a set of peers. The black-box nature of most CF applications leave the user wondering how the system arrived at its recommendation. This note introduces *PeerChooser*, a collaborative recommender system with an interactive graphical *explanation interface*. Users are provided with a visual explanation of the CF process and opportunity to manipulate their neighborhood at varying levels of granularity to reflect aspects of their current requirements. In this manner we overcome the problem of redundant profile information in CF systems, in addition to providing an explanation interface. Our layout algorithm produces an exact, noiseless graph representation of the underlying correlations between users. *PeerChooser's* prediction component uses this graph directly to yield the same results as the benchmark. User's then *improve* on these predictions by tweaking the graph to their current requirements. We present a user-survey in which *PeerChooser* compares favorably against a benchmark CF algorithm.

## ACM Classification Keywords

H.5.4 Information Interfaces and Presentation (e.g., HCI): Hypertext/Hypermedia—*architectures*

## Author Keywords

Recommender systems, Visualisation, Interaction

## INTRODUCTION

The idea behind recommender systems is to provide a user with a useful recommendation. Many recommender systems use item or user-based [5] collaborative filtering (CF) techniques to generate their recommendations. CF models the social process of asking a friend for a recommendation, making suggestions for some target user  $u$  from the items that are liked by users similar to  $u$  (user-based) or from items that have received similar ratings to the items that  $u$  likes (item-based). There are problems with the standard CF approach however. The so-called “grey-sheep” problem causes difficulties

when a particular user's ratings history does not help to identify a set of similar recommendation partners. In addition, it is common for the user-item ratings matrix to be sparsely populated, again making it difficult to confidently identify similar users and items, due to a lack of ratings overlap. CF techniques find it difficult to cope well with new or short-lived items since such items cannot be recommended until they have been rated by a critical mass of users.

In this paper we focus on the way CF systems identify user communities or *peer groups* as the basis for recommendation. Very often this is opaque to the users of a system. We suggest that users need more control in order to tailor recommendations to their current moods and influences; for example, in a movie recommendation system a user may want to select Stephen Spielberg's profile as a recommendation partner if they were in the mood for a science fiction movie, however if a user could assess Mr Spielberg's recommendations on a set of items well known to that user, the decision to include him as a peer may change. Accordingly we show how users can influence peer group formation by manipulating a visual representation of their current neighborhood, in order to fine-tune their recommendations. Figure 1 shows the architecture of *PeerChooser* with the feedback mechanism in the shaded area. At each tuning step, users are provided dynamic feedback in the form of recommendations on items well known to them. As such our *PeerChooser* system serves as explanation interface that provides users with a better understanding of the source of their recommendations and that helps to enhance overall user trust in the system; see [2, 4].

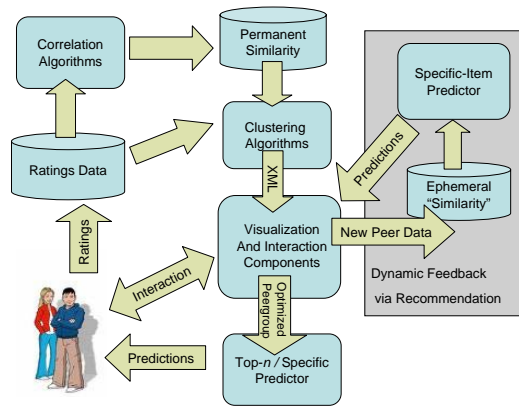
## VISUALIZING COLLABORATIVE FILTERING

Figure 2 shows a visualization of a CF process in *PeerChooser*. The main display includes the peer-graph which is centered on the active user. The active user's neighbors are represented by the connected nodes such that the length of the connection between the active user and a neighbor is based on the similarity of their ratings profiles (using the standard Pearson's Correlation Coefficient). The precise positioning of nodes is governed by using a force-directed graph layout technique, similar to that presented in [1]. This models node connections as ‘springs’ which exert a force on their nodes that is inversely proportional to node similarity and the systems settles to a zero-energy equilibrium state as the graph is visualized; in practice we have found this to provide accurate graph layouts in which

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2008, April 5 - 10, 2008, Florence, Italy.

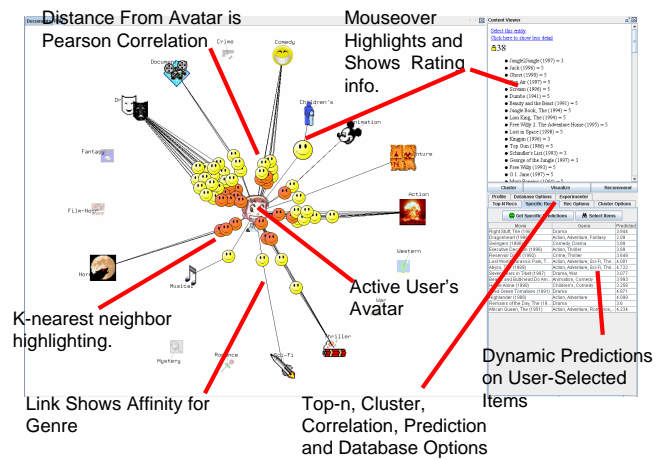
Copyright 2008 ACM 978-1-60558-011-1/08/04...\$5.00.



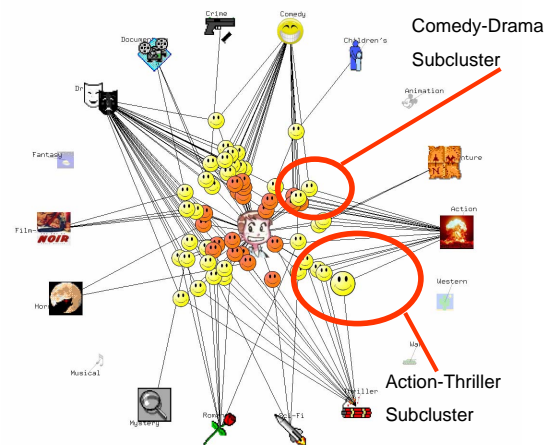
1: Architecture of the PeerChooser Visual Interactive Recommender System.

on-screen distance is closely correlated with node similarity, thus providing the system user with an accurate visual representation of their peer group or neighborhood.

We are interested in allowing a user to manipulate a neighborhood in order to allow then to benefit from improved recommendations that more accurately reflect their current preferences and mood. To achieve this the interface must convey meaningful information to the user about their neighborhood structure and the interests of their neighbors. This is a challenging problem as each neighborhood represents a high-dimensionality preference space which is difficult to render and even more difficult for a user to interpret. To address this, while maintaining ease of use for the end user, we have chosen to map the complex ratings data onto the space of movie *genres*, which are presented as a set of user-connected *genre nodes* alongside their neighborhood representation; see Figures 2 and 3. Very briefly, an edge is created between a genre node  $g$  and a user  $u$  if  $u$  displays an affinity for  $g$  in their ratings; in practice we use a simple thresholding technique to identify affinities between users and genres if more than 10% of the user's positive ratings reflect a given genre. These genre nodes are also displayed using the force-directed graph layout algorithm, which causes a natural clustering of users to genres as shown in Figure 3; in this example we allow each user to be connected to their two most dominant genres but in principle each user can be connected to any number of genres, subject to presentation clarity issues. The genre distribution in the MovieLens data has a massive bias towards three genres: Drama, Action, and Comedy. As a result, our initial clustered graphs tended to only show connections between non-active user nodes and nodes representing these three genres. To counter this problem and provide a more diverse set of neighbours we apply a scaling function to our edge drawing threshold based on number of occurrences of a genre in the database. In addition to manipulating individual neighbour nodes on the graph, users can also click and drag the genre nodes. These actions are translated into neighbor-node movements as follows: When a genre node is moved toward or away from the active user's avatar in the centre of the graph, each connected person-node is moved by a relative distance in the same direction. This allows users to make



2: Annotated Screenshot of PeerChooser's Interactive Interface.



3: MovieLens data visualized using *PeerChooser* with multiple genre associations.

more coarse-grained statements about current requirements.

*PeerChooser* uses OpenGL technology on a Java platform, making it visually appealing to the end user. An active user can interact with the graph by moving or deleting icons. Genre nodes are positioned around the outside of the graph and can be clicked on and moved freely. When this occurs, each connected neighbor node is moved by a relative distance in the same direction. This allows the user to make a bold statement about her current opinion on a particular genre. For example, if the comedy icon is moved towards the active user's avatar then all users who like comedy will be drawn closer and become "more similar" to the active user for this session. We term this new value *ephemeral similarity*. Furthermore, users can make fine-grained statements by moving neighbor icons individually. On mouseover, nodes are highlighted and associated ratings are displayed in the right panel, shown in Figure 2.

### GRAPH-BASED PREDICTION

We now explain how "hints" provided by the user through the interface translate in to actual recommendation influences. In *PeerChooser* the  $k$  nearest user nodes are

selected as the neighborhood for the purpose of recommendation; these are the more shaded nodes in Figures 2 and 3. Our benchmark CF prediction algorithm uses Resnick’s formula which is reproduced below as Equation 1; see also [5]. In this formula  $c(i)$  is the rating to be predicted for item  $i$  for an active profile  $c$  (the user receiving the recommendation), and  $p(i)$  is the rating for item  $i$  by a peer profile  $p$  who has rated  $i$ .  $\bar{c}$  and  $\bar{p}$  refer to the mean ratings for  $c$  and  $p$  respectively. The weighting factor  $sim(c, p)$  is a measure of the *similarity* between profiles  $c$  and  $p$ , which is traditionally calculated as Pearson’s correlation coefficient and has a range of -1 to +1. In our evaluation section we test the performance of this standard benchmark algorithm against our visualization-based approaches, which allow user hints to influence the similarity values used during recommendations as we shall now discuss.

$$c(i) = \bar{c} + \frac{\sum_{p \in P(i)} (p(i) - \bar{p}) sim(c, p)}{\sum_{p \in P_i} |sim(c, p)|} \quad (1)$$

### Distance Weighting

To incorporate user hints into the recommendation process we simply replace the standard similarity values (based on user-user ratings correlations) with a new similarity value that is based on the inverse *Euclidean distance* between the active user node and each of the  $k$  peer nodes that have been manipulated by the user. This is our *ephemeral similarity* value and is given by Equation 2. Here, Euclidean distance between pixels on the graph is normalized to the Pearson’s correlation range of (-1, +1), *max\_dist* is the maximum possible distance between the active user node and a peer node, while *node\_dist* is the distance between the active node (ie: the center of the graph) and each peer node. Equation 3 shows the original Resnick prediction formula using ephemeral similarity in place of the standard Pearson correlation. The nomenclature is similar to that in Equation 1 with  $c(i)$  being the predicted rating for an active user  $c$  on item  $i$ .

$$eph\_sim(c, p) = 1 - \frac{2(node\_dist)}{max\_dist} \quad (2)$$

$$c(i) = \bar{c} + \frac{\sum_{p \in P(i)} (p(i) - \bar{p}) eph\_sim(c, p)}{\sum_{p \in P_i} eph\_sim(c, p)} \quad (3)$$

### EVALUATION

The majority of experiments involving recommender system algorithms are based on some form of automated testing, for example, predicting ratings for some “hidden” subset of the rated data. This is only possible in absence of interactive components such as the ones of our *PeerChooser* system. Visualization and interaction are additional “tiers” to the process of collaborative filtering which prohibit the standard automated testing procedure since real people must explore the graph and interact with it to attain results.

### Procedure

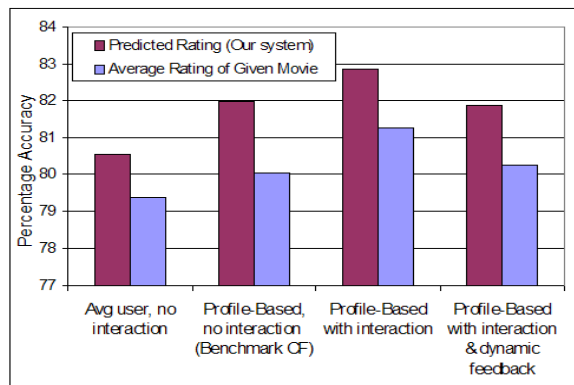
To test the performance of our system we conducted twenty five user trials which examined four techniques for generating recommendations with *PeerChooser*, two

with interaction (hints) and two without. The source data for this survey was the smaller MovieLens[3] dataset, which contains 943 users ratings on 1682 items. Participants were asked to generate recommendations using the techniques listed below. Three values per recommendation were recorded: the predicted rating, the actual rating, and the average rating for that item across the entire database.

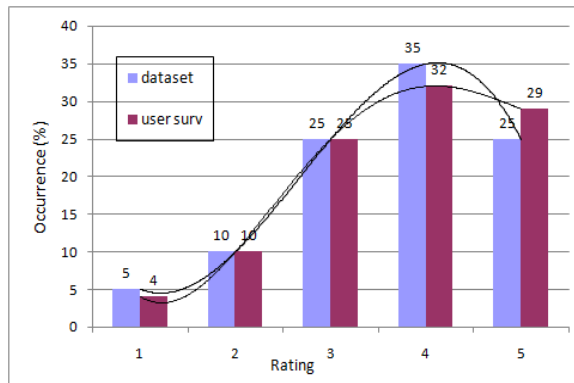
1. *Average Layout* - (non-interactive) The graph was laid out based on an “average user”. This profile was created by taking the average rating for the 50 most rated items. Participants were then asked to rate recommendations generated for this user. This technique was expected to yield the worst results as it contained no personalized information.
2. *Profile-Based Layout* - (non-interactive) This is our benchmark CF algorithm. Users rated 30 items in the right hand panel. Correlations computed from these were used to generate predictions.
3. *Profile-Based Layout with Manipulation* - (interactive) Same as above but the user can manipulate the graph to provide information on current requirements.
4. *Profile-Based Layout with Manipulation and Feedback* - (interactive) Same as above except the user receives dynamic recommendations on their salient items with each graph movement. We expected this to exhibit the best performance.

In all cases, the system’s predicted rating was not shown to participants until after they had provided their rating for each predicted item. This follows from work by Swearingen et al. in [6] which suggests that users tend to rate towards the machine-provided ratings. In all cases the number of neighbors was set to  $k=30$ , an optimal value for CF on our dataset reported in [4] For associating or *clustering* non-active user nodes to genre nodes we used a liked-item threshold of 3 (ie: liked-items had a rating of 4 or 5). For each test, the 400 most similar users based on Pearson’s correlation were displayed on the graph. In all tasks where the user interacted with the graph, predictions were made using our distance-based version of Resnick’s prediction formula, given by Equation 3, in all others the standard Resnick prediction formula, Equation 1 was used.

In the final task, *Profile-Based Layout with Manipulation and Feedback* the graph layout was initially built from correlation over the profile data. Users were asked to select checkboxes next to the 5 movies they really liked and the 5 that they disliked. Users selected a button marked “specific predictions” to see the benchmark CF predictions on those items. Users were then told to take some time to manipulate the graph of connected peers to try and tweak the recommendations for their chosen items to a value that most suited their needs and preferences. With no time constraints imposed, all users in the survey reported that they had arrived at a satisfactory position within 2 mins. With each interaction users were provided with dynamic recommendations based on their “salient” item-set and according to the current neighborhood configuration. This allowed the users to dynamically assess the goodness of the evolving neighborhood space as each interaction was performed. Once satisfied with predictions that the “tweaked” graph generated on the salient item sets, a list of top-n recommendations was presented and the user was asked to rate them individually.



4: Error results for each technique in the user trials.



5: Comparison of ratings distributions between MovieLens and user trial data.

### Recommendation Accuracy

To evaluate the accuracy of the techniques, mean absolute error was computed between the predicted rating and the users actual rating for each of the methods. Results of our analysis are presented in Figure 4 for four techniques. It must be noted that the y-axis on this graph is non-standard- the graph is intended to highlight relative differences between each technique. As expected, the average predictions- that is, predictions based on the average user described earlier, exhibited the worst performance, producing an accuracy of 80.5%. It is important to distinguish between this “average user” technique, and the “average ratings” presented in Figure 4. Predictions based on an average user (column 1 in Figure 4) have high accuracy, this is not surprising if we take a look at the rating distribution graph in Figure 5. Users tended to rate only movies they liked, and the average user was constructed from a list of the most commonly rated movies, which as it turns out were generally the most highly rated movies. This may be attributed to the fact that the movies were not new and users tended to remember them in a good light. Our profile-based technique (column 3) with manipulation beats the benchmark (column 2) achieving a small relative increase of 1.05%. A single factor between groups ANOVA shows that these differences are significant in each case with  $p = 0.006$ ,  $F = 3.87$ . This small increase is an important result because it indicates that manipulation does increase recommendation accuracy.

The most surprising result was that the dynamic feed-

back technique performed worse than the other profile-based techniques. After much analysis of the graphs it was determined that users tended to *over-tweak* the system to achieve desired results for their salient item sets. In doing this, many of the profile-based correlations were overwritten and the resulting layout was overfitted to the specific item sets. When this occurs, the system loses the inherent benefits of collaborative filtering, causing accuracy to drop. A solution to this may be to ensure diversity within the salient item sets. To assess the effects of users interaction with the system a pre and post study questionnaire was answered by each participant. Detailed discussion of this survey is not possible due to space restrictions. Briefly, participants were asked a range of questions to assess their experience with graphical interfaces, recommenders and visualizations, they were then asked to rate 10 statements about the *PeerChooser* interface. The salient findings from this survey were that 78% of users felt that the system provided a good explanation of collaborative filtering. and that more than 80% of participants felt they benefitted overall from interacting with the system.

### CONCLUSIONS

This note has introduced *PeerChooser*, an interactive visualization system for collaborative filtering. A precise representation of the CF process is presented to the user allowing openness and explanation of the CF process. Feedback is enabled through manipulation of the visualized data to facilitate expression of current mood and requirements. Results of our preliminary tests with real-world users were presented and discussed. Our results indicate that the visual-interactive approach can help produce better accuracy, more information, and an enhanced user experience with the recommender system.

### ACKNOWLEDGMENTS

This research was supported in part by Science Foundation Ireland under Grant No. 03/IN.3/I361, and the Intelligence Technology Innovation Centre (ITIC) through its Knowledge Discovery and Dissemination Program, as NSF grant #IIS-0635492.

### REFERENCES

1. R. A. Becker, S. G. Eick, and A. R. Wilks. Visualizing network data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):16–28, 1995.
2. J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Computer Supported Cooperative Work*, pages 241–250, 2000.
3. B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, and J. Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 263–266, New York, NY, USA, 2003. ACM Press.
4. J. O'Donovan and B. Smyth. Trust in recommender systems. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174. ACM Press, 2005.
5. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, Sharing Information and Creating Meaning, pages 175–186, 1994.
6. R. Sinha and K. Swearingen. The role of transparency in recommender systems. In *CHI '02 extended abstracts on Human factors in computing systems*, pages 830–831. ACM Press, 2002.