

**A Hybrid Rule-based and Case-based Approach
to Collaborative Filtering**

*A dissertation submitted in partial fulfillment
of the requirement for the award
of the degree of*

Master of Technology

In
Computer Science and Technology

By
Shweta Tyagi

Under the Supervision of
Prof. K. K. Bharadwaj

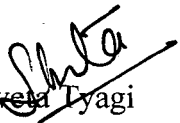


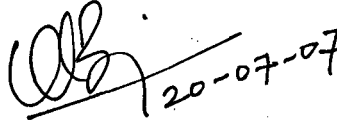
**School of Computer and Systems Sciences
Jawaharlal Nehru University
New Delhi-110067
July-2007**


Certificate

This is to certify that the dissertation entitled “**A Hybrid Rule-based and Case-based Approach to Collaborative Filtering**” submitted by **Ms. Shweta Tyagi** for the award of the degree of **Master of Technology (M. Tech.) in Computer Science and Technology** to the Jawaharlal Nehru University, Delhi is a bonafide record of work carried out under the guidance and supervision of **Prof. K. K. Bharadwaj**.

The results embodied in the dissertation have not been submitted to any other University or Institute for the award of any Degree or Diploma.


Shweta Tyagi
(Student)

 20-07-07
Prof. K. K. Bharadwaj
(Supervisor)


Dean, SC&SS

Jawaharlal Nehru University

New Delhi-67

India.

Prof. P. Anitha N:
Dean

School of Computer & Systems Sciences
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

Acknowledgement

It is my privilege and great pleasure to convey my gratitude to those who have directly or indirectly, helped me in completion of this academic endeavor. I also acknowledge the extremely stimulating discussions I have had with my guide **Prof. K. K. Bharadwaj**. It would have been impossible to complete this dissertation without his encouragement, support and invaluable guidance. His assistance was indispensable to the completion of this dissertation.

I also gratefully acknowledge Dean, Prof. S. Balasundaram for providing the necessary infrastructure to carry out this dissertation. I must mention the invaluable appreciation and encouragement by my all friends, to keep me enthusiastic during this dissertation in the face of many adversities. I regret forgetting to express my gratitude to anyone else who has contributed to this work. The gratitude does live on in my heart.

Finally, I wish to thank my husband, Mr. Janesh Kumar Kaushik and my parents for their invaluable help, constant support and encouragement.

Shweta Tyagi

Abstract

Recommender systems have become an important research area and are in widespread use. Various techniques have been proposed for providing recommendations. The main techniques that are used for the recommendations are: collaborative-based, content-based and knowledge-based. Each of these techniques has its own strengths and limitations. To overcome such limitations, these techniques have sometimes been combined in hybrid recommender systems.

In this work we propose a hybrid recommender system using rule-based and case-based reasoning approach to collaborative filtering. An architecture, appropriate for domains that are understood reasonably well but still imperfectly, is proposed combining rule-based and case-based reasoning. A set of rules is used to acquire the preference of a user; it then draws analogies from cases to deal with the exceptions to the rule. The rules together with the cases provide the preferences of a user in an innovative way and with an improved accuracy. The proposed knowledge-based recommender system, that combines rule-based and case-based reasoning, was applied to the task of movie recommendation. The experimental results reveal that the proposed hybrid approach out performs either of the rule-based or case-based reasoning alone.

Contents

	Page
1. Introduction.....	1
1.1. Recommender Systems.....	1
1.2. Recommendation Techniques.....	2
1.2.1. Content Based Recommender Systems	2
1.2.2. Collaborative Recommender Systems	3
1.2.3. Demographic Recommender Systems	4
1.2.4. Utility Based Recommender Systems	5
1.2.5. Knowledge Based Recommender Systems	5
1.3. Limitations of Various Recommendation Techniques	6
1.4. Hybrid Recommender Systems.....	8
1.4.1. Weighted.....	8
1.4.2. Switching.....	8
1.4.3. Mixed.....	9
1.4.4. Feature Combination	9
1.4.5. Cascade.....	10
1.4.6. Feature Augmentation.....	10
1.4.7. Meta level.....	10
1.5. Hybridization of Knowledge Based and Collaborative Filtering.....	11
1.6. Organization of the Thesis.....	11
2. Knowledge Based Techniques.....	13
2.1. Knowledge Engineering.....	13
2.2. Rule-based Reasoning.....	14
2.3. Case-based Reasoning.....	15
2.4. Combination of Rule-based and Case-based reasoning.....	16
2.5. Methodology.....	17
2.5.1. RBR.....	18
2.5.2. Indexing.....	18
2.5.3. CBR.....	18

2.5.4. Combination.....	19
2.5.5. Threshold Setting.....	22
2.6. Accuracy of a System.....	23
3. Combination of Rule-based and Case-based reasoning approach with Collaborative Filtering.....	24
3.1. Cascade Hybrid Recommender System.....	24
3.2. Movielens dataset.....	25
3.3. Ruleset.....	27
3.4. Case Library.....	28
3.5. Case Indexing.....	29
3.6. Similarity Measure.....	29
3.7. Proposed Scheme.....	30
3.8. Collaborative Filtering.....	32
4. Experiments and Results.....	34
4.1. Experiments on Threshold Setting Procedure.....	34
4.2. Experiments on RC-Hybrid Procedure.....	36
4.3. Analysis of Results.....	38
5. Conclusion.....	39
Appendix A.....	40
Appendix B.....	50
References.....	53

List of Figures

	Page
Figure 4-1 Distributions of Analogies by similarity score.....	35
Figure 4-2 Distributions of Analogies by accuracy score.....	35
Figure 4-3 Distributions of Analogies by significance score.....	36
Figure 4-4 Results for experiment 1.....	37
Figure 4-5 Results for experiment 2.....	37
Figure 4-6 Results for experiment 3.....	37
Figure 4-7 Results for experiment 4.....	38

List of Tables

	Page
Table 1-1 Tradeoffs between recommendation techniques.....	7
Table 2-1 Terms used in calculation of accuracy & significance.	20
Table 3-1 Movielens data set files.....	26
Table 3-2 Some cases from the case library.....	28

1. Introduction

Recommender Systems emerged as an independent research area in the mid-1990s. These have become fundamental applications in e-commerce and information access. A lot of work has been done both in the industry as well as in academia to develop new approaches of recommender systems. Still it has a problem-rich research area and the abundance of practical applications, that help users to deal with information overloads and provides personalized recommendations, content, and services [G. Adomavicius et al. 2005].

1.1 Recommender Systems

Recommender systems are designed as new types of internet-based software tools, to help users find their way through today's complex on-line shopping & entertainment websites. These systems provide suggestions to the user based on their need, requirement or preferences. By providing suggestions, these systems effectively shrink large information spaces so that users are directed toward those items that best meet their needs and preferences. The area of Recommender systems has an abundance of practical applications. Examples of such applications include recommending books, CDs, and other products at Amazon.com [G. Linden et al. 2003], movies by MovieLens [B.N. Miller et al. 2003], and news at AdaptiveInfo.com [D. Billsus et al. 2002]. Various techniques have been proposed for performing recommendation. All recommendation techniques have their own capabilities and limitations. These methods have sometimes been combined in hybrid recommendations to avoid their weaknesses and to improve performance [G. Adomavicius et al. 2005, R. Burke 2002].

1.2 Recommendation Techniques

Recommendation techniques have a number of possible classifications [G. Adomavicius et al. 2005, R. Burke 2002]. According to Robin Burke the classification of various techniques of Recommender systems is based on the sources of data on which recommendation is based and the use to which that data is put. On this basis, we can distinguish five different recommendation techniques as follows:

1.2.1 Content Based Recommender Systems

To make recommendations, Content-based methods analyze the description of the items that have been rated by the user and the description of items to be recommended. A wide range of algorithms have been proposed to analyze the content based recommendations. The recommendation problem can be formulated as follows [G. Adomavicius et al. 2005, M. J. Pazzani 1999]:

Let C be the set of all users and let S be the set of all possible items that can be recommended, such as books, movies, or restaurants. Let u be a rating (utility) function that measures the usefulness of item s to user c , i.e., $r : C \times S \rightarrow R$, where R is a totally ordered set (e.g., nonnegative integers or real numbers within a certain range). Then, for each user $c \in C$, we want to choose such item $s' \in S$ that maximizes the user's rating (utility).

$$\forall c \in C, s_c = \arg \max r(c, s)$$

More formally the central problem is to evaluate new ratings of the not-yet-rated items which can be estimated in many different ways using methods from machine learning, approximation theory, and various heuristics. In content-based recommendation methods, the rating $r(c, s)$ of item s for user c is estimated based on the ratings $r(c, s_i)$ assigned by user c to items $s_i \in S$ that are "similar" to item s .

The content-based approach to recommendation has roots in information retrieval and information filtering research [N. Belkin et al. 1992]. Many current content-based systems use documents, Web sites (URLs), and news messages etc and Others use users' tastes, preferences, and needs. According to Fab [M. Balabanovic et al. 1997] System and Syskill & Webert system [M. Pazzani et al. 1997], the importance of word k_i in document d_j is determined with some weighting measure w_{ij} that can be defined in different ways. Rocchio's [T. Joachims 1997] method uses the frequency/inverse document frequency (TF-IDF) weight for each informative keyword weights in Information Retrieval Besides the traditional heuristics, other techniques for content-based recommendation are Bayesian classifiers and various machine learning techniques, including clustering, decision trees, and artificial neural networks. These methods use models learned from the underlying data using statistical learning and machine learning techniques, rather than heuristics [G. Adomavicius et al. 2005].

1.2.2 Collaborative Recommender Systems

Collaborative filtering [J. S. Breese et al. 1998, R. Burke 2002] makes recommendations by finding correlations among users of a recommendation system. Collaborative recommender systems refer to multiple users, "sharing" recommendations, in the form of ratings.

The rating (utility) $r(c,s)$ of item s for user c is estimated based on the ratings $r(c,s)$ assigned to item s by those users $c_j \in C$ who are "similar" to user c . According to Gediminas Adomavicius various algorithms for collaborative recommendations can be grouped into two general classes given below:

Memory-based methods

Memory-based algorithms [J. S. Breese et al. 1998] essentially are heuristics that make rating predictions based on the entire collection of previously rated items by the users. In most of the approaches, the similarity between two users is based on their ratings of items

that both users have rated. The two most popular approaches are correlation and cosine-based. Many performance-improving modifications, such as default voting, inverse user frequency, case amplification, and weighted-majority prediction, have been proposed as extensions to these standard correlation-based and cosine-based techniques.

Model-based Methods

Model-based algorithms [J. S. Breese et al. 1998] use the collection of ratings to estimate or learn a model, which is then used to make rating predictions. From a probabilistic perspective, the collaborative filtering task can be viewed as calculating the expected value of a rating, given what we know about the user. For the active user, we wish to predict ratings on as-yet unobserved items. To estimate this probability, there are two alternative probabilistic models: (i) Cluster models and (ii) Bayesian networks. Moreover, another collaborative filtering methods using social information filtering, graph theoretical approach, artificial immune system have been studied. Collaborative filtering by personality diagnosis presents a combine study of model & memory based approaches.

1.2.3 Demographic Recommender Systems

Based on personal attributes demographic recommender systems intend to classify the user and make recommendations based on demographic classes [M. J. Pazzani 1999, R. Burke 2002]. Demographic information (age, gender, occupation, etc.) can be used to identify the types of users that like a certain object. The benefit of a demographic approach is that it may not require a history of user ratings of the type needed by collaborative and content-based techniques.

The 'Grundy system' recommends books using this technique of recommender system [E. Rich 1979]. Demographic techniques form people-to-people correlations like collaborative ones, but use different data. There are varieties of representation of demographic information in a user model. The features can be hand-crafted or extracted

from user's home pages. Some systems use machine learning approaches to arrive at a classifier based on demographic data [R. Burke 2002].

1.2.4 Utility Based Recommender Systems

Utility-based recommenders make suggestions based on a utility function which gives the utility of each object for the user. The utility-based recommender system takes care of non-product attributes, such as vendor reliability and product availability unlike other recommender systems. However the computation of Utility function is its major problem. Tête-à-Tête and the e-commerce site PersonaLogic2 (www.personalogic.aol.com/go/gradschools) are the examples of utility-based recommender system each having different utility function [R. Burke 2002].

1.2.5 Knowledge Based Recommender Systems

Knowledge-based recommender systems attempt to suggest objects based on knowledge about users and objects [R. Burke 2002]. Knowledge-based approaches are different from other recommendation techniques as they have functional knowledge about how a particular item meets a particular user need, and can therefore reason about what product meet the user's requirements. The knowledge structure may be a user profile, a query made by the user or a representation of user's needs.

The Restaurant recommender Entree system and several other recent systems employ techniques from case-based reasoning for knowledge-based recommendation. The Entree system makes recommendation by finding restaurants in a new city similar to restaurants the user knows and likes. The knowledge used by a knowledge-based recommender can also take other forms [R. Burke 2000, B. Towle et al. 2000].

1.3 Limitations of Various Recommendation Techniques

Various recommendation techniques have been developed as discussed above and each of these approaches has its strengths and weaknesses [R. Burke 2002, G. Adomavicius et al. 2005] as given in table 1-1. As a collaborative filtering system rely solely on user's preferences to make recommendation, it collects more ratings from more users. However, a collaborative filtering system suffers from a 'ramp-up' problem: *new item problem*- until the new item is rated by a sufficient number of users the system would not be able to recommend it, and *new user problem*- until a sufficient number of rated items have been collected, the system cannot be useful for a particular user. This technique also suffers from a *sparsity problem*- until there are large numbers of users whose habits are known, the system cannot be useful for most users.

Content-based techniques also have a 'new user' ramp-up problem as they must accrue enough ratings to build a trustworthy recommendation to a new user. This technique also has the problem of *limited content analysis* as they are limited by the features that are explicitly coupled with the items that they recommend.

Demographic recommenders do not have the 'new user' problem, because they do not have need of a record of ratings from the user. Instead it has the problem of gathering the requisite demographic information: those users are hesitant to disclose.

Utility-based recommendation techniques make suggestions based on a computation of the utility of each object for the user. Therefore, the central problem for such a technique is how to create a utility function for each user.

A knowledge-based recommender system avoids some of these drawbacks. It does not have ramp-up and sparsity problems as its recommendations do not depend on a foundation of user ratings. Knowledge-based approaches have functional 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. It does not have to collect information about a particular user since its judgments are not dependent of individual tastes. Such characteristics make knowledge-based recommenders not only

reliable and worthy systems on their own, but also highly complementary to other types of recommender systems.

Technique	Pluses	Minuses
Collaborative filtering	A. Can identify cross genre niches B. Domain Knowledge not needed C. Adaptive: quality improves over time D. Implicit feedback sufficient	I. New user ramp-up problem J. New item ramp-up problem K. Quality dependent on large historical data set
Content-based	B, C, D	I, K
Demographic	A, B, C	I, K L. Must gather demographic Information
Utility-based	E. No ramp-up required F. Sensitive to changes of preference G. Can include non-product Features	M. User Must input utility function N. Suggestion ability static
Knowledge-based	E, F, G H. Can map from user needs to Products	N O. Knowledge engineering Required
Hybrid Approach (Knowledge-based and Collaborative Filtering)	A, C, D, E, F, G, H	O

Table 1-1 Tradeoffs between recommendation techniques.

1.4 Hybrid Recommender Systems

To give better performance with fewer of the drawbacks of any individual approach, hybrid recommender systems combine two or more recommendation techniques. Different ways to combine recommendation techniques into a hybrid recommender system [G. Adomavicius et al. 2005, R. Burke 2002] can be classified as follows:

1.4.1 Weighted

A weighted hybrid recommender is one in which the ratings of several recommendation techniques are combined together to produce a single recommendation. For example, the P-Tango system [Claypool et al. 1999] uses a linear combination of collaborative and content-based recommenders. The benefit of a weighted hybrid is that we can adjust the weights as predictions about user ratings are confirmed or disconfirmed. It is a very straightforward process and easy to perform. This technique of hybridization avoids a New Item Problem of collaborative filtering. We can take the combination of collaborative filtering with some other recommendation technique by giving a small weight to collaborative, in case of a new item added to the recommender system.

1.4.2 Switching

A switching hybrid switches between recommendation techniques using some criterion. The DailyLearner system uses a hybridization of content-based and collaborative recommenders. This switching hybrid does not completely avoid the ramp-up problem, since both the collaborative and the content-based systems have the 'New user' problem. In a switching hybrid the collaborative technique provides the ability to cross genres. Switching hybrids adds an additional complexity into the recommendation process. This is because of the switching criteria that must be determined. However, the sensitivity to the strengths and weaknesses of its constituent recommenders is the benefit of the system.

Tran & Cohen (2000) used a switching hybrid recommender system for electronic commerce.

1.4.3 Mixed

Recommendations from several different recommendation techniques are presented (combined) at the same time. The PTV system [B. Smyth 2000] uses this approach to build a recommended program of television viewing. The program is suggested by combining recommendations from the two techniques, content-based techniques based on textual descriptions of TV shows and collaborative information about the preferences of other users. Such a technique avoids the ‘new item’ start-up problem as new shows (items) can be recommended on the basis of their descriptions even if they have not been rated. It does not get around the ‘new user’ start-up problem, but if such a system is integrated into a digital television, it can build its profiles by tracking what shows are watched and for how long. Other examples of the mixed hybrid recommendation are ProfBuilder [Wasfi 1999] and PickAFlick [R. Burke et al. 1997, R. Burke et al. 2000].

1.4.4 Feature combination

This kind of recommender system collects features from different recommendation data sources and combined them into a single recommendation algorithm. In this way to achieve the content/collaborative feature combination hybrid, this system treats collaborative information as simply additional feature data associated with each example and uses content-based techniques over this augmented data set. It does not rely on collaborative data exclusively, so it reduces the sensitivity to the number of users who have rated an item. For example, Basu, Hirsh and Cohen (1998) use such a hybrid recommender system for the task of recommending movies using both user ratings and content features.

1.4.5 Cascade

The cascade hybrid involves a staged process, unlike the previous hybridization methods. In this technique, one recommendation technique refines the recommendations given by another technique. Cascading allows the system to give the priority to a technique if the items are well-differentiated by this technique and then employs the other one, because the second step focuses only on those items for which additional discrimination is needed. For example, EntreeC, the restaurant recommender [R. Burke 2000, R. Burke 2002] is a cascaded knowledge-based and collaborative recommender. It uses its knowledge of restaurants to make recommendations of equal preference, and the collaborative technique is employed to break ties.

1.4.6 Feature augmentation

Output from one technique is used as an input feature to another. Ratings or classification of an item are produced by one technique and other technique then processes that information. For example, the Libra system makes recommendations of books based on data found in Amazon.com. The GroupLens research team working with Usenet news filtering also employed feature augmentation [Sarwar et al. 1998]. Augmentation is attractive because it offers a way to improve the performance of a system.

1.4.7 Meta-level

It is another way to combine two recommendation techniques in which the entire model generated by one is used as input for another. The meta-level method uses the learned model from one technique as a compressed representation of a user's interest, and other technique can operate on this information-dense representation more easily. Fab [M. Balabanovic et al. 1997] was the first meta-level hybrid web filtering system. Another meta-level hybrid that focuses exclusively on recommendation is given by Pazzani

(1999) as ‘collaboration via content’. More recently, a content-based naive Bayes classifier has been used meta-level hybrid technique for recommendations.

1.5 Hybridization of Knowledge Based and Collaborative Filtering

In order to improve recommendation accuracy and to address some of the limitations (e.g., new user, new item problems) of recommender systems, a hybrid recommender systems can also be augmented by knowledge-based techniques, such as case-based reasoning. For example, ‘EntreeC’, a knowledge-based recommender system with collaborative filtering approach produces a recommender system with some of the best characteristics of both to recommend restaurants to its users [R. Burke 2000, R. Burke 2002].

A “cascading” hybrid recommender system which combines the two: collaborative filtering technique and knowledge-based technique, uses its knowledge to produce the best possible set of recommendations, and then uses collaborative filtering to break ties between those recommendations. The differing strengths of these approaches suggest that they may be complementary rather than competing approaches for the improved production of recommendations. In spite of the necessary investment in knowledge engineering, such a hybrid could offer good performance and benefits of collaborative filtering as well as of knowledge-based.

The bottleneck of such hybrid recommender systems is knowledge acquisition. However when a knowledge-based technique such as case-based reasoning is added to rule-based reasoning, a solution to the above problem of knowledge engineering can be proposed.

1.6 Organization of the Thesis

In the first chapter, we have summarized various recommender systems, their limitations and hybridization of recommendation techniques to avoid the limitations of these

techniques. The accuracy of a knowledge-based system can be improved by combining rule-based and case-based reasoning techniques. Second chapter will present a methodology to combine rule-based and case-based reasoning techniques. This combined approach was applied to the MovieLens dataset and discussed in chapter three. The chapter ends with proposing a scheme of hybridization of rule-based and case-based reasoning approach, to get the preference of each user together with the set of recommendations. The combination of rule-based and case-based reasoning techniques will be implemented and the results will be presented in chapter four. The conclusion of the dissertation and future directions will be discussed in the last chapter.

2. Knowledge Based Techniques

Knowledge-based or expert systems are very successful in the research field of Artificial Intelligence [J. Giarratano et al. 1993, E. Rich et al. 1999]. Knowledge based systems are nothing but computer programs that contain huge amounts of knowledge, rules and reasoning methodologies to provide solutions to practical problems. In a recent survey the UK Department of Trade & Industry found over 2000 knowledge based systems in commercial operation.

2.1 Knowledge Engineering

Knowledge engineering is a field within artificial intelligence that develops knowledge based systems. The main objective of knowledge engineering is to build, maintain and develop the knowledge-based systems. In the early years of knowledge engineering, knowledge acquisition was considered as the bottleneck in development of an expert system. Acquisition of knowledge, to build a robust and useful system, was a very long and expensive exercise. So, knowledge acquisition has become a challenging research field within knowledge engineering. However, despite the undoubted success of knowledge based systems in various sectors knowledge engineers have faced several problems. Together with Knowledge acquisition, Development and maintaining are also the problems of knowledge based systems.

However, over the last few years an alternative reasoning paradigm and computational problem solving method has increasingly attracted more and more attention of researchers in the field of knowledge engineering. Rule-based reasoning and case-based reasoning approach are the two examples of such methods. These reasoning

methodologies are widely used in the area of knowledge based systems and providing the solution to the knowledge engineering problems. There are a number of techniques that are used in the field of knowledge based systems but we are concerned to these two reasoning techniques.

2.2 Rule-based Reasoning

Rule-based reasoning (RBR) system represent knowledge in terms of a bunch of rules (facts) that tell us what one should do or what one could conclude in different situations. These rules are in the form of IF THEN rules such as

IF some condition **THEN** some action

If the 'condition' is satisfied, the rule will take the 'action'. In many systems, the conditions are expressed logically as conjunctions (occasionally, disjunctions) of predicates.

Today a wide range of knowledge based expert systems have been built and many of these knowledge based expert systems have been built by expressing the knowledge in the form of rules. Rule-based reasoning techniques can answer the 'What if' type questions by providing the reasoning. To create a rule-based system for a given problem, we must have (or create) the following:

- A set of facts to represent the initial working memory. This should be anything relevant to the beginning state of the system.
- A set of rules. This should encompass any and all actions that should be taken within the scope of a problem, but nothing irrelevant. The number of rules in the system can affect its performance, so we don't want any that aren't needed.

- A condition that determines that a solution has been found or that none exists. This is necessary to terminate some rule-based systems that find themselves in infinite loops otherwise.

The most widely used knowledge representation scheme for expert systems is rules. Rule-based systems can be either *goal driven* or *data driven*. The goal driven rule-based systems use *backward chaining* to test whether some hypothesis is true, however the data driven rule-based systems use *forward chaining* to draw new conclusions from existing data.

2.3 Case-based Reasoning

Case-based Reasoning (CBR) is a relatively recent problem solving technique that is attracting the attention of Knowledge Engineers. However, the numbers of people with first hand theoretical or practical experience of CBR is still small. Case-based reasoning solves new problems by adapting previously successful solutions to similar problems. A case is a contextualized piece of knowledge representing an experience. It contains the past lesson that is the content of the case and the context in which the lesson can be used [M. M. Richter et al. 2006, Watson 1999]. Typically a case comprises:

- the *problem* that describes the state of the world when the case occurred,
- the *solution* which states the derived solution to that problem, and/or
- the *outcome* which describes the state of the world after the case occurred.

A new problem is matched against cases in the case base (case library) and one or more similar cases are *retrieved*. A solution suggested by the matching cases is then *reused* and tested for success. Unless the retrieved case is a close match the solution will probably have to be *revised* producing a new case that can be *retained*. Therefore the process of finding solution to the new problem consists of the following steps [Watson 1999]:

- Retrieve the most similar case.
- Reuse the case to attempt to solve the problem;
- Revise or adapt the proposed solution if necessary, and
- Retain the new solution as a part of a new case.

These are the well known steps used in the process of case-based reasoning. Basically case-based reasoning provides the solution of various problems of knowledge engineering. Now days various knowledge based systems are using this approach to give the better solution and to limit the knowledge engineering problems.

2.4 Combination of Rule-based and Case-based reasoning

As discussed above the knowledge of any system can be represented in terms of cases as well as rules. But the accuracy of a system can be improved when we bring together knowledge in two forms: rules and cases. This idea is used for domains that are understood well but not perfectly. It uses a set of approximately correct rules, to obtain a preliminary answer (solution) for a given problem. To deal with the exceptions to the rules, it then finds analogies from cases. A procedure for solving a problem by a combination of RBR and CBR can be explained by a RC-Hybrid Procedure proposed by Andrew R. Golding and Paul S. Rosenbloom in 1996. The algorithm for combining rule-based and case-based reasoning is given below:

Algorithm for RC-Hybrid Procedure

Until problem is solved do:

Step 1. RBR: Use the rules to attempt a solution to the problem.

Step 2. CBR: Look for analogies that contradict the solution suggested by RBR.

Step 3. Combination: Decide between the solutions suggested by RBR and CBR.

The RC-Hybrid procedure allows innovative ways of doing case-based reasoning what it could have achieved with rules or cases alone. This approach has been applied to the *Anapron* system (a name pronunciation system) with minimal knowledge engineering [A. R. Golding et al. 1996] and it was found to perform almost at the level of the best commercial name pronunciation systems. Moreover, it was investigated that *Anapron's* accuracy was exceeded by using the RC-Hybrid procedure, rather than using rules or cases alone. This shows the enhancement in accuracy acquired by combining rule-based and case-based reasoning.

Together with the development in the field of recommender systems research, the awareness to make the recommendation process more transparent for users is increasing. CBR systems examine multi-dimensional or semantic ratings so that system gets information about the reason behind a preference with which the recommendation process can be explained and the system's recommendations can be justified [D. Mcsherry 2005]. Also the RBR systems have explanation facilities that allow the user to ask *why* it asked some question, and *how* it reached some conclusion. These questions are answered by referring to the system goals, the rules being used, and any existing problem data.

2.5 Methodology

A data set is a collection of data related to the problem domain which is considered to understand the methodology of RC-Hybrid Procedure. This procedure is applied to the problem domain using four sources: a set of rules, a case library, a similarity metric, and a set of thresholds. First of all a set of rules is obtained using the data set The rules are applied to the target problem to get an approximate solution and then analogies are found from cases to cover exceptions to the rules. A case library is a collection of cases taken from the data set of the problem domain, where a case consists of a problem and its solution. The similarity metric is used to find the similarity between two problems. The

thresholds are applied at the combination step to decide whether an analogy should override the rules.

Steps required to understand the RC-Hybrid procedure are as follows:

2.5.1 RBR

Rule-based reasoning is used to find the rules from the set of rules, which can match against the target problem. Rule matching corresponds to first step of RC-Hybrid procedure. Any Rule matching the attributes of the target problem is taken as the 'matching rule'. The matching rule is taken as a 'provisional rule' and is not actually applied at this point. It will only be applied if not overridden by CBR.

2.5.2 Indexing

The indexing step stores the cases as a negative or a positive exemplar. A case is stored as a positive exemplar for a rule if it illustrates a place where the rule makes a correct prediction for the case. It stores the case as a negative exemplar for a rule if it illustrates a place where the rule makes an incorrect prediction for the case. Indexed cases are further accessed by the CBR step to critique the prediction made by RBR.

After all of the cases have been, they can be used as source exemplars of the analogies. The indexed cases will also be used to judge analogical compellingness.

2.5.3 CBR

The prediction made by RBR step is criticized by the CBR step. By looking for an analogy between the target problem and a negative exemplar of the rule, it tries to show that the target problem is an exception of the provisional rule. The list arranged by the

indexing scheme provides the negative exemplars of the provisional rule. The CBR checks each analogy in the list of indexed cases, one at a time and the combination step decides one of the analogies to be compelling.

A metric is considered during the step of CBR. This metric takes three arguments: a source problem, a target problem and a solution-to-be-transferred from source to target. Here the source problem will be negative exemplar, and the solution-to-be-transferred will be the solution that was obtained from this exemplar. The solution establishes a context for comparing the problems. Using these three arguments, the metric returns two values the similarity score and the implicit rule behind the analogy. The rule associated with the analogy is called the analogical rule. The analogical rule that any analogy makes is that a particular set of features that were found to be in common between source and target problems determines the same outcome for the two problems. Accordingly, the left-hand side of the analogical rule gives the features that were found by the metric to be shared by the two problems, and the right-hand side of the analogical rule gives the solution-to-be-transferred. The analogical rule will be used for judging whether analogy is compelling or not.

2.5.4 Combination

The combination step decides which of the other modules to follow, RBR or CBR. It evaluates the analogies proposed by CBR. If any analogy seems compelling, then CBR wins; else RBR wins. The similarity score of the analogy takes part in making the decisions of compellingness. The similarity score is the degree to which the source and the target problems match on relevant attributes, according to similarity metric. The combination step does not rely on the similarity metric exclusively because the metric is only a heuristic. Thus to check the compellingness of an analogy an empirical verification is required. This is a test of how well the analogical rule works for other exemplar in the case library. The test gives two results- the analogical rule's accuracy & the significance of the accuracy rating. Thus compellingness can now be expressed as a conjunction of the

two factors: the analogy must have a high similarity score, and it must perform well in the empirical verification. The conjunction provides more robust judgments of compellingness. To evaluate the accuracy and significance of the analogical rule let us consider:

o	Solution suggested by the analogical rule (action of the analogical rule)
m	Number of exemplar that the analogical rule applies to and that were found to give solution o
n	Total number of exemplars that the analogical rule applies to
M	Number of exemplars of the provisional rule that were observed to give solution o
N	Total number of exemplar of the provisional rule

Table 2-1 Explanation of terms used in calculation of an accuracy & significance.

Therefore the accuracy and significance can be calculated using these formulas given below:

$$\text{Accuracy of analogical rule} = m/n$$

$$\text{Significance of analogical rule} = 1 - \sum_{m \leq k \leq n} \binom{n-1}{k-1} r^{k-1} (1-r)^{n-k}$$

$$\text{where } r = M/N$$

An analogy between two apparently similar problems will be rejected if the similarity not to be predictive for other examples; and an analogy that works by spurious coincidence on the available examples will be rejected if there is not also a plausible similarity between its source and target. The compellingness of an analogy can be understood more precisely using the algorithm given below:

Algorithm to evaluate the compellingness of an Analogy

Step 1. If (similarity-score(Analogy) \geq S1)

Go to step 2 else go to step 5.

Step 2. If (accuracy(Analogy) \geq A)

Go to step 3 else go to step 5.

Step 3. If (significance(Analogy) \geq S or similarity-score(Analogy) \geq S2)

Go to step 4 else go to step 5.

Step 4. Analogy is found to be compelling.

Stop.

Step 5. Analogy is found to be unconvincing.

Stop.

7/11-14.555

Where S1, A, S, S2 are thresholds for deciding when a value is high enough; they can be provided from the outside, or set by the threshold setting procedure.



2.5.5 Threshold setting

The thresholds are used to determine whether an analogy is compelling or not. The point of compellingness enables the architecture to decide when it should listen to an analogy i.e., when the analogy is right and the rules are wrong. The threshold setting procedure provides a standard way of choosing values for the thresholds. The goal of threshold setting procedure is to decide on values for the thresholds that will result in analogies being classified as compelling whenever they would correct wrong solutions provided by the rules. It should accept the analogies that correct wrong solutions provided by the rules and conversely it should reject analogies that spoil correct solutions provided by the rules. For this purpose it generates training analogies and tries to select the thresholds so as to do the right thing for these training analogies. The approach is semi-automatic in that the user has the final say of what values to pick based on the procedure.

It generates its training analogies from the case library. It pretends that each exemplar in the case library in turn is a target problem, and it finds all analogies to it from other exemplars. It classifies the training analogy as

Helpful Analogies

It suggests the right solution for the target problem, where the rules would have suggested the wrong solution.

Harmful Analogies

It suggests the wrong solution for a problem where the rules would have suggested the right solution.

Misclassified Analogies

These are the helpful ones that are found unconvincing plus the harmful ones that are found convincing.

Thus the thresholds are selected to minimize the number of misclassified analogies. Besides setting all the thresholds simultaneously all the four thresholds were set one at a time. All the analogies when taken on the similarity axis, the similarity thresholds were set S_1 was chosen on the similarity axis to minimize the number of misclassified analogies and s_2 was chosen to exclude all harmful analogies. After choosing the similarity thresholds, all training analogies whose similarity scores fall below this value can be discarded. The dropped analogies are of no use as they are classified as the misclassified analogies and have no information about how to set rest of the thresholds.

Accuracy and significance thresholds are taken on the accuracy and significance axis respectively, to minimize the number of remaining misclassified analogies.

2.6 Accuracy of a System

The complementary properties of CBR and RBR can be advantageously combined to solve some problems to which only one technique fails to provide a satisfactory solution. There is an abundance of practical applications which demonstrate the improvement of the systems accuracy.

Beside the well known ANAPRON system [A. R. Golding et al. 1996] other practical examples [N. Cercone et al. 1999] are CASEY, INRECA, CABARET and RISE. The integration of rule-based and case-based reasoning is also useful in medical applications. A practical application of such kind of integration is used for making a decision support tool for Insulin Dependent Diabetes Mellitus management [R. Bellazzi et al. 1999]. This multi-modal reasoning system aims at providing physicians with a suitable solution to the problem of therapy planning by exploiting, in the most flexible way, the strengths of the two selected methods.

3. Combination of Rule-based and Case-based reasoning approach with Collaborative Filtering

In order to improve recommendation accuracy and to address some of the limitations (e.g., new user, new item problems) of recommender systems, a hybrid recommender system can also be augmented by knowledge-based techniques, such as case-based reasoning. For example, 'EntreeC', a knowledge-based recommender system with collaborative filtering approach produces a recommender system with some of the best characteristics of both to recommend restaurants to its users [R. Burke 2002]. A novel cascade hybrid recommender system can be considered by taking the combination of rule-based and case-based reasoning approach with collaborative filtering.

3.1 Cascade Hybrid Recommender System

A "cascading" hybrid recommender system which combines the two: collaborative filtering technique and knowledge-based technique, uses its knowledge to produce the best possible set of recommendations, and then uses collaborative filtering to break ties between those recommendations. The differing strengths of these approaches suggest that they may be complementary rather than competing approaches for the improved production of recommendations [R. Burke 2000, B. Towle et al. 2000]. In spite of the necessary investment in knowledge engineering, such a hybrid could offer good performance and benefits of collaborative filtering as well as of knowledge-based [G. Adomavicius et al. 2005]. The bottleneck of such hybrid recommender systems is knowledge acquisition. However when a knowledge-based technique such as case-based reasoning is added to rule-based reasoning, a solution to the above problem of knowledge engineering can be proposed.

Table 1-1 [R. Burke 2002] contrasts the collaborative filtering and knowledge-based approaches identifying the improved performance of an ideal hybrid. The positive and negative aspects of each approach are given in the table. The last row in the table suggests what might be achieved in an ideal hybrid that combines the techniques. In spite of the necessary investment in knowledge engineering, such a hybrid could offer good performance even with little or no user data, and the benefits of collaborative filtering as data is collected.

To test the proposed cascade hybrid recommendation technique, it was applied to the MovieLens data set available at <http://www.MovieLens.umn.edu>. It is a well known research movie recommendation website. To crack the puzzle of movie recommendation with MovieLens data set, the taste or preference of each user was extracted using RC-Hybrid procedure. Such a system was found to perform better what it could have performed with rules or cases alone. The experiments and results demonstrate the enhancement in accuracy very well, which will be discussed later.

3.2 MovieLens Dataset

MovieLens data sets were collected by the GroupLens Research Project at the University of Minnesota. This dataset has been used widely for the experiments of collaborative recommendation technique. But in this dissertation we have used this dataset for a hybrid recommender system. For this purpose we made some modifications in the dataset accordingly. The dataset is properly arranged in separate files. The description of the MovieLens dataset files is given in the table below:

Data File	This file consists of 100,000 ratings by 943 users on 1682 items and each user has rated at least 20 movies.
User File	The demographic information about each user is given in this file.
Genre File	It contains a list of the genres.
Item File	This file consists of the nineteen fields, representing the genres, a Boolean value, 0 or 1 indicates whether the movie belongs to the specific genre; a movie can be in several genres.

Table 3-1 MovieLens data set files.

This data set consists of 100,000 ratings (1-5) from 943 users on 1682 movies. Each user has rated at least 20 movies. Simple demographic info for the users (age, gender, occupation, zip-code). The nineteen genres of movies are given as unknown, Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War and Western. Occupation of the users are classified as administrator, artist, doctor, educator, engineer, entertainment, executive, healthcare, homemaker, lawyer, librarian, marketing, none, other, programmer, retired, salesman, scientist, student, technician and writer. Before presenting the scheme of hybrid technique we will consider some important factors.

- Ruleset
- Case Library
- Similarity Measure
- Case Indexing

For providing a ruleset for the MovieLens dataset we have found the three most preferable genres for each user. The algorithm is given below:

Algorithm for finding three most preferable genres

```
For each user u=1:1:943
  For each genre g=1:1:19
    M(g,1) = no of movies having genre 'g' and rating 5;
    If ((M(g,1)==0) S(1) = 0 Else S(1) = 1 End;
    M(g,2) = no of movies having genre 'g' and rating 4;
    If ((M(g,2)==0) S(2) = 0 Else S(2) = 1 End;
    M(g,3) = no of movies having genre 'g' and rating 3;
    If ((M(g,3)==0) S(3) = 0 Else S(3) = 1 End;
    Weighted_sum(g) = (5 * M(g,1) + 4 * M(g,2) + 3 * M(g,3));
    If (Weighted_sum(g) != 0)
      Weighted_avg(g) = Weighted_sum(g) / (5 * S(1) + 4 * S(2) + 3 * S(3));
    Else
      Weighted_avg(g) = 0;
    End
  End
End

Find the three largest values of the Weighted_avg(g=1:1:19), which gives
the three most preferable genres.

End
```

3.3 Ruleset

As we have the simple demographic information (age gender occupation and zip code) of each user available in the MovieLens dataset Based on this demographic information and

the three most preferable genres corresponding to each user, three rulesets were formed. The rulesets were formed by using the software See5 (Appendix A). This is a system that extracts informative patterns from the given dataset. The ruleset was generated by taking top 50 users who made the ratings of the maximum number items (movies). In this way the most representable users were selected for generating the ruleset. The three rulesets (Appendix B), used the demographic information of the users and provided the three most preferable genres respectively.

3.4 Case Library

As the demographic information of each user is provided in the dataset and the three most preferable genres are also have been found. A case library is constructed using the demographic information and the genres. The case library of the problem consists of the users name or id, age, gender, occupation, zip code, first preferred genre (Genre1), second preferred genre (Genre2) and third preferred genre (Genre3). A toy version of the case library is given in the table 3-2.

User-id	Age	Gender	Occupation	Zip code	Genre1	Genre2	Genre3
1	24	M	Technician	85711	Drama	Comedy	Action
2	53	F	Other	94043	Drama	Romance	Comedy
3	23	M	Writer	32067	Drama	Horror	Thriller
4	24	M	Technician	43537	Comedy	Thriller	Action
5	33	F	Other	15213	Comedy	Action	Sci-Fi

Table 3-2 Some cases from the case library

3.5 Case Indexing

The ruleset is used for indexing the cases in the case library. The case indexing procedure [A. R. Golding et al. 1996] can be explained by the algorithm given below:

Algorithm for case indexing procedure

Until case is solved do

- Use the ruleset to predict which rule is applicable (only if part of the rule). Let r be the predicted rule.
- Compare the genre suggested by the rule r and of the case.
- If the two genre matches, the case is stored as a positive exemplar, else as a negative exemplar.
- Store the case in the indexed case library and proceed.

Indexing procedure stores each case as a positive or negative exemplar of the predicted rule. The indexing procedure is applied to the case library for all three rulesets independently. These indexed cases are stored in the indexed case library.

3.6 Similarity Measure

Similarity measure is also a major step when we find the similarity between two cases during the process of case-based reasoning. The formula to compute the similarity between two cases C_1 and C_2 is as follows [D. W. Aha 1991]:

$$Similarity(C_1, C_2, P) = \frac{1}{\sqrt{\sum_{i \in P} Feature_dissimilarity(C_{1_i}, C_{2_i})}}$$

Where P is the set of predictor features and

$$Feature_dissimilarity(C_{1_i}, C_{2_i}) = \begin{cases} (C_{1_i} - C_{2_i})^2 & \text{if feature } i\text{'s values are numeric} \\ 0 & \text{if } C_{1_i} = C_{2_i} \\ 1 & \text{otherwise} \end{cases}$$

Thus the similarity can be defined as the inverse of Euclidean distance for numeric features and a simple matching test for symbolic features. To find the feature dissimilarity a slight modification was made for the fourth feature: zip code (Appendix B). The formula for similarity given above was used to calculate the similarity between two cases.

3.7 Proposed Scheme

Our aim was to find the three most preferable genres corresponding to the test case (active user) using the RC-Hybrid procedure given in chapter 2. Once all the three genres have been found, a movie having these three genres and high ratings is recommended by the system. This scheme is explained by the algorithm given below:

Algorithm for the proposed scheme

1. Find the provisional rule for the test case.
2. Consider all the exemplars (e_i) of the provisional rule and corresponding analogical rules a_i .
3. Find the similarity, accuracy and significance of each exemplar (e_i).
4. For each exemplar e_i
 - 4(a). Treat exemplar e_i as a test case
 - 4(b). Find all helpful, harmful and misclassified analogies.
 - 4(c). Find their similarity, accuracy and significance.
 - 4(d). Plot all the analogies on the similarity, accuracy and significance axis independently.
 - 4(e). Similarity threshold ($S1$) = $\left\{ \begin{array}{l} \text{A point on the similarity axis} \\ \text{that minimizes the number} \\ \text{of misclassified analogies} \end{array} \right.$
 - 4(f). Similarity threshold ($S2$) = $\left\{ \begin{array}{l} \text{A point on the similarity} \\ \text{axis that excludes all} \\ \text{harmful analogies} \end{array} \right.$
 - 4(g). Accuracy threshold (A) = $\left\{ \begin{array}{l} \text{A point on the accuracy axis} \\ \text{that minimizes the number} \\ \text{of misclassified analogies} \end{array} \right.$

4(h). Significance threshold (S) = $\left\{ \begin{array}{l} \text{A point on the significance axis} \\ \text{that minimizes the number} \\ \text{of misclassified analogies} \end{array} \right.$

5. Use the thresholds to test the compellingness of each analogy a_i

If analogy is found to compelling

Genre = genre proposed by the CBR.

Else

Genre = genre proposed by the RBR.

6. Repeat steps 1 to 5 for the three rulesets to obtain the three most preferable genres: Genre1, Genre2, Genre3
7. Find the set of all movies having the three genres

The scheme was applied to the MovieLens dataset to provide the recommendation. The results and experiments are given in the next chapter. Also the results of Rule-based and case-based reasoning are compared with the RC-Hybrid approach to generate the three most preferable genres and the RC-Hybrid procedure was found to perform at the best rather than using the rule-based and case-based reasoning alone.

3.8 Collaborative Filtering

From the study of recommendation techniques, we know that the collaborative filtering predicts the ratings of the unseen items. But here the goal of collaborative filtering is not strictly to predict ratings but rather to improve the quality of the recommendations made

by the knowledge-based system. The above defined RC-Hybrid approach provides a set of recommendations. Thus some kind of discrimination strategy is required to give the final recommendation from the set of recommendations. Suppose the system is providing 20 recommendations. The recommendation from this set can be made by randomly selected an item from the recommendation set of 20 items. On the other hand to give a better recommendation we may require some more knowledge about the system so that further discrimination can be made. The requirement of more knowledge adds the problems of more knowledge engineering. Thus the well known collaborative filtering engine can be added to the system to give a better recommendation from the set of top 20 recommendations. Thus collaborative filtering is applied only to the set of recommendations to improve the accuracy of the system.

Research in the field of collaborative filtering suffers from the start-up problems. However the cascading of collaborative filtering to the knowledge engineering technique avoids such drawbacks. The collaborative technique predicts the ratings made by the active user to the items suggested by the knowledge based technique. Any standard collaborative technique can be used to predict the ratings based on the entire collection of previously rated items by the users. But the hybrid approach of recommender system predicts the ratings for those items that were recommended by the knowledge-based. Once the ratings are predicted, the system makes the recommendation of highly rated item, by the knowledge-based technique.

4. Experiments and Results

The MovieLens dataset consisting of 943 users was divided into test and training sets for experiment purpose. The ruleset was formed by taking the top 50 users who rated maximum number of movies. The case library consists of all the users together with their demographic information.

4.1 Experiments on Threshold Setting Procedure

When the RC-Hybrid procedure was applied to the test set, the RBR step proposed a genre from ruleset and the CBR step proposed a genre from analogies. Thresholds were generated to decide a genre, obtained from RBR & CBR. Here we are considering an experiment on the threshold generating procedure. The figure 4-1 given below gives all analogies (Misclassified, Harmful and Helpful) on the similarity axis for a test case. The similarity threshold S_1 is chosen on the similarity axis to minimize the number of misclassified analogies. The second similarity threshold S_2 is chosen to exclude all harmful analogies.

Once the similarity thresholds have been found, all analogies whose similarity scores fall below the threshold S_1 were excluded. The analogies below the threshold S_1 are all unconvincing and offer no information about how to set rest of the thresholds. Therefore each subsequent step of the threshold setting procedure has fewer analogies to process further. The figure 4-2 and figure 4-3 shows the subsequent procedure of choosing accuracy threshold A and significance threshold S . The procedure of choosing the thresholds A and S is same as the procedure of choosing the threshold S_1 (but not as for the threshold S_2).

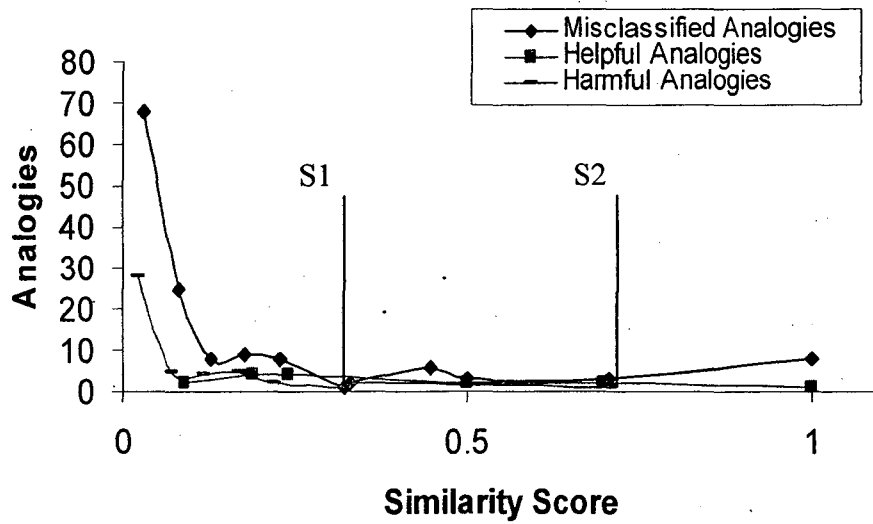


Figure 4-1 Distributions of Analogies by similarity score.

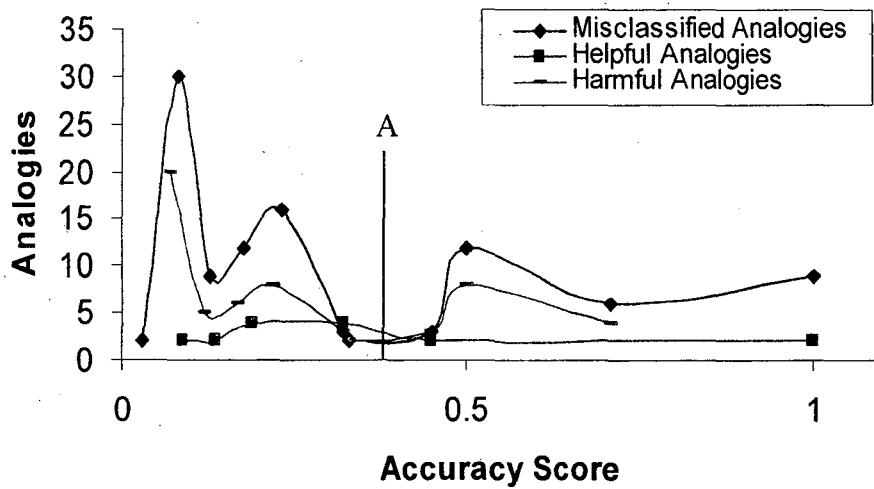


Figure 4-2 Distributions of Analogies by accuracy score.

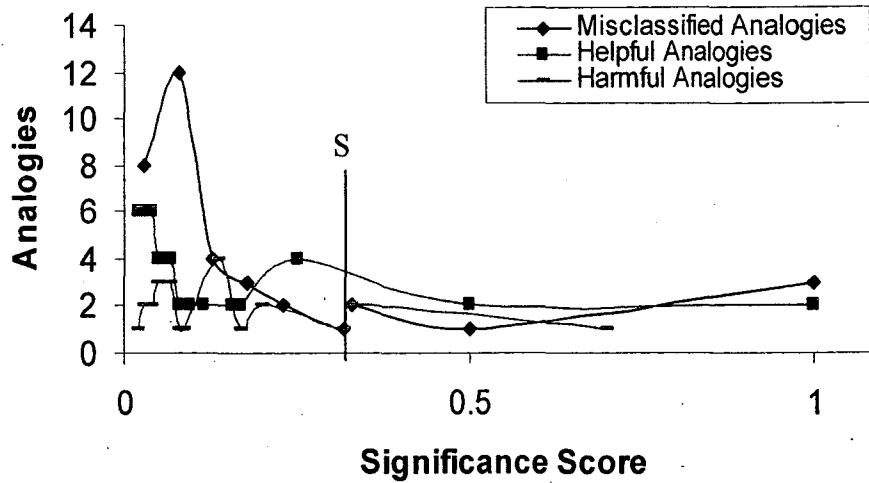


Figure 4-3 Distributions of Analogies by significance score.

4.2 Experiments on RC-Hybrid Procedure

To test the improved performance of RC-Hybrid procedure over the RBR and CBR procedures alone, a randomly chosen test set of 50 active users was considered. The four experiments on this set were as follows:

Experiment 1: The test set was used to predict the first preferred genre.

Experiment 2: The test set was used to predict the second preferred genre.

Experiment 3: The test set was used to predict the third preferred genre.

Experiment 4: The test set was used to predict all the three genres.

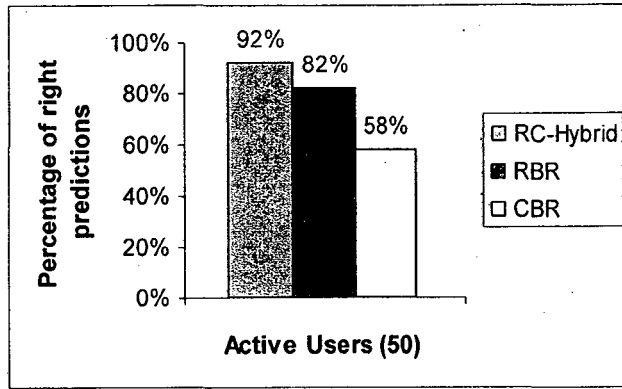


Figure 4-4 Results for experiment 1

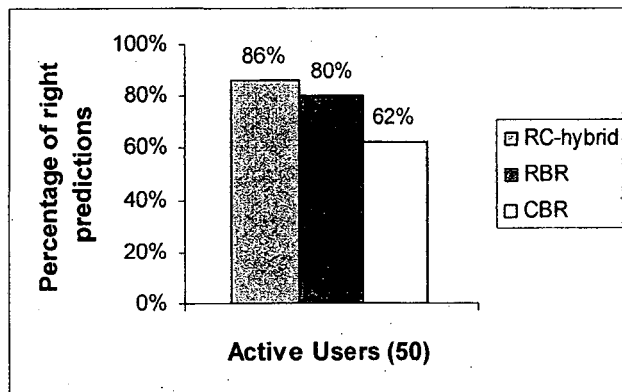


Figure 4-5 Results for experiment 2

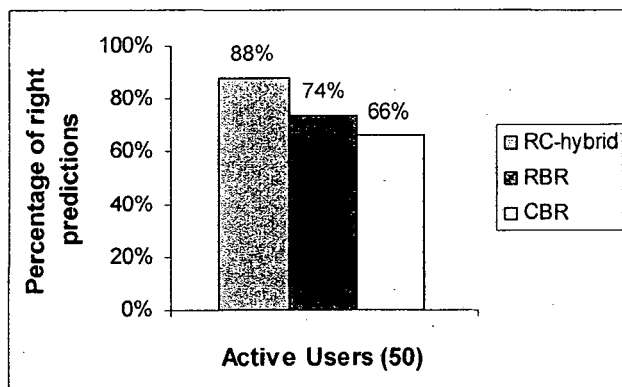


Figure 4-6 Results for experiment 3

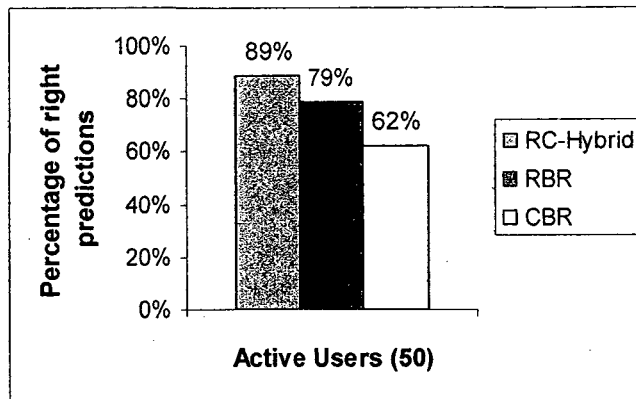


Figure 4-7 Results for experiment 4

4.3 Analysis of Results

Figure 4-4 shows the prediction of first preferred genre by the test set of 50 active users. The accuracy of RC-Hybrid procedure was found 92%, while the RBR alone had the prediction accuracy as 82% and the CBR had the lower one 58% of accuracy. Similarly Figure 4-5 and 4-6 shows the predictions of second preferred genre and third preferred genre respectively. Figure 4-7 gives an average accuracy of predicting all three genres. Figure 4-4, 4-5, 4-6 and 4-7, all shows that the accuracy of prediction can be improved when we use a combined approach of Rule-based reasoning and Case-based reasoning rather than using these techniques alone. Prediction accuracy was increased by using the RC-Hybrid procedure for predicting the genres preferred by the active user.

Conclusion

This dissertation presents a hybrid recommender system using rule-based and case-based reasoning approach to collaborative filtering. In this technique, the combination of Rule-based and Case-based reasoning approach is applied to the task of getting the taste of user (genres) based on their demographic information.

Through the experiments it is established that there is a considerable increase in the accuracy of prediction by taking a combined approach of Rule-based and Case-based reasoning. Clearly the results show that the proposed hybrid approach out performs either of the Rule-based or Case-based reasoning technique alone.

Future work

In the present work we have considered all the features equally important. By adding the feature importance factor to similarity measure, the accuracy of the knowledge-based recommender system can be improved further. Therefore one of the possible extensions would be to take into consideration the weights of the features.

In this dissertation, the main emphasis of the proposed RC-Hybrid knowledge-based recommender system was on getting the taste or preference of the user to give a set of recommendations. Further a discrimination strategy is required to break ties between the set of recommendations provided by the proposed technique and therefore any well-known collaborative technique can be applied to the set of recommendations.

Appendix A

This appendix gives a brief explanation about the software See5, used for constructing the rulesets. See5 is a system that extracts informative patterns from data. See5's job is to find how to predict a case's class from the values of the other attributes. See5 does this by constructing a *classifier* that makes this prediction. See5 can construct classifiers expressed as *decision trees* or as sets of *rules*.

A1.1. Preparing Data for See5

See5 can be illustrated by using a medical application -- mining a database of thyroid assays from the Garvan Institute of Medical Research, Sydney, to construct diagnostic rules for hypothyroidism. Each case concerns a single referral and contains information on the source of the referral, assays requested, patient data, and referring physician's comments. Here are three examples:

<u>Attribute</u>	<u>Case 1</u>	<u>Case 2</u>	<u>Case 3</u>
age	41	23	46	
sex	F	F	M	
on thyroxine	f	f	f	
query on thyroxine	f	f	f	
on antithyroid medication	f	f	f	
sick	f	f	f	
pregnant	f	f	not applicable	
thyroid surgery	f	f	f	
I131 treatment	f	f	f	
query hypothyroid	f	f	f	
query hyperthyroid	f	f	f	

lithium	f	f	f
tumor	f	f	f
goitre	f	f	f
hypopituitary	f	f	f
psych	f	f	f
TSH	1.3	4.1	0.98
T3	2.5	2	unknown
TT4	125	102	109
T4U	1.14	unknown	0.91
FTI	109	unknown	unknown
referral source	SVHC	other	other
diagnosis	negative	negative	negative
ID	3733	1442	2965

This is exactly the sort of task for which See5 was designed. Each case belongs to one of a small number of mutually exclusive classes (negative, primary, secondary, compensated). Properties of every case that *may* be relevant to its class are provided, although some cases may have unknown or non-applicable values for some attributes. There are 24 attributes in this example, but See5 can deal with any number of attributes.

A1.2. Application files

Every See5 application has a short name called a filestem; the filestem “hypothyroid” will be used for this illustration. All files read or written by See5 for an application have names of the form filestem.extension, where filestem identifies the application and extension describes the contents of the file. Here is a summary table of the extensions used by See5:

names	description of the application's attributes	[required]
data	cases used to generate a classifier	[required]
test	unseen cases used to test a classifier	[optional]
cases	cases to be classified subsequently	[optional]
costs	differential misclassification costs	[optional]
tree	Decision tree classifier produced by See5	[output]
rules	ruleset classifier produced by See5	[output]
out	report produced when a classifier is generated	[output]
set	settings used for the last classifier	[output]

The case of letters in both the filestem and extension is important -- file names APP.DATA, app.data, and App.Data, are all different. The extensions must be written in lower case as shown above, otherwise See5 will not recognize the files for the application.

A1.3. Names file

Two files are essential for all See5 applications and there are three further optional files, each identified by its extension. The first essential file is the names file (e.g. hypothyroid.names) that describes the attributes and classes. There are two important subgroups of attributes:

- The value of an explicitly-defined attribute is given directly in the data. A *discrete* attribute has a value drawn from a set of nominal values, a *continuous* attribute

has a numeric value, a *date* attribute holds a calendar date, a *time* attribute holds a clock time, a *timestamp* attribute holds a date and time, and a *label* attribute serves only to identify a particular case.

- The value of an implicitly-defined attribute is specified by a formula. (Most attributes are explicitly defined, so we may never need implicitly-defined attributes.)

The file `hypothyroid.names` looks like this:

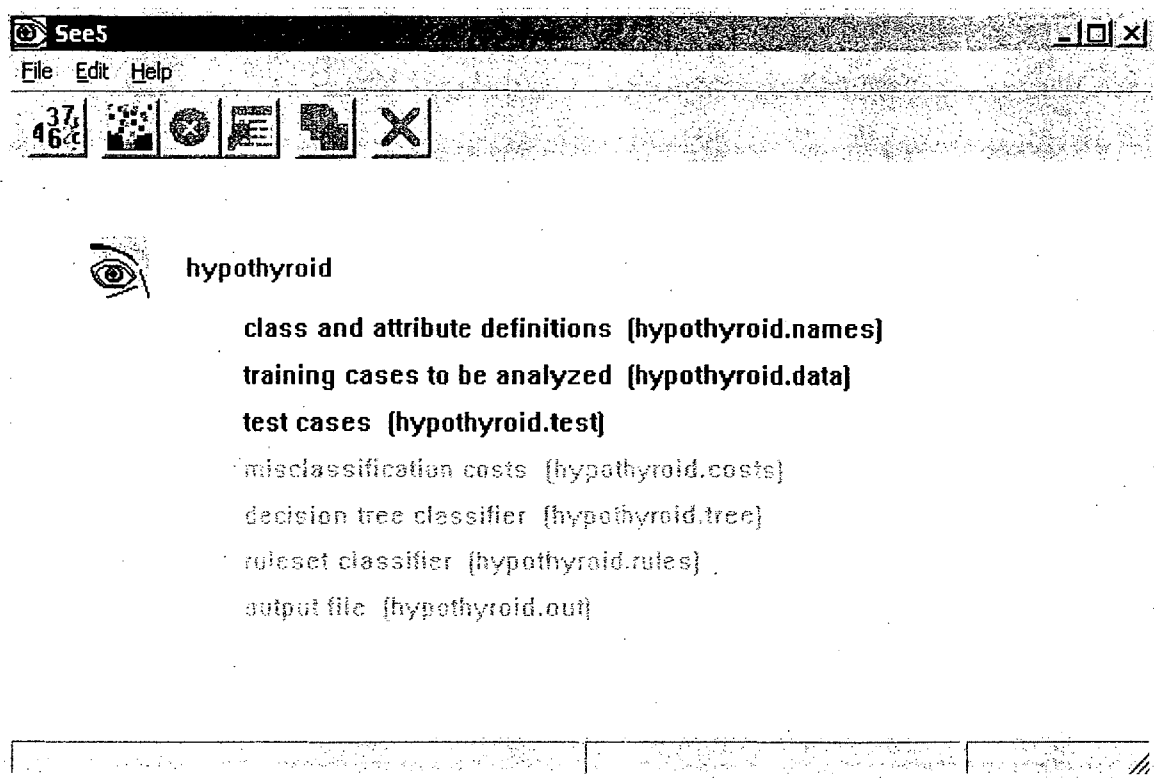
diagnosis.	the target attribute
age:	continuous.
sex:	M, F.
on thyroxine:	f, t.
query on thyroxine:	f, t.
on antithyroid medication:	f, t.
sick:	f, t.
pregnant:	f, t.
thyroid surgery:	f, t.
I131 treatment:	f, t.
query hypothyroid:	f, t.
query hyperthyroid:	f, t.
lithium:	f, t.
tumor:	f, t.
goitre:	f, t.
hypopituitary:	f, t.
psych:	f, t.
TSH:	continuous.
T3:	continuous.
TT4:	continuous.
T4U:	continuous.
FTI:=	TT4 / T4U.
referral source:	WEST, STMW, SVHC, SVI, SVHD, other.
diagnosis:	primary, compensated, secondary, negative.
ID:	label.

A1.6. Costs file (optional)

The last kind of file the costs files (e.g. hypothyroid.costs), is also optional and sets out differential misclassification costs. In some applications there is a much higher penalty for certain types of mistakes. In this application, a prediction that hypothyroidism is not present could be very costly if in fact it is. On the other hand, predicting incorrectly that a patient is hypothyroid may be a less serious error. See5 allows different misclassification costs to be associated with each combination of real class and predicted class.

A1.7. User Interface

As a simple illustration, here is the main window of See5 after the hypothyroid application has been selected.



The main window of See5 has six buttons on its toolbar. From left to right, they are

Locate Data invokes a browser to find the files for the application, or to change the current application;

Construct Classifier selects the type of classifier to be constructed and sets other options;

Stop interrupts the classifier-generating process;

Review Output re-displays the output from the last classifier construction (if any);

Use Classifier interactively applies the current classifier to one or more cases; and

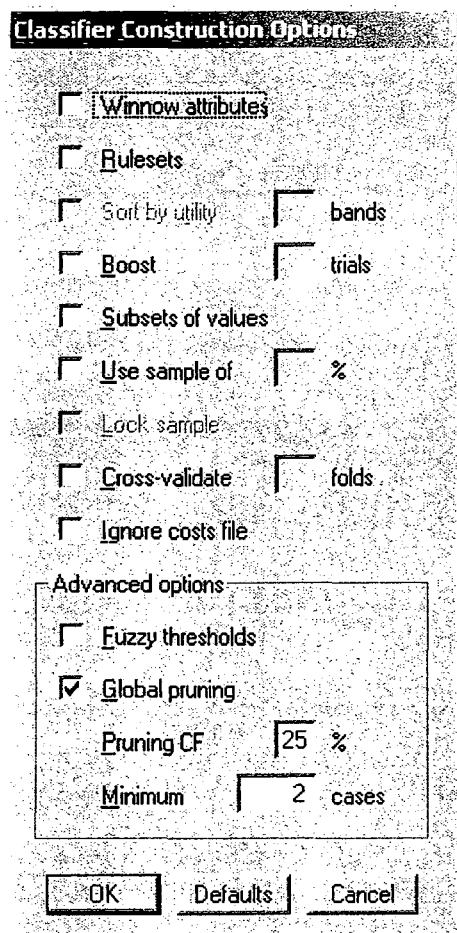
Cross-Reference shows how cases in training or test data relate to (parts of) a classifier and vice versa.

These functions can also be initiated from the File menu.

The Edit menu facilities changes to the names and costs files after an application's files have been located. On-line help is available through the Help menu.

A1.8. Constructing Classifiers

Once the names, data, and optional files have been set up, everything is ready to use See5. The first step is to locate the data using the Locate Data button on the toolbar (or the corresponding selection from the File menu). Assume that the hypothyroid data above has been located in this manner. There are several options that affect the type of classifier that See5 produces and the way that it is constructed. The Construct Classifier button on the toolbar (or selection from the File menu) displays a dialog box that sets out these classifier construction options:



Many of the options have default values that should be satisfactory for most applications.

A1.9. Rulesets

Decision trees can sometimes be quite difficult to understand. An important feature of See5 is its ability to generate classifiers called *rulesets* that consist of unordered collections of (relatively) simple if-then rules. The Rulesets option causes classifiers to be expressed as rulesets rather than decision trees, here giving the following rules:

```

Rule 1: (31, lift 42.7)
  thyroid surgery = f
  TSH > 6
  TT4 <= 37
  -> class primary [0.970]

```

```

Rule 2: (63/6, lift 39.3)
    TSH > 6
    FTI <= 65
    -> class primary [0.892]

Rule 3: (270/116, lift 10.3)
    TSH > 6
    -> class compensated [0.570]

Rule 4: (2225/2, lift 1.1)
    TSH <= 6
    -> class negative [0.999]

Rule 5: (296, lift 1.1)
    on thyroxine = t
    FTI > 65
    -> class negative [0.997]

Rule 6: (240, lift 1.1)
    TT4 > 153
    -> class negative [0.996]

Rule 7: (29, lift 1.1)
    thyroid surgery = t
    FTI > 65
    -> class negative [0.968]

Default class: negative

```

Each rule consists of:

- A rule number -- this is quite arbitrary and serves only to identify the rule.
- Statistics $(n, \text{lift } x)$ or $(n/m, \text{lift } x)$ that summarize the performance of the rule. Similarly to a leaf, n is the number of training cases covered by the rule and m , if it appears, shows how many of them do not belong to the class predicted by the rule. The rule's accuracy is estimated by the Laplace ratio $(n-m+1)/(n+2)$. The lift x is the result of dividing the rule's estimated accuracy by the relative frequency of the predicted class in the training set
- One or more conditions that must all be satisfied if the rule is to be applicable.
- A class predicted by the rule.

- A value between 0 and 1 that indicates the confidence with which this prediction is made. (Note: If boosting is used, this confidence is measured using an artificial weighting of the training cases and so does not reflect the accuracy of the rule.)

There is also a *default class*, here *negative*, that is used when none of the rules apply.

Rulesets are generally easier to understand than trees since each rule describes a specific context associated with a class. Furthermore, a ruleset generated from a tree usually has fewer rules than the tree has leaves, another plus for comprehensibility. (In this example; the first decision tree with 14 leaves is reduced to seven rules.) Finally, rules are often more accurate predictors than decision trees -- a point not illustrated here, since the ruleset has an error rate of 0.5% on the test cases. For very large datasets, however, generating rules with the Ruleset option can require considerably more computer time.

Appendix B

To construct the rulesets, Zip-code representation was converted into the text format as given in the following table:

First digit of Zip-Code	Text Representation
0	A
1	B
2	C
3	D
4	E
5	F
6	G
7	H
8	I
9	J

Table B1- Representation of Zip-Codes

The rulesets are generated using the See5 software for the top 50 users who made the ratings of maximum numbers of movies (items). The sample of the ruleset generated for the first preferred genre is given below:

```
Rule 1: (1, lift 3.2)
  Age > 39
  Occupation = engineer
  -> class Action [0.667]
```

```
Rule 2: (1, lift 3.2)
  Age <= 27
  Occupation = none
  -> class Action [0.667]
```

Rule 3: (1, lift 3.2)
Age <= 27
Zip-code = A
-> class Action [0.667]

Rule 4: (1, lift 3.2)
Occupation = student
Zip-code = E
-> class Action [0.667]

Rule 5: (4/1, lift 3.2)
Age > 21
Age <= 25
-> class Action [0.667]

Rule 6: (1, lift 16.0)
Occupation = educator
Zip-code = E
-> class Adventure [0.667]

Rule 7: (1, lift 16.0)
Occupation = other
Zip-code = F
-> class Adventure [0.667]

Rule 8: (1, lift 32.0)
Occupation = writer
Zip-code = I
-> class Children [0.667]

Rule 9: (1, lift 4.0)
Occupation = administrator
Zip-code = I
-> class Comedy [0.667]

Rule 10: (1, lift 4.0)
Occupation = programmer
-> class Comedy [0.667]

Rule 11: (1, lift 4.0)
Age <= 19
Gender = F
Occupation = student
-> class Comedy [0.667]

Rule 12: (18/12, lift 2.1)
Occupation = student
-> class Comedy [0.350]

Rule 13: (4, lift 2.5)
Occupation in {executive, librarian, retired}
-> class Drama [0.833]

Rule 14: (4, lift 2.5)
Occupation = educator
Zip-code in {A, B, C, H}
-> class Drama [0.833]

References

- Gediminas Adomavicius, Alexander Tuzhilin, (2005) “*Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions*”, IEEE Trans. on Knowledge and Data Engineering, vol. 17, no. 6.
- Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen and Alexander Tuzhilin (2005), “*Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach*”, ACM Transactions on Information Systems, Vol. 23, No. 1, pp. 103–145.
- G. Linden, B. Smith, and J. York (2003), “*Amazon.com Recommendations: Item-to-Item Collaborative Filtering*,” IEEE Internet Computing, pp. 76-80.
- B.N. Miller, I. Albert, S.K. Lam, J.A. Konstan, and J. Riedl (2003), “*MovieLens Unplugged: Experiences with an Occasionally Connected Recommender System*”, Proc. Int’l Conf. Intelligent User Interfaces, pp. 263-266.
- D. Billsus, C.A. Brunk, C. Evans, B. Gladish, and M. Pazzani (2002), “*Adaptive Interfaces for Ubiquitous Web Access*”, Comm. ACM, vol. 45, no. 5, pp. 34-38.
- Robin Burke, (2002), “*Hybrid Recommender Systems: Survey and Experiments*”, User Modeling and User-Adapted Interaction 12: pp. 331-370.
- N. Belkin and B. Croft (1992), “*Information Filtering and Information Retrieval*”, Comm. ACM, vol. 35, no. 12, pp. 29-37.
- M. Balabanovic and Y. Shoham (1997), “*Fab: Content-Based, Collaborative Recommendation*”, Comm. ACM, vol. 40, no. 3, pp. 66-72.
- M. Pazzani and D. Billsus (1997), “*Learning and Revising User Profiles: The Identification of Interesting Web Sites*”, Machine Learning, vol. 27, pp. 313-331.
- T. Joachims. (1997), “*A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization*”, In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 143–151, San Francisco, CA. Morgan Kaufman.
- J. S. Breese, D. Heckerman and C. Kadie (1998), “*Empirical Analysis of Predictive Algorithms for Collaborative Filtering*”. Proc. 14th Conf. Uncertainty in Artificial Intelligence.
- M. J. Pazzani (1999), “*A Framework for Collaborative, Content-Based and Demographic Filtering*”. Artificial Intelligence Review, 13 (5/6), 393-408.

R. Burke (2000), "*A Case-Based Approach to Collaborative Filtering*", E. Blanzieri and L. Portinale (Eds), pp. 370–379.

E. Rich (1979), "*User Modeling via Stereotypes*", *Cognitive Science*, vol. 3, no. 4, pp. 329-354.

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

B. Towle and C. Quinn (2000), "*Knowledge Based Recommender Systems Using Explicit User Model*". In *Knowledge-Based Electronic Markets, Papers from the AAAI Workshop*, AAAI Technical Report WS-00-04. pp. 74-77. Menlo Park, CA: AAAI Press.

M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes and M. Sartin (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>

T. Tran and R. Cohen (2000), "*Hybrid Recommender Systems for Electronic Commerce*". In *Knowledge-Based Electronic Markets, Papers from the AAAI Workshop*, AAAI Technical Report WS-00-04. pp. 78-83. Menlo Park, CA: AAAI Press.

B. Smyth and P. Cotter (2000), "*A Personalized TV Listings Service for the Digital TV Age*", *Knowledge-Based Systems* 13: pp. 53-59.

A. M. Wasfi (1999), "*Collecting User Access Patterns for Building User Profiles and Collaborative Filtering*". In: *IUI '99: Proceedings of the 1999 International Conference on Intelligent User Interfaces*, Redondo Beach, CA, pp. 57-64.

R. Burke, K. Hammond and B. Young (1997), "*The FindMe Approach to Assisted Browsing*". *IEEE Expert*, 12 (4), pp. 32-40.

C. Basu, H. Hirsh and W. Cohen (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.

B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller and J. Riedl (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.

Joseph Giarratano & Gary Riley (1993), *Expert Systems*, PWS Publication.

Elaine Rich & Kevin Knight (1999), *Artificial Intelligence*, TMH Publication.

Michael M. Richter and Agnar Aamodt (2006), "*Case-based reasoning foundations*". The Knowledge Engineering Review, Vol. 20:3, pp. 203–207.

Watson (1999), "*Case-based reasoning is a methodology not a technology*", Knowledge-Based Systems 12, pp. 303–308

Andrew R. Golding, Paul S. Rosenbloom, (1996), "*Improving accuracy by combining rule-based and case-based reasoning*", Artificial Intelligence 87, pp. 215-254.

David Mcsherry, Springer (2005), "*Explanation in Recommender Systems*", Artificial Intelligence Review, 24: pp. 179–197.

Nick Cercone, Senior Member, IEEE, Aijun An, and Christine Chan (1999), "*Rule-Induction and Case-Based Reasoning: Hybrid Architectures Appear Advantageous*", IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 1, pp. 166-174.

Riccardo Bellazzi, Stefania Montani, Luigi Portinale, Alberto Riva (1999), "*Integrating Rule-Based and Case-Based Decision Making in Diabetic Patient Management*", In Proc. ICCBR-99, LNAI 1650, pp. 386-400. Springer-Verlag, 1999.

D. W. Aha (1991), "*Case-Based Learning Algorithm*", In Proceedings of the DARPA Case-Based Reasoning Workshop.