On User Recommendations Based on Multiple Cues

G. Dudek and M. Garden
Centre for Intelligent Machines, McGill University
3480 University St, Montréal, Québec, Canada H3A 2A7
{dudek,mgarden}@cim.mcgill.ca

Abstract*— *In this paper we present an overview of a recommender system that attempts to predict user preferences based on several sources including prior choices and selected user-defined features. By using a combination of collaborative filtering and semantic features, we hope to provide performance superior to either alone. Further, our set of semantic features is acquired and updated using a learning-based procedure that avoids the need for manual knowledge-engineering. Our system is implemented in a web-based application server environment and can be used with arbitrary domains, although the test data reported here is restricted to recommendations of movies.

I. Introduction

This paper outlines a recommendation system [1] that combines aspects of user-based and item-based methodologies to attempt to avoid some of the difficulties encountered with either approach in isolation. Web-based systems that attempt to recommend products to users based on knowledge of their personal preferences have become relatively commonplace. The preferences in question may simply be items in a current web-based shopping cart, but more typically they are based on historical shopping data or explicit responses to questions. As such, recommendation systems represent one of the most successful and unique application spaces of the world wide web; prior to the web, such recommendation systems were unheard of in this form and essentially infeasible.

Recommendation systems are typically based on one of two key paradigms: collaborative filtering, which makes recommendations to a user based on what similar users have done, or item-based filtering, which makes recommendation to a user based on inferred connections between the objects in the domain of interest without explicitly modeling other users (which are typically used implicitly to create the inter-object links). An idealized example of collaborative filtering would be to recommend to a user the films that his or her best friend is known to have enjoyed, and which the user has not seen before; in practice the "best friend" is computed using statistical methods, most commonly as a form of cross-correlation. A simple example of item-based filtering would be to recommend a film to a user based on its genre (e.g. comedy, family, action, etc.) given prior knowledge of the genre of the films the user has enjoyed in the past; in practice, the genre is one of several features that can be used, but choosing the features to use can be problematic.

[1] the system can be accessed at http://q.cim.mcgill.ca

Each of these standard paradigms has been demonstrated to be successful, but each suffers from several shortcomings. In addition, we have identified several shortcomings that seem to be shared by *both* approaches and which we are attempting to address.

Collaborative filtering suffers from the fact that the basis for user preferences is not modeled whatsoever: one user might like a film for its humor, another might like it for its action. As such, the correlation between their tastes may be incidental [1]. For pairs of users that have only rated a small number of items, the risk of incidental correlations based on insufficient data is significant. On the other hand, for users that have rated a very large number of items, the risk of incidental correlations is also substantial.

Item-based recommendations can be based on several approaches. We will restrict our attention to those based on semantic features such as the genre of the items. For such systems, the set of features used to classify the items is critical and it needs to be both expressive and tractable. If the features in question (e.g *action movie*) are too broad or subjective, then the recommendations will suffer (for example, a husband and wife might disagree on the definition of a "dramatic film" or a "violent film"; sometimes this disagreement can itself be dramatic). On the other hand, if the features are too narrow to be applicable to many items, then either the rating matrix will become too sparse or ratings will be interpreted inconsistently in order to force them to apply more broadly.

Finally, it is often the case that the recommendations suitable to a user will be context dependent. The context of a user's search often has a significant bearing on what should be recommended, yet to our knowledge is has not been considered in the context of filtering applications. For example, if one is looking for a book, it is often for a specific purpose, be it light reading, self-improvement, or academic research, and a book that is appropriate to a reader in the light reading context will probably not be useful for research. Similarly, if one is looking for a film, then those that are appropriate for viewing with one's family may be different from those one would view when alone.

In this paper we outline the design of a recommender system that combines aspects of collaborative filtering and item-based recommendation. Our collaborative filtering algorithm and item-based components are based on statistical pattern matching methodologies. In the development of our item-

based model, however, we use a non-deterministic selection mechanism to define the attributes of interest that relate items to one another. Our approach is domain independent and should be suitable for most domains, but in the context of this paper we will use movie recommendations as our example domain.

A. Outline

In the remainder of this paper we discuss related work, and the problem of providing personalized recommendations. We continue with an exposition of the methodology and architecture, provide an illustrative example, and discuss some of the realized benefits and challenges. Finally, we close with a discussion of open problems, directions for future work and conclusions/findings from our work.

II. Related Work

Several different approaches have been considered for automated recommendation systems. Very broadly, the bulk of these can be classified into three major categories: those based on user-to-user matching and referred to as collaborative filtering, those based on item content information, and hybrid methods.

Collaborative filtering was first developed and identified as a methodology in the Tapestry email and Usenet filtering system [2]. In that work the emphasis was on the use and transmission of manually generated annotations of articles. In the GroupLens system the collaborative filtering paradigm was automated to provide automatic filtering of Usenet news articles [3].

Recommender systems are often described as being either memory- or model-based. Memory-based systems make predictions based on the entire raw data set, while model-based systems perform predictive calculations based on a version of the data which has been reduced in size [4]. A memory-based system might calculate nearest neighbors for each user and make predictions based on the preferences of those neighbors. In such systems, the similarity between users is often defined in terms of Pearson Correlation or the Vector Similarity measure used in Information Retrieval [4], [5]. Examples of model-based systems include Goldberg's "Eigentaste" framework [6] and Canny's "Mender" system [5], both of which map ratings data to a lower-dimensional subspace before making predictions. Systems such as the probabilistic Personality Diagnosis use a hybrid of the memory and model-based approaches [7].

In most recommender systems the overall opinion of an item is given as a integer value on some discrete scale. In the Entree recommender system, however, the user indicates a feature in which the item is lacking [1]. The system then determines which items are rich in the qualities being searched for, based on the responses of other users.

Due to the large number of items in most recommender systems, the ratings data will tend to be very sparse [8]. In such situations knowing a user's opinion of an arbitrary item may not help in determining the user's relationship with other users [9], [10]. An important aspect of a recommender system therefore is having a principled way of suggesting items to be rated. Approaches include using Partially Observable Markov Decision Processes [11] or Expected Value of Information [9] to determine which ratings will provide the most information. A simpler approach is to prompt the user to rate items which have been rated with a high variance [6].

In developing our recommender system we have taken all of these issues into consideration.

III. Approach

Typically in a recommender system a user will be given a list of items and prompted to indicate his or her preference for each. Preferences are indicated with a value in some range of integer values, e.g. $[1, 10]$, or by choosing "like" or "dislike". In Burke's Entree restaurant recommender system, instead of a numeric rating, the user selects a *semantic rating* which describes some aspect of the item [1]. For instance the user can select a rating such as "less expensive" or "nicer" to indicate an attribute he or she is looking for but feels is lacking in the current restaurant. In that work the set of possible semantic ratings a user can choose from is predefined when the system is created. Additionally, distances representing the similarity between the ratings are determined in advance by knowledge-engineering (i.e. manual intervention).

Other systems use information regarding the content of items in order to infer reasons behind a user's preferences. For instance if a user consistently exhibits a preference for movies which the system knows are classified under the action genre, then the system will automatically infer that the user enjoys action films.

In our system, we wish to have information about which items a user prefers, but also to collect information about which features of the item the user liked or disliked, i.e. which features contributed to the user's preference. In addition, we wish to learn a large set of features suitable for classification. To do this, we allow the user to suggest arbitrary features to the system at their discretion, and to classify items using these new *ad hoc* features. This avoids the need to guess all suitable features in advance. It also allows for the use of specialized features for specific sub-domains where specialized vocabulary may be appropriate. Finally, it permits new jargon to be introduced as it develops (for example in technical domains, or in domains related to popular culture). The obvious disadvantage of this is that (a) obscure or useless features may be introduced, (b) feature selection may become onerous due to the excess of available features and (c) redundant features may be introduced (such as both "funny" and "amusing" in the context of movies). We address these issues below.

In our system, the user indicates overall liking or disliking of an item with an integer rating on a scale of 1 to 10 (where 1 indicates an extremely negative opinion and 10 indicates an extremely positive opinion). To complete the rating the user is required to specify at least one feature of the item which was important to his or her overall rating. The user is presented with a list of possible features for rating purposes,

and is also given the opportunity to add new features to the system. For each feature chosen, the user specifies whether the feature contributes positively or negatively to the overall rating of the item, again on a scale of 1 to 10.

Both item-based and collaborative filtering-based recommendations have advantages [12]. Aside from the empirical data, item-based methods can be used to provide recommendations when the number of viewers (in the case of films) is too small to use collaborative filtering reliably (for example for new films). On the other hand, collaborative filtering can sometimes provide recommendations for items where the features have not been clearly defined, or in cases where existing features are not adequate descriptors. Thus, we have selected a hybrid approach that combines the two approaches as described above, with a weighting factor between 0.1 and 0.9 (i.e. one source has a weight of α and the other has a weight of $1-\alpha$).

A. Infrastructure

The system consists of a website implemented using the Zope application server to provide a dynamic HTML front end. User and item data is supported with a MySQL database. The more intensive computations are implemented in C and are connected to Zope and the website via Python. The movie database used is the EachMovie data set [2] (only the movie information was used).

The structure of the system, including the Zope web presentation, the database schema, and the recommendation code have all been designed to be domain-independent, so that domains other than movies can be easily used.

B. User similarity

The collaborative filtering component of our methodology hinges on the ability of define the similarity between a target user for whom we are to make a recommendation, and any other user. Once this similarity is defined it allows us to find users who are nearby to each other in the sense of preferences. The preferences of nearby users can then be used to compute a prediction for the unseen preferences of the user requesting recommendations. We use Pearson Correlation to compute similarity, and since we collect data both regarding the items themselves and the features of the items, we can compute similarity in either item or feature rating space.

Each user j will input a set of ratings in which $r_{ij} \in [1,10]$ is his or her overall rating of item i. The system contains a set of user-specified features $F = \{\phi_1, \phi_2, ..., \phi_t\}$. For each feature k felt to be important to the overall rating of item i, user j specifies a rating f_{ij}^k in the range $[1,10]$.

Given this ratings set, computing user similarity consists of two parts. First, we compute user similarity based purely on the "overall" ratings. The similarity between users p and q with respect to overall ratings is defined as the Pearson Correlation between their ratings:

$$s_r(p,q) = \frac{\sum_i (r_{ip} - \bar{r}_p)(r_{iq} - \bar{r}_q)}{\sqrt{\sum_i (r_{ip} - \bar{r}_p)^2 \sum_i (r_{iq} - \bar{r}_q)^2}} \quad (1)$$

where $\bar{r}_p$ and $\bar{r}_q$ are the mean overall item ratings for all items rated by users p and q, respectively, and the summations are over each item i rated by both users p and q.

Next, we compute the similarity between users based on their preference for features. We calculate the mean rating $\bar{f}_j^k$ given to feature k by user j, and $\bar{f}_j$, the mean feature rating assigned by user j across all ratings. Then we compute the similarity between two users p and q as the Pearson correlation between these feature rating statistics:

$$s_f(p,q) = \frac{\sum_k (\bar{f}_p^k - \bar{f}_p)(\bar{f}_q^k - \bar{f}_q)}{\sqrt{\sum_k (\bar{f}_p^k - \bar{f}_p)^2 (\bar{f}_q^k - \bar{f}_q)^2}} \quad (2)$$

where the summations are over all features k which have been used by both users p and q.

Given s_r and s_f and a weight $\alpha \in [0.1, 0.9]$ we can compute the hybrid similarity between users p and q as

$$s_\alpha(p,q) = (1-\alpha) s_r(p,q) + \alpha s_f(p,q) \quad (3)$$

Using equation 3 we can define the nearest neighbors of user j to be N_j^α, the set of n users with the highest similarity s_α to user j for a given α. The effect of α is to allow us to give more or less weight to the overall ratings or the feature ratings when determining user similarity.

Having chosen a value of α and computed the set of nearest neighbors to user j, we can predict the rating user j will give to item i:

$$r_{ij}^p = \bar{r}_j + \kappa \sum_{u \in N_j^\alpha} s_\alpha(j,u)(r_{iu} - \bar{r}_u) \quad (4)$$

where κ is a normalizing term.

C. User-specified features

In preliminary tests of the system a limited user population rapidly increased the number of features to over 100. Having to wade through this list to make each rating would be unacceptable to most users. Furthermore, several of the features only apply to certain classes of film, and some are too esoteric to be of interest to most users. Our approach, instead, is to select a useful subset of the features at any given time and present these to the user. On the other hand, it is worthwhile to collect *some* data on even those features of little apparent utility since new features need an opportunity to become useful, and even less useful features provide useful cues to inter-item associations. Thus, while the user is able to access and choose from the complete list of features, should that be desired, the system attempts to make the choice easier by probabilistically selecting a subset of features to be suggested for use. To suggest features for a particular item, the system computes the variance in ratings for each feature which has been used to rate the item. The features with the highest variance are considered to be informative because while users tend to agree the feature

[2] http://research.compaq.com/SRC/eachmovie/

is applicable to the item, they disagree about whether it is a positive or negative aspect of the item. On the other hand, low variance features are considered to be less informative. The highest variance features will always be suggested to the user, while a given low-variance feature t with variance σ_{it}^2 for item i will be suggested with the following probability:

$$P_i(\phi_t|\sigma_{it}^2) = \frac{1}{e^{g(\sigma_{it}^2)}} \quad (5)$$

where g is a normalizing function.

While the system automatically matches misspelled or similar features when they are added to the system, it is likely that users will enter features which are *semantically equivalent* to features already in the system. These features will be redundant and should be merged. By building and using a table of synonyms, we believe we can reduce the effect of these mutually redundant features. Such a table could be built in part by examining correlations in feature use as in Table II, i.e. features which are used for the same items very often can be flagged as possible synonyms by the system. The problem of redundant features may also admit more complex solutions and seems to be amenable to a correlation-based analysis and may be linked to issues of granularity [13].

IV. Preliminary Results

At the time of this writing, our results remain rather tentative as we are collecting further data. One difficulty we faced is that since our rating system is novel, any existing dataset we use can only be used for item information, and not for rating information. As a consequence the results presented below are preliminary, being based on the rating information we have been able to collect so far. At the time of writing the system consists of 48 users, 1016 ratings entered using 125 features, and the database contains 1641 movie entries.

We have estimated the optimum weight for our limited user population using leave-one-out cross validation. That is, for each user we ran a series of trials. In each trial, all but one of the items rated by the user were used to calculate a predicted rating for the remaining item (if possible), and then the error between the predicted and actual rating was calculated. We repeated this process for each user in the system, and calculated the mean of all absolute errors (MAE) over these trials. For cases where the number of nearest neighbors (as per Equation 4) is 12, the cross-validation error as a function of the weight given to feature information, α is illustrated in Fig. IV. It appears that optimum performance is achieved for a weight of 0.3. For some parameter settings, it appears that the optimum weight takes on different values, including cases where the optimum recommendation is produced by maximizing the influence of the feature-based recommendations. Due to the limited size of the user population these results should be regarded as tentative, but they suggest that item-based and collaborative filtering-based recommendations can be combined profitably. Further, the eclectic features suggested by a sample user population seem to provide effective recommendations, despite redundancy and some features of minimal utility.

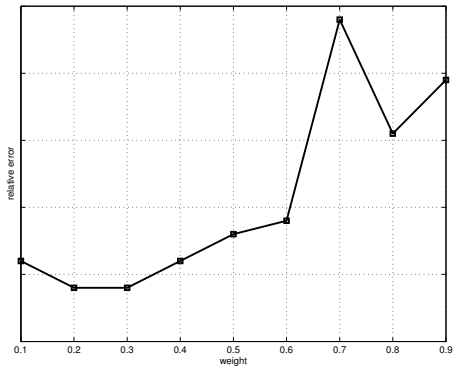

Fig. 1. Mean Absolute Error (MAE) of cross-validation test as a function of the weight α given to features (versus other users) when making predictions.

Presented in Table I is a list of some of the more popular features and the frequency with which they were used.

TABLE I
A selected sample of user-created features and a relative frequency indicator measuring how often each had been used.

Feature	Frequency of use
comedy	55
action	37
sci-fi	36
stupid	34
violence	22
long	15
black comedy	15
social commentary	13
dark	12
Ahnold-esque one-liners	11
complex	7
Ingrid Bergman	1

The features in the system consist of information about genre, actors, directors, as well as more high-level judgments, for instance "stupidity" and "complexity". Information about genre, actors, directors, and even plot, can be obtained mechanically from the description of the item (i.e. databases such as the EachMovie set contain genre information), and most people would agree about whether such a feature applies to an item or not. On the other hand, the higher-level features would typically only be available from subjective reviews of the item, and would be subject to debate.

Some of the items in the list might appear useless at first. Take for instance the feature "stupid". We expected that all users who used "stupid" to describe an item would use it negatively, but in fact it was used by many users, and as both a negative and positive feature. The same behavior was true for the feature "complex". In followup interviews that were conducted with a subset of users, some people indicated that they genuinely enjoyed movies with a "stupid" component (much to the amazement of one of the authors of this paper).

We also examined the correlation between pairs of features (i.e. pairs of features which were both used to describe the same item). Table II lists some of the more highly correlated and interesting combinations. Such a table will help a human

system administrator to identify semantically equivalent features since they will presumably be used to describe the same items.

TABLE II
A SAMPLE OF PAIRS OF USER-CREATED FEATURES WHICH ARE USED TO RATE THE SAME ITEMS.

horror	grotesque
space wars	futuristic
artsy	stylized
complex	Time Travel
complex	interesting plot
woody allen	intellectual
children	amusing
animation	children
historical	complex content
atmospheric	photography
surreal	dystopia
action	violence
artsy	lots of questions unanswered
very moving	character drama

V. DISCUSSION

In this paper we have outlined the design of a recommender system we have developed that combines collaborative filtering with continuously modifiable user-suggested feature-based recommendations. In preliminary trials it appears to provide good recommendations and addresses some issues with standard systems in terms of how items are rated.

One issue we are currently addressing is the need to adjust the recommendation processes conditionally as a function of the context in which the recommendation will be used. We are currently evaluating allowing users to define the context in which a recommendation should be considered. This will allow us to conditionally rate items and, hence, to subsequently make conditional recommendations (i.e. "this film would be good to watch with your children"). The absence of existing data sets has hampered the quantitative evaluation of this approach. Further elaboration is outside the scope of this paper and so at this time we can only suggest that it appears promising.

Examining the features used to classify films, it was clear to us that several of the features suggested by users were ones we would not have inserted ourselves, and yet they proved appealing to users and useful; in fact one might speculate that these whimsical features improved the level of user satisfaction (although we have no firm data to corroborate that). Further, the mere fact that users could add additional features to address perceived shortcomings in the system seems to enhance the sense of satisfaction and community, and reduce frustration – all important factors in a recommendation system that uses collaborative filtering.

Our results based in stochastic presentation of features based on utility seems to allow us to construct an item-based recommendation component of our system that uses a rather large number of user-defined features, while only requiring users to select from a limited list when indicating their preferences. This seems to lead to good performance of the recommendation system. On the other hand, users occasionally expressed frustration (in followup interviews) because features they had expected to find based on past experience were not present sometimes. We have addressed this on an interim basis by allowing users to manually type in additional features, and if these match an existing feature then that feature is used.

REFERENCES

[1] R. Burke, "Semantic ratings and heuristic similarity for collaborative filtering," in *AAAI Workshop on Knowledge-based Electronic Markets*, pp. 14–20, AAAI, 2000.

[2] D. Goldberg, D. Nichols, B. M. Oki, and D. B. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. CACM 35, no. 12, pp. 61–70, 1992.

[3] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," in *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, (Chapel Hill, North Carolina), pp. 175–186, ACM, 1994.

[4] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 43–52, 1998.

[5] J. Canny, "Collaborative filtering with privacy via factor analysis," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 238–245, ACM Press, 2002.

[6] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.

[7] D. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles, "Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach," in *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, UAI 2000*, (Stanford, CA), pp. 473–480, 2000.

[8] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the tenth international conference on World Wide Web*, pp. 285–295, 2001.

[9] C. Boutilier and R. S. Zemel, "Online queries for collaborative filtering," in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics* (C. Bishop and B. Frey, eds.), 2003.

[10] S. Dasgupta, W. Lee, and P. Long, "A theoretical analysis of query selection for collaborative filtering," *Machine Learning*, vol. 51, pp. 283–298, 2003.

[11] C. Boutilier, "A POMDP formulation of preference elicitation problems," in *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pp. 239–246, 2002.

[12] J. Herlocker, J. Konstan, A. Borchers, , and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in infofrmation retrieval*, pp. 230–237, 1999.

[13] P. Paulson and A. Tzanavari, "Combining collaborative and content-based filtering using conceptual graphs," in *Modeling with Words* (J. Lawry, J. Shanahan, and A. Ralescu, eds.), Lecture Notes In Artificial Intelligence Series, Springer-Verlag, 2003.