Hybrid Systems for Personalized Recommendations

Robin Burke

School of Computer Science, Telecommunications and Information Systems,
DePaul University, 243 S. Wabash Ave., Chicago, Illinois
rburke@cs.depaul.edu

Abstract. A variety of techniques have been proposed and investigated for delivering personalized recommendations for electronic commerce and other web applications. To improve performance, these methods have sometimes been combined in hybrid recommenders. This chapter surveys the landscape of actual and possible hybrid recommenders, and summarizes experiments that compare a large set of hybrid recommendation designs.

1 Introduction

Recommender systems are personalized information agents that provide recommendations, suggestions for items likely to be of use to a user [17, 25, 26]. They have been applied to many information access problems from news and email filtering to e-commerce product selection.

Recommendation techniques can be distinguished on the basis of their knowledge sources: where does the knowledge needed to make recommendations come from? In some systems, this is the knowledge of other user's ratings of the items in question. In others, it is ontological or inferential knowledge about the domain, added by a human knowledge engineer.

Specifically, recommender systems have *background data* that the system has before the recommendation process begins; *input data* that the user must communicate to the system in order to generate a personalized recommendation; and an *algorithm* that combines background and input data. On this basis, we can distinguish three commonly-used recommendation techniques: collaborative, content-based, and knowledge-based.[1]

1.1 Collaborative Recommendation

Collaborative recommendation is probably the most familiar, most widely implemented and most mature of the technologies. As shown in Figure 1, the collaborative recommender uses a profile from the current user and a profile database

[1] My earlier survey [7] includes two additional techniques: demographic and utility-based. Utility-based recommendation is a special case of knowledge based recommendation. Demographic recommendation is rarely used in a web context because users are generally reluctant to provide the personal data that would make such a technique effective.

B. Mobasher and S.S. Anand (Eds.): ITWP 2003, LNAI 3169, pp. 133 – 152, 2005.

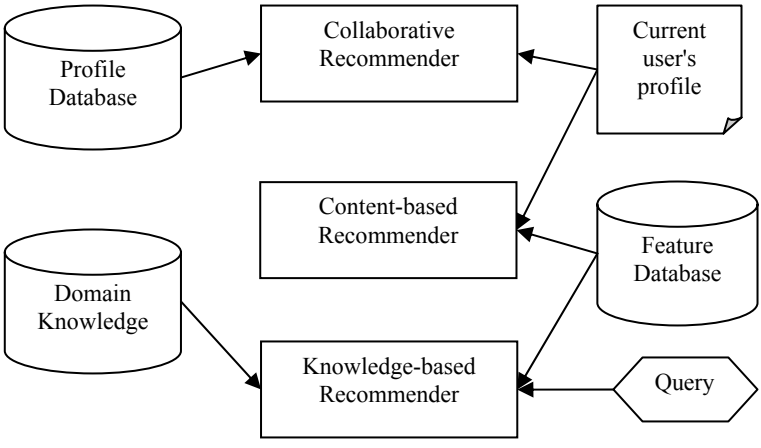

Fig. 1. Recommendation techniques and their knowledge sources

of other users. The system recognizes commonalities between the current user and those in the profile database, and generates new recommendations based on inter-user comparisons. A typical user profile in a collaborative system consists of a vector of items and their ratings, continuously augmented as the user interacts with the system over time. Some of the most important systems using this technique are GroupLens/NetPerceptions [25], Ringo/Firefly [30], Tapestry [14] and Recommender [17].

The greatest strength of collaborative techniques is that they are completely independent of any machine-readable representation of the objects being recommended, and work well for objects whose qualities are relatively intangible such as music and movies where variations in taste are responsible for much of the variation in preferences.

1.2 Content-Based Recommendation

Content-based recommendation is an outgrowth and continuation of information filtering research and is based on the idea of recommendation as classification [3]. Figure 1 shows the content-based recommender drawing from the user's profile and also from a feature database. For example, text recommendation systems like the newsgroup filtering system NewsWeeder [20] use the words of their texts as features. A content-based recommender learns a profile of the user's interests based on the features present in objects the user has rated. The type of user profile derived by a content-based recommender depends on the learning method employed. Decision trees, neural nets, and vector-based representations have all been used. As in the collaborative case, content-based user profiles are long-term models and updated as more evidence about user preferences is gathered.

1.3 Knowledge-Based Recommendation

Knowledge-based recommendation is similar to content-based recommendation in that it draws from the features of the recommended objects, but a knowledge-based

recommender also makes use of an explicit query formulated by the user. The knowledge-based recommender uses the query to make recommendations based on inferences about a user's needs and preferences. In some sense, all recommendation techniques could be described as doing some kind of inference. Knowledge-based approaches are distinguished in that they typically have some form of functional knowledge: they have knowledge about how a particular item meets a particular user need, and can therefore reason about the relationship between a need and a possible recommendation. By contrast, a content-based system has only ratings on which to base its conclusions. The Entree system (described below) and several other recent systems (for example, [29]) employ techniques from case-based reasoning for knowledge-based recommendation.

1.4 Strengths and Weaknesses

All recommendation techniques have strengths and weaknesses. Perhaps the best known weakness of the collaborative filtering technique is the "ramp-up" or "cold-start" problem [19]. This term actually refers to two distinct but related problems.

- **New User:** Because recommendations follow from a comparison between the target user and other users based solely on the accumulation of ratings, a user with few ratings becomes difficult to categorize.
- **New Item:** Similarly, a new item that has not had many ratings cannot be easily recommended.

Collaborative recommender systems depend on overlap in ratings across users and have difficulty when the space of ratings is sparse: few users have rated the same items. The sparsity problem is somewhat reduced in model-based approaches, such as singular value decomposition, which can reduce the dimensionality of the space in which comparison takes place [12, 27].

Content-based techniques also have a start-up problem in that they must accumulate enough ratings to build a reliable classifier. Relative to collaborative filtering, content-based techniques also have the problem that they are limited by the features that are explicitly associated with the objects that they recommend. For example, content-based movie recommendation can only be based on written materials about a movie: actors' names, plot summaries, etc. because the movie itself is opaque to the system. This puts these techniques at the mercy of the descriptive data available. Collaborative systems rely only on user ratings and can be used to recommend items without any descriptive data. Even in the presence of descriptive data, some experiments have found that collaborative recommender systems can be more accurate than content-based ones [1].

Knowledge-based recommenders do not have ramp-up or sparsity problems, since they do not base their recommendations on accumulated statistical evidence. Knowledge-based recommender systems are prone to the drawback of all knowledge-based systems: the need for knowledge acquisition. There are three types of knowledge that are involved in such a system: catalog knowledge, knowledge about the objects being recommended and their features; functional knowledge, needed to map between the user's needs and the object that might satisfy those needs; and, user

knowledge, knowledge of the user's need and preferences, sufficient to drive the recommendation process.

Despite this knowledge acquisition problem, knowledge-based recommendation has some beneficial characteristics. It does not involve a start-up period during which its suggestions are low quality. While a knowledge-based recommender cannot "discover" user niches, the way collaborative systems can, it can make recommendations as wide-ranging as its knowledge base allows.

These considerations show that no recommendation technique is perfect:

- Collaborative techniques have the unique capacity to identify cross-genre niches and can entice users to jump outside of the familiar. Knowledge-based techniques can do the same but only if such associations have been identified ahead of time by the knowledge engineer.
- Both of the learning-based techniques (collaborative and content-based) suffer from the ramp-up problem in one form or another. The converse of this problem is the stability vs. plasticity problem for such learners. Once one's profile has been established in the system, it may be difficult to change one's preferences.
- The learning-based technologies work best for dedicated users who are willing to invest some time making their preferences known to the system. Knowledge-based systems have fewer problems in this regard because they do not rely on having historical data about a user's preferences.

1.5 Hybrid Recommendation

The desire to avoid the weaknesses of individual recommendation methods has led researchers to consider hybrid recommendation systems, systems which combine techniques of different types with the expectation that the strengths of one will compensate for the weaknesses of another. Researchers have typically evaluated these hybrids by showing their benefits over non-hybridized systems, but there is little work that compares different hybrid designs against each other.

This chapter summarizes the results from a comparative study [8] that examines a large subset of the hybrid recommendation design space using a single data set. The next sections conform to the following outline. First, a taxonomy of recommendation hybrids is outlined and the landscape of possible hybrid designs articulated, based on the survey in [7]. Then, the data set and experimental methodology are described, followed by a summary of results from the large-scale study, and conclusions.

2 Hybrid Recommender Systems

There are seven basic ways that recommender systems can be combined to build hybrids:

– Mixed: Results for different recommenders are presented together either in a combined presentation or in separate lists.
– Weighted: Scores from the recommenders are combined using weights to derive a single score.

Table 1. Two-Component Hybrid Recommendation Designs

	Weighted	Switching	Feature Combination	Cascade	Feature Aug.	Meta-level
CF/CN						
CF/KB						
CN/CF						
CN/KB						
KB/CF						
KB/CN						

(CF = collaborative, CN = content-based, KB = knowledge-based)

Redundant
Not possible

- Switching: The system uses some decision criteria to choose a recommender based on the context and uses the results from only the chosen source.
- Cascade: One recommender refines the recommendations produced by another.
- Feature Combination: Data from different source types are combined together and treated using one recommendation algorithm.
- Feature Augmentation: The output from one technique is used as an input feature to another.
- Meta-level: One recommender produces a model, which is then used as input for the second recommender.

If we consider these possible hybridization strategies and the three recommendation techniques discussed above, we can derive a matrix of possible hybrid recommender designs. Table 1 shows this matrix. For the sake of simplicity, the table omits "Mixed" hybrids, which do not combine evidence from their components and are not really comparable to other hybrids.

There are three hybridization techniques that are order-insensitive: Weighted, Switching and Feature Combination. With these hybrids, it does not make sense to talk about the order in which the techniques are applied: a CN/CF weighted system would be no different from a CF/CN one. The redundant combinations are marked in gray.

The cascade, augmentation and meta-level hybrids are inherently ordered. For example, a feature augmentation hybrid that used a content-based recommender to contribute features to be used by a second collaborative process, would be quite different from one that used collaboration first. With these techniques all permutations must be considered and these columns do not contain any redundancies.

There are 27 non-redundant spaces in the table, but some combinations are not possible. Since a knowledge-based technique may take into account any kind of data, feature combination does not really represent a possible hybrid. The illogical hybrids are marked in black. The white areas of the table enumerate 25 different possible hybrid recommender systems.

3 Experiments with Hybrid Recommendation

This paper examines some of the results from a large comparative study of hybrid recommender systems [8], looking at a large subset of Table 1 using the same data and recommendation problem. To understand the evaluation methodology employed in this study, we will need to examine the characteristics of the Entree data set. Entree [4, 5] is a knowledge-based restaurant recommendation system that uses case-based reasoning [18] techniques to select and rank restaurants. It was implemented to serve as a guide to attendees of the 1996 Democratic National Convention in Chicago and operated as a web utility for approximately three years. The system is interactive, using a critiquing dialog [9, 31] in which users' preferences are elicited through their reactions to examples that they are shown.

Fig. 2. Results of a query to the Entree restaurant recommender

Figures 2 and 3 show a user's interaction with the system. An initial query based on a favorite restaurant yields a similar Chicago establishment. When this is deemed too expensive, a click on the "Less $$" critique navigates to a less expensive option.

The data set produced by this interaction is in the form of user histories: $<u, h>$ where each history h is a set of pairs $<r_i, s_i>$ where r_i is a restaurant, and s_i is one of 9 possible responses: entry point, exit point, or one of the seven critiques. This data set has some substantial differences from the standard collaborative-filtering data sets

frequently used in the recommendation literature, and is more similar in some respects to the log data used in web personalization [22]. The sessions are short, with only a small percentage containing more than a dozen interactions. User tracking technology was not employed when the system was deployed, but it is possible to heuristically join sessions into long-term user profiles of larger size. However, the task constraints of the restaurant search problem are such that long-term profiles are less likely to be valuable – an intuition borne out by experiment. With such short sessions, we cannot expect to get a large amount of information about a user, and this limits how well a recommender can be expected to perform.

Fig. 3. Results of the "Less $$" critique

Explicit rating data and standard web mining data such as dwell time are not available. However, we do have the evidence of the user's critiques. The critiques can be interpreted as negative ratings; they result in the user moving away from the suggestion that is shown. There are few actions that can be interpreted as positive ratings. The system allows a user to input a favorite restaurant as a starting point for recommendation, which can definitely be considered positive. However, this only occurs in about 10% of the sessions. End points may constitute either successful recommendations (which should be positive ratings) or abandoned sessions (noise). To determine the effects of this noise, we experimented using only the definite positive (starting point) ratings in the subset of the data in which these ratings are available and compared these results with the experiments treating end points also as positive ratings.

Because this data is much small than the full one, the algorithms are less accurate, but the correlation between the two conditions was extremely high (0.92). In the experiments described here, both start and end points are used as positive ratings, with the understanding that there is some noise in the positive rating data.

Despite its problems, the Entree data has the unique advantage in that a working knowledge-based recommender exists for the restaurant domain (the Entree recommender itself), the hard work of knowledge acquisition having already been performed. This means that it is possible to compare the designs shown in Table 1. Without a knowledge-based component, only nine of these designs can be evaluated. In this article, we will not discuss the six possible meta-level hybrids, which are somewhat more complex in design and for which experimental results are still incomplete. See [8] for additional details.

3.1 Evaluation Metrics

Herlocker and colleagues have studied a variety of evaluation techniques for collaborative filtering systems [16]. Different criteria may be important in different contexts. In the restaurant recommendation domain, "Find Good Recommendations" is the appropriate task context. This work identifies three basic classes of evaluation measures: discriminability measures (such as ROC-derived measures) , precision measures (such as mean absolute error) and holistic measures (that work best when all user ratings and system predictions are pooled and evaluated as a group). In each of these groups, a wide variety of different metrics were found to be highly correlated, effectively measuring the same property. For the restaurant recommendation task, we are interested in a precision-type measure, and Herlocker's results tell us that we need not be extremely picky about how such a measure is calculated.

With short sessions and a dearth of positive ratings, there are some obvious constraints on how the Entree sessions can be employed and recommendations evaluated. An evaluation technique that requires making many predictions for a given user will not be applicable, because there would not be enough of a profile left for a collaborative system to use. This rules out such standard metrics as precision/recall and mean absolute error. Ultimately, in order to find good recommendations, the system must be able to prefer an item that the user rated highly. How well the system can do this is a good indicator of its success in prediction, so our evaluation will concentrate on those items the user likes. What we do is to record the rank of a positively-rated test item in the recommendation set returned by a given recommender. Averaging over many trials we can compute the "average rank of the correct recommendation" or ARC. The ARC measure provides a single value for comparing the performance of the hybrids, focusing on how well each can discriminate the item liked by the user from the others.

The evaluation of the recommenders proceeded as follows. The set of sessions was divided randomly into training and test parts of approximately equal size. This partition was performed five times and results from each test/training split averaged. Each algorithm was given the training part of the data as its input, each handling this data in its own way, and in some cases, such as with the knowledge-based recommender, it was ignored. Evaluation was performed on each session of the test data, simulating the interaction of the system with a single user. From the session, a single item with a

positive rating was chosen.[2] This item was the test item for which the ARC value was calculated. All of the other ratings were considered part of the user profile.

The recommendation algorithm was then given the user profile without the positively-rated test item, and made its recommendations. The result of the recommendation process was a ranked subset of the product database containing those items possibly of interest to the user. From this ranked list, the rank of the test item was recorded.

There are two variables that can be manipulated with respect to session size. One is the number of ratings the user has provided; the other is whether the profile consists of a single session or multiple interactions. There are only a small percentage of single sessions of size 15 or greater, so for long profiles, we must turn to multi-session ones. To examine how the hybrids fared on both types of profiles, we examine each system's performance with single sessions of size 5, 10 and 15 and multi-session profiles of 10, 20 and 30. These conditions are indicated as "5S", "10S", "15S" and "10M", "20M", "30M", respectively.

The content-based and collaborative components fit well into this evaluation paradigm. They are designed to accept a profile as input and produce a recommendation. A knowledge-based component is different. The Entree recommender needs a query in order to produce output, and in order to use such a component we must decide where its queries will come from. One possibility would be to use the features of all of the restaurants that appear in the profile and derive a composite representation that would serve as a query. This did not work well in practice, no doubt because the Entree interface encourages users to explore, and cumulative profiles contain many digressions. A better alternative is to pick the last item in the profile and use it as the query. This item represents in some sense the user's progress toward the final recommendation destination, and perhaps the convergence of their preferences. The last item in the profile is not necessarily the critique immediately prior to the end point, since we evaluate profiles at fixed sizes as discussed above and extra (negative) ratings are truncated.

4 Results

To provide a starting point for analysis of the hybrids, we can examine the four basic algorithms, including the performance of the "average" recommender, which recommends restaurants based on their average rating from all users, and does not take the user profile into account. The three basic algorithms are CF, a correlation-based collaborative algorithm using Pearson's r [15]; CN, a content-based recommender implemented using the Naive Bayes technique [13]; and KB, the Entree recommender system used as a knowledge-based component.

Figure 4 shows the average rank of the correct recommendation (ARC) for each of the basic algorithms over the six different session size conditions. We should note that this recommendation task is actually rather difficult, especially for the multi-session profiles. The best any of these basic algorithms can manage is average rank of

[2] If there are no positive ratings, the session is discarded. We cannot evaluate the quality of recommendation if we have no information about what the user prefers.

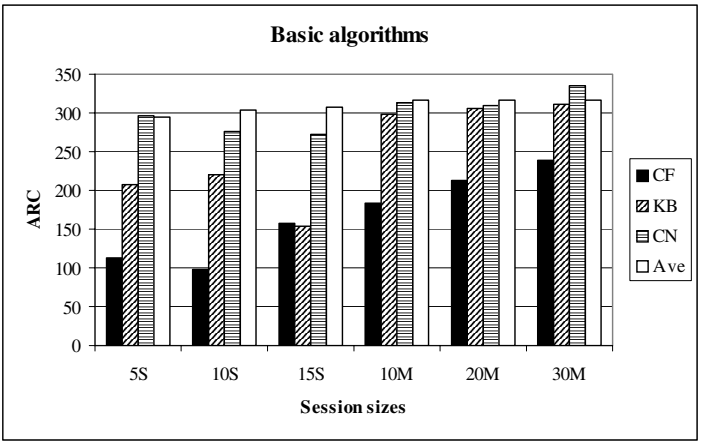

Fig. 4. Average rank results for the basic recommenders

approximately 100 for the correct answer, not inspiring in an e-commerce context where the user might be expected only to look at the first dozen results or so. The techniques vary widely in their performance on the Entree data. On the multi-visit profiles, all of the algorithms do worse, with the knowledge-based technique in particular falling back considerably from its single-session performance.[3]

4.1 Weighted

Perhaps the simplest design for a hybrid system is a weighted one. Each component of the hybrid scores a given item and the scores are combined using a linear formula [11]. Entree returns integer scores,[4] which are normalized to the range 0..1. To derive the optimum weighting, we examine all possible weightings (in discrete increments) and determine which weighting, over the training data, would yield the best ARC value.

Three weighted combinations of the algorithms were possible: CF/CN, CF/KB, CN/KB. The results for this hybrid were rather surprising. Figure 5 shows the average rank results. In all but five of the 18 conditions, the performance of the combined recommenders was worse than the stronger recommender alone. In two of the conditions, the weighted hybrid was either the same or worse than the weakest recommender of the hybrid pair. Only in five conditions is the weighted hybrid superior to its components separately, with the remaining conditions showing performance in-between the two components.

---

[3] The content-based algorithm may suffer due to the skewed nature of the profiles: with many negative ratings and few positive ones, but recall that the ARC metric measures where a recommendation is placed relative to others in the list and all items are effected by the negative baseline. More likely the paucity of data is responsible for the poor performance of the content-based technique. Because the evaluation paradigm for the knowledge-based technique assumes consistency in preferences, it is particularly affected in the multi-session case.

[4] A peculiarity dictated by the system's similarity metrics [4].

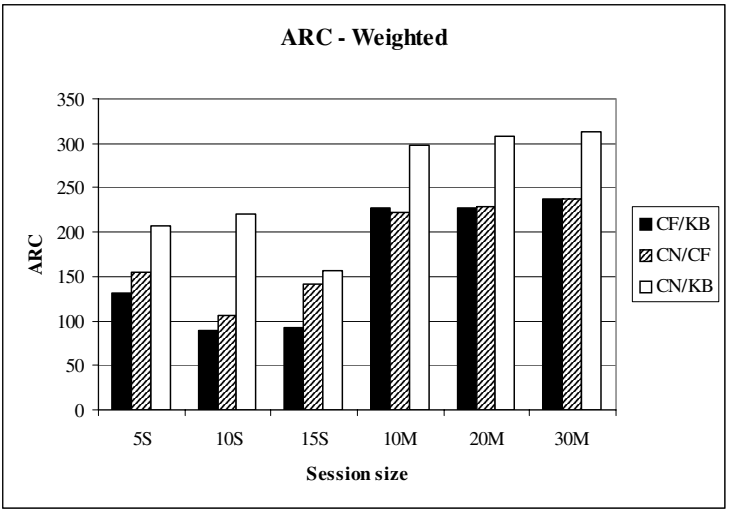

Fig. 5. ARC results for weighted hybrids

There are several reasons why this result might occur. One is that the recommendation components are not sufficiently independent for their combination to be different that any one alone. However, we know this is not the case since the measures use very different knowledge sources and as we will see below, other types of hybrids using these components are successful. More likely, the problem is that the recommenders, especially KB and CF, do not have uniform performance across the product and user space and the underlying assumption behind a linear weighting scheme is at fault. This suggests the application of the next type of hybrid, one in which the hybrid switches between its components depending on the context.

4.2 Switching

A switching hybrid is one that chooses a single recommender from among its constituents in each recommendation situation. In order to implement such a hybrid, there must be some criterion available to enable the switching decision. We can think of this value as a "confidence" value.

Ideally, we would survey the confidence values computed by each algorithm and choose the most confident. However, this would assume comparability between confidence values computed in different ways, and experiments showed this was not a valid assumption. An alternative is to select one component of the hybrid as the primary recommender and let it determine the confidence in its own prediction. If the primary recommender has confidence above some threshold, its recommendation will be used; otherwise, the secondary recommender takes over.[5] This distinction between

[5] There are other possibilities for implementing a switching hybrid. The system might look at the difference between confidence values or their ratio, for example. An alternative is to have a third metric outside of the hybrids as the switching criterion. Mobasher and Nakagawa [23] describe a system in which a metric of site connectivity is used to determine which of two usage mining techniques to employ.

primary and secondary recommenders makes the switching hybrid an order-sensitive system. Like the weighted hybrid, all possible switching thresholds are tried and the threshold that maximizes ARC performance on the training set is used.

Each of the recommenders in the experimental set required a different confidence calculation, normalized to the range [0-1]. For the collaborative algorithms, the confidence value is computed using the inverse of the average distance between these peer profiles and the user, since the closer the peers to the user, the more likely it should be that they are good predictors. For the naive Bayes algorithm, the choice of confidence metric is fairly straightforward – the value returned by the naive Bayes classifier is supposed to represent the probability that the classified object is a member of the given class. For the knowledge-based algorithm, the confidence is computed by finding the overlap in features between the top recommendation and the query. The intuition here is that if the knowledge-based system did not have to go far afield (and thereby making many inferences) to make its retrieval, then the results returned will be more confident.

To be a good primary recommender in this switching paradigm, an algorithm must have a reliable assessment of its own accuracy. Otherwise, it will turn over control to the secondary recommender when its own results might be more correct and make recommendations when the secondary one might be better. In some cases, when a range of confidence thresholds were attempted, the best accuracy was achieved when the primary algorithm had a threshold equal to 1.0, meaning that the recommender would have to compute a confidence > 1 in order for its recommendations to be used. This is an impossibility, and so in these degenerate cases, the primary recommender is essentially ignored and the recommender becomes a non-hybrid made up only of the secondary component. This happens when the primary recommender is particularly weak.

For the sake of brevity, Figure 6 and the rest of the graphs only show those hybrids that achieve synergy: that is, the hybrid together performs better than either of its

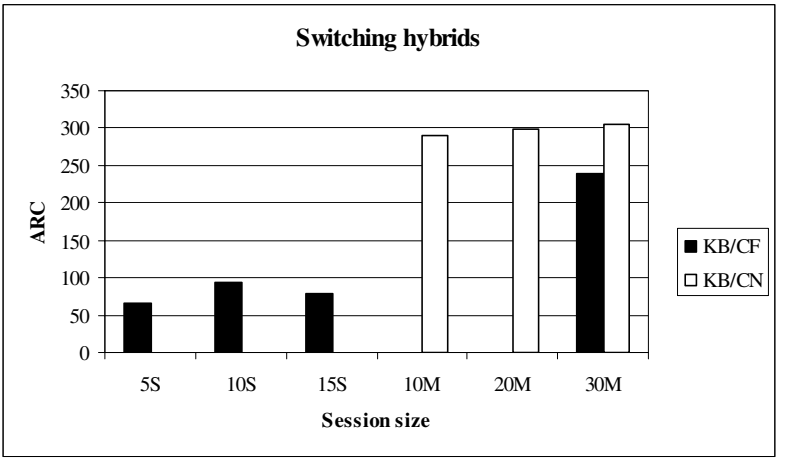

Fig. 6. ARC results for switching hybrids

components on their own. Also, for switching hybrids, the degenerate (threshold = 1.0) cases have been omitted, so this chart shows only a subset of the systems tested. Only in the KB/CF case do we see true synergy: the CF algorithm achieved an ARC around 150 in the 15S condition, but the KB/CF hybrid is close to half that value in the same condition.

The relatively poor performance of these hybrids indicates the difficulties presented in accurately computing the confidence that should be associated with an algorithm's prediction. See [10] for a discussion of the difficulties of calculating confidence in a prediction context.

4.3 Cascade Hybrids

The idea of a cascade hybrid is to create a strictly hierarchical hybrid, one in which a weak recommender cannot overturn decisions made by a stronger one, but can merely refine them. A cascade recommender uses a secondary recommender only to break ties in the scoring of the primary one.

Many recommendation techniques have real-valued outputs and so the probability of identical scores is small. This would give the secondary recommender in a cascade little to do. In fact, the literature did not reveal any other instances of the cascade type at the time that the original hybrid recommendation survey [7] was completed. However, the cascade hybrid raises the issue of the appropriate confidence interval for the score returned by recommendation algorithms, presumably much less than the full 32 bits in the computational representation. And, if the scoring of our algorithms is somewhat less precise, then there may be space in which a cascade can operate. Figure 7 shows the ARC graph comparing the CF and CN recommenders with full 32-bit precision against the same algorithms truncated to two digits of precision (the LP versions). We see that overall the differences are very small and not always in the

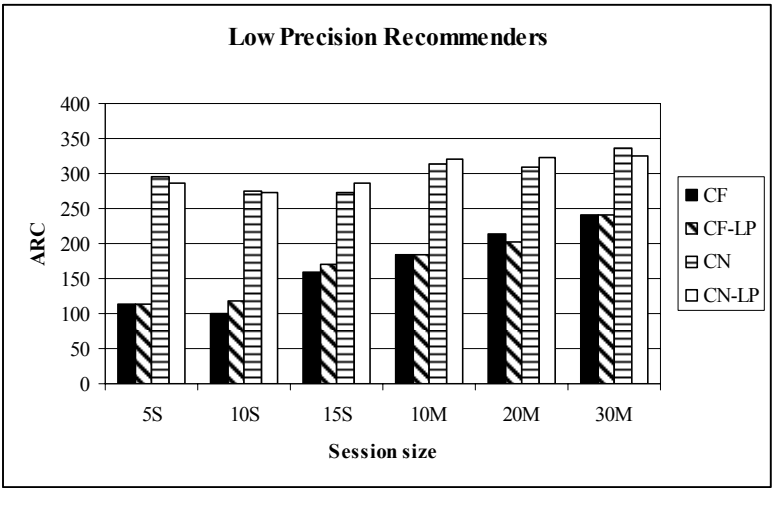

Fig. 7. ARC results for low-precision recommenders compared with full-precision implementations

favor of the higher-precision algorithm. Our cascade hybrids therefore use these low-precision versions of the algorithms, generating ties that a secondary recommender can break.

With this result in mind, we can turn to the cascade recommenders. Figure 8 shows the ARC results for these hybrids. Again, only those recommenders demonstrating synergy are shown. The successful cascade combinations are very good. (Note the change in scale in the y-axis.) Most significant is the behavior on the multi-profile sessions, for which most systems examined so far have been inadequate.

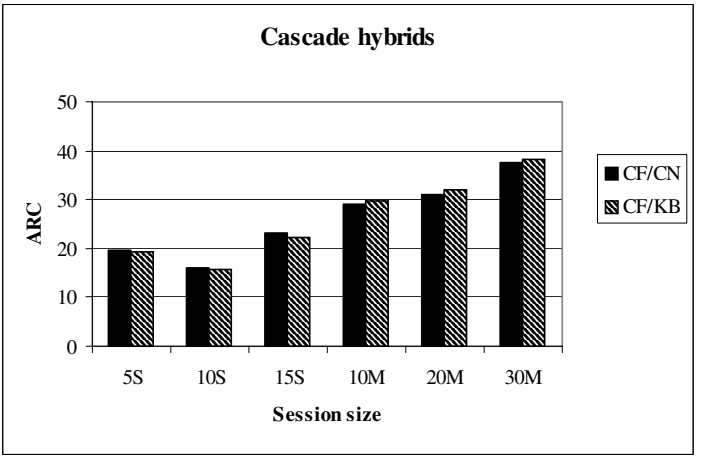

Fig. 8. ARC results for cascade recommenders

4.4 Feature Combination Hybrids

The idea of feature combination is to inject features associated with one recommendation type (such as collaborative recommendation) into an algorithm designed to process data with a different source (such a content-based recommendation). This is a way to expand the capabilities of a well-understood and well-tuned system, by adding new kinds of features into the mix [2, 24].

The content-based recommender with a contributing collaborative part (CF/CN) was built by augmenting the representation of each restaurant with new features corresponding to the reaction of each profile in the training data to that restaurant. For example, if profiles A and B had negative ratings for restaurant X and profile C had a positive rating, the representation of restaurant X would be augmented with three new features, which can be thought of as A-, B- and C+. Now an ordinary classification algorithm can be employed using the test user's profile to learn a model of user interests, but this model will now take into account similarities between restaurants that have a collaborative origin.

A feature combination hybrid uses only one recommendation algorithm, but this should not disqualify it from consideration as a hybrid. The original taxonomy presented above and illustrated in Figure 1 defines recommendation techniques based

on their knowledge sources: what it is that they need to know in order to generate recommendations. This is a most important consideration for system design and implementation, because it determines what data the system must be able to store and track and what outside data must be supplied to it. In this definition, a content-based recommender is one that is aware of one user's profile and the product database. A collaborative hybrid in which the entire profile database is also brought to bear is a very different type of system, regardless of the algorithm by which it is achieved.

The naive Bayes implementation performed poorly with this augmented model, including as it does over 5,000 new collaborative features in addition to the 256 content ones. So, for this hybrid only, I examined the Winnow algorithm [21], another classification learning algorithm that scales better to large numbers of features. (Naives Bayes outperformed Winnow in the non-hybridized case.)

A collaborative recommender with a contributing content-based component turns this process around and creates artificial profiles corresponding to particular content features; these are sometimes called "pseudo-users" [28]. For example, all of the restaurants with Tex-Mex cuisine would be brought together and a profile created in which the pseudo-user likes all of the Tex-Mex restaurants in the database. Similar profiles are generated for all the other content features.

Other possibilities for feature combination hybrids turn out to be either illogical or infeasible. The features that the knowledge-based recommender uses are the same as those used by the content-based recommender; they are just used in a different way. A feature combination hybrid with a knowledge-based contributing part would therefore be no different from the content-using one described above. A knowledge-based hybrid with a collaborative contributing recommender would be theoretically possible: a knowledge engineer could write rules that make inferences about the preferences of the users in our test set, but such an enterprise would be wholly impractical in any fielded application and would run counter to the intent that as little as possible extra work would be done on the base recommenders in order to implement the hybrids.

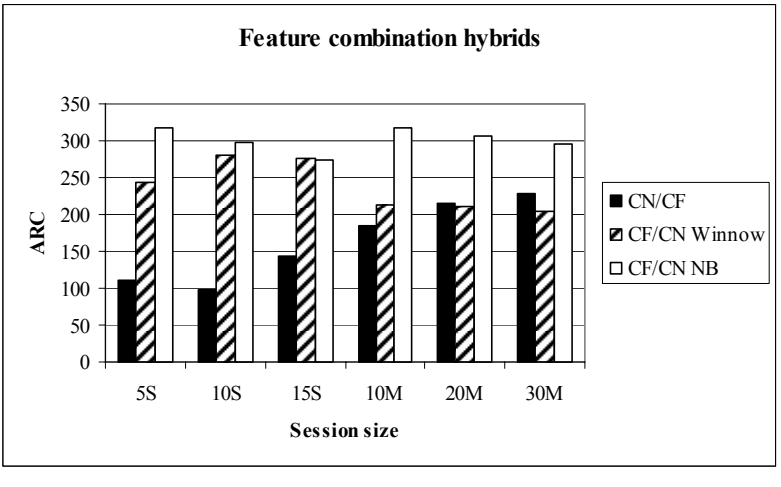

Fig. 9. ARC results for feature combination hybrids

Figure 9 shows the average rank data for these hybrids. The CN/CF hybrid is nearly identical to the unhybridized CF algorithm. The CF/CN hybrid with Winnow shows modest success for the multi-session profiles.

4.5 Feature Augmentation Hybrids

Feature augmentation is a strategy for hybrid recommendation in which a contributing recommender generates a new feature or set of features for each item, augmenting the data for the primary recommender with its own contribution. The augmentation can usually be done off-line, making this approach attractive when trying to strengthen an existing recommendation algorithm by adjusting its input.

The integration of components in a feature augmentation arrangement is somewhat trickier than in the hybrids seen so far. The contributing recommender must actually modify the input of the primary recommender and this is different than merely producing a score. So, the KB component must produce features associated with items that the other components can use, in order to be a contributing component, and it must reason with the features produced by the other components in order to be a primary recommender.

The feature generation technique used in these experiments is clustering. To use clustering techniques, we can envision the set of user profiles as a set of sparse vectors. Each user is a row and each restaurant a dimension. Each user will only have rated a few restaurants, which is what makes each vector sparse. Clustering these vectors will group similar users together. An alternate representation would be one in which there is a vector for each restaurant with the users being the dimensions. Clustering the restaurant vectors yields information on what restaurant are similar to others, based on their patterns of preference across users. The result of a clustering computation is a cluster id, which is effectively a collaboratively-derived feature that can be associated with each restaurant. Restaurants that share a cluster will have the same id, and this can be part of the input to the CN component, which is expecting input consisting of restaurants and their features.

When the KB component is the primary recommender, we use these ids in a very simple manner: an additional similarity metric is added to the recommender that prefers restaurants in the same cluster. This does require modification of the Entree recommender, but it is the minimal amount required to make use of the new features.

When the KB component is the contributing recommender, there is no underlying data that can be used as features for this type of hybrid. If we want to build a KB/CN feature augmentation query, we need the KB component to contribute restaurant features derived from its similarity knowledge. Sticking with the idea of clustering, we perform the following operation. For each restaurant, use the KB component to compute the most similar restaurant. Build a "user profile" for each restaurant, where each similar restaurant is given a positive rating. These profiles can then be input to the clustering algorithm described above, yielding a cluster id based on the system's similarity knowledge, which are then added to the features used by the CN component.

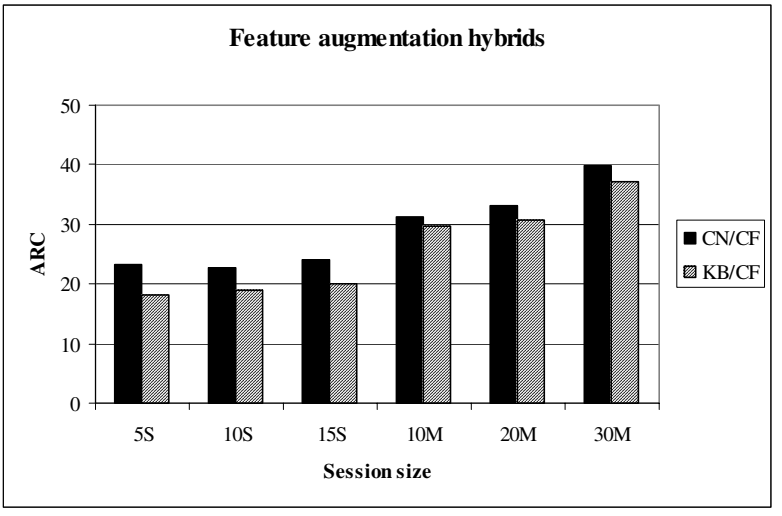

Fig. 10. ARC results for feature augmentation hybrids

The KB/CF feature augmentation hybrid is slightly different. The features that the CF component "understands" are profiles, lists of restaurants/rating pairs. A KB/CF hybrid can be achieved through pseudo-users created through retrieval. We perform retrieval based on the user profile and then create a pseudo-user, who gives positive ratings to all the restaurants returned by the query and negative ratings to the others.

The results for these recommenders shown in Figure 10 are impressive for the both of the cases shown. While other combinations did not achieve synergy, the successful implementations were those in which the collaborative component was the one receiving augmented data. Here we see very strong performance, especially the KB/CF recommender which has an ARC below 40 throughout all of the profile types. This follows the pattern seen in the cascade results, where a combination of a collaborative and a knowledge-based component produced the best results.

5 Conclusion

The experiments outlined in this paper cover a large portion of the space of possible two-component hybrid recommender systems based on three basic recommendation algorithms: content-based, collaborative, and knowledge-based and five types of combinations: weighted, switching, cascade, feature combination, and feature augmentation. The work surveyed 19 different systems in all, including 10 designs without previous extant examples.

Of course, any such study is by its nature limited by the peculiarities of the available data and the recommendation domain. Of course, any such study is by its nature limited by the peculiarities of the available data and the recommendation domain. The Entree data set is relatively small (just over ¼ million ratings), the

profiles are short and the ratings are implicit and heavily skewed to the negative. It would be valuable to repeat this study in a different recommendation domain with different products and a set of user profiles with different characteristics. The most substantial hurdle to such a study is the availability of a knowledge-based recommendation component.

Three results, however, can be seen, which may have general applicability and are worthy of further study. First is the utility of a knowledge-based recommender itself. Such recommenders have been explored as a knowledge-based solution to the problem of product search, and have been deployed in a number of e-commerce applications [6, 27]. The experiments presented here show that a knowledge-based recommendation engine may also be combined in numerous ways to build hybrids and in fact, some of the best performing recommenders seen in these experiments were created by using a knowledge-based component.

In examining the hybrids themselves, we see that cascade recommendation, although rare in the hybrid recommendation literature, turns out to be a very effective means of combining recommenders of differing strengths. Adopting this approach requires treating the scores from a primary recommender as rough approximations, and allowing a secondary recommender to fine-tune the results. In addition, feature augmentation also is shown to be highly effective, and this technique has the added efficiency that the contributing recommender can produce its augmenting features off-line.

Acknowledgements

Entree was developed at the University of Chicago in collaboration with Kristian Hammond, with the support of the Office of Naval Research under grant F49620-88-D-0058. Parts of the experimental design were done with the assistance of Dan Billsus at the University of California, Irvine. Thanks to Michael Pazzani and Bomshad Mobasher for helpful comments on an early draft of this paper.

References

[1] Alspector, J., Koicz, A., and Karunanithi, N.: 1997, 'Feature-based and Clique-based User Models for Movie Selection: A Comparative Study'. *User Modeling and User-Adapted Interaction* 7, 279-304.

[2] Basu, C., Hirsh, H. and Cohen W.: 1998, 'Recommendation as Classification: Using Social and Content-Based Information in Recommendation'. In: *Proceedings of the 15th National Conference on Artificial Intelligence*, Madison, WI, pp. 714-720.

[3] Belkin, N. J. and Croft, W. B.: 1992, 'Information Filtering and Information Retrieval: Two Sides of the Same Coin?' *Communications of the ACM* 35(12), 29-38.

[4] Burke, R.: 1999, 'The Wasabi Personal Shopper: A Case-Based Recommender System'. In: *Proceedings of the 11th National Conference on Innovative Applications of Artificial Intelligence*, pp. 844-849.

[5] Burke, R.: 2000, 'Knowledge-based Recommender Systems'. In: A. Kent (ed.): *Encyclopedia of Library and Information Systems*. Vol. 69, Supplement 32.

[6] Burke, R.: 2001, *Case-Based Reasoning in Electronic Commerce*. Workshop held at ICCBR 2001, Vancouver, Canada. Proceedings available at http://www.aic.nrl.navy.mil/-papers/2001/AIC-01-003/ws3/ws3.htm

[7] Burke, R.: 2002, 'Hybrid Recommender Systems: Survey and Experiments'. *User Modeling and User Adapted Interaction*, 12 (4), 331-370.

[8] Burke, R.: in preparation, 'Hybrid Recommender Systems: Comparative Studies'.

[9] Burke, R., Hammond, K., and Young, B.: 1997, 'The FindMe Approach to Assisted Browsing'. *IEEE Expert*, 12 (4), 32-40.

[10] Cheetham, W and Price, J.: 2004, 'Measures of Solution Accuracy in Case-Based Reasoning Systems'. In: P. Funk and P.A. Gonzalez-Calero (eds.) *Advances in Case-Based Reasoning (ECCBR 2004)*, pp. 106-118.

[11] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M.: 1999, 'Combining Content-Based and Collaborative Filters in an Online Newspaper'. *SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA. <URL: http://www.cs.umbc.edu/~ian/sigir99-rec/papers/claypool_m.ps.gz>

[12] Foltz, P. W.: 1990, 'Using Latent Semantic Indexing for Information Filtering'. In: R. B. Allen (ed.): *Proceedings of the Conference on Office Information Systems*, Cambridge, MA, pp. 40-47.

[13] Friedman, N., Gieger, M., and Goldszmidt, M.: 1997, 'Bayesian Network Classifiers'. *Machine Learning*, 29, 131-163.

[14] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D.: 1992, 'Using collaborative filtering to weave an information tapestry'. *Communications of the ACM*. 35 (12), 61-70;

[15] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. T.: 1999, 'An Algorithmic Framework for Performing Collaborative Filtering'. In *ACM SIGIR 1999*, pp. 230-237.

[16] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T.: 2004, 'Evaluating Collaborative Filtering Recommender Systems'. *ACM Transactions on Information Systems*. 22 (1).

[17] Hill, W., Stead, L., Rosenstein, M. and Furnas, G.: 1995, 'Recommending and evaluating choices in a virtual community of use'. In: *CHI '95: Conference Proceedings on Human Factors in Computing Systems*, Denver, CO, pp. 194-201.

[18] Kolodner, J.: 1993, *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann.

[19] Konstan, J. A., Riedl, J., Borchers, A. and Herlocker, J. L.: 1998, 'Recommender Systems: A GroupLens Perspective.' In: *Recommender Systems: Papers from the 1998 Workshop (AAAI Technical Report WS-98-08)*. Menlo Park, CA: AAAI Press, pp. 60-64

[20] Lang, K.: 1995, 'Newsweeder: Learning to filter news'. In: *Proceedings of the 12th International Conference on Machine Learning*, Lake Tahoe, CA, pp. 331-339.

[21] Littlestone, N. and Warmuth, M.: 1994, 'The Weighted Majority Algorithm'. *Information and Computation* 108 (2), 212-261.

[22] Mobasher, B.: 2004, 'Web usage mining and personalization'. In *Practical Handbook of Internet Computing*, Munindar P. Singh (editor), Chapman Hall & CRC Press.

[23] Mobasher, B. and Nakagawa, M.: 2003, ' A Hybrid Web Personalization Model Based on Site Connectivity'. In *Proceedings of the WebKDD Workshop at the ACM SIGKKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, August 2003.

[24] Mooney, R. J. and Roy, L.: 1999, 'Content-Based Book Recommending Using Learning for Text Categorization'. *SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA. <URL: http://www.cs.umbc.edu/~ian/sigir99-rec/papers/ mooney_r.ps.gz>

[25] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J.: 1994, 'GroupLens: An Open Architecture for Collaborative Filtering of Netnews'. In: *Proceedings of the Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, pp. 175-186.

[26] Resnick, P. and Varian, H. R.: 1997, 'Recommender Systems'. *Communications of the ACM*, 40 (3), 56-58.

[27] Rosenstein, M. and Lochbaum, C.: 2000, 'Recommending from Content: Preliminary Results from an E-Commerce Experiment.' In: *Proceedings of CHI'00: Conference on Human Factors in Computing*, The Hague, Netherlands.

[28] Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J. Miller, B. and Riedl, J.: 1998, 'Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System'. In: *Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work*, Seattle, WA, pp. 345-354.

[29] Schmitt, S. and Bergmann, R.: 1999, 'Applying case-based reasoning technology for product selection and customization in electronic commerce environments.' *12th Bled Electronic Commerce Conference*. Bled, Slovenia, June 7-9, 1999.

[30] Shardanand, U. and Maes, P.: 1995, 'Social Information Filtering: Algorithms for Automating "Word of Mouth"'. In: *CHI '95: Conference Proceedings on Human Factors in Computing Systems*, Denver, CO, pp. 210-217.

[31] Shimazu, H.: 2001, 'ExpertClerk: Navigating Shoppers' Buying Process with the Combination of Asking and Proposing'. In B. Nebel, (ed.) *Proceedings of the Seventeenth International Joint Conference on Artification Intelligence*, pp. 1443-1448.