Hybrid Web Recommender Systems

Conference Paper · January 2007

DOI: 10.1007/978-3-540-72079-9_12

CITATIONS
467

READS
3,775

2 authors, including:

Robin Burke
University of Colorado Boulder
212 PUBLICATIONS **9,754** CITATIONS

SEE PROFILE

12

Hybrid Web Recommender Systems

Robin Burke

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

Abstract. Adaptive web sites may offer automated recommendations generated through any number of well-studied techniques including collaborative, content-based and knowledge-based recommendation. Each of these techniques has its own strengths and weaknesses. In search of better performance, researchers have combined recommendation techniques to build hybrid recommender systems. This chapter surveys the space of two-part hybrid recommender systems, comparing four different recommendation techniques and seven different hybridization strategies. Implementations of 41 hybrids including some novel combinations are examined and compared. The study finds that cascade and augmented hybrids work well, especially when combining two components of differing strengths.

12.1 Introduction

Recommender systems are personalized information agents that provide recommendations: suggestions for items likely to be of use to a user [18, 41, 42]. In an e-commerce context, these might be items to purchase; in a digital library context, they might be texts or other media relevant to the user's interests.[1] A recommender system can be distinguished from an information retrieval system by the semantics of its user interaction. A result from a recommender system is understood as a recommendation, an option worthy of consideration; a result from an information retrieval system is interpreted as a match to the user's query. Recommender systems are also distinguished in terms of personalization and agency. A recommender system customizes its responses to a particular user. Rather than simply responding to queries, a recommender system is intended to serve as an information agent.[2]

[1] In this chapter, I use the e-commerce term "products" to refer to the items being recommended, with the understanding that other information-seeking contexts are also pertinent.

[2] Techniques such as relevance feedback enable an information retrieval engine to refine its representation of the user's query, and therefore can be seen as a simple form of recommendation. The search engine Google (http://www.google.com) blurs this distinction further, using "authoritativeness" criteria in addition to strict matching [6].

P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.): The Adaptive Web, LNCS 4321, pp. 377 – 408, 2007.

A variety of techniques have been proposed as the basis for recommender systems: collaborative, content-based, knowledge-based, and demographic techniques are surveyed below. Each of these techniques has known shortcomings, such as the well-known cold-start problem for collaborative and content-based systems (what to do with new users with few ratings) and the knowledge engineering bottleneck in knowledge-based approaches. A hybrid recommender system is one that combines multiple techniques together to achieve some synergy between them. For example, a collaborative system and a knowledge-based system might be combined so that the knowledge-based component can compensate for the cold-start problem, providing recommendations to new users whose profiles are too small to give the collaborative technique any traction, and the collaborative component can work its statistical magic by finding peer users who share unexpected niches in the preference space that no knowledge engineer could have predicted. This chapter examines the landscape of possible recommender system hybrids, investigating a range of possible hybridization methods, and demonstrating quantitative results by which they can be compared.

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 knowledge is the knowledge of other users' preferences. In others, it is ontological or inferential knowledge about the domain, added by a human knowledge engineer.

Previous work [10] distinguished four different classes of recommendation techniques based on knowledge source[3], as shown in Figure 12.1:

- Collaborative: The system generates recommendations using only information about rating profiles for different users. Collaborative systems locate peer users with a rating history similar to the current user and generate recommendations using this neighborhood. Examples include [17, 21, 41, 46].
- Content-based: The system generates recommendations from two sources: the features associated with products and the ratings that a user has given them. Content-based recommenders treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on product features [14, 22, 25, 38].
- Demographic: A demographic recommender provides recommendations based on a demographic profile of the user. Recommended products can be produced for different demographic niches, by combining the ratings of users in those niches [24, 36]
- Knowledge-based: A knowledge-based recommender suggests products based on inferences about a user’s needs and preferences. This knowledge will sometimes contain explicit functional knowledge about how certain product features meet user needs. [8, 9, 44].

Each of these recommendation techniques has been the subject of active exploration since the mid-1990's, when the first recommender systems were pioneered, and their capabilities and limitations are fairly well known.

[3] It should be noted that there is another knowledge source: context, which has not yet become widely used in web-based recommendation, but promises to become important particularly for mobile applications. See, for example, [7].

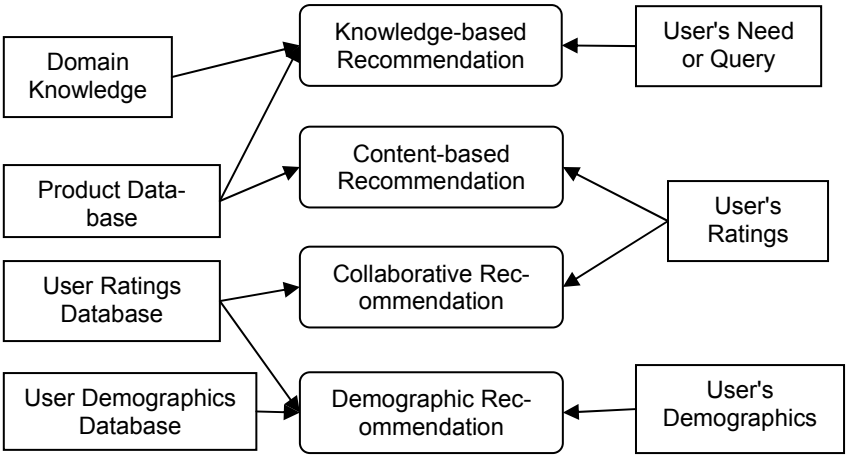

Fig. 12.1. Recommendation techniques and their knowledge sources

All of the learning-based techniques (collaborative, content-based and demographic) suffer from the *cold-start* problem in one form or another. This is the well-known problem of handling new items or new users. In a collaborative system, for example, new items cannot be recommended to any user until they have been rated by some one.. Recommendations for items that are new to the catalog are therefore considerably weaker than more widely rated products, and there is a similar failing for users who are new to the system.

The converse of this problem is the stability vs. plasticity problem. Once a user's profile has been established in the system, it is difficult to change one's preferences. A steak-eater who becomes a vegetarian will continue to get steakhouse recommendations from a content-based or collaborative recommender for some time, until newer ratings have the chance to tip the scales. Many adaptive systems include some sort of temporal discount to cause older ratings to have less influence [4, 45], but they do so at the risk of losing information about interests that are long-term but sporadically exercised. For example, a user might like to read about major earthquakes when they happen, but such occurrences are sufficiently rare that the ratings associated with last year's earthquake might no longer be considered by the time the next big one hits. Knowledge-based recommenders respond to the user's immediate need and do not need any kind of retraining when preferences change.

Researchers have found that collaborative and demographic 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. However, the cold-start problem has the side-effect of excluding casual users from receiving the full benefits of collaborative and content-based recommendation. It is possible to do simple market-basket recommendation with minimal user input: Amazon.com's "people who bought X also bought Y" but this mechanism has few of the advantages commonly associated with the collaborative filtering concept. The learning-based tech-

nologies 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.

Hybrid recommender systems are those that combine two or more of the techniques described above to improve recommendation performance, usually to deal with the cold-start problem.[4] This chapter will examine seven different hybridization techniques in detail and evaluate their performance. From a large body of successful research in the area, we know that hybrid recommenders can be quite successful. The question of interest is to understand what types of hybrids are likely to be successful in general or failing such a general result, to determine under what domain and data characteristics we might expect different hybrids to work well. While this chapter does by necessity fall short of providing a definitive answer to such questions, the experiments described below do point the way towards answering this important question for recommender system design.

12.2 Strategies for Hybrid Recommendation

The term *hybrid recommender system* is used here to describe any recommender system that combines multiple recommendation techniques together to produce its output. There is no reason why several different techniques of the same type could not be hybridized, for example, two different content-based recommenders could work together, and a number of projects have investigated this type of hybrid: NewsDude, which uses both naive Bayes and kNN classifiers in its news recommendations is just one example [4]. However, we are particularly focused on recommenders that combine information across different sources, since these are the most commonly implemented ones and those that hold the most promise for resolving the cold-start problem.

The earlier survey of hybrids [10] identified seven different types:

- Weighted: The score of different recommendation components are combined numerically.
- Switching: The system chooses among recommendation components and applies the selected one.
- Mixed: Recommendations from different recommenders are presented together.
- Feature Combination: Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
- Feature Augmentation: One recommendation technique is used to compute a feature or set of features, which is then part of the input to the next technique.
- Cascade: Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.
- Meta-level: One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

[4] Some hybrids combine different implementations of the same class of technique – for example, switching between two different content-based recommenders. The present study only examines hybrids that combine different types of recommenders.

Table 12.1. The space of possible hybrid recommender systems (adapted from [10])

	Weight.	Mixed	Switch.	FC	Cascade	FA	Meta
CF/CN							
CF/DM							
CF/KB							
CN/CF							
CN/DM							
CN/KB							
DM/CF							
DM/CN							
DM/KB							
KB/CF							
KB/CN							
KB/DM							

FC = Feature Combination, FA = Feature Augmentation
CF = collaborative, CN = content-based, DM = demographic, KB = knowledge-based

	Redundant
	Not possible
	Existing implementation

The previous study showed that the combination of the five recommendation approaches and the seven hybridization techniques yields 53 possible two-part hybrids, as shown in Table 12.1. This number is greater than 5x7=35 because some of the techniques are order-sensitive. For example, a content-based/collaborative feature augmentation hybrid is different from one that applies the collaborative part first and uses its features in a content-based recommender. The complexity of the taxonomy is increased by the fact that some hybrids are not logically distinguishable from others and other combinations are infeasible. See [10] for details.

The remainder of this section will consider each of the hybrid types in detail before we turn our attention to the question of comparative evaluation.

12.2.1 Weighted

> *The movie recommender system in [32] has two components: one, using collaborative techniques, identifies similarities between rating profiles and makes predictions based on this information. The second component uses simple semantic knowledge about the features of movies, compressed dimensionally via latent semantic analysis, and recommends movies that are semantically similar to those the user likes. The output of the two components is combined using a linear weighting scheme.*

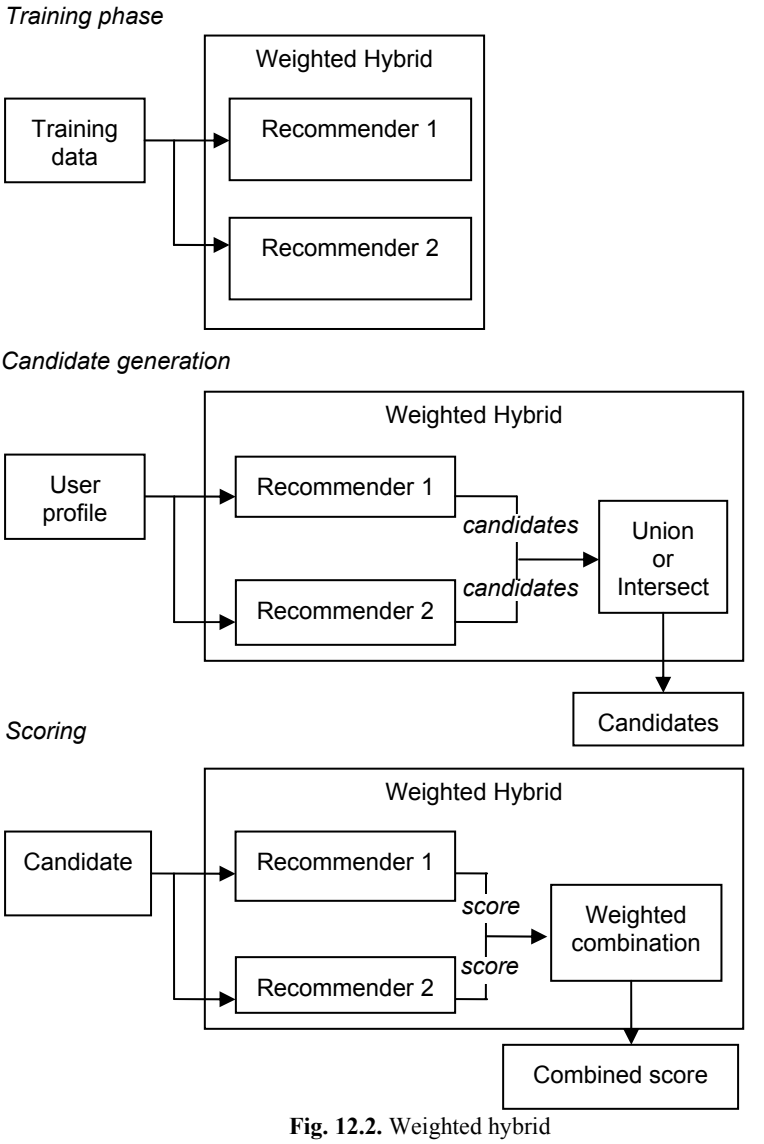

Fig. 12.2. Weighted hybrid

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. Examples of weighted hybrid recommenders include [15] as well as the example above. This type of hybrid combines evidence from both recommenders in a static manner, and would therefore seem to be appropriate when the component recommenders have consistent relative power or accuracy across the product space..

We can think of a weighted algorithm as operating in the manner shown in Figure 12.2. There is a training phase in which each individual recommender processes

the training data. (This phase is the same in most hybrid scenarios and will be omitted in subsequent diagrams.) Then when a prediction is being generated for a test user, the recommenders jointly propose candidates. Some recommendation techniques, such as content-based classification algorithms, are able to make predictions on any item, but others are limited. For example, a collaborative recommender cannot make predictions about the ratings of a product if there are no peer users who have rated it. Candidate generation is necessary to identify those items that will be considered.

The sets of candidates must then be rated jointly. Hybrids differ in how candidate sets are handled. Typically, either the intersection or the union of the sets is used. If an intersection is performed, there is the possibility that only a small number of candidates will be shared between the candidate sets. When union is performed, the system must decide how to handle cases in which it is not possible for a recommender to rate a given candidate. One possibility is to give such a candidate a neutral (neither liked nor disliked) score. Each candidate is then rated by the two recommendation components and a linear combination of the two scores computed, which becomes the item's predicted rating. Candidates are then sorted by the combined score and the top items shown to the user.

Usually empirical means are used to determine the best weights for each component. For example, Mobasher and his colleagues found that weighting 60/40 semantic/collaborative produced the greatest accuracy in their system [32]. Note that there is an implicit assumption that each recommendation component will have uniform performance across the product and user space. Each component makes a fixed contribution to the score, but it is possible that recommenders will have different strengths in different parts of the product space. This suggests the application of the next type of hybrid, one in which the hybrid switches between its components depending on the context.

12.2.2 Mixed

> *PTV recommends television shows [48]. It has both content-based and collaborative components, but because of the sparsity of the ratings and the content space, it is difficult to get both recommenders to produce a rating for any given show. Instead the components each produce their own set of recommendations that are combined before being shown to the user.*

A mixed hybrid presents recommendations of its different components side-by-side in a combined list. There is no attempt to combine evidence between recommenders. The challenge in this type of recommender is one of presentation: if lists are to be combined, how are rankings to be integrated? Typical techniques include merging based on predicted rating or on recommender confidence. Figure 12.3 shows the mixed hybrid design.

It is difficult to evaluate a mixed recommender using retrospective data. With other types of hybrids, we can use user's actual ratings to determine if the right items are being ranked highly. With a mixed strategy, especially one that presents results side-by-side, it is difficult to say how the hybrid improves over its constituent components

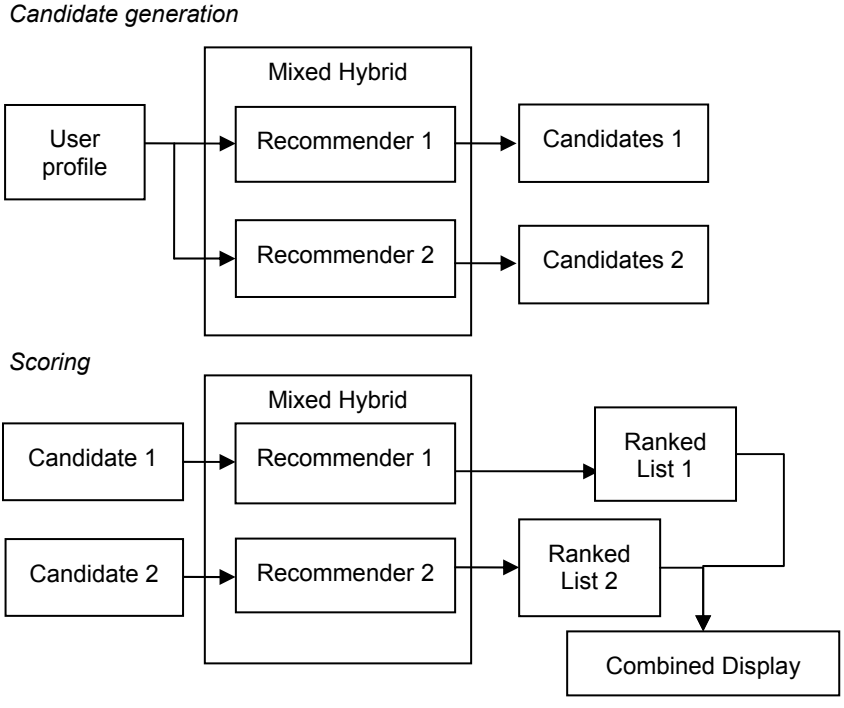

Fig. 12.3. Mixed hybrid (Training phase omitted, same as Weighted Hybrid)

without doing an on-line user study, as was performed for PTV. The mixed hybrid is therefore omitted from the experiments described below, which use exclusively retrospective data.

12.2.3 Switching

> *NewsDude [4] recommends news stories. It has three recommendation components: a content-based nearest-neighbor recommender, a collaborative recommender and a second content-based algorithm using a naive Bayes classifier. The recommenders are ordered. The nearest neighbor technique is used first. If it cannot produce a recommendation with high confidence, then the collaborative recommender is tried, and so on, with the naive Bayes recommender at the end of line.*

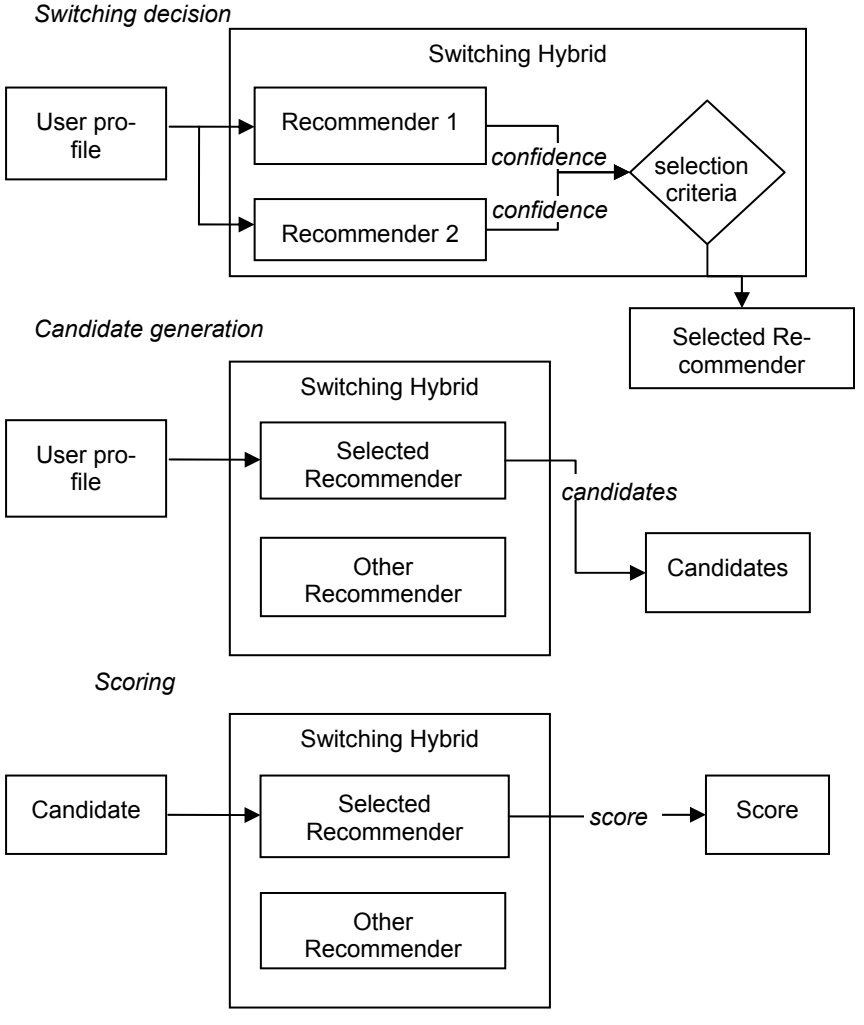

Fig. 12.4. Switching hybrid

A switching hybrid is one that selects a single recommender from among its constituents based on the recommendation situation. For a different profile, a different recommender might be chosen. This approach takes into account the problem that components may not have consistent performance for all types of users. However, it assumes that some reliable criterion is available on which to base the switching decision. The choice of this switching criterion is important. Some researchers have used confidence values inherent in the recommendation components themselves as was the case with NewsDude and van Setten's Duine system [49]; others have used external criteria [33]. The question of how to determine an appropriate confidence value for a recommendation is an area of active research. See [13] for recent work on the assessing the confidence of a case-based recommender system.

As shown in Figure 12.4 the switching hybrid begins the recommendation process by selecting one of its components as appropriate in the current situation, based on its switching criteria. Once that choice is made, the component that is not chosen has no role in the remaining recommendation process.

A switching recommender requires a reliable switching criteria, either a measure of the algorithm's individual confidence levels (that can be compared) or some alternative measure and the criterion must be well-tuned to the strengths of the individual components.

12.2.4 Feature Combination

> *Basu, Hirsh and Cohen [3] used the inductive rule learner Ripper [16] to learn content-based rules about user's likes and dislikes. They were able to improve the system's performance by adding collaborative features, thereby treating a fact like "User1 and User2 liked Movie X" in the same way that the algorithm treated features like "Actor1 and Actor2 starred in Movie X".*

The idea of feature combination is to inject features of one source (such as collaborative recommendation) into an algorithm designed to process data with a different source (such a content-based recommendation). This idea is shown schematically in Figure 12.5. Here we see that in addition to a component that actually makes the recommendation, there is also a virtual "contributing recommender". The features which would ordinarily be processed by this recommender are instead used as part of the input to the actual recommender. 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 [3, 34].

The feature combination hybrid is not a hybrid in the sense that we have seen before, that of combining components, because there is only one recommendation component. What makes it a hybrid is the knowledge sources involved: a feature combination hybrid borrows the recommendation logic from another technique rather employing a separate component that implements it. In the example above from Basu, Hirsh and Cohen, the content-based recommender works in the typical way by building a learned model for each user, but user rating data is combined with the product features. The system has only one recommendation component and it works in a content-based way, but the content draws from a knowledge source associated with collaborative recommendation.

12.2.5 Feature Augmentation

> *Melville, Mooney and Nagarajan [30] coin the term "content-boosted collaborative filtering." This algorithm learns a content-based model over the training data and then uses this model to generate ratings for unrated items. This makes for a set of profiles that is denser and more useful to the collaborative stage of recommendation that does the actual recommending.*

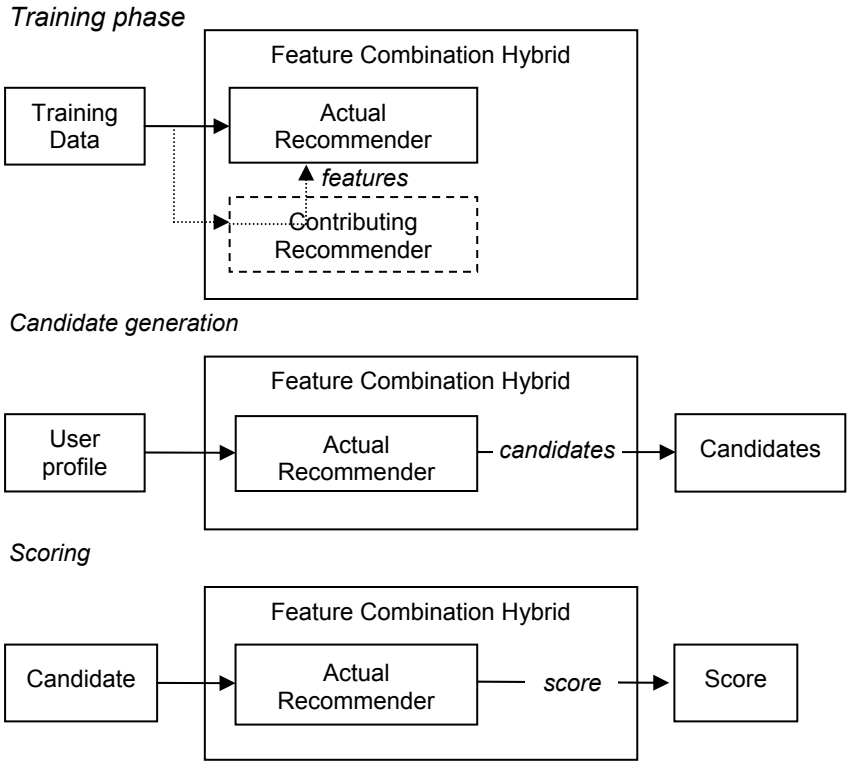

Fig. 12.5. Feature combination hybrid

Feature augmentation is a strategy for hybrid recommendation that is similar in some ways to feature combination. Instead of using features drawn from the contributing recommender's domain, a feature augmentation hybrid generates a new feature for each item by using the recommendation logic of the contributing domain. In case-based recommendation, for example, Smyth and his colleagues [35, 50] use association rule mining over the collaborative data to derive new content features for content-based recommendation.

This difference can be seen in the schematic diagram (Figure 12.6). At each step, the contributing recommender intercepts the data headed for the actual recommender and augments it with its own contribution, not raw features as in the case of feature combination, but the result of some computation. A feature augmentation recommender would be employed when there is a well-developed strong primary recommendation component, and a desire to add additional knowledge sources. As a practical matter, the augmentation can usually be done off-line, making this approach attractive, as in the case of feature combination, when trying to strengthen an existing recommendation algorithm by adjusting its input.

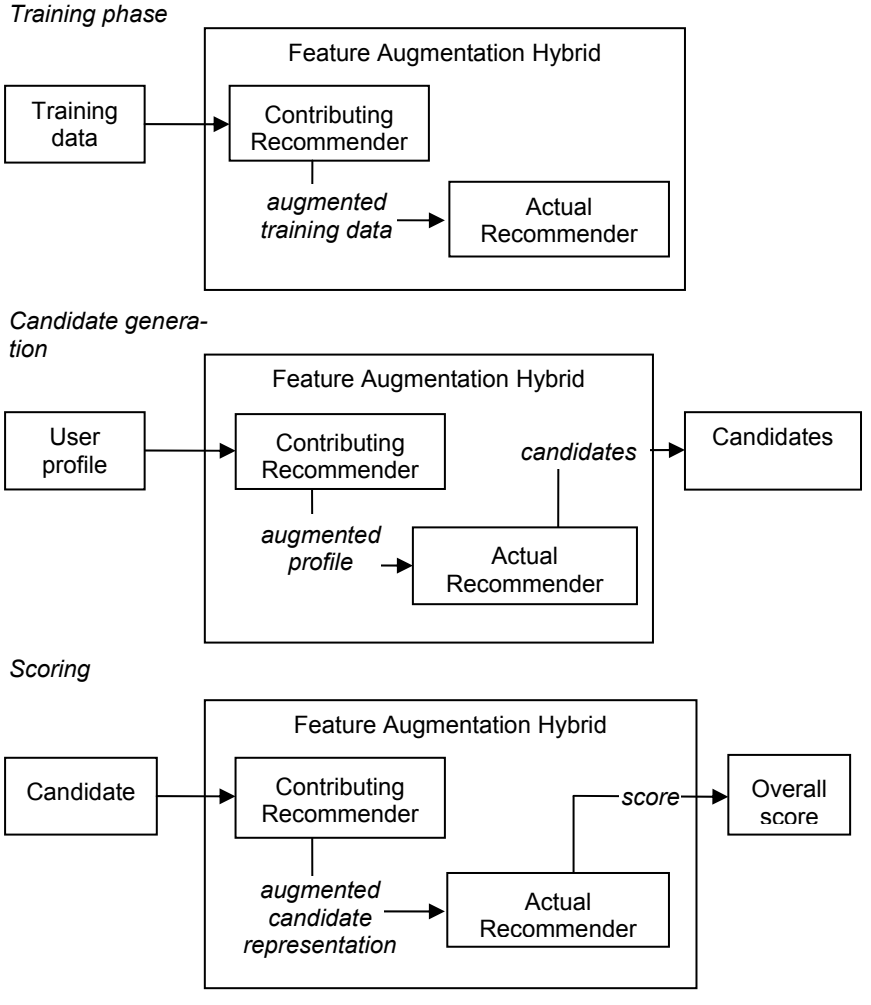

Fig. 12.6. Feature augmentation hybrid

There are a number of reasons why a feature augmentation hybrid might be preferred to a feature combination one. It is not always easy or even possible to create a feature combination hybrid for all possible hybrid combinations: the feature augmentation approach is more flexible. Also, the primary recommender in a feature combination hybrid must confront the added dimensionality of the larger training data, particularly in the case of collaborative ratings data. An augmentation hybrid adds a smaller number of features to the primary recommender's input.

Still, it is not always immediately obvious how to create a feature augmentation recommender for any two recommendation components. A recommendation component, after all, is intended to produce a predicted score or a ranking of items, not a feature for consumption by another process. What is required, as in feature com-

bination, is attention to the knowledge sources and recommendation logic. In the example above, a content-based recommender uses the features of the items in a profile to induce a classifier that fits a particular user. The classifier can then be used to rate additional items on the user's behalf, making for a denser and more fleshed-out set of ratings, which then become input for a collaborative algorithm – we can more precisely describe this as a content-based / collaborative feature augmentation hybrid.

12.2.6 Cascade

> *The knowledge-based Entree restaurant recommender [10] was found to return too many equally-scored items, which could not be ranked relative to each other. Rather than additional labor-intensive knowledge engineering (to produce finer discriminations), the hybrid EntreeC was created by adding a collaborative re-ranking of only those items with equal scores.*

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. In its order-dependence, it is similar to the feature augmentation hybrid, but it is an approach that retains the function of the recommendation component as providing predicted ratings. A cascade recommender uses a secondary recommender only to break ties in the scoring of the primary one. Figure 12.7 shows a schematic depiction of this style of hybrid.

Many recommendation techniques have real-valued outputs and so the probability of actual numeric ties 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 was completed in 2002. In the case of EntreeC, the knowledge-based / collaborative cascade hybrid described above, the knowledge-based component was already producing an integer-valued score, and ties were observed in every retrieval set, so the cascade design was a natural one.

The cascade hybrid raises the question of the uncertainty that should be associated with the real-valued outputs of a recommendation algorithm. It is certainly not the case that our confidence in the algorithms should extend to the full 32 bit precision of double floating point values. And, if the scoring of our algorithms is somewhat less precise, then there may be ties in ranks to which the cascade design can be applied. As we shall see below, recommenders operating at reduced numeric precision do not suffer greatly in accuracy and so the cascade hybrid is a reasonable option. McSherry [29] uses a similar idea in creating regions of similarity in which scores vary no more than a given ε to satisfy the goal of increasing recommendation diversity.

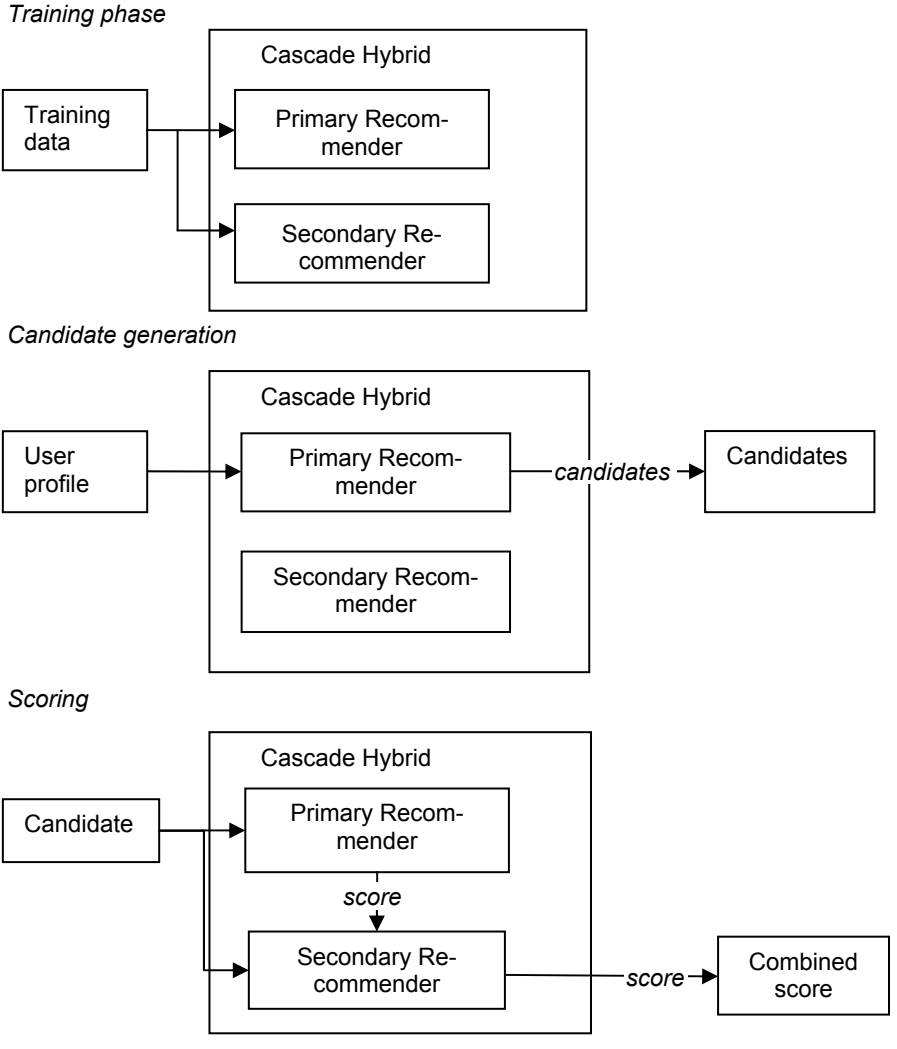

Fig. 12.7. Cascade recommender

12.2.7 Meta-level

Pazzani [36] used the term "collaboration through content" to refer to his restaurant recommender that used the naive Bayes technique to build models of user preferences in a content-based way. With each user so represented, a collaborative step was then be performed in which the vectors were compared and peer users identified.

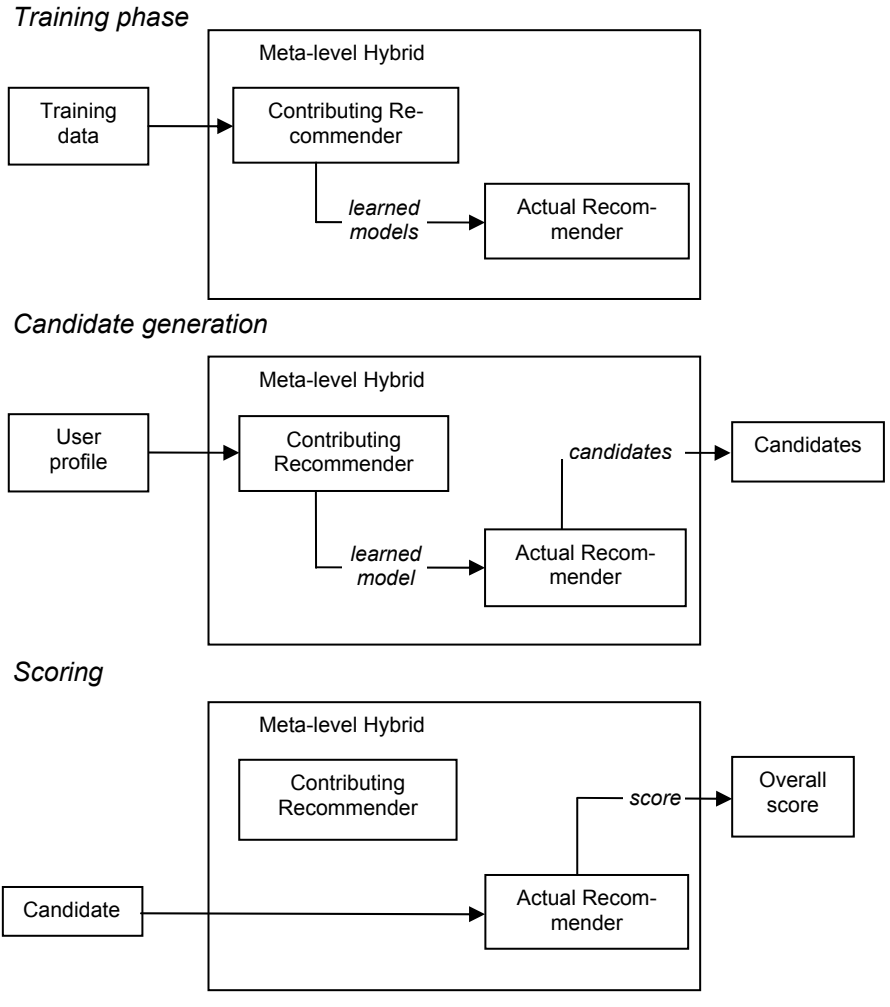

Fig. 12.8. Meta-level hybrid

A meta-level hybrid is one that uses a model learned by one recommender as input for another. Another classic example is Fab [1], a document recommender that used the same "collaboration through content" structure. Figure 12.8 shows the general schematic for this type of recommender. Note that this type is similar to the feature augmentation hybrid in that the contributing recommender is providing input to the actual recommender, but the difference is that in a meta-level hybrid, the contributing recommender completely replaces the original knowledge source with a learned model that the actual recommender uses in its computation. The actual recommender does not work with any raw profile data. We can think of this as a kind of "change of basis" in the recommendation space.

It is not always straightforward (or necessarily feasible) to derive a meta-level hybrid from any given pair of recommenders. The contributing recommender has to produce some kind of model that can be used as input by the actual recommender and not all recommendation logics can do so.

12.3 Comparing Hybrids

There have been a few studies that compared different hybrids using the same data. Pazzani's study is notable for comparing both a meta-level and a weighted scheme for hybrid recommenders using content, collaborative and demographic data. He found a significant improvement in precision for both hybrid techniques. Good and colleagues [18] examined an assortment of hybrids involving collaborative, content-based and very simple knowledge-based techniques in the movie recommendation domain. The study did find that a hybridized recommender system was better than any single algorithm and that multi-part hybrids could be successful.

To compare the full scope of the hybrid design space from Table 12.1 would require recommendation components of each of the four types: collaborative, content-based, knowledge-based and demographic. Given appropriate rating and product data, collaborative and content-based components can easily be constructed and most studies of hybrid recommendation have looked at just these components. Constructing a demographic recommendation component is more difficult as it requires access to users' personal demographic data, which is not found in the commonly-used ratings data sets used for evaluating recommender systems, such as MovieLens[5]. Constructing a knowledge-based recommendation component is a matter of knowledge engineering, and while there are a number of extant examples, there is only one that is associated with publicly-available user profile data, namely the Entree restaurant recommender system [8, 9]. [6]

The benefit of using the Entree data is that it allows us to examine some of the particularly under-explored portions of the hybrid design space – those with knowledge-based components. The tradeoff is that this data set has some peculiarities (discussed in detail below), which may limit the applicability of the results. However, the experiments do allow us to examine some of the interactions between recommendation approaches and hybridization techniques, and hopefully to provide some guidance to researchers and implementers seeking to build hybrid systems.

12.3.1 The Entree Restaurant Recommender

To understand the evaluation methodology employed in this study and the operation of the knowledge-based recommendation component, we will need to examine the characteristics of the Entree restaurant recommender and the Entree data set. Entree is a restaurant recommendation system that uses case-based reasoning [23] techniques to select and rank restaurants. It operated as a web utility for approximately three years starting in 1996. The system is interactive, using a critiquing dialog [11, 47] in which

[5] The MovieLens data sets are at http://www.cs.umn.edu/research/GroupLens/index.html.
[6] The Entree data set is available from the UC Irvine KDD archive at http://kdd.ics.uci.edu/databases/entree/entree.html

Fig. 12.9. Results of a query to the Entree restaurant recommender

users' preferences are elicited through their reactions to examples that they are shown. Recent user studies [39] have shown this technique to be an effective one for product catalog navigation, and the refinement of this model is an area of active research. See, for example, [28, 40].

Consider a user who starts browsing by entering a query in the form of a known restaurant, Wolfgang Puck's "Chinois on Main" in Los Angeles. As shown in Figure 12.9, the system finds a similar Chicago restaurant that combines Asian and French influences, "Yoshi's Cafe," as well as other similar restaurants that are ranked by their similarity. Note that the connection between "Pacific New Wave" cuisine and its Asian and French culinary components is part of the system's knowledge base of cuisines. The user might however be interested in a cheaper meal, selecting the "Less $$" button. The result would be a creative Asian restaurant in a cheaper price bracket, if one could be found. Note that the critiques are not "narrowing" the search in the sense of adding constraints, but rather changing the focus to a different point in the feature space. The user can continue browsing and critiquing until an acceptable restaurant has been located.

12.3.2 The Entree Data Set

Each user session in the Entree data set therefore consists of an entry point, which may be a restaurant or a query, a series of critiques, and finally an end point. For example, the session that began in Figure 12.9 might consist of three actions

(<"Chinois on Main", entry>, <"Yoshi's", too expensive>, <"Lulu's", end>). To turn this action sequence into a rating profile, we make the simplifying assumption that the entry and ending points are "positive" ratings and the critiques are "negative" ones. (Earlier research showed that a more nuanced interpretation of the critiques was not helpful [10].) If we look at a session consisting of ten interactions, we would have eight or nine negative ratings and one or two positive ratings. This is quite different than the typical recommender system that has a more even mix of ratings and usually more positive than negative ratings [45]. The Entree data set is also much smaller than some other data sets used for collaborative filtering research, containing about 50,000 sessions/users and a total of just under 280,000 ratings. The small number of ratings per user (average 5.6) means that collaborative and especially content-based algorithms cannot achieve the same level of performance as is possible when there is more training data.

Another way to look at the data set however is that it foregrounds the most vexing problems for recommender systems, the twin "cold start" problems of new users (short profiles) and new items (sparse ratings). Since the major motivation for using recommendation hybrids is to improve performance in these cold start cases, the Entree data set is a good trial for the effectiveness of hybrids in precisely these conditions. It is also the case that users are often reluctant to allow lengthy personal profiles to be maintained by e-commerce sites, so good performance with single session profiles is important.

The assumption that the end point is a positive rating is a rather strong assumption. Effectively, this assumption amounts to the proposition that most users are satisfied with the recommendations that they receive. It is of course possible that users are abandoning their searches in frustration. To examine the validity of this assumption, we experimented with a subset of the data that contains entry point ratings. Entry points can be confidently labeled as implicit positive ratings – users would not ask for restaurants similar to those they did not like. Experiments found extremely strong correlation (0.92) between the two conditions, demonstrating that the behavior of the algorithms does not differ markedly when exit points are treated as positive ratings. Therefore, in the experiments below, we will use the full data set and assume that both entry and exit points are positive ratings, with the understanding that there is some noise associated with this assumption.

12.3.3 Evaluation

[20] is a recent survey that compares a variety of evaluation techniques for collaborative filtering systems, and although this article looks at a larger class of recommendation systems, these results are still informative. Herlocker and colleagues identify three basic classes of evaluation measures: discriminability measures, precision measures and holistic measures. In each group, many different metrics were found to be highly correlated, effectively measuring the same property. For restaurant recommendation, 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 if many ratings are held out for testing, there would not be enough of a profile left on which a recommender could base its prediction. 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. We would like to measure how well each system is able to give a good item as a recommendation. So, the method used here is to record the rank of a positively-rated test item in a recommendation set. 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 an item known to be liked by the user from the others.[7]

To calculate this value for each recommender system design, the set of sessions is 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 is given the training part of the data as its input and handles it in its own way. Evaluation is performed on each session of the test data. From the session, a single item with a positive rating is chosen to be held back.[8] This item will be the test item on which the recommender's performance will be evaluated. All of the other ratings are considered part of the test profile.

The recommendation algorithm is then given the test profile without the positively-rated item, and must make its recommendations. The result of the recommendation process is a ranked subset of the product database containing those items possibly of interest to the user. From this set, we record the rank of the positively-rated test item. Ideally, that rank would be a low as possible – the closer to the front the preferred item is placed, the more precisely the recommender is reflecting the user's preferences.

12.3.4 Sessions and Profiles

The Entree data contains approximately 50,000 sessions of widely differing lengths. Some sessions consist of only an entry and exit point, others contain dozens of critiques. To examine differences in recommender performance due to profile size, we fix the session size for each evaluation test set, discarding sessions shorter than this size and randomly discarding negative ratings from longer sessions.

Longer profiles are available if we examine user behavior over multiple visits. There are approximately 20,000 multi-session profiles. These longer multiple-visit profiles are somewhat less reliable as user profiles because they are collated using IP address alone [31]. So, we understand that they will be noisier than the ones derived from single visits.

The evaluation examined six different session sizes: three from single visits and three from multi-visit profiles. We used 5, 10 and 15 rating sessions from single vis-

[7] The significance of ARC results is computed with paired ANOVA analysis using the Bonferroni t test for rank with $\alpha = 0.01$. The significance calculations were performed in SAS 8.0 using the Generalized Linear Model procedure. (http://www.sas.com/)

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

its; and 10, 20 and 30 rating sessions from multi-visit profiles. In the figures below, the single-visit profiles will be marked with a capital "S" and the multi-visit profiles with a capital "M". In the case of 5-rating sessions, we used a 50% sample of the data for testing due to the large number of profiles of this size.

12.3.5 Baseline Algorithms

Four basic algorithms were used in the study.

Collaborative Pearson – CFP. This algorithm recommends restaurants based on a collaborative filtering algorithm using Pearson's correlation coefficient to compute the similarity between users [18]. A threshold is used to select similar users and the top 50 are retained as the user's peer group. The restaurants rated by this peer group and not rated by the user are considered the candidate set. These candidates are scored using the average rating from the peer group.[9]

Collaborative Heuristic – CFH. This recommender uses a collaborative variant that computes the similarity between users, taking into account the semantics of the Entree ratings. This algorithm is described more fully in [9]. Rather than treating all of the critiques in each user session as negative ratings (as is done in the CFP algorithm), the heuristic algorithm has a distance matrix for comparing critiques directly. For example, a "nicer" critique and a "cheaper" critique are considered dissimilar, while a "nicer" and "quieter" critique are considered similar. Earlier experiments suggested that this variant was more effective than methods that treat the ratings as binary-valued.

Content-Based – CN. This technique uses the naive Bayes algorithm to compute the probability that a restaurant will be liked by the user. The training data is used to compute prior probabilities and the test session data is used to build a user-specific profile. In most recommender systems, the profile is then used to classify products into liked and disliked categories and the liked category becomes the candidate set, with the classification score becoming the rating. Because of the skewed distribution of ratings, however, this approach was not found to be effective – too few restaurants are rated as "liked". In these experiments, I instituted a candidate generation phase that retrieves all those restaurants with some features in common with the "liked" vector of the naive Bayes profile. Some of these restaurants would not be rated as "liked", but restaurants that do not have at least one such feature cannot be assigned to the "liked" category. The ranking of candidates is then determined by the prediction of the "liked" classifier.

Knowledge-Based (KB). The knowledge-based recommender recommends restaurants using Entree's knowledge-based retrieval. Entree has a set of metrics for knowledge-based comparison of restaurants. It knows, for example, that Thai and Vietnamese food are more similar to each other than Thai and German food would be. Other

[9] Weighting user's rating by the proximity to the test user as some authors suggest [5] was not found to be effective.

Table 12.2. Average rank of correct recommendation (ARC) for basic recommendation algorithms at each session size.

	5S	10S	15S	10M	20M	30M
CFH	80	83	124	229	231	230
CFP	113	99	158	183	213	240
KB	207	220	154	298	305	311
CN	296	276	273	313	310	336
Ave	294	304	307	316	317	317

knowledge enables it to reason about price, atmosphere and other characteristics of restaurants. In order to evaluate this component from historical user sessions, the system reissues the last query or critique present in the session and returns the candidate set and its scores. Because longer sessions are truncated, the query will rarely correspond to the one immediately prior to the exit point (which may or may not be the test item) but it will be the available rating chronologically closest to the exit point.

12.3.6 Baseline Evaluation

A starting point for analysis of the hybrids is the evaluation of the four basic algorithms, and for a baseline, we can also examine the performance of the "average" recommender, which recommends restaurants based on their average rating from all users, and does not take individual user profiles into account.

Table 12.2 shows the average rank of the correct recommendation (ARC) for each of the basic algorithms over the six different session size conditions. Figure 12.10 shows the same data in graphical form. (Brackets above the bars indicate places where differences between algorithm performance are not significant.) There are several points to make about these results. First, we should note that this recommendation task is, as expected, rather difficult. The best any of these basic algorithms can manage is average rank of 80 for the correct answer. The primary reason is the paucity of data. With only a small number of ratings to work from, collaborative algorithms cannot narrow their matching neighborhoods to precise niches, and the content-based algorithm has fewer patterns from which to learn. It is not surprising that the results are not exactly inspiring in an e-commerce context where the user might be expected only to look at the first dozen results or so. The top result for single-visit profiles is obtained by the heuristic collaborative algorithm. However, when we look at multiple visit profiles, the standard collaborative algorithm is preferred. In three of the six cases, however, the differences are not significant.

This data also demonstrates something of the task-focused nature of the Entree data, a characteristic that it shares with other consumer-focused recommendation domains. Users coming to the Entree system are planning for a particular dining occasion and their preferences undoubtedly reflect many factors in addition to their own particular tastes. (Since restaurant meals are often taken in groups, the task is effectively one of group recommendation [27].) These extra-individual factors may change radically from session to session and therefore add to the difficulty of extracting a

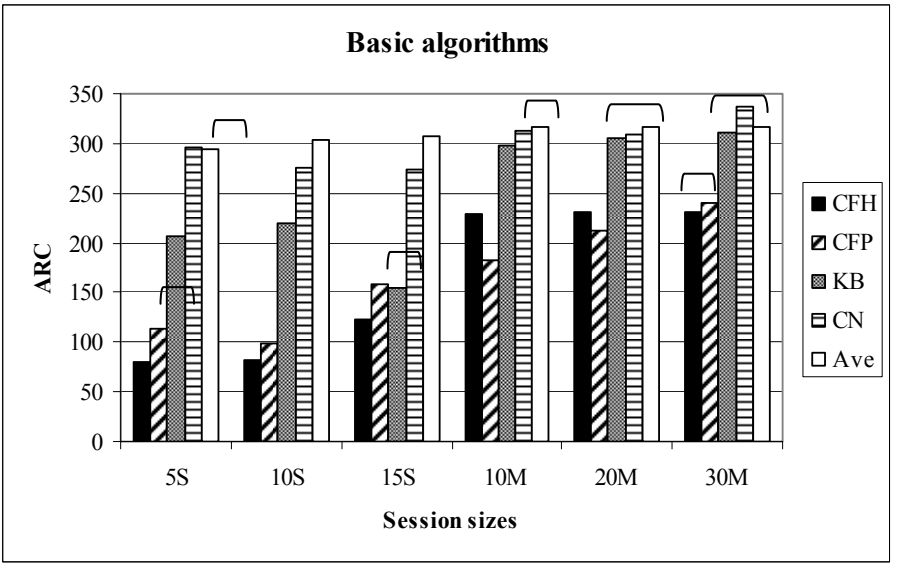

Fig. 12.10. Average rank of correct answer, basic algorithms. Non-significant differences between results are indicated with brackets

consistent multi-session profile. We can see this in the performance of the naive Bayes (CN) recommender across the different session sizes from 5S to 15S, where it improves steadily, but as we step up to the multi-session profile of size of 30, we see that the performance of this recommender actually goes down, not statistically better than the simple average. Also, the performance of the knowledge-based recommender is weaker in the multi-session profiles, due most likely to the same lack of user consistency across visits: the constraints that the recommender can use to find restaurants in one session may not be valid in a later one.

There are two conclusions to be drawn from the performance of the basic algorithms. One is that the techniques vary widely in their performance on the Entree data. The content-based technique is generally weak. The knowledge-based technique is much better on single-session profiles than on multi-session ones. The heuristic collaborative technique may have a relative advantage over the correlation-based one for short profiles but does not have it for multi-visit ones. The second point is that there is much room for improvement in the results shown by these algorithms acting alone. This is particularly the case for the multi-session profiles.

12.4 Results

This section describes the general findings of a comparative study, examining 41 different types of hybrid recommenders using the four basic components evaluated above. Full details are omitted for resasons of space, but can be found in [11]. There are no results for "Mixed" hybrids as it is not possible to evaluate retrospec-

tively its effectiveness from usage logs. There is no demographic data in the Entree data set, and so no demographic components were examined. The study did not examine hybrids combining recommenders of the same type, such as CFP/CFH, and some designs that are theoretically possible were not implemented due to constraints of the existing algorithms and knowledge-bases. In the interest of a level playing field, no enhancements were added that would specifically benefit only certain hybrid types.

Table 12.3 shows the two best results from each hybrid type. Grey cells indicate conditions in which no synergy was found: the hybrid was no better than one of its components taken individually. Figure 12.11 shows the ARC results for the top hybrid of each type.

12.4.1 Weak Performers

The weighted, switching, feature combination and meta-level hybrids were not particularly effective designs for this data set. They showed only scant and spotty improvement over the unhybridized algorithms, and in the case of meta-level designs, no synergy whatsoever.

Weighted. The weighted hybrid was created by setting the component weights empirically, determining which weighting yielded the best ARC value over the training data. The results were rather surprising, although [49] found a similar effect. In only 10 of the 30 conditions was the performance of the combined recommenders better than the best component working alone. The CN/CFP hybrid does show consistent synergy (5 of 6 conditions), as [36] also found. The most likely explanation is that the recommenders, especially KB and CFH, do not have uniform performance across the product and user space.

Switching. A switching hybrid is one that selects a single recommender from among its constituents and uses it exclusively for a given recommendation situation. For a different profile, a different recommender might be chosen. This approach takes into account the problem that components may not have consistent performance for all types of users. However, it assumes that some reliable criterion is available on which to base the switching decision: a confidence value. Each of the recommenders in the experimental set required a different confidence calculation. (See [11] for additional details.) However, none of the switching hybrids were particularly effective, perhaps due to the lack of reliable switching criteria: only the KB/CFP hybrid showed overall synergy.

Feature Combination. Feature combination requires that we alter the input of a recommendation component to use data from another knowledge source. Such an arrangement will, by necessity, be different for each pair of techniques being combined. 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 content-based

Table 12.3. Top two best results for each hybrid type. (Meta-level omitted.) Grey cells indicate conditions in which synergy was not achieved.

		5S	10S	15S	10M	20M	30M
Basic	**CFH**	80	83	124	229	231	230
	CFP	113	99	158	183	213	240
Weighted	**CN/CFP**	102		142	168	202	224
	CFP/KB		90	92			238
Switching	**CFH/KB**	65		65	205	203	211
	KB/CFP	65	93	79			239
Feature Comb.	**CN/CFP**	111	98	143	184	215	228
	CN/CFH	80	83	117	233	224	228
Feature Aug.	**CN/CFP**	23	23	24	31	33	40
	KB/CFP	18	19	20	30	31	37
Casacade	**CFP/CN**	20	16	23	29	31	38
	CFP/KB	19	16	22	30	32	38

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. The CF/CN feature combination hybrid showed modestimprovement over the hybridized CN algorithm but it still falls far short of the basic collaborative algorithms. [10]

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" [43] or "genreBots" (Good et al. 1999). 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.[11] The CN/CFH and CN/CFP results are nearly identical to the non-hybrid results as one might expect given that the pseudo-users add only about 1% more data to the training set. Even when the training data was downsampled, the contribution of the psuedo-users was minimal.

[10] The naive Bayes implementation performed poorly with this augmented model, including over 5,000 new collaborative features in addition to the 256 content ones. So, for this hybrid only, the Winnow algorithm was used [26], because of its ability to handle large numbers of features. Winnow was found to be inferior to naive Bayes for the other content-based recommendation tasks.

[11] Other possibilities for feature combination hybrids turn out to be either illogical or infeasible.

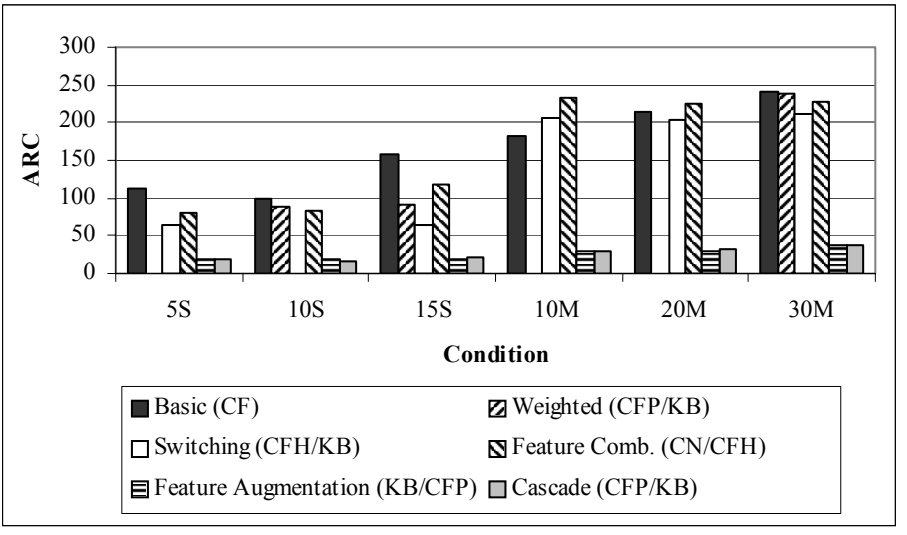

Fig. 12.11. Best results for each hybrid type.

Meta-level. The construction of a meta-level hybrid is highly dependent on the characteristics of the recommendation components being combined. It is not always feasible to derive a meta-level hybrid from any given pair of recommenders. Because none of the basic recommenders are particularly good on their own, we might expect low reliability in any learned model they might produce. This expectation is borne out by experiment: none of the six meta-level hybrids examined in the study achieved synergy in any condition, and these results are omitted from Table 12.3 and Figure 12.11. It is evident that to build a working meta-level hybrid, both recommendation components must be strong individual performers.

12.4.2 Cascade

Our recommendation components all produce real-valued outputs, making them unsuitable at first glance for the use of the cascade, which uses a secondary recommender to break ties. However, reduced-precision versions of these algorithms were implemented in which predicted ratings were limited to two decimal digits. The reduction in precision was found to have minimal impact on accuracy, and these versions of the algorithms were used to produce output that could contribute to a cascade.

The cascade hybrid was designed for a strong recommender to get a boost from a weaker one. It assumes that the primary recommender is the more reliable of its components. Therefore, it is not surprising that the collaborative-primary cascade hybrids work well. These implementations, especially the CFP/KB and CFP/CN hybrids, show great improvement over the other hybrids seen so far and over the basic recommenders.

12.4.3 Feature Augmentation

In feature augmentation, we seek to have the contributing recommender produce some feature or features that augment the knowledge source used by the primary recommender. To preserve the recommendation logic of our recommenders in this study required some ingenuity to create the eight hybrids studied. Several different methods are used as described in detail below. In each, the goal was to preserve the comparative nature of the study, to avoid adding new recommendation logic to the hybrid – where it was unavoidable, the simplest possible technique was employed, which in some cases was unsupervised clustering.

Content-Based Contributing / Collaborative Actual – CN/CF. A content-based recommender uses the features of the items in a profile to induce a classifier that fits a particular user. The features that such a classifier can produce are classifications of items into liked / disliked categories. This capability is used as follows:

1. The content-based algorithm is trained on the user profile.
1. The collaborative algorithm retrieves candidate restaurants from users with similar profiles.
2. These candidates are rated by the content-based classifier and those ratings are used to augment the profile thus filling it out with more ratings.
3. Then the collaborative recommendation process is performed again with a new augmented profile. This is Melville's "content-boosted collaborative filtering" [30].

Collaborative Contributing – CF/CN and CF/KB. A collaborative recommender deals with similarities between users. The other recommenders are interested in comparing restaurants, so the problem for a collaborative recommender contributing to a knowledge-based or content-based hybrid is how to turn user data into features associated with restaurants. One way to do this is to cluster restaurants into groups based on user preferences about them. The cluster to which a given restaurant belongs can be considered a new feature that augments the restaurant representation. To incorporate these new features into the knowledge-based recommendation, the recommender's domain knowledge was augmented with a simple metric that prefers restaurants that share the same cluster id.

Knowledge-Based Contributing – KB/CF and KB/CN. A knowledge-based recommender can be used like the content-based one to classify restaurants into liked / disliked categories by assuming that the restaurants retrieved by the recommender are in the "liked" category and all others are disliked. The algorithm given above for the CN/CF hybrid can then be employed.

This is not an adequate solution for a KB / CN feature augmentation hybrid where the knowledge-based recommender needs to augment the representation of restaurants rather than user profiles. In this case, however, we treat the knowledge-based system as a source of "profiles". For each restaurant, we retrieve a set of similar restaurants known to the system. Each such set is treated like a user profile, and this profile matrix can be transposed and clustered as in the collaborative case.

Results. The feature augmentation hybrids show the best performance seen so far, particularly where the content-oriented recommenders are contributing to the collaborative ones. The KB and CN recommenders did not make good primary components, as might be expected from the performance of the basic algorithms and none of these hybrids showed synergy. Both strong components were greatly enhanced by the addition of content-derived features. Performance is particularly good for the multi-session profiles for which none of the previous hybrids were adequate.

12.5 Discussion

Given this survey of 41 different hybrid recommenders, we can return to our initial purpose in this survey, to determine the best hybrids for the Entree data and to determine if any lessons can be learned that apply to the construction of hybrids in the more general case.

It is quite clear, as others have also shown, that there is an unqualified benefit to hybrid recommendation, particularly in the case of sparse data. This can be seen in Figure 12.11. Nowhere was this effect more striking than in the noisy multi-session profiles, which proved so much more difficult for even the stronger basic algorithms. Where the best results obtained on the 30-rating sessions by a basic algorithm was only an ARC of 227, the top hybrids all have ARC scores under 40. Note that this synergy is found under the twin difficulties of smaller profile size and sparse recommendation density, showing that hybridization does help conquer the cold start problem.

Of course, not all hybrid designs were successful, leading to a second question: What is the best hybrid type? This answer can be found by examining the relative performance over all the hybrids on the different conditions. If we rank the hybrids by their ARC performance and look at the top hybrids in each condition, feature augmentation and cascade recommenders dominate. None of the other hybrid types achieve a rank higher than 9th best for any condition, and the only non-FA or cascade hybrids that appears twice in the top ten are two switching recommenders: CFH/KB and KB/CFP. Table 12.4 shows the top ten hybrids ranked by their average ARC over all conditions. Beyond the top four (two feature-augmentation and two cascade), performance drops off markedly.

In retrospect, given the performance of the basic algorithms, the performance of the cascade recommenders is fairly predictable. The KB and CN algorithms are relatively weak, but do take into account different knowledge sources than the collaborative algorithms. A cascade design allows these recommenders to have a positive impact on the recommendation process with little risk of negative impact – since they are only fine-tuning the judgments made by stronger recommenders. What is particularly interesting is that this performance was achieved by explicitly sacrificing numeric precision in the scoring of the primary recommender. The other top performing hybrids were the feature augmentation hybrids. Again, we see that the feature augmentation design allows a contributing recommender to make a modest positive impact without the danger of interfering with the performance of the better algorithm.

Generalizing from these results is by necessity speculative, since all we have are results in a particular product domain with a somewhat sparse and unorthodox data set. These experiments show that standard recommenders with widely varying per-

Table 12.4. Top ten hybrids by ARC

Type	Recommenders used	Average ARC
FA	KB/CFP	25.8
Cascade	CN/CFP	26.2
Cascade	KB/CFP	26.3
FA	CN/CFP	29.0
FA	KB/CFH	79.9
Cascade	CN/CFH	91.1
Cascade	KB/CFH	92.2
FA	CN/CFH	95.1
Switching	CFH/KB	139.1
Switching	KB/CFP	155.6

formance can be combined to achieve strong synergies on a fairly difficult recommendation task with limited data. In particular, it is clear that even recommendation algorithms with weak knowledge sources can have a strong positive impact on performance if they are combined in an appropriate hybrid.

No hybrids were tested in which both components could be considered strong, and while it seems likely that the feature augmentation and cascade designs would work well in this best case strong-strong scenario, it is seems likely that other techniques such as the meta-level hybrid would also succeed. Clearly, other researchers have had success with meta-level designs [1, 36, 45].

We see significant differences between the hybridization techniques, particularly their sensitivity to the relative strength and consistency of each component part. Some hybrids can make the most of a weak-strong combination; others cannot. Some hybrids work under the assumption that their components have uniform performance across the recommendation space (weighted, augmentation, meta-level); others are effective even if this is not true. In choosing a hybrid recommendation approach, therefore it seems particularly important to examine the design goals for a hybridized system (overall accuracy, cold-start performance, etc.) and evaluate the relative performance of each component of the hybrid under those conditions. For example, consider an implementer interested in improving collaborative recommendation results for cold-start users by building a hybrid that adds a content-based technique. We know that new users would have small usage profiles and the content-based recommender would be weak in these cases. This situation would suggest a cascade or feature augmentation approach.

Another consideration in the choice of hybridization techniques for recommendation is efficiency, particularly run-time efficiency, since recommendations are typically made on the fly to users expecting a quick interactive response. Of the basic algorithms, the collaborative algorithms are the slowest since they must compare the user's profile against the database of other users. A number of approaches have been developed to improve the efficiency of collaborative algorithms, for example clustering and indexing [31] and these would be of interest in any hybrid scheme as well. Of the hybrid designs, the weighted approach is the least efficient since it requires that

both recommenders process every request; depending on the implementation, a meta-level hybrid may have the same drawback. Among the strong performers, the cascade hybrid also requires computation from both recommenders, but since the secondary recommender is only breaking ties, it is not required to retrieve any candidates and need only rate those items that need to be further discriminated. This can be done on demand as the user requests portions of the retrieval set. On the other hand, the other top performing hybrid, the feature augmentation hybrid, the contributing recommender operates by adding features to the underlying representation. This step can be performed entirely off-line. So, the feature augmentation hybrid offers accuracy on par with the cascade hybrid with virtually no additional on-line computation.

12.6 Conclusion

This chapter has more fully characterized each of 53 hybrid types shown in Table 12.1 and described experiments that compare the performance of a subset of the design space. The experiments cover the space of possible hybrid recommender systems available with four basic recommendation algorithms: content-based, standard collaborative, heuristic collaborative and knowledge-based. Six types of combinations were explored: weighted, switching, feature combination, feature augmentation, cascade and meta-level, for a total of 41 different systems. Due to data and methodological limitations, demographic recommendation and mixed hybrids were not explored. Because two different collaborative algorithms were explored, the 41 systems evaluated represent 24 of the 53 spaces in this table, including 12 recommenders with no previous known examples.

Of course, any such study is by its nature limited by the peculiarities of the 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. In particular, it is unfortunate that the circumstances of this study allow only very limited findings with respect to meta-level recommendation.

Three general results, however, can be seen. First, the utility of a knowledge-based recommendation engine is not limited strictly to its ability to retrieve appropriate products in response to user queries. Such a component can 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 the knowledge-based component as a secondary or contributing component rather than as the main retrieval component. Second, 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. None of the weak/strong cascade hybrids that were explored ranked less than eighth in any condition, and in the average results, they rank in four of the top seven positions. This is despite the fact that the primary recommender was operating in a state of reduced precision. Finally, the six hybridization techniques examined have very different performance characteristics. An implementer should evaluate the rela-

tive accuracy and consistency of each component of the hybrid to determine its best role in a hybrid system.

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 Bomshad Mobasher and Alfred Kobsa for helpful comments on early drafts of this chapter.

References

1. Balabanovic, M.: An Adaptive Web Page Recommendation Service. In: Agents 97: Proceedings of the First International Conference on Autonomous Agents, Marina Del Rey, CA, pp. 378-385. (1997)
2. Balabanovic, M.: Exploring versus Exploiting when Learning User Models for Text Representation. UMUAU 8(1-2), 71-102. (1998)
3. Basu, C., Hirsh, H. and Cohen W.: Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In: ProC. of the 15th Natl. Conf. on AI, Madison, WI, 714-720. (1998)
4. Billsus, D. and Pazzani, M.: User Modeling for Adaptive News Access. UMUAI 10(2-3), 147-180. (2000)
5. Breese, J. S., Heckerman, D. and Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proc. of the 14th Annual Conf. on Uncertainty in AI, 43-52. (1998)
6. Brin, S. and Page, L The anatomy of a large-scale hypertextual web search engine. Comp. Networks and ISDN Systems, 30 (1–7):107–117. (1998)
7. Brunato, M. and Battiti, R.: 2003. A Location-Dependent Recommender System for the Web. In: Proceedings of the MobEA Workshop, Budapest, May 20, 2003. Accessed at http://www.science.unitn.it/~brunato/pubblicazioni/MobEA.pdf. (2003)
8. Burke, R.: The Wasabi Personal Shopper: A Case-Based Recommender System. In: Proc. of the 11th Nat. Conf. on Innovative Appl. of AI, 844-849. (1999)
9. Burke, R.: Knowledge-based Recommender Systems. In: A. Kent (ed.): Encyclopedia of Library and Information Systems, 69, Sup. 32. (2000)
10. Burke, R.: Hybrid Recommender Systems: Survey and Experiments. UMUAI 12 (4), 331-370. (2002)
11. Burke, R. Hybrid Recommender Systems: A Comparative Study. CTI Technical Report 06-012. 2006. (Available at http://www.cs.depaul.edu/research/technical.asp.)
12. Burke, R., Hammond, K., and Young, B.: The FindMe Approach to Assisted Browsing. IEEE Expert, 12 (4), 32-40. (1997)
13. Cheetham, W., and Price J.: Measures of Solution Accuracy in Case-Based Reasoning Systems. In Proc. of the 6th Eur. Conf. on CBR. Lecture Notes in Computer Science 3155. Springer-Verlag, London, pp. 106 – 118. (2004)
14. Chen, L. and Sycara, K.: WebMate: A personal agent for browsing and searching. In Proc. of the 2nd Intl Conf. on Autonomous Agents (Agents'98), 132-139, New York. ACM Press. (1998)
15. Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M.: Combining Content-Based and Collaborative Filters in an Online Newspaper. SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation. Berkeley, CA. Accessed at http://www.cs.umbc.edu/~ian/sigir99-rec/papers/claypool_m.ps.gz (1999)
16. Cohen, W. W.: Fast effective rule induction'. In Machine Learning: Proc. of the 12th Intl Conf., Lake Tahoe, CA, 115-123. (1995)

17. Goldberg, D., Nichols, D., Oki, B, & Terry, D. Using collaborative filtering to weave an information tapestry. 35(12):61–70. CACM. (1992)
18. Good, N., Schafer, B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J.: Combining Collaborative Filtering With Personal Agents for Better Recommendations. In: Proceedings of the AAAI'99 Conf., 439-446. (1999)
19. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An Algorithmic Framework for Performing Collaborative Filtering. Proc. of the 1999 Conf. on Research and Development in Information Retrieval. Berkeley (SIGIR '99), 230-237. (1999)
20. Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T.: Evaluating Collaborative Filtering Recommender Systems. ACM Trans. on Inf. Sys. 22 (1). (2004)
21. Hill, W., Stead, L., Rosenstein, M. and Furnas, G.: Recommending and evaluating choices in a virtual community of use. In: CHI ’95: Conf. Proc. on Human Factors in Computing Sys., Denver, CO, 194-201. (1995)
22. Jennings, A. and Higuchi, H.: A User Model Neural Network for a Personal News Service. UMUAI 3, 1-25. (1993)
23. Kolodner, J.: Case-Based Reasoning. San Mateo, CA: Morgan Kaufmann. (1993)
24. Krulwich, B.: Lifestyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data. AI Magazine 18 (2), 37-45. (1997)
25. Lang, K.: Newsweeder: Learning to filter news. In: Proc. of the 12th Intl Conf. on Machine Learning, Lake Tahoe, CA, 331-339. (1995)
26. Littlestone, N. and Warmuth, M.: The Weighted Majority Algorithm. Information and Computation 108 (2), 212-261. (1004)
27. McCarthy, J. F. and Anagnost, T. D.: MUSICFX: An Arbiter of Group Preferences for Computer Supported Collaborative Workouts. In Proc. of the ACM 1998 Conf. on Comp. Support. Coop. Work (CSCW 98), 363-372. (1998)
28. McCarthy, K., McGinty, L. Smyth, B. and Reilly, J.: A Live-User Evaluation of Incremental Dynamic Critiquing'. In H. Muñoz-Avila and F. Ricci (eds.): CBR Research and Develop. (6th Intl Conf., ICCBR 2005). Chicago, IL, 339-352. (2005)
29. McSherry, D.: Diversity-Conscious Retrieval. In S. Craw & A. Preece (eds.): Advances in CBR (6th Eur. Conf., ECCBR 2002). Aberdeen, Scotland, 219-233. (2002)
30. Melville, P., Mooney, R. J. and Nagarajan, R.: Content-Boosted Collaborative Filtering for Improved Recommendations. In: Proc. of the 18th Natl. Conf. on AI (AAAI-2002), pp. 187-192. (2002)
31. Mobasher, B., Dai, H., Luo, T. and Nakagawa, M.: Discovery and evaluation of aggregate usage profiles for web personalization. Data Mining and Knowledge Discovery, 6:61-82. (2002)
32. Mobasher, B., Jin, X., and Zhou, Y.: Semantically Enhanced Collaborative Filtering on the Web'. In B. Berendt, et al. (eds.): Web Mining: From Web to Semantic Web. LNAI Volume 3209, Springer. (2004)
33. Mobasher, B. and Nakagawa, M.: A Hybrid Web Personalization Model Based on Site Connectivity. In Proc. of the WebKDD Workshop at the ACM SIGKDD Intl Conf. on Knowledge Discovery and Data Mining, Washington, DC. (2003)
34. Mooney, R. J. and Roy, L.: Content-Based Book Recommending Using Learning for Text Categorization. SIGIR ’99 Workshop on Recommender Systems: Algorithms and Evaluation. Berkeley, CA. Accessed at http://www.cs.umbc.edu/~ian/sigir99-rec/papers/mooney_r.ps.gz (1999)
35. O'Sullivan, D., Smyth, B., Wilson, D. C.: Preserving recommender accuracy and diversity in sparse datasets. Intl J on AI Tools 13(1): 219-235. (2004)
36. Pazzani, M. J.: A Framework for Collaborative, Content-Based and Demographic Filtering. AI Review 13 (5/6), 393-408. (1999)
37. Pazzani, M. and Billsus, D.: Learning and Revising User Profiles: The Identification of Interesting Web Sites. Machine Learning 27, 313-331. (1997)

38. Pazzani, M., Muramatsu, J., and Billsus, D.: Syskill & Webert: Identifying Interesting Web Sites. In Proc. of the 13th Natl Conf. on AI, 54-61. (1996)
39. Pu, P. H. Z. and Kumar, P.: Evaluating Example-based Search Tools. In EC'04; Proc. of the 5th ACM Conf. on Electronic Commerce. New York, 208-215. (2004)
40. Reilly, J., McCarthy, K, McGinty, L. and Smyth, B.: Incremental Critiquing. In M. Bramer, et al. (eds.): Research and Devel. in Int. Sys. XXI. AI-2004, Cambridge, UK. (2004)
41. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In: Proc. of the Conf. on Comp. Supp. Coop. Work, Chapel Hill, NC, 175-186. (1994)
42. Resnick, P. and Varian, H. R.: Recommender Systems. Comm. of the ACM 40 (3), 56-58. (1997)
43. Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J. Miller, B. and Riedl, J.: Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System. In: Proc. of the ACM 1998 Conf. on Comp. Supp. Coop. Work, Seattle, WA, 345-354. (1998)
44. Schmitt, S. and Bergmann, R.: Applying case-based reasoning technology for product selection and customization in electronic commerce environments. In: 12th Bled Electronic Commerce Conf. Bled, Slovenia, June 7-9, (1999)
45. Schwab, I. and Kobsa, A.: Adaptivity through Unobstrusive Learning. Künstliche Intelligenz 16(3): 5-9. (2002)
46. Shardanand, U. and Maes, P.: Social Information Filtering: Algorithms for Automating "Word of Mouth". In: CHI '95: Conf. Proc. on Human Factors in Comp. Sys. Denver, CO, 210-217. (1995)
47. Shimazu, H.: ExpertClerk: Navigating Shoppers' Buying Process with the Combination of Asking and Proposing. In B. Nebel, (ed.): Proceedings of the 17th Intl J Conf on AI, 1443-1448. (2001)
48. Smyth, B. and Cotter, P.: A Personalized TV Listings Service for the Digital TV Age. Knowledge-Based Systems 13: 53-59. (2000)
49. Van Setten, M.: Supporting People in Finding Information: Hybrid Recommender Systems and Goal-Based Structuring. Report No. 016 (TI/FRS/016). Enschede, the Netherlands: Telematica Institut. (2005)
50. Wilson, D. C., Smyth, B., O'Sullivan, D.: Sparsity Reduction in Collaborative Recommendation: A Case-Based Approach. Intl J of Pattern Recognition and AI 17 (5): 863-884. (2003)