DIVA: Applying Decision Theory to Collaborative Filtering

Hien Nguyen*, Peter Haddawy*†

Decision Systems and Artificial Intelligence Lab*
Dept. of EE &CS
University of Wisconsin-Milwaukee
Milwaukee, WI 53201
{nguyen,haddawy}@cs.uwm.edu

Intelligent Systems Lab†
Faculty of Science & Technology
Assumption University
Bangkok 10240, Thailand
haddawy@isl.s-t.au.ac.th

Abstract

This paper describes DIVA, a decision-theoretic agent for recommending movies that contains a number of novel features. DIVA represents user preferences using pairwise comparisons among items, rather than numeric ratings. It uses a novel similarity measure based on the concept of the probability of conflict between two orderings of items. The system has a rich representation of preference, distinguishing between a user's general taste in movies and his immediate interests. It takes an incremental approach to preference elicitation in which the user can provide feedback if not satisfied with the recommendation list. We empirically evaluate the performance of the system using the EachMovie collaborative filtering database.

Introduction

Collaborative filtering has become a popular approach for eliciting user preferences in order to recommend items of interest[1]. Representation and elicitation of preferences have long been studied in Decision Theory (Keeyney & Raiffa 1976), but surprisingly no work in collaborative filtering has made use of the wealth of techniques and formalism available. This paper describes the Decision-Theoretic Interactive Video Advisor (DIVA), a collaborative filtering system that provides movie recommendations. The guiding principle behind the system design is the accurate representation and efficient elicitation of user preferences, following decision-theoretic principles wherever they apply. Following this design methodology has lead to a number of novel features that provide DIVA with distinct advantages over other collaborative filtering systems.

All other collaborative filtering systems to date represent user preferences with numerical ratings. In contrast, DIVA uses the standard decision-theoretic notion of a preference order, i.e., pairwise comparisons among movies. This provides a fine-grained representation of preference without forcing the user to think about a large number of rating categories.

Most collaborative filtering systems, e.g. (Hill et al. 1995, Shardanand & Maes 1995), determine similarity between preferences of different users following the technique used in GroupLens (Resnick 1994), which is based on the Pearson correlation coefficient. In contrast, DIVA uses a measure based on the concept of the probability of conflict between pairwise rankings. We show that our measure has several practical advantages over the GroupLens measure. We empirically demonstrate that use of our similarity measure results in more accurate recommendations than use of the GroupLens measure.

While people can be characterized as having a general taste in movies, what they are interested in seeing on any one occasion may deviate from this. DIVA supports this dynamic notion of preference by distinguishing between long- and short-term preferences.

In order to minimize the amount of time the user needs to spend providing preference information to the system, DIVA supports the notion of incremental elicitation of preferences by permitting the user to provide feedback if he is not satisfied with the list movies DIVA recommends. This feedback takes into account the distinction between long- and short-term preferences.

The rest of this paper is organized as follows. Section 1 provides an overview of DIVA's functionality. Section 2 describes the overall architecture and key algorithms. Section 3 presents the results from our empirical evaluation of the system. Section 4 discusses related work in collaborative filtering and incremental preference elicitation, and section 5 presents conclusions and directions for future research.

[1] AAAI Workshop on Recommender Systems, Madison, July 1998. AAAI Technical Report WS-98-08.

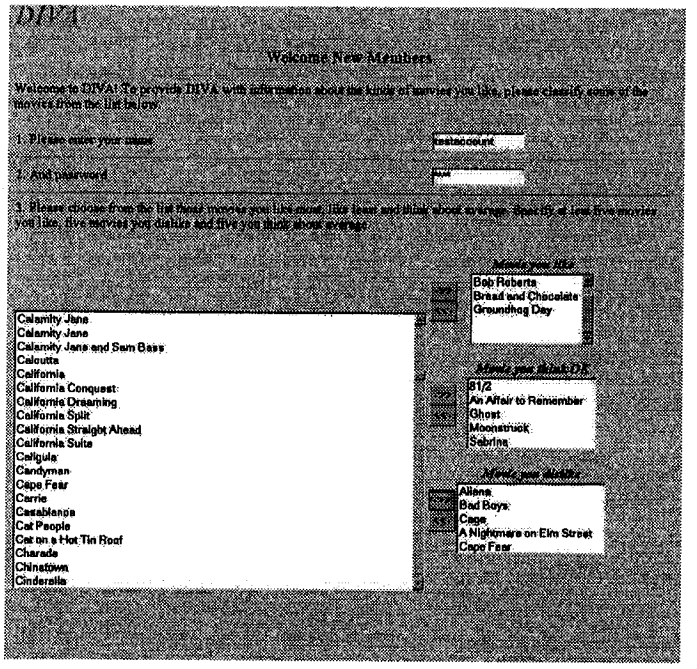

Figure 1: A screenshot of the registration window.

Overview of DIVA

DIVA's design emphasizes ease and accuracy of preference elicitation. In order not to overburden the user with unnecessary questions, DIVA takes an incremental approach to preference elicitation. It starts by eliciting some preference information and then quickly provides a list of recommendations. If the user is not satisfied with the recommendations, the user can critique them in order to provide additional preference information.

To accurately represent user preferences, we attempt to account for the dynamic nature of preferences. While people have general taste in movies, the type of movie they would like to see will typically vary from one occasion to another, based on many environmental factors, such as what movies they recently saw. Thus we separate the elicitation of long- and short-term preferences and we combine the two types of preference in a natural and intuitive fashion.

A new user to DIVA is asked to provide a user login name and a password, which are used to index his preference information whenever he accesses the system. The user is then shown an alphabetical list of all movies in the system and asked to indicate some he particularly liked, some he particularly disliked, and some he thought were about average (Figure 1). Experience has shown that a user must classify a minimum of about 5 movies in each category to get reasonable recommendations from the system. This information is used to build an initial preference structure for the user.

The user may add or change some preferences at any time by clicking the "View/Update Your Profile" button at the top-right corner of Figure 2. After submitting his preferences, the user may request recommendations. This is done via a page that permits the user to constrain the search, as shown in Figure 2. The user may specify actors and actresses, directors, genres, professional star ratings, countries of production, release years, MPAA ratings, and the running time that he is particularly interested in. These constraints give the user a way of indicating what kind of movie he is particularly interested in watching at the current time, i.e. his short-term preferences. They act as an initial filter on the entries in the movie database. The movies that satisfy the constraints are then ranked according to the user's preferences. The behavior we have endeavored to embody in DIVA is that of someone who knows you well acting on your behalf with some instruction from you. For example, suppose you ask a close friend to go to the video store and rent an adventure film that takes place in Africa. Within the context of that constraint, she would use her knowledge of your taste in movies, e.g. a preference for films with excellent

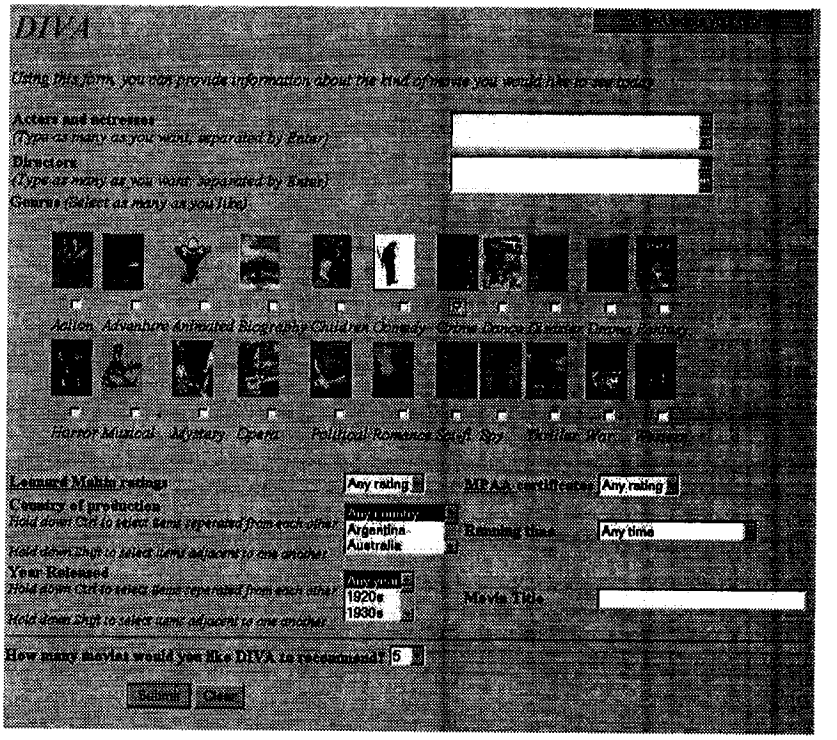

Figure 2: A screenshot of the search window

cinematography, to select the film she thinks you would be most likely to enjoy.

The recommendation list is displayed on a separate page, as shown in Figure 3. By clicking any of the movie titles in the recommendation list, the user can see all the attribute information about that film. If the user is not interested in any of the films in the recommendation list, he can provide feedback to the system and then request another search. We distinguish between feedback concerning long-term preferences (second feedback column) and feedback concerning short-term preferences (first feedback column). If the user has seen any movie in the list, he can indicate whether he particularly liked or disliked it. Even though he may not have seen a film, the user may know enough about it to be quite confident that he would not enjoy seeing it. This is all feedback concerning long-term preferences and is added to the stored user preference model. For movies the user has not seen, DIVA asks that he indicate if they are close to what he is interested in watching currently or if they are far from that. This is feedback concerning short-term preferences and is used only in the current search session. After providing feedback, the user clicks on the "Continue Search" button (Figure 3) to obtain a new list of recommendations.

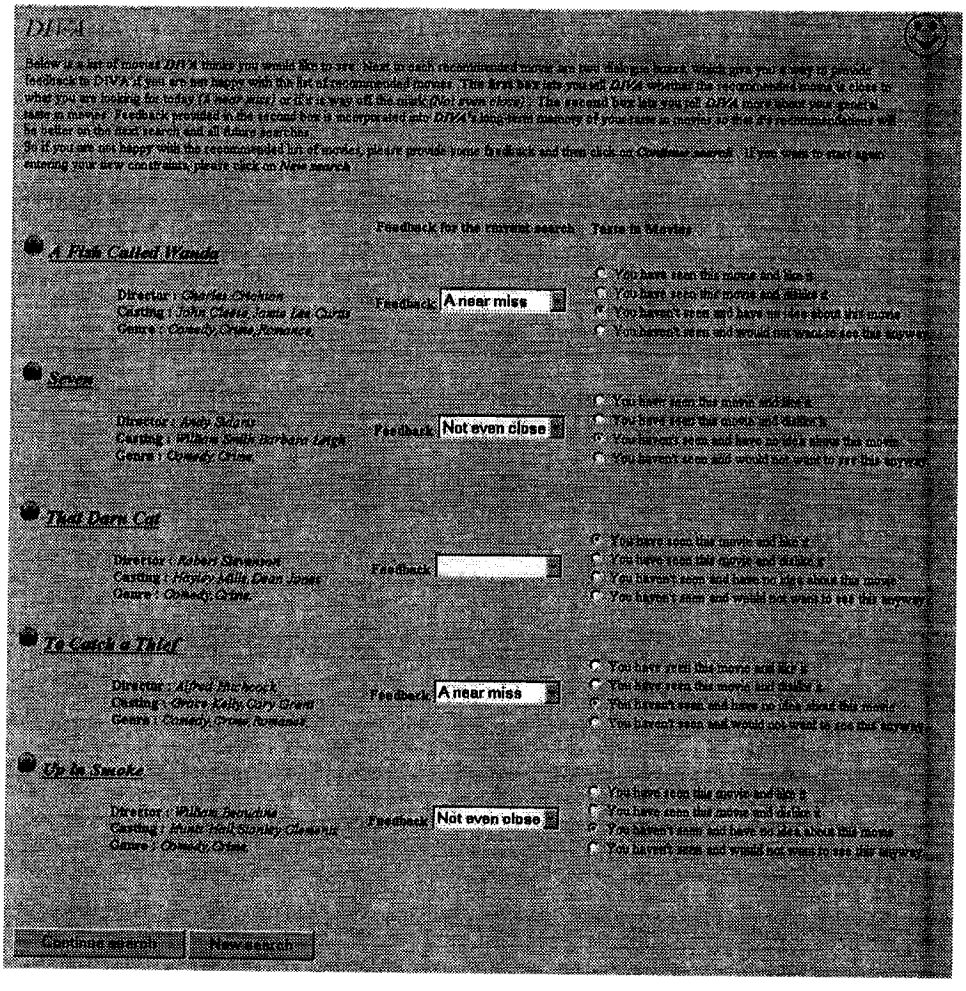

Figure 3: The screenshot of the search result.

Architecture and Algorithms

We retrieve movies by first computing the preference ranking over all movies in our movie database. Then we remove any movies that do not satisfy the user's short-term constraints and display the top n movies. If there are not many movies that satisfy all the constraints, we relax the constraints by disjoining them.

Initial preference elicitation

A user's complete set of preferences in a domain with no uncertainty (such as the movie domain) can be represented as a total ordering over the items in the domain. A subset of the user's preferences then corresponds to a partial order over the items. An initial set of pairwise preferences among movies is obtained from the user's Like, OK, and Dislike lists (Figure 1). This gives us a partial order over movies: every movie in the Like list is considered preferred to every movie in the OK list, which is in turn preferred to every movie in the Dislike list. Figure 4(b) (without the dashed links and the filled node) shows the preferences obtained from the Like, OK, and Dislike lists of the user in Figure 1.

Case retrieval

The directly elicited user preferences are augmented with preference information from a case base of user preference information, using the technique described in (Ha & Haddawy 1998). The case base contains partial preference structures for all users of the system. We compute the similarity between the active user's initial partially elicited preference structure and the preference structures in the case base. We then use the preferences of the most similar stored structure to supplement the directly elicited preferences.

In order to find the closest matching preference structure, we need a method of computing the similarity between two preference structures. We define the *dis-*

similarity between two complete preference structures as the probability that two randomly chosen elements are ranked differently by the two preference structures. This similarity measure satisfies all the properties of a distance metric and has range [0,1].

But the preference information in our case base and the preference information for the active user are only partial. So we need a similarity measure over partially specified preference structures. A partial preference structure can be thought of as the set of linear extensions consistent with the partial order. We define the *dis*-similarity between two partial preference structures simply as the average dis-similarity between all pairs of linear extensions in the two sets. This measure also has range [0 1] and satisfies the triangel inequality, but it is not a metric because the distance between two identical partial orders that are not complete orders is always positive. But this is desirable if the two orders represent the preferences of two different users, since the complete preference orders of the two may actually differ.

Computing this distance measure is closely related to the problem of computing the number of linear extensions of a finite partial order. That problem is known to be #P – complete (Brightwell & Winkler 1991). We use the Markov chain based approximation algorithm of (Bubley & Dyer 1998), which almost uniformly samples the space of linear extensions and runs in polynomial time. To generate each sample linear extension, the algorithm involves running a Monte Carlo simulation of a Markov chain for a fixed number of iterations. The complexity of the distance measure algorithm used in DIVA is $O(nm^2)$ in which n is the number of users in the case base and m is the number of movies in the movie database.

After computing the distance between the active user and each preference structure in the database, we choose the most similar preference structure and then choose the sampled linear extension for the active user that is most similar to that preference structure. In this way we retain all the directly elicited user preferences and obtain a complete preference structure for the active user, guided by the preferences in the case base. In effect, the preference structures in the case base are used as attractors, indicating in which direction to complete the partial preference structure of the active user. In order to save computation time, we compute the similarity only over the top 100 ranked movies in each pair of partial preference structures.

Incorporating user feedback

DIVA permits the user to provide feedback concerning long-term and short-term preferences. The long-term preference feedback is added to the stored partial preference structure obtained from the original Like/OK/Dislike list. If the user saw a movie and liked it, the movie is added to the Like list, as shown in Figure 4(b). If the user saw a movie and disliked it or if he didn't see a movie but is confident he wouldn't like it, the movie is added to the Dislike list. The short-term preference feedback is stored separately from the long-term preferences and is forgotten after the current search is ended. Movies the user tags as being a *near miss* are considered preferred to those tagged as being *not even close*. A preference link is created from every *near miss* movie to every *not even close* movie, as shown in Figure 4(a).

The new preference structure representing the user is now the union of the updated long-term preference structure and the structure resulting from the short-term feedback. The process of generating a recommendation list is now rerun, using the new updated preference structure as the input to the case retrieval algorithm.

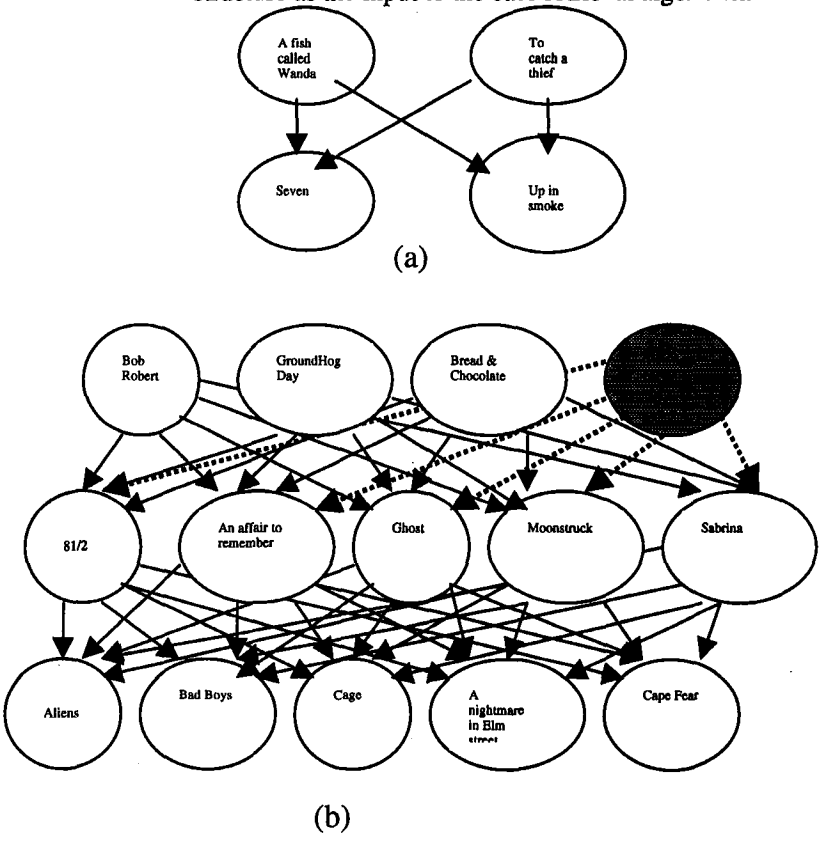

Figure 4: Preference structure of the user feedback as seen in Figure 3. (a) Temporary feedback information (b) Permanent feedback information is incorporated into the user's initial preferences.

Databases

DIVA contains both a movie database and a user preference case base. We built the initial case base using the EachMovie collaborative filtering database provided by Digital Equipment Corporation. The Digital systems research center assembled the database by running a collaborative filtering system for 18 months. The EachMovie database contains 72916 users, 2811983 numeric ratings and 1628 movies. To represent user preferences, EachMovie use numeric rating scale ranging from 0 (awful) to 1 (excellent), in increments of 0.2. To populate our case base, we chose 500 users out of the 72916 users such that each user rated at least 20 movies. The average number of ratings per user over all 500 users is around 70 movies.

Since the EachMovie database did not contain sufficient attribute information for our purposes, we obtained movie attributes from the Microsoft Cinemania 94 CDROM. Our movie database contains 2000 movies with 9 attributes including actors/actresses, director, genre, professional film critic star rating, MPAA rating, country of production, running time, year released and title. Of the 1628 movies in EachMovie, only 500 appear in the Cinemania database. So we removed from the EachMovie rankings for all those movies that are not present in Cinemania. We then converted the preference structures represented as numeric ratings in Eachmovie to partial order preference structures in the obvious way: For each user, a movie s_1 is preferable to movie s_2 if the user's rating for movie s_1 is greater than that for movie s_2.

Empirical Analysis

We evaluated the quality of DIVA's recommendations and those generated using the GroupLens (Resnick 1994) collaborative filtering technique over our database of 500 users. We used the precision and recall metrics commonly used in information retrieval (Salton 1983) as our evaluation criteria. In the present context, precision indicates how many movies in a recommended list were actually liked by the user. Recall indicates how many movies out of all movies liked by the user were predicted correctly. We formed our test set by randomly taking out 10 users from the case base. For each user in the test set, we divided the user's partial preference structure into two sets that are observed set O_1 and unobserved set O_2. Since the partial value function used by DIVA adds information not in the preference structure stored in the case base, we disabled this portion of the system. Hence, the recommendations were generated using only the case-based elicitation portion of the system.

We ran the evaluation of DIVA by using the preference structure of the observed set to simulate elicitation from a new user. Based on the closest matching preference structure, we predicted unobserved items in O_2, i.e. the bulk of the Like list. We generated a recommendation list of length 1/6 of all movies that the user rated, which for all users in the experimental set was a subset of the number of items in O_2. We chose the value 1/6 in order to maximize precision at the expense of recall because from a user standpoint, the desirability of the items in the recommendation list is more important than finding all items of interest in the data base.

We ran 100 experiments with each of 10 users. We set the number of iterations in our sampling algorithm to 50, 100, and 150 and the number of linear set to 10, 30, and 50. For each user, we randomly kept 3 movies from his "Like" list and tried to predict the remaining movies in his original "Like" list. The results are shown in Table 1, averaged over all 10 users.

We observed that if the number of linear extensions is greater than 30 and the number of iterations is greater than 100, the precision and recall do not change much.

We ran the same set of experiments using the GroupLens collaborative filtering technique. GroupLens (Resnick et al 1994, Konstan et al 1998) is a collaborative filtering system that helps News readers to find articles of interest. Each News reader rates a number of articles on a 5 point numeric scale. The system uses these ratings to determine which users are most similar to each other and then predicts how much the user will like new articles based on ratings from similar users.

Number of linear extensions	10	30	50
Number of iterations	50,100,150	50,100,150	50,100,150
Precision	80%,83%,83%	81%, 86%,86%	84%,85%,86%
Recall	38%,40%,40%	38%,40%,40%	40%,40%,40%

Table 1: Average precision and recall for DIVA's case-based elicitation algorithm.

GroupLens uses a similarity metric based on the Pearson correlation coefficient (Pindyck & Rubinfeld 1991). The measure assumes that preferences are represented with numeric ratings. Thus, for this portion of the experiment we worked with the original numeric rating representation of the preferences in our case base of 500 users. The correlation between the user K and any user L in the case base defined as follows:

$$r_{KL} = \frac{\sum_i (Ki - Kavg)(Li - Lavg)}{\sqrt{\sum_i (Ki - Kavg)^2} \sqrt{\sum_i (Li - Lavg)^2}}$$

In which K_{avg}, L_{avg} are the average ratings of user *K* and *L*, respectively. K_i, L_i are the rating of the users *K* and *L* to the article *i*. The summations over *i* are over the items for which both users *K* and *L* have rated. The correlation coefficient weights between -1 (*disagree*) and 1 (*completely agree*). It indicates how much the user *K* tends to agree with the user *L* on those articles that they both rated. Predictions for movies that one user did not rate are then produced by using the correlation to produce a weighted sum over all users in the case base. The precision and recall using GroupLens technique were 65% and 35% respectively. The best precision and recall for DIVA were 86% and 40%, respectively, so the case-based elicitation technique in DIVA outperformed the GroupLens technique along both dimensions.

This result is not surprising given the properties of the similarity measures. The GroupLens measure has an advantage that it is fast and easy to implement. However, it has a disadvantage that it is insensitive to the size of the set of movies that both users have rated. In an extreme case, the preferences of two users can be maximally similar if they have only one movie in common that they rated and they agreed on that rating. In contrast, the distance measure used in DIVA considers all movies that the two users have rated. Experiments we conducted to examine the set of users ranked similar to a given user showed that the DIVA measure does not rate two users as being similar unless their preferences agree over a set of reasonable size. As a consequence of this property, the GroupLens measure does not satisfy the triangle inequality. In addition to being intuitively desirable, the triangle inequality can be exploited to reduce computational effort by computing bounds on similarity.

Related Work

The Automated Travel Assistant (ATA) (Linden 1997) is a system for recommending airline flights. It uses decision-theoretic techniques to determine user preferences and takes an incremental approach to elicitation. The ATA system elicits some user preferences concerning airline, price, and non-stop vs indirect, and combines this with default preferences (prefer cheaper flights, prefer fewer stops) to obtain a complete preference structure. It then presents the top ranked flights to the user, as well as some extreme solutions: cheapest flight, most direct flight. If the user is not satisfied with the recommendations, he can modify the model of his preferences by manipulating a graphical representation of the function used to represent the preferences. ATA and DIVA differ in their approach to incremental elicitation. While DIVA incrementally obtains increasing amounts of preference information from the user, ATA allows the user to incrementally modify a always complete preference structure. The user provides feedback to ATA by directly manipulating the system's internal representation of preferences. In contrast, the feedback mechanism in DIVA is intended to allow the user to communicate about the proposed solutions in a way that is natural for the application domain. Finally, whereas ATA applies a single set of defaults to every user, the defaults used in DIVA are determined by searching the case base, using the initially elicited preferences.

Basu et al (Basu 1998) describe an approach to generating recommendations that combines content information and collaborative information. The content information includes value for 26 features in the movie selection domain such as genre, actors/actresses, and director. The collaborative information is a triple (user, movie, rating). The hybrid feature is created based on the observation that genre is the content feature that users most often think of while choosing a movie. After creating the hybrid features, they then use *Ripper* (Cohen 1995) to learn a user's preferences. The case base contains 260 users and 45,000 movie ratings on scale of 1 – 10. The average precision and the recall of the system are 83% and 34%, respectively.

Conclusions and Future Research

This paper has described the first attempt to use decision-theoretic techniques in design of a collaborative filtering system. DIVA addresses several difficult problems in building recommender systems, including the distinction between a user's general tastes and his immediate interests, and provision for the user to provide feedback if unsatisfied with the recommendations. Several difficult technical problems remain to be solved. Our representation of short-term preferences is rather crude. We would like to be able to represent short-term preferences using as rich a representation as that for long-term preferences and to be able to merge the two even when conflicts exist. In addition to distinguishing between long and short-term preferences, a recommender system should be able to account for the fact that people's preferences evolve over time. This would require a system to keep track of the time that each piece of preference information was obtained and to notice when newly expressed preferences conflict with older preferences.

A problem for DIVA and other collaborative filtering systems is the computational cost of computing similarity when the case base becomes very large. We are examining the user of hierarchical clustering of the

case base in order to reduce the computational cost of case retrieval. An interesting question is how much accuracy we lose in the recommendations produced *vs* how much computation time we save.

To save computation time, we currently use the rough heuristic of computing similarity over only the top 100 ranked movies. We would like to experiment with other heuristics that involve sampling the preference structure in other ways.

The elicitation of pariwise preferences using the Like, OK, and Dislike lists results in a ranking of movies into only three categories. But the internal representation supports an arbitrary number of categories. We are currently working on augmenting the interface so that the user can graphically specify a richer set of preferences by directly positioning movies within each list on a vertical axis. We expect this will improve the accuracy of the recommendations.

Acknowledgements

This work was partially supported by NSF grant IRI-9509165. The EachMovie database was generously provided by Digital Equipment Corporation. Thanks to Vu Ha, Preerapol Meomeeg and Dr. Hanh Pham for help with the implementation and for useful discussions.

References

Basu, C.; Hirsh, H.; and Cohen, W. 1998. Recommendations as classifications: Using social and content-based information in recommendation. In *Proceedings of the fifteenth National Conference on Artificial Intelligence,* 714-720.

Brightwell, G., and Winkler, P. 1991. Counting linear extensions is #P-complete. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing,* 175-181.

Bubley, R., and Dyer, M. 1998. Faster random generation of linear extensions. In *Proceedings of the Ninth Annual ACM-SIAM Symposium of Discrete Algorithm,*175-186.

Ha, V., and Haddawy, P. 1998. Toward case-based preference elicitation: Similarity measures on preference structure. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence,* 193-201.

Keeyney, R. L., and Raiffa, H. 1976. *Decision with Multiple Objectives: Preferences and Value Tradeoffs.* John Wiley and Sons.

Konstan, J. A.; Rield, J.; Borchers, A.; Herlocker, J. L. 1998. Recommender systems: A GroupLens perspective. In *Working notes of the 1998 AAAI workshop on Recommender Systems,* 60-64.

Hill, W; Stead, L, Rosenstein, M; Furnas, G. 1995. Recommending and Evaluating Choices in a Virtual Community of Use. In *Proceedings of the CHI-95 Conference.* Denver, Colorado.

Linden, G; Hanks, S.; Lesh, N. 1997. Interactive assessment of user preference models: the automated travel assistant. In *User Modeling,* June 1997.

Resnick, P.; Iacovou, N.; Sushak, M.; Bergstrom, P.; and Rield, J. 1994. GroupLens: An open architechture for collaborative filtering of netnews. In *Proceedings of the 1994 Computer Supported Cooperative Work Conference,*175-186.

Salton, G., and McGill, M. J. 1983. *Introduction to Modern Information Retrieval.* McGraw-Hill.

Shardanand, U. and Maes, P. 1995. Social Information Filtering: Algorithms for Automating "Word of Mouth". In *Proceedings of the CHI-95 Conference.* Denver, Colorado.