THE DESIGN AND STUDY OF PEDAGOGICAL PAPER RECOMMENDATION

A Thesis Submitted to the College of

Graduate Studies and Research

In Partial Fulfillment of the Requirements

For the Degree of Doctor of Philosophy

In the Department of Computer Science

University of Saskatchewan

Saskatoon

By

TIFFANY YA TANG

PERMISSION TO USE

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science

University of Saskatchewan

Saskatoon, Saskatchewan, S7N 5C9

ABSTRACT

For learners engaging in senior-level courses, tutors in many cases would like to pick some articles as supplementary reading materials for them each week. Unlike researchers ‘Googling’ papers from the Internet, tutors, when making recommendations, should consider course syllabus and their assessment of learners along many dimensions. As such, simply ‘Googling’ articles from the Internet is far from enough. That is, learner models of each individual, including their learning interest, knowledge, goals, etc. should be considered when making paper recommendations, since the recommendation should be carried out so as to ensure that the suitability of a paper for a learner is calculated as the summation of the fitness of the appropriateness of it to help the learner in general. This type of the recommendation is called a Pedagogical Paper Recommender.

In this thesis, we propose a set of recommendation methods for a Pedagogical Paper Recommender and study the various important issues surrounding it. Experimental studies confirm that making recommendations to learners in social learning environments is not the same as making recommendation to users in commercial environments such as Amazon.com. In such learning environments, learners are willing to accept items that are not interesting, yet meet their learning goals in some way or another; learners’ overall impression towards each paper is not solely dependent on the interestingness of the paper, but also other factors, such as the degree to which the paper can help to meet their ‘cognitive’ goals.

It is also observed that most of the recommendation methods are scalable. Although the degree of this scalability is still unclear, we conjecture that those methods are consistent to up to 50 papers in terms of recommendation accuracy.

The experiments conducted so far and suggestions made on the adoption of recommendation methods are based on the data we have collected during one semester of a course. Therefore, the generality of results needs to undergo further validation before more certain conclusion can be drawn. These follow up studies should be performed (ideally) in more semesters on the same course or related courses with more newly added papers. Then, some open issues can be further investigated.

Despite these weaknesses, this study has been able to reach the research goals set out in the proposed pedagogical paper recommender which, although sounding intuitive, unfortunately has been largely ignored in the research community.

Finding a ‘good’ paper is not trivial: it is not about the simple fact that the user will either accept the recommended items, or not; rather, it is a multiple step process that typically entails the users navigating the paper collections, understanding the recommended items, seeing what others like/dislike, and making decisions. Therefore, a future research goal to proceed from the study here is to design for different kinds of social navigation in order to study their respective impacts on user behavior, and how over time, user behavior feeds back to influence the system performance.

ACKNOWLEDGMENTS

I am so deeply grateful for my supervisor Prof. Gordon McCalla for his encouragement, support, openness, guidance and inspiration throughout the years during my PhD studies, particularly after my return to Hong Kong five years ago. Without his numerous e-efforts and e-understandings, I would never been able to finish writing this thesis.

Thanks also go to my advisory Committee members, whose constructive suggestions have helped me reshape this thesis.

The efforts and support from Jan, Heather and Linda are greatly appreciated.

The financial support from the Canadian NSERC research grants is greatly acknowledged for the research documented in this thesis.

The work discussed in this thesis has also been inspired by many valuable comments and suggestions from those people whom I met, particularly former and current ARIES-ers Ben and Chris, and a lot more whom I have not when my early work were submitted for conference and journals. Thanks also go to my best friends Xiaolin, Golha, Yongli, Zhaitao and Yamini who are always there to support me.

My son, Kevin has always been there to accompany me, especially during the many boring and rather hot summer holidays when he has to stay in the office with me: I am busy with the thesis, and he has to stay busy with ‘Googling’ more information on the Black Hole! The patience and love from my husband Pinata have also been able to pull me through during those long and painful nights when I had to stay awake after a full-day of work and push me to concentrate on the thesis. And, both of them have always tolerated me for being such a lousy cooker.

Finally, not the least, I would like to dedicate this thesis to my parents and my grandma who have always believed in me ever since I was a little girl, and kept calling me to check when I can finish writing the thesis and graduate. Without their unselfishness, I will never been able to find myself being capable of doing something that I am doing today.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Paper Recommendation in Active Teaching/Learning

When information overload intensifies, users are overwhelmed by the information pouring out from various sources including the Internet, and are usually confused by which information should be consumed; that is, users find it difficult to pick something appropriate to them when the number of choices increases. Fortunately, a recommender system offers a feasible solution to this problem. For example, if a user explicitly indicates that he/she favors action movies starring *Sean Penn*, then he/she could be recommended movies like *The Interpreter*. In this case, the system is able to match user preferences to content features of the movies, which is a content-based filtering approach. In another major recommendation approach called collaborative filtering, the system constructs a group of like-minded users with whom the target user shares similar interests and makes recommendations based on an analysis of them.

For learners engaging in senior-level courses, tutors in many cases would like to pick some articles as supplementary reading materials for them each week. Unlike researchers 'Googling' research papers matching their interests from the Internet, tutors, when making recommendations, should consider the course syllabus and their assessment of learners along many dimensions. As such, simply Googling articles from the Internet is far from enough. Suppose, a paper recommender system can carefully assess and compare both learner and candidate paper characteristics (through instructions from tutors), and make recommendations

accordingly. In other words, learner models of each individual, including their learning interest, knowledge, goals, etc., will be created. Models of the articles or papers will also be created based on the topic, degree of peer recommendation, etc. The recommendation is carried out by matching the learner characteristics with the paper topics to achieve appropriate pedagogical goals such as 'the technical level of the paper should not impede the learner in understanding it'. Therefore, the suitability of a paper for a learner is calculated in terms of the appropriateness of it to help the learner in general. This type of recommendation system is called a Pedagogical Paper Recommender.

1.2 Examples of Pedagogical Recommendation

Let us consider an application in which the goal is to make paper recommendations to adult learners with various backgrounds in a Software Engineering course. The characteristics that we should take into consideration include:

- Paper: each paper is defined by some main attributes or metadata such as Paper ID (acting as a primary key for each paper), Paper Title, Paper Authors, Paper Publication Place (including conference name, or journal name, or workshop name where the paper was published), Paper Publication Year, and Number of Pages.
- Learner: a learner model represents learners that have registered in the course and for whom papers are recommended; defined by Learner ID (primary key)[1]. Other information about a learner includes:

[1] Since the experiments in this thesis are conducted based on anonymous learners who signed the consent form, they cannot be identified. Instead, we will use a unique learner ID to represent them when necessary.

1. Learner Interest: represented in terms of the main topics defined in the course syllabus, such as software requirements engineering, software process, software project management, user interface design, software testing, web design and web engineering, case studies, etc.
2. Learner Goal: represents any specific learning goals the learner might have. Learner goals consist of a single attribute having values "software engineering in general", "any specific software engineering areas (students can specify)" and "others".
3. Job Experience: represents each learner's current job. In our implementation, learners are allowed to specify their job nature, which can include, programming, academic teaching, management, etc.
4. Background Knowledge: represents some knowledge items necessary for the learners to read some of papers with a certain level of technical difficulty. For example, some papers discuss statistical studies on small to medium sized software project management, which requires learners to have some knowledge about t-tests, standard deviation etc. Note that the fact that we should consider learners' background knowledge in making recommendations is one way our recommender system is differentiated from other systems.

Based on the above characteristics, the rating assigned to a paper by a learner depends not only on his/her interest (as commonly adopted in the majority of recommender systems), but also his/her goals, job experience and knowledge background. For example, the type of paper to recommend to learner Alex can differ significantly depending on whether or not he has enough background in statistics.

To better explain this, let us consider some examples.

Example 1.

Bob, a part time student, is now a Project Manager in a Hong Kong-based software company. He has rich hands-on knowledge of and experience in Programming, but his Mathematical Knowledge is poor. His interest in the course is to gain some knowledge in User Interface Design. Based on these "pedagogical" features, we recommend the following two papers to Bob to read as an assignment:

- The Windows ® UI: A Case Study in Usability Engineering, Kent Sullivan, Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI' 1996), 473-480, ACM Press, 1996.
- The Usability Engineering Life Cycle, Jakob Nielsen, IEEE Computer, Volume 25, No.3, 12-22, March 1992, IEEE Press, 1992.

The above papers focus more on the application/programming oriented part of the design of the user interface and its design process in the general software engineering lifecycle. Note here that the second paper has been regarded as a core paper in the usability engineering sub-area of software engineering, and its author Dr. Jakob Nielsen is one of the pioneers in the field. Hence, this paper might appear in many learners' recommended list. ■

Example 2.

Michelle, a part time student, is now a Junior Researcher with a research institute in a university in Hong Kong. Her Mathematical Skills are excellent, though she lacks hands-on programming knowledge. Her interest in the course is to gain general knowledge in Software Engineering. Based on these "pedagogical" features, we recommend reading the following three papers to her as an assignment:

- Blending CMM and Six Sigma to Meet Business Goals, Mala Murugappan and Gargi Keeni, IEEE Software, 42-48, March/April 2003, IEEE Press, 2003.
- How Software Engineers Use Documentation: The State of the Practice, Timothy Lethbridge, Janice Singer and Andrew Forward, IEEE Software, November/December 2003, 35-39, IEEE Press, 2003.
- The Usability Engineering Life Cycle, Jakob Nielsen, IEEE Computer, Volume 25, No.3, 12-22, March 1992, IEEE Press, 1992.

The third paper is recommended to Michelle for her to gain a general understanding of the topic of Usability Testing/Engineering in Software Engineering, while the other two are recommended due to their discussions of Requirements Engineering and Software Quality. The full understanding and appreciation of the first paper depends on the readers' mathematical skills, which Michelle has. ■

1.3 The Special Features of Pedagogical Paper Recommendations

A paper recommendation system for learners differs from other recommendation systems in at least three ways [Tang and McCalla 2005a, Tang and McCalla 2004]:

1. The first is that in an e-learning context, there is a course curriculum that helps to inform the system. Since pure collaborative filtering may not be appropriate because it needs a large number of ratings (sparsity issue), the availability of a curriculum allows the deployment of a hybrid technique, partly relying on curriculum-based paper annotations.
2. The second difference is the pedagogical issue. Beyond learner interests, there are multiple learner characteristics that should be considered in recommending learning material. For example, if a learner states that his/her interest is in Internet Computing,

then recommending only the highly cited/rated papers in this area is not sufficient, because the learner may not be able to understand such papers if their technical background is inappropriate.

3. The third difference concerns the evaluation strategies. Recommender systems are decision-support systems; thus, it is believed that measuring whether the recommendation mechanisms did assist users in selecting papers and support them pedagogically is more appropriate than measuring the error between the actual and predicted ratings. Thus, besides testing the 'accuracy' of the recommendations, assessments should be performed to determine the extent to which learners are satisfied pedagogically.

Other than these three major features, there are some additional ones. For instance, it is often mandatory for a learner to provide feedback including numerical ratings on various aspect of the paper after reading it; while in other domains, users voluntarily tell the system their ratings on the item (for example, a movie). Also, the order with which papers are recommended matters a lot, that is, a tutor might suggest a list of ordered reading [Tang and McCalla 2005b], which is different from recommendation in e-commerce, where site managers prefer to leave the list unordered to avoid leaving the impression that a specific recommendation is the best choice [Schafer et al. 2001].

These features characterize pedagogical paper recommendation and differentiate it from recommendation in other domains.

1.4 Thesis Objectives and Contribution

The objective of this thesis and its contribution are based on some issues related to pedagogical paper recommendation. In the following subsection we will discuss these issues.

1.4.1 Issues in Pedagogical Paper Recommendation

Since we have relatively little understanding of pedagogical paper recommendation, there are a number of challenging issues that need to be studied, among them:

1. Other than learner interest, which aspect(s) will influence the overall rating given by a learner to a paper? For instance, does new information/knowledge contained in the paper (Value-addedness) affect the overall rating? Or, does the helpfulness of reading the paper to aid learning matter more?
2. Since we need to consider multiple aspects related to both learners and papers during the recommendation process, as well as the limited number of available ratings by the learners, is traditional recommendation based on the overall rating still suitable? Are there more appropriate recommendation techniques?
3. Among the possible recommendation techniques (including our proposed ones) an understanding of their performance in a pedagogical context is essential to answer questions such as which technique(s) would be the most suitable one(s) under a given learning scenario to generate appropriate recommendations?
4. The suitability of the various recommendation techniques must be explored as to their effect on learners. This may result in an over-fitting problem. Over-fitting happens when parameter optimization on a model fits a training dataset yet biases the prediction on a separate testing dataset. Therefore, we wonder whether doing parameter optimization on possible recommendation techniques is worthy or not. If yes, what are the most appropriate parameters in order to avoid the over-fitting problem? If not, what are the most appropriate parameters in order to yield satisfying results?

5. What would be the real user acceptance after using the paper recommender? Are users satisfied with the performance of the system? How would the system adjust the recommendation strategies based on user feedback? When both new papers and users keep coming in to use the system, how would the system respond to maintain its performance?
6. Since the paper recommender is a social system involving users (including tutors and learners), papers, and user interactions with both the system and the paper, how would the system be designed to enhance users' paper-seeking activities in the information space? Two key issues related to the user interface and interaction design of the system are:
    a. Users should be able to know how the recommendation is made (the transparency issue).
    b. Users should easily be able to discern the popularity of the paper in terms of its pedagogical features (say, the degree of the paper's usefulness or technical difficulty) and make decisions accordingly.

The above issues are not exhaustive. This thesis, to a large extent, will only address the first three issues, slightly discuss the fourth and the fifth issue, and leave the rest for our future work.

1.4.2 Thesis Objectives

In this thesis, we will study some of the above issues. In particular, this thesis proposes a pedagogical paper recommender system and explores its characteristics.

First, we will study some pedagogical factors that may affect learner overall satisfaction with a paper in terms of an overall rating. Then, we will also propose a set of recommendation techniques that are unique in this domain, and look into their effects on learners. After that, we

will compare and evaluate these techniques using real data collected from our human subject experiments. Our experiments and evaluation strategies are also designed to explore the hypothesis that '*accurate recommendations alone do not guarantee users of recommender systems an effective and satisfying experience. Instead, systems are useful to the extent that they help users complete their tasks*' [Herlocker et al. 2004]. In particular, we will describe an evaluation framework that takes into account user acceptance as well as overall accuracy.

Finally, we will show a prototypical recommender system which contains all the proposed algorithms. We will show the usage guideline of this prototype under various conditions, including a re-evaluation strategy and an ensemble method when more than one recommendation method is available. In the fully implemented system, this prototype may not be shown to the learner but to the tutors or system administrators only.

1.4.3 Thesis Contributions

This thesis makes the following contributions:

1. We identify the uniqueness of pedagogical paper recommendation and make comparison with other recommendation domains.
2. We put forward a set of pedagogical paper recommendation techniques, including pure content-based filtering, multi-dimensional collaborative filtering, and a user-model based filtering technique that makes use of both user models and paper models to start up the recommender system.
3. Through a number of structural statistical analysis methods including partial correlation, structural equation model, principal components regression and partial least square

regression analysis, we determine what aspect(s) of the paper matter the most in the paper recommender based on the ratings provided by learners.

4. Through experimental studies, we test the performance of the recommendation techniques based on data collected from a one-semester course on Software Engineering.
5. Based on the experimental studies, we create a prototype system that adopts the recommendation techniques and show how to use them appropriately in a range of pedagogical contexts.
6. Although our study is based on data collected in one course, we suggest how to generalize the recommendation techniques to other courses.

The problem of making pedagogical paper recommendation can be of interest to researchers in areas such as artificial intelligence in education, user modeling, recommender systems, social network analysis etc. Hence, this thesis will contribute to these research communities.

1.5 Thesis Organization

This thesis is organized as follows. In the next chapter, we will start by presenting a literature review on recommender systems, including design issues and evaluation strategies. Prior research on paper recommendation will also be included in order to position our work in this context, as well as differentiate our work from that of other researchers in the area. In Chapter 3, through statistical analysis, we attempt to correlate the relationships between learner interest, learner perception of paper quality, and learner characteristics. In Chapter 4, we propose and explain in detail a set of recommendation strategies, some of which are unique to our domain. The analysis of the performance of these recommendation strategies is discussed. In Chapter 5,

the system prototype is described along with some typical scenarios of going through the system with an aim to gain an overview of the general flow of the system. In Chapter 6, we discuss next steps following up on the studies in this thesis.

CHAPTER 2
RECOMMENDER SYSTEMS: LITERATURE SURVEY

Recommender systems (RSs) can roughly be divided into three major types based on the techniques used: Collaborative Filtering (CF), content-based filtering and hybrid filtering ([Jameson et al. 2002; Adomavicius and Tuzhilin 2005; Tang et al. 2005]). Some researchers have added a fourth major type called 'knowledge-based filtering' or 'conversational' [Burke 2002; Smyth et al. 2004]. In this thesis, we will discuss all of the four major types of RSs. Regardless of approach, the key idea is personalization of the recommendation and at the core of personalization is the task of building a model of the user. Content-based approaches build user models that link the contents of the information a user has consumed about the artifacts to be recommended to the preferences of the user concerning those artifacts; CF approaches build user models that link the information preferences of a user to those of other users with similar tastes or preferences; hybrid approaches use a mixture of CF and content-based modeling; and knowledge-based approaches construct user profiles more gradually using many 'interactive' forms of knowledge structure. In all approaches, the success of the item recommended is represented by the *utility* of the item, usually captured by a rating specified by the user based on how much the user liked the item [Adomavicius and Tuzhilin 2005].

2.1 Recommendation Techniques

2.1.1 Content-based Personalized Recommendation

As its name signifies, content-based approaches recommend items based on the contents of the items a user has experienced before. Obviously, to ensure 'high-quality' recommendations, the system should conduct a rather delicate analysis on the content features of the target item (the item to be recommended) in an attempt to establish the relationship between what the user likes and the target item.

2.1.1.1 How the Content-based Approach Works

Generally, the content-based recommendation approach has its roots in *information retrieval* (IR) [Baeza-Yates and Ribeiro-Neto 1999; Salton 1989; Kobayashi and Takeda 2000] and *information filtering* [Belkin and Croft 1992] approaches. Thanks to the significant advances made by IR and information filtering researchers, the majority of current content-based techniques are able to associate the content aspect of items such as books, movies, documents, news articles etc, with the elements that are the most probably attractive to users (see among others, [Woodruff et al. 2000; Melville et al. 2002; Hofmann et al. 1999; Torres et al. 2004; Basu et al. 2001; Ziegler et al. 2005]). Content-based filtering in RSs not only utilizes the content aspect of the items but also *user profiles* that contain information (usually textual) about users' tastes, preferences, etc. And the user profile models are normally constructed *explicitly* from users' own specified keywords from a list of pre-defined keywords on a topic; or implicitly from the system's long-term observations of user behaviors [Woodruff et al. 2000; Melville et al. 2002; Hofmann et al. 1999; Ziegler et al. 2005].

It is clear that content-based approaches are designed mostly to recommend items with easy-to-obtain or easy-to-elicit content information, such as keywords extracted

from books [Woodruff et al. 2000] and web sites [Balabanovic and Shoham 1997; Pazzani and Billsus 1997], document titles and abstracts [Torres et al. 2004], CD titles [Shardanand and Mae 1995], movie titles and main cast members [Basu et al. 1998; Melville et al. 2002] etc. Take, for example, the content-aspect of the recommender for Fab [Balabanovic and Shoham 1997]; the keywords that are used to represent the candidate web pages are the 100 most important words. Similarly, the keywords used for research paper recommendation in [Torres et al. 2004] include topical keywords extracted from each paper's title and abstract; while in the movie recommender discussed in [Good et al. 1999], the content information is obtained from sources in the international movie database at www.imdb.com (which contains the most comprehensive and up-to-date information about each movie), including each movie's descriptive keywords, title and cast members.

Among the various content-based RSs, *News Dude* [Billsus and Pazzani 1999] and *WebWatcher* [Joachims et al. 1997] are two of the most representative. The former performs a content-based approach to learn users' news-reading preferences and recommend a new story [Billsus and Pazzani 1999]. To accomplish this, the system builds and maintains two kinds of user models. The first is a short-term user model that measures the interestingness of a story by how close it is to the stories that the user has read before. In this case, the similarity measurement is based on the *co-occurrences* of words appearing in these stories. The second user model carries a probabilistic classifier that assigns a probability of interest to a new story by comparing how *frequently* its words occur in those stories the user regards as interesting when contrasted with those the user regards as of no interest. When a new article arrives, News Dude will first consult

with the first user model to predict whether the user will be interested in it; if not, the second user model is consulted. Results show that this kind of double user model performs better than either model in isolation.

WebWatcher [Joachims et al. 1997] is another content-based system, in which on entering the site, a user's browsing is monitored by a learning agent which can learn from what the user has stated as their interests when first entering the system, and the subsequent contents of the pages he/she has visited. *WebWatcher* tries to link the contents of a page to the user's interests. Experimental results show, however, that *WebWatcher*'s ability to recommend hyperlinks of interest to the user is relatively low (only 48% [Joachims et al. 1997]) due to the drifting interests of users.

2.1.1.2 The Disadvantages of the Content-based Approach

Despite the success of some early RSs that are content-based, there are several major drawbacks that prevent the widespread application of the content-based approach:

1. Since user profiles in the content-based approach are built through an association with the contents of the items, the approach tends to be quite narrowly focused and with a bias towards highly scored items. Thus, a user might be restricted to those items that are very similar to the ones he/she has read before, which is known as the issue of 'over-specialization' of the recommendations [Domavicius et al. 2005] or being trapped into a so-called 'similarity hole' [McNee et al. 2006a]. In addition, the content-based recommendation approach works well only in domains where the content features of the item can be extracted automatically mostly through various machine learning techniques [Pazzani and Billsus 1997].

2. The content-based approach only considers the preferences of a single user. For News Dude, even if multiple users' preferences are given (in the form of the co-

occurrences of words best representing the preferences of the users), the system cannot learn across this cluster of users to inform the prediction process. The content-based approach is thus incapable of exploiting community endorsement.

3. Content-based recommendation is domain dependant. If a content-based RS needs to be applied in domains other than those where the recommendation mechanism was developed, substantial modifications need to be done before the RS can be used. For instance, a content-based RS originally developed for the movie domain needs to be extended to make recommendations on books, and the recommendation algorithm(s) need to be modified. In fact, it is mainly this feature of the content-based RS that prevents it from being widely used in commercial systems.

Some of these problems that content-based approach suffers from can be remedied by the CF approach that is discussed below.

2.1.2 The Collaborative Filtering Approach (CF)

Collaborative filtering (CF) makes recommendations by observing like-minded groups. It starts with the assumption that users who enjoyed certain things in the past will enjoy similar things in the future. CF build user profiles by keeping user ratings on items without relying on the content of the items; a user-item rating matrix incorporating users and their ratings maintains this information. Thanks to its simplicity and application-independence, CF remains the most commonly adopted technique in commercial RSs [Linden et al. 2003], and the most studied in the academic community (see, for example, [Resnick et al. 1994, Shardanand and Maes 1995, Breese et al. 1998, Cosley et al. 2005, Ziegler et al. 2005, Herlocker et al. 2005, Tang et al. 2005]).

CF algorithms rely on similarity metrics computed between two users' ratings of items being recommended. The CF system has the potential to learn from a group of

similar users and arrive at appropriate recommendations without the need to construct a complete profile for each user. Therefore, the key to CF is to apply similarity measurements to identify users with similar tastes to a given user (i.e. target user). A number of similarity measurements have been applied including *Pearson correlation* [Resnick et al. 1994], *mean squared difference* [Shardanand and Maes 1995], and *vector similarity* [Breese et al. 1998]. Two commonly adopted similarity measurements in the literature include *Pearson-correlation* based and Cosine-based similarity. According to [Breese et al. 1998], the Pearson-correlation approach outperforms the Cosine-based approach. It has been used in many studies including [Melville et al. 2002, Tang et al. 2005, Aimeur et al. 2007], and this thesis.

GroupLens is a pioneering recommendation system which successfully adopts the CF approach [Resnick et al. 1994]. Developed for Usenet news and movie recommendation, GroupLens is a rating-based CF system, where user profiles are built based on their ratings of products or services. Firefly is another CF system which provides recommendations for music albums and artists [Shardanand and Maes 1995]. Results show that even if Firefly ignores the features of the music albums, simply matching one user against a cluster of similar users and making recommendations is surprisingly good at predicting a user's future music tastes.

A number of Internet applications and online retailers have employed CF for making recommendations to their clients [Schafer et al., 2001], among them, Alexa (a web browser), Amazon.com™, Netflix, CDNow.com™ and Levis.com™. The major reason is that CF can work in any domain, anytime, and anywhere, because it makes

recommendations by only considering user ratings which are independent of the domain content.

2.1.2.1 How Collaborative-Filtering Works: A General Formal Model

Forming a neighborhood of similar users

Formally, the key neighborhood determination process can be expressed in the following way:

Let us assume we have a target user a. The RS should find an ordered list of h neighboring users $N = \{N_1, N_2, \ldots, N_h\}$, such that $a \notin N$ and $sim(a, N_1) \geq sim(a, N_2) \geq \ldots \geq sim(a, N_h)$, where $sim(a, N_i)$ denotes the *similarity* of user model a to its neighbor N_i.

There are two commonly adopted similarity measurements in the literature: *Pearson-correlation based* and *Cosine-based similarity*. According to [Breese et al. 1998], the Pearson-correlation approach outperforms the Cosine-based approach. It has been used in many studies including [Yu et al. 2001, Melville et al. 2002, Tang and McCalla 2004; Tang et al. 2005]:

Let I_i be the set of items which user i has rated. Then the mean vote for user i is defined as:

$$\bar{r}_i = \frac{1}{|I_i|} \sum_{k \in I_i} r_{i,k} \qquad (2\text{-}1)$$

where $r_{i,k}$ is the rating by user i on item k, and $\bar{r}_i$ is the mean rating by user i to all items. The *Pearson correlation* between users a and b is given by:

$$P(a,b) = \frac{\sum_{k \in K} (r_{a,k} - \bar{r}_a)(r_{b,k} - \bar{r}_b)}{\sqrt{\sum_{k \in K} (r_{a,k} - \bar{r}_a)^2 \sum_{k \in K} (r_{b,k} - \bar{r}_b)^2}} \qquad (2\text{-}2)$$

where K is the set of items co-rated by both a and b. Co-rated items are those items rated by both users. The value of the Pearson correlation is between 0 and 1.

Moreover, as suggested in [Herlocker et al. 1999], the Pearson correlation will normally be de-valued by $|K|/50$ if $|K| < 50$. The reason is that a neighbor whose number of co-rated items is relatively small is less interesting than a neighbor who has rated many similar items to the target user. Therefore, the denser the rating database is, the more accurate and efficient the predicted ratings become.

Making predictions

After computing the (adjusted) Pearson correlation between the target user profile and each of the other user profiles, the system will select the top n-most correlated users as the relevant neighbors of each target user. After that, it can calculate the predicted rating of target user a on item j using the following equation (2-3):

$$P_{a,j} = \bar{r}_a + \frac{\sum_{B} sim(a,b) \times (r_{b,j} - \bar{r}_b)}{\sum_{B} sim(a,b)} \qquad (2\text{-}3)$$

where B is the set of all neighbors being considered, $|B| = n$. Note here that the value of n is usually set to be 30 in CF literature (among them, [Herlocker and Konstan 2002, Tang et al. 2005, Herlocker et al. 2004, Lekakos and Giaglis 2006, McLaughlin and Herlocker 2006]); in other words, 30 closest neighbors for a target user. The similarity between two users, $sim(a, b)$ can be $P(a, b)$ or $\cos(\vec{a}, \vec{b})$.

A prediction for the target user is computed as a *weighted sum* of the average rating of the best-n closest neighbors provided for that user.

2.1.2.2 Output from a CF System

In general, there are two kinds of output strategies in a CF system:

- Recommended items are presented to the user one-at-a-time (representative systems include PHOAKS [Terveen et al. 1997] and SiteSeer [Rucker and Polanco 1997]). When a user only wants to find a single 'decent' book to read, or an 'interesting' movie to watch, this type of the recommendation is the most appropriate [Herlocker et al. 2000].

- Recommended items are presented to the user as an ordered list (top-*N* recommendation). [Deshpande and Karypis 2004] treat the recommendation problem (termed "making a prediction") as a top-*N* recommendation problem where the system needs to identify a set of *N* items that a certain user will be interested in. Obviously, these *N* items are those with the highest predicted ratings, that is, those which a user will find the most interesting.

In this thesis, we will consider both one-at-a-time and top-*N* output strategies.

2.1.2.3 The Drawbacks of CF

CF's dependency on user ratings causes three key drawbacks to the overall performance of the system:

1. If an item has not received enough ratings from users, or if many users have not rated each item, correlation computations cannot be performed. These two problems, the *first-item* problem and the *first-rater* problem respectively, are collectively referred to as *cold-start* problems. A cold-start problem prevents users from seeking recommendation information on new items and serendipitous items (items that nobody or only a few users have rated). This will be discussed in the next section.

2. In addition, it can be challenging for the system to form neighbors when users have *unusual* tastes, since the correlation of ratings between these unusual users and 'mainstream normal' users could be low, resulting in poor recommendations. The performance of the CF-based approach relies heavily on finding neighboring users who have co-related enough items.
3. As CF largely relies on the correlations between user ratings to form neighborhood users for target users, the sparsity of the rating database can severely compromise the overall performance of the recommendations. Solutions to this problem come in two major types. The first is to directly aim at the sparsity problem by generating artificial ratings [Mellville et al. 2002], while the second focuses on selecting quality candidate items to alleviate system's dependency on the ratings [Yu et al. 2001, Tang et al. 2005].

2.1.3 Hybrid Approach

Hybrid recommendation mechanisms attempt to deal with some of these issues and smooth out the drawbacks of the CF and content-based approaches. A purely content-based approach fails to consider community endorsement, and is concerned with only the significant features describing the content of an item, whereas, a purely CF approach ignores the contents of the item, and makes recommendations only based on comparing the user against clusters of other similar users. Consider, however, the possibility that item information (features that would best categorize the item) can be obtained through a content-based approach, and user information (relative distance of the user to other clusters of users, and users' opinions) can be obtained from CF. By combining these two techniques, we can smooth out the drawbacks of both the pure

content-based and pure CF approach and obtain both individual as well as collective experiences with respect to the items being recommended.

The majority of hybrid RSs combines collaborative and content-based approaches by learning and constructing a unified user profile for recommendations. For example, Fab [Balabanović and Shoham 1997] can be regarded as a two-layered filtering system. The first layer is created by a *content-based* approach, which ranks documents by topic, and then ranked documents are sent to a user's personal filter. In the second layer, a user's relevance feedback is used to modify both the personal profile filter and the topic filter. It is obvious that only filtered documents are added to the list of candidate documents to be recommended. In the case when there are no ratings for a target item, Fab can still recommend it appropriately based on content filtering. [Claypool et al. 1999] and [Pazzani 1999] attempt to build separate user profiles based on the content-based and collaborative mechanisms. Then, the outputs from these two approaches are incorporated either by a linear combination of ratings [Claypool et al. 1999] or a voting scheme [Pazzani 1999]. [Schein et al. 2002] employ an associative retrieval technique to combine models obtained from both content-based and CF techniques with an emphasis on dealing with the sparsity problem. [Melville et al. 2002] create '*pseudo user ratings vectors*' for each user in the database in an attempt to generate a dense ratings matrix, which can then be used to make recommendations. These pseudo user ratings are obtained through the analysis of the contents of the movies. After the artificial rating generation, all of the ratings give the so-called *dense pseudo user-ratings matrix*, on which CF is performed. Obviously, when the user has rated many items, content-based rating injections are worthwhile and can fairly reflect the '*consistent*' tastes of the user; on the other hand, if

the user has rated only a few items, the pseudo-rating will not be accurate, and thus can yield a misleadingly high correlation between the target user and candidate users [Melville et al. 2002]. To cope with this, Melville *et al.* proposed several weighting factors to devalue the correlation, i.e. *harmonic mean weighting* and *self weighting* [Melville et al. 2002]. Experimental results demonstrated that the artificial content-boosted CF performs better than a pure content-based predictor, a pure CF and a hybrid of the two [Melville et al. 2002]. Obviously, by injecting these pseudo user ratings, some of the challenging issues in RSs can be overcome, among them, the first-rater and sparsity issues. Other systems which apply a hybrid approach to make movie recommendations include [Basu et al. 1998, Grant and McCalla 2001]. [Tang et al. 2005] refers to both the content features and applies a temporal window analysis on the candidate items in an attempt to scale down the number of sets.

2.1.4 Knowledge-based Approach

In fact, all recommendation approaches should do some kind of inference to know about the many dimensions of the user profile, e.g. interest etc. A knowledge-based approach builds the user profile *gradually* [Burke 2002]. Such approaches include preference-based feedback [Smyth and McGinty 2003] and critiques [Smyth et al. 2004; McGinty and Smyth 2003; McCarthy et al. 2004], which are a more complex form to build a user profile than the more common *rating-based* feedback (as in [Resnick et al. 1994; Shardanand and Maes 1995; Tang et al. 2005; Herlocker et al. 2004; Miller et al. 2004]).

The 'Find-Me system" is a typical knowledge-based RS [Burke et al., 1997; Burke 2000; Burke 2002]. The restaurant recommender Entrée allows users to provide incremental and refined critiques (such as "*show me more like Reco but less expensive*")

of the system's suggestions through rounds of interactions until an acceptable option is reached. Smyth *et al.* later refer to this type of the system as a *conversational RS* [Smyth et al. 2004; McGinty and Smyth 2003; McCarthy et al. 2004] to reflect its characteristics as well as differentiate it from other types of RSs. Table 2-1 below lists a slightly modified example of critiques (to be used in recommending PCs) originally shown in Figure 1 of [Smyth et al. 2004].

Table 2-1. A comparison among the three main selection strategies in [Rashid et al. 2002]

	Current case	Case *c* from CB	Critique pattern
Manufacturer	Compaq	Sony	!=
Monitor (inches)	14'	12'	<
Type	Desktop	Desktop	=
Price	1500	3000	>

As shown in the above table, when purchasing a PC, we would consider many different features, ranging from the manufacturer, monitor size (or even monitor type), to price. As such, users can indicate that they are looking for a PC with 'bigger-size monitor'; or select that they prefer a 'more powerful' PC. The former critique is referred to as a unit preference over a single dimension; while the latter is a compound preference over a possibly multiple dimensions including 'processor', 'memory', etc. Hence, it is obvious that this type of RS can afford users an opportunity to provide more informative feedback through incremental critiques provided to the system.

Although *critique-like* feedback has limited ability to guide the recommendation process (when compared with the explicit numerical ratings in other RSs) and therefore solicit user preferences, in certain complex domains where the dimensions of a product include many compound features, it can be of help to encourage user involvement

through many interactions between the RS and the user. For example, in cases when users have limited knowledge of the item being considered, it is difficult for them to key-in features they would not know. For instance, consider buying a digital camera; features like resolution (in pixels), optical zoom, digital zoom, storage type, etc., would be difficult for novice users to specify. As such, an adaptive user interface with automated generation of compound critiques can relax users' burden of manually inputting these features [McCarthy et al. 2005]. However, the most challenging issues in implementing this type of RS is the construction of the feature space along which users are able to critique and the adaptive identification and generation of the compound features. For instance, generating the following compound critique in which the user is looking for a PC with *lower speed*, *less memory* and *lower price* than current one: *{[Speed <], [Memory <], [Price<]}*

It is not trivial to identify compound features as above, especially when the number of the features that should be considered increases. Although Smyth *et al.* propose the adoption of association rule mining to dig out the historical critique database to find '*associated patterns*' to present to the target user [Smyth et al. 2005], the recommender can still suffer from performance complexity. The situations can further be compounded by the fact that there will be cost incurred when human decision makers are facing so many choices [Shugan 1980], i.e. '*the cost of thinking*'. In other words, the cognitive costs associated with evaluating numerous alternatives cannot be ignored, which prevent these semi-automated knowledge-based recommendation approaches from gaining widespread popularity in commercial systems, in which it is usually difficult to

encourage users to contribute their time and effort to participate in a seemingly un-rewarding recommendation process [Rashid et al. 2006a, Ling et al. 2005].

It must be pointed out that among all of these major types of RSs, knowledge-based ones have received the least attention among researchers in the RS and machine learning areas. In addition, while CF-based recommenders have already found their way from the research community to the design of commercial RSs such as Amazon.com, few commercial systems exist that makes use of the knowledge-based techniques due to the inherent complexity of the feature as well as decision space.

2.2 Evaluation Methodologies

Since the first automated rating-based RS was proposed in 1994 [Resnick et al. 1994], the accuracy of RSs remained the ultimate evaluation goal in the research literature until early 2004 when several researchers began to explore other ways to evaluate the performance of RSs [Herlocker 2004; McNee et al. 2006a; Riedl and Dourish 2005; Adomavicius and Tuzhilin 2005].

Although the metrics adopted in previous RSs differ, there are some commonly used metrics which have been acknowledged in the community. In this thesis, we will evaluate two broad classes of evaluation approaches: objective and subjective measures. Objective approaches are then sub-classified into two main categories: *predictive accuracy metrics* and *classification accuracy metrics* [Herlocker et al. 2004].

2.2.1 Objective Evaluation Metrics

2.2.1.1 Predictive Accuracy Metrics

Predictive accuracy metrics examine how close the RS's predicted ratings are to the true user ratings. Among the many flavors of these metrics, Mean Absolute Error or MAE is the most popular (e.g, [Melville et al. 2002; Sarwar et al. 2001; Shardanand and Maes

1995; Sarwar et al. 1998; Good et al. 1999; Claypool et al. 1999; Herlocker et al. 1999; Hofmann 2004; Miller et al. 2004; Tang et al. 2005; Adomavicius and Tuzhilin 2005; Lekakos and Giaglis 2006]). MAE measures the average absolution deviation between the user's true rating and the system's predicted rating. Inherently, however, the accuracy of MAE depends heavily on how well and carefully '*true preference*' is determined, that is, whether a rating of 3 or 4 should be regarded as 'good' by both the system and the user. This is especially true when the preference scale is small, say, from 0 to 3. Errors will be inadvertently introduced into the system in erroneously classifying a 'good' item as a 'bad' one, or vice versa. For a larger scale, say 0-5 with 3.5 as the cut-off value differentiating good from bad items, then predicting a 4 as 5, or a 2 as 3, makes little difference to the users.

Obviously, the metric is of particular value for evaluating tasks where the predicted rating will be displayed to the user, in an attempt to establish a trust between the system and the user, so as to encourage the user to come to rely on the subsequent ratings given by the system [Herlocker et al. 2004]. For instance, [Dahlen et al. 1998] make movie predictions and display them to the user (along with the number of the stars). Obviously, if the predicted ratings deviate from users' true ratings, it could compromise the credibility of the system.

In addition to its practical disadvantage, MAE can be less applicable when the RS generates a set of *ranked* items to users who would normally view the top *N* of them and will not even care that some items are uninteresting as long as the top N items also have at least some interesting items. As such, MAE becomes impractical and has been criticized by McLaughlin and Herlocker [2004], Herlocker et al. [2004], McNee et al.

[2006a], on the grounds that it ignores users' experiences with the performance of the system, as earlier efforts have been too much invested in simply testing the accuracy of the RSs without considering the aftermath of the recommended items: whether or not they will be well received by users.

2.2.1.2 Classification Accuracy Metrics

According to [Herlocker et al. 2004], classification accuracy metrics measure the *ability* of a RS that makes correct or incorrect decisions to *determine whether an item is good*. Thus, this type of the measurement is usually regarded as a *decision-support* accuracy metric [Herlocker et al. 1999; Herlocker et al. 2004; Tang et al. 2005], which examines how well a RS can make predictions that help users select high-quality items.

One assumption of these metrics is that user preferences in RSs should be *binary*, that is, making recommendations is a *binary classification process*: either users will like it; or they will not. Suggested by Herlocker et al. [1999], and widely adopted in the research community (e.g. [Good et al. 1999; Meville et al. 2002; Schein et al. 2002; Schein et al. 2005; Tang et al. 2005]) is ROC (Receiver Operating Characteristic) *sensitivity*, which was originally introduced into the IR community by Swets [1963, 1969] with the name 'relative operating characteristics'. Generally, the probability of a randomly selected good item being accepted by the user is referred to as *sensitivity*; while the probability of a randomly selected bad item being rejected by the user is referred to as *specificity* [Good et al. 1999]. Thus, when adopted for a RS, the ROC model measures the decision-support aspect of accuracy: how the system differentiates between 'good' items and 'bad' items.

2.2.1.3 Precision and Recall

Recall and precision metrics that are popular in the IR community, are also used by some researchers in the RS area [Sarwar et al. 2000; Mooney and Roy 2000; Huang et al. 2002; Miller et al. 2004; Adomavicius and Tuzhilin 2005; McLaughlin and Herlocker 2004; Esmaili et al. 2006; O'Mahony and Smyth 2007], although they might be slightly modified. Generally, the data set is divided into the training and test set. A RS will work on the training set and generate the top-N recommendations. Recall and precision are then used to look into the number of matching elements in the test set and the top-N set. Those items in both sets constitute a hit set [Sarwar et al. 2000]. Recall is defined as the ratio of hit set size to test set size, while precision is the ratio of hit set size to top-N set size.

The accuracy of precision and recall relies on the differentiation between relevant and irrelevant items. Unfortunately, the definition of '*relevance*' itself and an appropriate way to compute it has been in dispute in the IR community [Harter 1996; Kobyashi and Takeda 2000]. To date, the majority of evaluation in the IR community has been focused on an objective definition of relevance, that is, whether the returned documents match the user's query, and is mostly independent of the user's current information task and the user himself/herself [Herlocker et al. 2004; Kobyashi and Takeda 2000; Rieh 2002]. It is not usually that useful for the system to set the boundary between relevant and irrelevant items without the participation of users.

2.2.1.4 Limitations of Objective Evaluation Metrics

Earlier studies in the RS community have focused on the design of algorithms to compute the above mentioned evaluation metrics. For instance, many studies with an emphasis on improving the 'accuracy' of CF algorithms make predictions on 'withheld

ratings' (referring to those ratings in the test set that were hidden during evaluation, [Herlocker et al. 1999; Resnick et al. 1994; Konstan et al. 1997; Good et al. 1999; Melville et al. 2002; Schein et al. 2002, 2005; Tang et al. 2005; Lekakos and Giaglis 2006]) by comparing the predicted ratings with the actual ones in order to arrive at conclusion about the accuracy of the algorithms. In the past, some researchers have realized that when RSs are supposed to support users' decision making, it could be more interesting to see how often recommendations can lead users to make wrong choices [Shardanad and Maes 1995], but this is not a topic of much study.

Recently, there have been quite a few criticisms of the heavy research emphasis on improving the '*accuracy*' of recommendations alone [Herlocker et al. 2004; Riedl and Dourish 2005, McNee et al. 2006a, Ziegler et al. 2005], since the algorithmic improvements of the RSs alone cannot guarantee users a satisfying experience. Simply put, '*the bottom-line measure of RS success should be user satisfaction*' (page 6, [Herlocker et al. 2004]). Proceeding from this assumption it is therefore crucial to conduct effective and meaningful user evaluations. For instance, [Sinha and Swearingen 2002] looked into to what extent RSs can explain their recommendations to users, since users in many real applications need to know how the recommended items have been selected in order to improve the transparency of the RS. Fortunately, during the past several years, various new works have appeared opening up interesting and creative directions to better understand recommendation technologies including multi-dimensional recommendations [Adomavicius and Tuzhilin 2005] and the user interface and interaction aspect of a RS [Terveen and McDonald 2005; Herlocker et al. 2004; McNee et al. 2006a].

2.2.2 Subjective Evaluation Metrics

It is speculated that there are properties other than accuracy that could have a greater impact on user's acceptance of the recommendations. Of the many published works attempting to conduct more effective evaluation of RSs, most are in the area of paper recommendation. Next, we are going to review these approaches, in order to motivate how we are going to evaluate our paper recommender.

[McNee et al. 2002] investigated the paper recommendation issue for experienced researchers, helping them find relevant research articles. The recommended papers might be authored by the users themselves. The evaluation of the performance of the paper recommender comes in two stages. During the first stage, users were first instructed to answer two questions after the recommendation list is presented to them:

1. "how relevant is this citation to your paper and its related work?"
2. "how familiar are you with this citation?"

After obtaining users' answers on each of the items in the recommendation list, McNee et al invited the users to assess the following aspects of the recommendation quality of all items as a whole: the overall quality, usefulness and novelty of the recommendations. Usefulness and novelty of the recommended items can be regarded as a finer-grained break-down of the overall rating of users towards the items according to [Rieh and Belkin 2000; Rieh 2002].

[Torres et al. 2004] studied similar issues, although the target users included students with less research experience. The first two were later interpreted as 'satisfied', while the last two were interpreted as 'dissatisfied' for evaluation purposes. To carry out the evaluation, seven questions were asked, ranging from the degree of familiarity that a user has with the recommended paper, to asking users to label each paper as one of the

five pre-classified ones: novel, authoritative, introductory, specialized, survey/overview. Rating choice comes in four elements: strongly agree, agree, disagree and strongly disagree. Obviously, the evaluation is very closely related to the intended use of the paper recommender: to help researchers find papers that they are interested in, that can help them find related papers. One of the most notable impressions that Torres et al. [2004] get from the evaluation is that, in contrast to other most-studied RSs for movies, CDs, books etc, it is not necessary for the paper recommender to generate a set of high quality recommendations consistently as long as there are some good ones among the set.

Fundamental to the conclusions arrived at by Torres et al. [2004] is that their studies confirmed that paper recommendation is vitally different from recommendation in other domains due to many aspects, including the nature of the domain, the user and the tasks. In particular for paper recommendation, we often have fewer users than the number of the papers. User satisfaction is very closely related to their tasks, and can vary based on user tasks and goals.

Cosley et al. [2002] present a framework to evaluate user acceptance rather than predictive accuracy of research paper recommendation. The evaluation framework examines some statistics of recommended papers such as the *number of times* a paper is downloaded, an implicit interest indicator [Claypool et al. 2001]. Downloading a recommended paper can be regarded as a relatively appropriate way to determine a user's acceptance of the item, but to date, a strong correlation between this implicit interest indicator and the user's preference for the item is yet to be demonstrated and must be further investigated [Claypool et al. 2001].

Kalas, a food recipe system supports social navigation to help users find information through various channels, including RS functionality (recommendations computed from others' ratings), chatting facilities, and real-time broadcasting of user navigation trail in the interface: their comments on recipes, the number of downloads per recipe, for personalized recipe recommendations [Svensson et al. 2005].

Among the various evaluations [Svensson et a2005] have carried out, there are only a few which are related to the theme of this thesis: to evaluate the effectiveness of recommendations. We will only discuss experiments relevant to our work. In one experiment, questionnaires and interviews were conducted to obtain subjective views from the users or, in the researchers' own words, '*the utility of the recommender*' (page 9, [Svensson et al. 2005]) in terms of how users act on the recommendations. After using Kalas, users were prompted to fill in a post-questionnaire consisting of Likert-scale questions for users to either agree or disagree with (on a scale from 0 to 7). Table 2-2 below lists some questions from the study [Svensson et al. 2005].

Among the above sample questions from the study, the first two tend to focus more on the general 'usability' of the system in terms of *ease of use* and *time to learn*; while the last three questions focus more on how other users' opinions and system's recommendations can affect a user's choice to act on the recommendations.

Evaluation results confirmed the researchers' hypothesis that users tend to be influenced by the recommended recipes in that they go to the densely populated space where there are the most people commenting on a recipe [Svensson et al. 2005]. This influence of other users' opinion of the recommendations has been one of the most reliable metrics to test the users' acceptance of the recommendations.

Table 2-2. A list of subjective evaluation questions for Kalas [Svensson et al. 2005]

How easy/difficult was Kalas to learn?
How easy/difficult was Kalas to use?
How important is it that others have chosen the recipe in your choice of recipe?
How important is it that a recipe was recommended in the chat in choosing a recipe?
How important are a recipe's comments in your choice of recipe?

[Brusilovsky et al. 2005] followed similar assessment procedures for the social-navigation-based educational resource recommender, *KnowledgeSea II*. With *KnowledgeSea II*, users can explicitly rate part of the resource by clicking 'thumbs-up' (interpreted as a positive rating), 'thumbs-down' (interpreted as a negative rating) and 'question-mark' (interpreted as a neutral rating). Post-questionnaire based evaluations are then conducted to assess users' perceived subjective perspectives of the recommender. Similar conclusions were drawn as those in [Svensson et al. 2005].

Thus, we conclude that questionnaire-based subjective evaluations can continue to provide us insights into more reliable and practical ways of measuring user satisfaction of RSs. Generally, some observations can be made from most of the subjective evaluations of the RSs. The first one is that since it is not easy to construct an automated tool to accomplish the evaluations, questionnaires and interviews remain the most reliable methodologies and easiest to implement. The second is that these evaluations would usually examine how the users would act following the presentation of the recommended items in terms of whether or not they will recommend it to others, whether or not they are interested to comment on it, and whether or not they are willing to download it. Interestingly, in commercial systems, users' aftermath actions (such as the number of

products purchased/viewed, the degree of willingness to re-recommend the recommended items to others), matter.

In this thesis, we mainly adopt the subjective evaluation methodology, similar to [Svensson et al. 2005, Torres et al. 2004; McNee et al. 2002]. In particular, we administrate questionnaires (both pre-and post-), designed to examine the performance of the recommender according to the domain, the user, as well as the user's tasks. It is only through such studies that we can probe deeply into how a paper recommender can help users complete their tasks. A detailed discussion of these issues will appear in the next two chapters of this thesis.

2.3 Related Work on Paper Recommendation

Since this thesis mainly focuses on the recommendation of papers, in this section, we present related works concerning tracking and recommending technical papers.

2.3.1 Paper Recommendation on Paper × Reviewer Matrix: A Query-Language Approach

Basu et al. [2001] studied paper recommendation in the context of assigning conference paper submissions to reviewing committee members. In particular, their approach to making paper recommendations is to represent the two types of entities, i.e. papers and their reviewers, by a variety of characteristics. For instance, papers can be represented by their titles, abstracts and a set of user-specified keywords from a pre-specified list; while reviewers' information is obtained through the analysis of the reviewers' own papers in an attempt to extract information about the reviewers' expertise. After this information is obtained, a paper reviewer matrix is generated, and then any recommendation can be made through querying the matrix data base. Essentially, the recommendation process is cast as extracting information about reviewers' expertise and

papers' main topics, and formulating queries to obtain the recommended paper(s). For instance, the following query taken from [Basu et al. 2001] can be made to match a paper with a reviewer:

SELECT Reviewer.Name, Paper.ID

FROM Paper AND Reviewer

WHERE Reviewer.Descriptor SIM Paper.Abstract

It is obvious that the context is different from ours, where we are interested in making recommendations to learners in the classroom.

2.3.2 CiteSeer: Paper Searching and Recommendation

Bollacker et al. [1999] refine CiteSeer[1] through an automatic personalized paper-tracking module which retrieves each user's interests from well-maintained heterogeneous user profiles. Some commonly known text processing algorithms were adopted, such as TF-IDF, keyword matching, the use of papers' metadata, citation analysis, to track and maintain a user's notion of 'relatedness' when submitting a query to search for documents. Among the techniques used in CiteSeer, Citation/co-citation analysis and bibliographic coupling [Salton 1963; Small 1973; Garfield 1980] has been used to harness the decisions made by researchers/authors/users to include references to similar previous documents in the scientific literature. *Citation analysis* makes use of endorsement (including citing and being cited) by one author to another to establish subjective similarity among their works. *Co-citation* is used to establish a *subject similarity* between two documents. If papers A and B are both cited by paper C, they

[1] CiteSeer is a publicly available paper searching tool, and can be accessed at: http://citeseer.ist.psu.edu/cs

might be related to one another, even if there is no direct citing relationship between paper A and B. *Co-citation coupling* measures the degree of subjective similarity between paper A and B. That is, if they are both cited by many other papers, then they are said to have a stronger relationship. The more papers they are jointly cited by, the stronger their relationship is.

A profile is then maintained of the user's behavior while searching for papers when he/she begins to use the interface, as a cookie will be uniquely assigned to a user once he/she begins surfing through the CiteSeer user interface. Superficially, CiteSeer is similar to Google™ in its ability to find a set of papers similar to a user's query. As such, CiteSeer can be regarded as a general paper searching system, requiring a user to formulate appropriate keywords to facilitate the searching. This makes such a system not that suitable for learners without much research experience.

2.3.3 Digital Book Content Recommendation based on a Spreading-Activation Mechanism

Woodruff et al. [2000] discuss an enhanced digital book with a spreading-activation-geared mechanism to make customized recommendations for readers with different types of background and knowledge. In particular, they combine *citation analysis* and *contextual information* (as shown in the following table) in order to recommend the next paper a user should read from within a digital book consisting of 43 articles. Scenarios are created for users with different goals and research experiences, as presented in Table 2-3 (this is Table 2 of [Woodruff et al. 2000]). The system then delivers lists of recommended articles the users should read next. Since the recommender mechanism also makes use of the citations a paper receives, the users are asked to explicitly list the papers they liked.

Table 2-3. User scenarios in the personalized reading recommender [Woodruff et al. 2000]

Topic	User Scenario	User's Liked List of Papers
Fisheye	I am a VLSI chip designer. I'm writing a tool that uses fisheyes to show circuit layout. I want to read everything I can about fisheyes. I have already read Furnas' paper listed. What should I read next?	Furnas, G. W. (1981). The Fisheye view: a new look at structured files. Murray Hill, NJ: Bell Laboratories.

Two conditions essentially support the performance of the recommender. The first is that the user should be able to specify the seed paper, based on which the recommendation process will start to apply the spreading-activation mechanism. The second condition lies in the fact that the user should be well motivated and have clear goals, and be able to specify the context in which they seek recommendation (as shown in the table the scenario is very clear and fine grained). The relaxation of either of these conditions would greatly compromise the effectiveness of the recommendations made. The paper recommender proposed in [Woodruff et al. 2000] is relatively inapplicable in the e-learning domain where students are relatively novice learners, it is not reasonable to assume that students are able to describe appropriate paper-seeking context.

2.3.4 Research Paper Recommendation: Collaborative Filtering Approach

Similar to [Woodruff et al. 2000], McNee et al. [2002] investigate the adoption of CF techniques to recommend papers for researchers, although researchers do not need to explicitly specify the details of the papers they like. However, McNee et al. did not address the issue of how to recommend a research paper; but how to recommend *additional references* for a *target research paper*. In the context of an e-learning system, additional readings cannot be recommended purely through an analysis of the citation

matrix of a target paper. A similar study of recommending research papers by Torres et al. [2004] investigates cross-culture, cross-language, and cross-research-experience research paper recommendations. In particular, in their study, they classify research papers into four main classes: novel, authoritative, introductory and survey/overview. They empirically show that different recommendation algorithms, including pure CFs, and hybrid CFs, should be used for different categories of papers.

Generally, the CF adopted in the study is the citation and co-citation analysis of the active paper and its citations. And the content-based filtering (CBF) applied in the paper recommender is based on the pure textual analysis of a paper's abstract and title only. In addition, all hybrid approaches have two components: *pure CF* and *CBF* in order to complementing each other's performance, and tend to augment the feature space to make more accurate recommendations. The recommendation techniques in [Torres et al. 2004] are quite similar to some of the authors' previous study [McNee et al. 2002] in which their core techniques fall into two categories:

- citation, co-citation analysis;
- pure paper content-based approach (TF-IDF similarity measures)

Torres et al. [2004] did not, however, study, how users responded to the paper classes; but only demonstrated that it is important for the RS to generate high quality recommendations consistently, although not every single recommendation has to be good. They did provide advice on the sort of algorithms that work best for these different classes of papers. The results, though, are confusing in that they only draw conclusions purely from the papers' perspective, and fail to associate the types of users with the classes of the papers. In other words, they fail to indicate what types of papers will be

most liked by what types of users, which is very important for research paper recommendations, since the research and knowledge contexts within which the user is seeking recommendation can greatly affect their perceptions of the RS. Indeed, in their post evaluations, they found out that 'professionals were not as happy as students toward the recommendations' [Torres et al. 2004], as professors have perhaps read and researched those recommended papers. Unfortunately, neither study (i.e. Torres et al. 2004, McNee et al. 2002) an went further to enhance the recommendation algorithms to solve these problems.

A similar study [Middleton et al. 2004] also focuses on recommending on-line research papers to academic staff and students with a major aim to search for relevant research papers and to continue to work as new papers come out. Their approach is to carry out a dynamic 'match-making' between user and paper ontology. Specifically, paper properties are represented using term vectors, where ontological topics on papers are selected. Meanwhile, user profiles are inferred dynamically by using built-in tools tracing users' browsing behaviors through the system interface. For example, a user is allowed to rate whether he/she is 'interested'/'not interested'/ 'no comment' regarding a specific paper. A user of the system can also manually adjust the classification label of a research paper by clicking on the topic of the paper and choosing another topic from a pull-down topic list. These 'unobtrusive behaviors' are then fed into the system and used to train new papers. Finally, user interest profiles are calculated by correlating previously browsed papers with their classification. Essentially, the recommendations are made through classical IR techniques.

2.3.5 Pedagogical Resource Recommendation

2.3.5.1 Pedagogical Web Resource Recommendation: Altered Vista

Recker et al. [2003] study the recommendation of educational resources in a system called *Altered Vista*, where teachers and learners can submit and review comments on educational web resources provided by learners. Learners are pre-categorized into different '*pedagogical*' groups: undergraduate, graduate, Grades 6-12, Grades K-5. An example comment might be how a learner from the undergraduate group would like the site 'www.doc.mmu.ac.uk/aric/eae.index.html', a site for the Encyclopedia of Atmospheric Environment. To facilitate group sharing and system recommendation, members of each group can then take part in group discussions, where both negative and positive comments can 'motivate' each individual member's rating towards the site. Table 2-4 presents a modified example adapted from Figure 1 in [Sumner et al. 2003].

Table 2-4. Evaluating a web resource to facilitate recommendation [Sumner et al. 2003].

Web resource	www.doc.mmu.ac.uk/aric/eae.index.html
Overall Quality of the Site	very low very high -3 -2 -1 0 1 2 3
Justification of the Rating	Positive factors
	Negative factors

Learners' overall ratings as well as their comments toward each web site will then be recorded and categorized by tutors to determine the selection category of the sites in classroom use: *quality content*, *advertising*, *scientific bias*, and *design and usability issues*. For instance, a K-5 subject giving a high score on atozteacherstuff.com complained (in the comments) that the presence of advertising in the site is a very negative feature, though the rating he provided was based on the positive aspects of the

site in support of the activities and lessons appropriate for younger children. Although, as admitted in [Sumner et al. 2003], some of these dimensions are difficult to define, categorize, and more importantly, can considerably vary depending on the subjective views of different levels of learners, the study stressed the importance recommending pedagogically appropriate learning materials in the classroom.

Altogether 18 different web-based educational resources were rated by learners of various levels. Experimental observations demonstrated that learners were able to judge the quality of a web resource and combine their considerations of several factors in reaching the overall rating. Meanwhile, learners also suggested some additional design elements that might contribute to the learners' perceptions of the value/quality of the educational web resources. In the group-level experiments, group members were able to identify and rank the evaluative criteria they, as a group, deemed the most important to educational web resources. Although many of the findings seem to be intuitive, their research is one of the very few projects that systematically explore the dimensions that are important to educational web resources. More importantly, their findings suggest that educational digital libraries should be '*more nuanced than simple presence or absence of filters, especially for libraries that serve multiple audiences*' (pp. 277 of [Sumner et al. 2003]).

Although they emphasize the importance of the pedagogical features of these educational resources as what O'Mahony and Smyth do in [2007], they also do not use the pedagogical features to make recommendations, which is one of our goals.

2.3.5.2 Pedagogical Web Resource Recommendation: Annotation-based Social Navigation Approach

Knowledge Sea II [Brusilovsky et al. 2005] makes use of the annotations learners have left behind as clues to help subsequent learners select those items of interest to them. In fact, when readers make annotations while reading documents, multiple purposes can be served: supporting information sharing [Marshall 1997], facilitating online discussions [Cadiz et al. 2000], encouraging critical thinking and learning [Davis and Huttenlocher 1995], and supporting collaborative interpretation [Cox and Greenberg 2000]. Annotations can be regarded as notes or highlights attached by the reader(s) to the article, and since they are either privately used or publicly shared by humans, they should thus ideally be in human-understandable format.

Another line of research on annotations focuses more on the properties (metadata) of the document as attached by editors (such as teachers or tutors in an e-learning context), e.g. using the Dublin Core metadata. Common metadata include Title, Creator, Subject, Publisher, References, etc. [Weibel 1999]. These metadata (sometimes referred to as item-level annotations) are mainly used to facilitate IR and interoperability of the distributed databases, and hence need only be in machine-understandable format. Some researchers have studied automatic metadata extraction, where parsing and machine learning techniques are adapted to automatically extract and classify information from an article [Han et al. 2003; Lawrence et al. 1999]. Others have also utilized the metadata for recommending a research paper [Torres et al. 2004], or providing its detailed bibliographic information to the user, e.g. in ACM DL or CiteSeer [Lawrence et al. 1999]. Since those metadata are not designed for pedagogical purposes, sometimes they are not informative enough to help a teacher in selecting learning materials [Sumner et al. 2003].

Instead of allowing users to write 'free-text' annotations, *Knowledge Sea II* includes a slightly different version of the annotation. Specifically, *Knowledge Sea II* [Brusilovsky et al. 2005] reports a user study on annotation-based social navigation support for making personalized recommendations to students engaging in learning a computer programming language. In particular, students are allowed to annotate tutorial pages and make highlights on the different sections of the page while reading it. *Reputation indicators* like a 'question mark', 'thumbs up' and 'sticky note' have been adopted to provide visual clues to students regarding the specific tutorial[2] instead of Lickert-scale more common in the RS community. Moreover, the color-intensity of these visual indicators highlights the density of annotations made. The most interesting issues regarding the annotation-based navigation support in such an environment lie in the examination made on the actual 'popularity' of these supports, and how students actually make use of them.

In order to examine the degree of usefulness and usage of these pedagogical tools, Brusilovsky et al. [2005] performed several experiments and observed that 'the presence of public annotation does influence students' navigation behavior and assists them in deciding which page to visit next' [Brusilovsky et al. 2005]. One of the most interesting results reveals the there aren't really significant differences among the three types of annotations. In other words, resource materials with all three types of opinions attract, on average, the same amount of traffic from the students. As such, the study indicates that

[2] A 'question mark' is associated with negative comments, a 'thumbs up' corresponds to positive comments, while a 'sticky note' icon will be shown associated with a neutral notes views from the viewer. In fact, these are similar to a 3 point Likert scale.

students find it interesting to read articles recommended (both positively and negatively) by other students, which characterizes one of the most important trends and factors in e-learning: to provide a study-space in which students can exchange their ideas and benefit from each other's insights.

2.3.5.3 Research Paper Recommendation: Research-Context Approach

Traditional digital libraries and online search engines in general encourage users to key in from simple to complicated keyword combinations when searching scientific papers, as discussed before. One of the basic driving forces behind these search mechanisms is that users should be aware of the features of the user interface (such as the Boolean operator 'and') and be able to formulate appropriate queries based on their information needs. Unfortunately, it is not a trivial task for many users, including experienced researchers. In addition, studies that investigate the searching behavior of people who use digital libraries [Jones et al. 1998] and other information systems [Jansen et al. 1998; Spink et al. 1998] have shown that people rarely form queries longer than three words. Meanwhile, users rarely use the advanced features of the query interface (e.g. the Boolean operator "and") to form complicated keyword combinations in order to facilitate the search engine to establish a more accurate matching against documents. As a result, documents returned are sometimes not what the user is looking for; they do include the keywords users specified, but generate documents that often are in a somewhat different context and/or domains. To resolve the problem, researchers have begun to look at other information contextually situated within users' information seeking spaces, in an effort to establish a more subtle relationship between the features of the papers and users' information seeking behaviors.

Bradshaw et al. [Bradshaw et al. 2000] designed *Rosetta*, a digital library system that indexes research articles based on the way they have been described when cited in other documents, thus capturing something about how they will be used by the users. To understand this, let us look at an example[3].

The paper by Pattie Maes, entitled "*Agents That Reduce Work and Information Overload*" has been cited extensively in many research communities. For example, consider the following two sentences taken from [Hook et al. 1997] and [Khoo and Chen 1995]:

1. "There is also the question of trust, as discussed by Maes (1994)."

2. "…computerized personal assistants which deal with meeting scheduling, email filtering and re-ordering, flight booking, selection of books, etc."

To readers of these other papers, these references indicate the main values of the Maes paper: the discussion of agents and trust in intelligent applications. Thus, the terms 'intelligent agent' and 'trust' analyzed and generated by *Rosetta* become the contextual information to help users determine the value of the paper. So, for example, if the user types in trust and agent, then the paper by Maes will be returned. The returned documents tend to have higher similarity with the users' information needs. In reality, though, this type of automated generation of the citation descriptions for documents will require indexing to be performed in a computationally costly way: the way how the potentially relevant papers cite Maes's paper needs to be studied, instead of just obtaining the list of the papers that referenced Maes's paper. Another notoriously difficult problem in this approach is that as more and more papers are cited by researchers from many different

[3] We use the examples from the paper [Bradshaw et al. 2000].

areas, the process of generating and enumerating the reference terms automatically might not be trivial.

The work proposed in this thesis is different than this work in that we not only recommend papers according to learners' interests, but also pick up those not-so-interesting-yet-pedagogically-suitable papers for them. In some cases pedagogically valuable papers might not be interesting and papers with significant influence on the research community might not be pedagogically suitable for learners. We argue that the main goal of recommending papers is to provide learners with necessary knowledge of a given topic and personalize the learning environment in order to motivate them to explore more.

2.3.6 A Multi-Dimensional Approach to Paper Recommendation: Previous Work

The majority of recommendation systems make recommendations purely based on item-item, user-user and/or item-user correlations without considering the contextual information where the decision making happens. In other words, the RS works as a black-box, only matching the interests of the target users, without probing into the details of why the users like it, or how the users will like it, as illustrated in Figure 2.1. For example, a user does not like animated movies, but likes to watch them during non-working days with his/her kids. Therefore, he/she should be recommended *The Incredibles* on Saturdays and Sundays. Take another example, Joe normally does not like romantic comedy movies, especially those starring *Meg Ryan*; but he will be willing and happy to watch one during holidays with his wife Sarah, who enjoys movies starring *Meg Ryan* (of any genre). Thus, on weekends, *You've Got Mail* can be recommended to Joe.

In the e-learning domain, a learner does not like software testing in general, but because he/she is taking a class on software engineering, and he/she is expecting 3 credits

to complete the class, he/she should be recommended an article on software testing. In these cases, incorporating *contextual information* is very important and helpful in informing the recommender to provide high quality recommendations to users because they vary in their decision making based on the 'usage situation, the use of the good or service (for family, for gift, for self) and purchase situation' [Lilien et al. 1992]. For instance, customers' *seasonal buying patterns* are classical examples of how customers change their buying habits based on the situation. A context-aware recommender can provide a smart shopping environment since the recommendations are location-aware, i.e. of the recommended shopping date/time, location of the stores, etc. As such, a shopper can receive personalized shopping recommendations in the stores of the neighborhood where he/she is. [Adomavicius et al. 2005] argue that dimensions of contextual information can include when, how and with whom the users will consume the recommended items, which, therefore, directly affect users' satisfaction towards the system performance. In particular, the recommendation space now consists of not only item and user dimensions, but also many other contextual dimensions, such as location, time and so on[4]. Let us assume that D_1, D_2, …, D_m are the dimensions under consideration. Then, the following formula shows how a white-box recommender makes recommendations:

$$\boldsymbol{R_{com} = f(D_1, D_2, ..., D_m)} \qquad (2\text{-}4)$$

When the dimensions only include *Items* and *Users*, then, the recommender becomes the traditional recommender, as follows:

(2-5)

[4] There are no agreed terms on what to call these additional aspects in recommender systems. Names that have been used include contextual information [Adomavicius et al. 2005], lifestyle [Lekakos and Giaglis 2006] and demographic data [Pazzani 1999].

$$R_{com} = f(I,U)$$

According to [Adomavicius et al. 2005], each of the dimensions can be a subset of a Cartesian product of various attributes A_{ij} where $j=1, \ldots k$:

$$D_i \subseteq A_{i1} \times A_{i2} \times \ldots \times A_{ik} \qquad (2\text{-}6)$$

Let us use an example to illustrate the general idea of the approach. Consider the three-dimensional recommendation space *User* × *Item* × *Time*[5]: Then, the rating computation will be computed on the three-dimensional space specifying how much user $u \subseteq User$ enjoyed item $i \subseteq Item$ at time $t \subseteq Time$. Hence, the rating is derived from these three dimensions, and thus is aggregated.

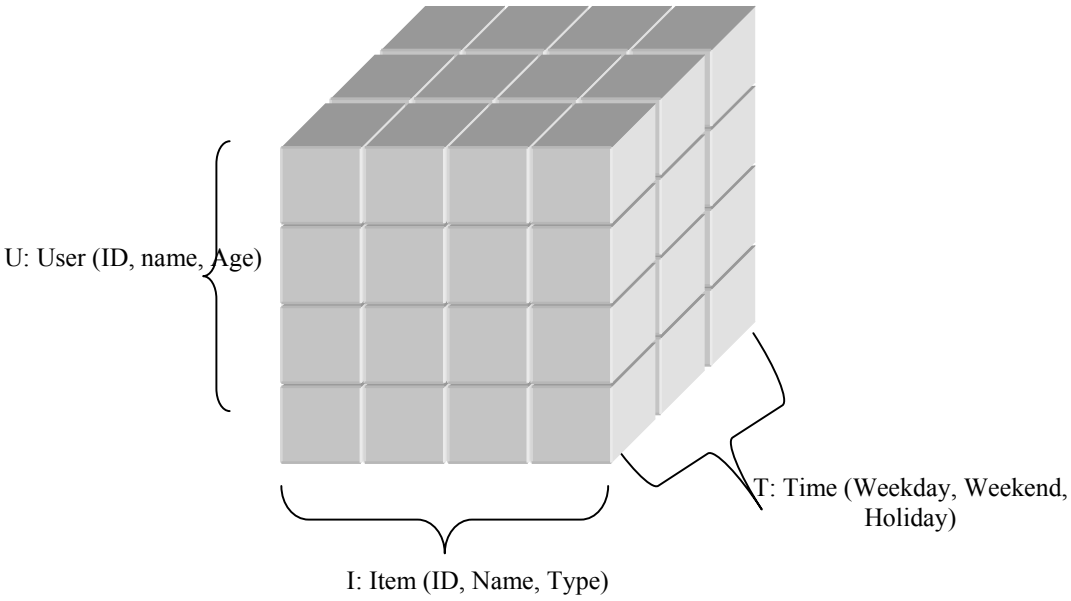


Figure 2-1 The Multi-dimensional RS [Adomavicius et al. 2005]

[5] The example was taken from [Adomavicius et al. 2005], with light modifications.

An example on user profile and interests could be: John, age 25, enjoys watching action movies in theatres during holidays. Hence recommendations can be of the following form: John is recommended *The Departed* during a Saturday night show at UA Pacific Place (a movie theater in one of the high-end areas in Hong Kong). To deal with the multi-dimensional CF, data warehouse and OLAP[6] application concepts drawn from database approaches are proposed [Adomavicius et al. 2005].

Essentially, we have to transform the multi-dimensional CF into traditional 2D recommendations. Simply put, using our previous 3D-CF examples, we can first eliminate the *Time* dimension by only considering votes delivered on *weekdays* from the rating database. The resulting problem becomes the traditional 2D users vs. items CF case. In fact, from a data warehousing perspective, this approach is similar to a slicing operation on a multi-dimensional database, as illustrated in Figure 2-2. The rationale behind the '*slicing*' operation is straightforward: if we only want to predict whether a user will prefer to, say, watch a movie on weekdays, we should only consider the historical 'weekday' ratings for this purpose.

Pazzani [1999] also studied an earlier 'version' of multi-dimensional CF through the aggregation of users' demographic information such as their gender, age, education, address etc. There are a number of ways to obtain demographic data either through explicit ways such as questionnaires or implicit ways such as analyzing their behavioral data (purchasing data). For instance, from users' browsing behaviors, we can easily know where the users come from; hence, the recommendations become '*location-aware*',

[6] Discussions on data warehouses and OLAP are beyond the scope of this thesis. Interested readers can refer to [Adomavicius et al. 2005] for more information.

which is widely used especially for recommendations to be served on mobile devices. In order to make predictions to a target user, the demographic based-CF [Pazzani1999] learns a relationship between each item and the type of people who tend to like it. Then, out of 'that' type of people, the CF identifies the neighbors for the target user, and makes recommendations accordingly. Clearly, the difference between traditional CF and demographically based CF (3D-CF) is the preprocessing step of '*grouping*' similar users.

[Pazzani 1999] stresses the importance of learning a user profile in informing the RS, and is one of earliest successful endeavors to shape subsequent research on multi-dimensional recommendations, despite the fact that the learning process in [Pazzani 1999] is quite easy and straightforward compared to that in [Adomavicius et al. 2005], and is still more like methods in IR.

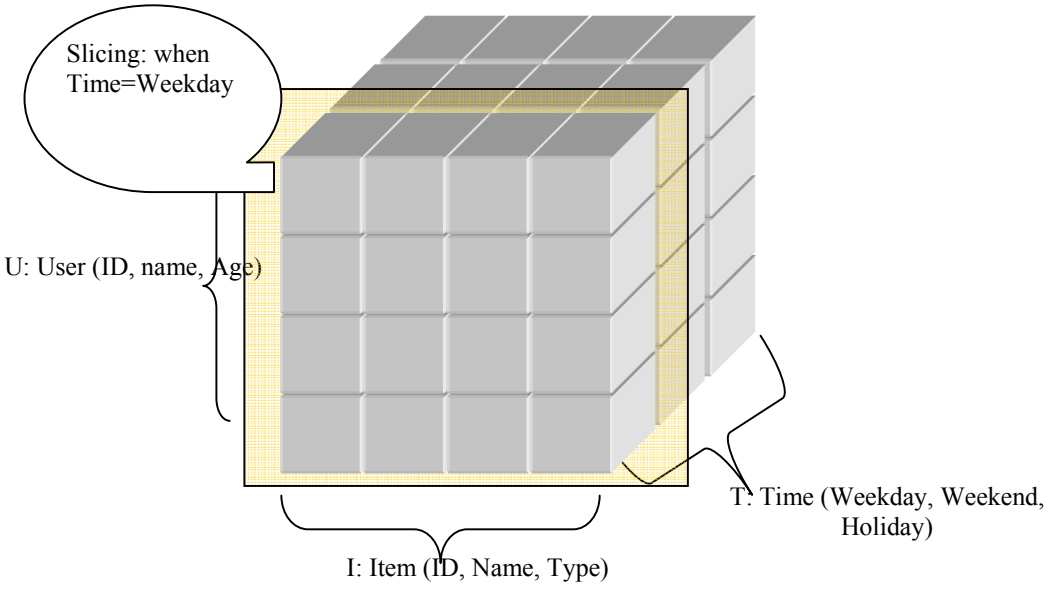


Figure 2-2 Demension Reduction in the Multi-dimensional RS

The most recent effort in incorporating context information in making a recommendation is a study by Lekakos and Giaglis [2006], in which users' *lifestyle* is

considered. Lifestyle includes users' living and spending patterns, which are in turn affected by some external factors such as culture, family, etc., and internal factors such as their personality, emotions, attitudes etc. Users are exposed to a number of advertisements picked up from seven product categories such as food and drink, books and magazines etc, to obtain their lifestyle information. The system will first compute the *Pearson correlation of users' lifestyles* to relate one user to another. In particular, the closeness between users is measured in terms of their lifestyle instead of ratings in traditional CF: the chance that users with the same lifestyle tend to have similar tastes will be higher. This process is formulated as follows (Equ. (3) in [Lekakos and Giaglis 2006]):

$$w(i,j) = \frac{\sum_k (I_{i,k} - \overline{I_i})(I_{j,k} - \overline{I_j})}{\sqrt{\sum_k (I_{i,k} - \overline{I_i})^2 (I_{j,k} - \overline{I_j})^2}} \quad (2\text{-}7)$$

where, $I_{i,k}$ and $I_{j,k}$ represent the kth common lifestyle indicator for the target user and jth users, $\overline{I_i}$ and $\overline{I_j}$. And the method to determine whether the target user and the candidate user have a common lifestyle indicator is through a simple keyword match. Based on the similarity measures, the system will select those users who score above a certain threshold. After this filtering process, the system will make predictions on items for the target user based on ratings from neighbors. This approach is essentially similar to that in [Pazzani 1999]: make use of the additional information (i.e. lifestyle, demography) to determine the closeness between users. Similar to these approaches, several of the recommendation techniques proposed and tested in this thesis for paper recommendation

make use of the learner profile to identify and group neighboring users. Details will be given out in the Chapter 4.

Table 2-5 below summarizes the three major approaches in multi-dimensional CF.

Table 2-5. A summarization of three major approaches in multi-dimensional CF

	Type of additional information	Data type of finding neighbors for a target user	Ease of applicability
Adomavicius et al. 2005	• users' demographic data, • item information, • consuming information	ratings	difficult
Pazzani 1999	• users' demographic data	user profile	easy
Lekakos and Giaglis 2006	• users' demographic • lifestyle data	user profile	medium

In fact, in non-traditional CF, we attempt to formulate the *neighborhood* of a target user through learning and combining the various aspects of their profile (demographic data, lifestyle etc). Then, based on the ratings of these neighbors, the system makes predictions on items.

2.4 Discussions on Evaluating Paper Recommendation Systems

Since the studies on paper recommenders, especially those taking into account pedagogical considerations, stress the necessity of satisfying users' needs instead of purely measuring the performance of a recommender based on those subjective evaluation techniques (such as precision, recall, MAE etc), the evaluation strategies are mainly subjective ones, through pre-, post- questionnaires[Recker et al. 2003; McNee et al. 2002; Torres et al. 2004; Bradshaw et al. 2000; Brusilovsky et al. 2005], interviews [Bradshaw et al. 2000], unobtrusive observations [Brusilovsky et al. 2005], etc. Basu et al. [2001] evaluate their paper recommender by comparing the predicted and actual ratings

of papers for a target reviewer. The Altered Vista system [Recker et al. 2003] was tested by 15 registered students in a class for 3 months, and evaluators were required to subjectively vote for each of the web sites filtered through Altered Vista on a 5 point Lickert-scale. [Woodruff et al. 2000] purely rely on *experts' ratings* (book authors in the area) towards the documents to determine the effectiveness of their recommender. In particular, since their recommender presents the recommendation in a ranked list, with the top one indicating the most suitable, precision is measured at each rank *r*. For instance, assume an algorithm returns a not useful top-ranked recommendation and a useful second-ranked recommendation, then the precision at rank 1 would be 0, and the precision at rank 2 would be 1 divided by the rank 2 (i.e. 1/2). Intuitively, the *precision at each rank* tells the percentage of good vs. not good ranked items made by the algorithm. When we want to calculate the *aggregate precision* for a given algorithm, we will have to sum up the precision values at all ranks divided by 10 (among a 10-ranked list). Clearly, this computation will bias those algorithms that return useful items in the early ranks.

The research paper recommenders in [McNee et al. 2002, Torres et al. 2004] make recommendations for a target paper based on citation analysis. [McNee et al. 2002] conducted both subjective and objective evaluations for their research paper recommender.

In the subjective evaluation, for a given research paper the recommender returns the author of the paper (him/her-self) was asked to evaluate the relevancy and familiarity of the returned citation in relation to the target paper that he/she has written through two questions: '*how relevant is this citation to your paper and its related work*?' and '*how familiar are you with this citation*'? They then compare the total number of relevant vs.

non-relevant recommendations to assess the quality of the recommender. In the objective evaluation, the '*all but one*' protocol is employed; that is, for each paper in the test dataset, McNee et al. [2002] randomly removed one citation, and the remaining citations were used to generate a list of recommended citations. They then compared how early (out of the ranked generated citations) the removed citation would appear. Intuitively, the earlier the removed citation appeared in the recommended list, the better the quality of the recommendation. This '*all but one*' protocol, first proposed by Breese et al. [1998], has been widely used in many other studies in the area as a way to compare the actual and predicted ratings of items (see, among others, [Herlocker et al. 1999; Tang et al. 2005; Torres et al. 2004; Lekakos and Giaglis 2006; Rashid et al. 2002]). [Torres et al. 2004] used similar evaluation methodologies as those in [McNee et al. 2002] except that ordinary users were invited to evaluate the relevancy of citations instead of expert authors of the papers.

[Torres et al. 2004] execute the same evaluations as those in [McNee et al. 2002], with more users with different research experiences including graduate students, professional researchers and professors. Obviously, by including more users with different research background, their judgments on the quality of citations vary. That is, the quality of a citation can be at different levels as a function of users' research experiences. As is pointed out by [Rieh 2002, Custard and Sumner 2005], people make judgments based on both the *information authority* and *cognitive authority* of the item. The former characterizes the extent to which users think that the item is '*useful, good, current and accurate*' [Rieh 2002], while the latter encompasses the extent to which users think that they can trust the item [Rieh 2002].

The effectiveness of *Rosetta* [Bradshaw et al. 2000] was measured through subjective evaluations by 6 graduate students. In particular, based on the abstract of a paper, subjects were asked to query *Rosetta* to find the paper. Results were drawn from subjects' answers to a questionnaire consisting of questions such as: 'did you find the paper?', 'what queries did you use?' 'what number was the paper ranked in the returned list?'. The last question is used to assess the ability of *Rosetta* in returning the paper just as the judgment we have toward Google™: how early on in the list of items returned does the returned result match our keywords.

Knowledge Sea II [Brusilovsky et al. 2005] performed both subjective and objective assessment to evaluate the quality of the social-navigation and annotation based educational resources. In the subjective assessment, 15 students were asked to agree or disagree with the annotated resources (thumbs-up or thumbs-down). In the objective evaluation, student activity logs were statistically analyzed to look at the usefulness of the annotated educational resources in the learning process, for instance, the total 'click' traffic of those densely annotated resources.

In summary, the evaluation protocols and methodologies used by previous work on paper recommendation tend to be more useful and appropriate to the purposes of this thesis when compared with those employed in movie or music recommendations. Table 2-6 below summarizes these evaluation approaches.

In fact, evaluations should be designed to appropriately reflect the tasks that the RS supports as well as the users of the system, for instance, how experienced researchers can find the most related citations in [McNee et al. 2002, Torres et al. 2004]; research papers can be matched to relevant users in Rosetta [Bradshaw et al. 2000]; the most

appropriate articles can be found in relation to the user's current task [Woodruff et al. 2000], etc. These studies [McNee et al. 2002, Torres et al. 2004, Woodruff et al. 2000, Brusilovsky et al. 2005, Bradshaw et al. 2000, Recker et al. 2003] tend to focus more on *user acceptance* rather than *pure prediction accuracy*, which aligns with *bottom-line measurement* of the quality of a RS [Herlocker et al. 2004].

Table 2-6. A summarization of evaluating a paper recommender

	Evaluators	Subjective evaluation	Objective evaluation	
Basu et al. 2001	expert	x	√	Mean Absolute Error
Recker et al. 2003	students	√	x	
Woodruff et al. 2000	expert	√	x	
McNee et al. 2002	expert	√	√	citation rank
Torres et al. 2004	• graduate student • professional researchers • professors	√	√	citation rank
Bradshaw et al. 2000	students	√	x	
Brusilovsky et al. 2005	students	√	√	click traffic

It is clear that there are a tremendous number of interesting issues to study in the RSs area ranging from algorithmic issues to evaluative strategies. This thesis will study the following two fundamental issues derived from some of the previous studies in the IR and RS areas: what do learners need in pedagogical RSs? and what are the important factors tutors should consider in making paper recommendations?

Fundamentally, the most important, interesting and yet intractable question in both IR and RS areas is: what do users want (which needs to be considered both before

and after of the recommendations)? More and more researchers and practitioners have come to realize the fact that RS is not merely about how to make the recommended item 'accurate' [Riedl and Dourish 2005; Adomavicius and Tuzhilin 2005; Herlocker et al. 2005; McNee et al. 2006a]. Indeed, the same intriguing issue arises in the IR area as to what constitutes the 'quality' and 'relevance' of the documents retrieved. Even in the data mining area [Kobayashi and Takeda 2000; Crestani et al. 1998; Rieh 2002; Paepcke et al. 2000; Rieh and Belkin 2000; Klobas 1995], researchers have also come to the core belief in determining how to value the recommendations after years of focus on the algorithmic aspect of RSs.

Drawn from these studies, in this thesis we attempt to design and execute appropriate evaluation protocols and methodologies for making recommendations.

2.5 The Ecological Approach to the Design of Recommender Systems in E-Learning Environments

[McCalla 2004, Tang and McCalla 2005a] pointed out that in the majority of web search and content development it is assumed that the contents are attached by editors (such as teachers or tutors in an e-learning context), e.g. using the Dublin Core metadata. Common metadata include Title, Creator, Subject, Publisher, References, etc. [Weibel 2999]. These metadata (sometimes referred to as item-level annotations) are mainly used to facilitate information retrieval and interoperabiity of the distributed databases, and hence need only be in machine-understandable format. Others have studied automatic metadata extraction, where parsing and machine learning techniques are adapted to automatically extract and classify information from an article [Han et al. 2003, Lawrence et al. 1999]. Others have also utilized the citations in a paper for recommending another [Torres et al. 2004], or providing its detailed bibliographic information to the user, e.g. in

ACM DL or CiteSeer [Lawrence et al. 1999]. Since those metadata are not designed for pedagogical purposes, sometimes they are not informative enough to help a teacher in selecting learning materials [Sumner et al. 2003]. In addition, McCalla [2004] argues that the metadata-based approach attached to the content suffer from a number of problems in education domain:

- it tend to focus more on the form and the content of the item, even though the item can be utilized for several purposes;
- it cannot reflect the pedagogical uses by different learners during their cognitive development;
- it cannot capture the changing value of the items 'viewed' by learners through their interactions with the system.

To deal with these, McCalla [2004] proposes an alternative called the *ecological approach*, where information about an item in the domain is attached as learners access it. The information includes:

- information about the learners regarding their cognitive, social, emotional characteristics and their goals;
- information about the usage of the content, including those through implicit interest indicators such as dwell time, number of click strokes etc,
- information about the social contextual of use, including access to a target learner and other learners' (similar or dissimilar to the target learner) experiences with the item.

In the ecological approach, user models should be constructed to reflect the purposed-based use [McCalla et al. 2000, Vassileva et al. 2003] of each item to support

learners pedagogically. And how the items are used by learners constitute the contextual information surrounding the same items, and thus very valuable for the system to take a number of actions such as recommending relevant content for a particular learner with a particular goal in mind, tailoring the item to meet a particular learner's goals and/or needs etc, determining the semantic relationship such as similarity between the item and others etc. As such, learner models are computed on-the-fly based on the various purposes of the learners; this type of learning modeling referred to as *active learning modeling* [Vassileva et al. 2003] is different from traditional method focusing on constructing the representation of a comprehensive single learner model, and thus is preferred.

The ecological approach can be regarded as an enhancement to the typical rating-based CF approach in which numerical rating is attached to each item, which motivates the work in this thesis: making paper recommendations based on the pedagogical features of learners. In particular, in the pedagogical paper recommender, each paper is annotated with a multiple dimension of ratings reflecting its pedagogical uses; in turn, a paper will be recommended to learners to meet their pedagogical goals instead of simply matching against their interest.

CHAPTER 3

ANALYSIS OF USERS AND THEIR RATINGS

In this thesis, our focus is to propose a Pedagogical Paper Recommender System and identify a set of issues that charaterize the system space. Then using a set of recommendation techniques that are unique in this domain, through an the experimental study, we explore these issues within the system space. One main goal of this study is to explore the relationship between learner satisfaction and paper features.

In this chapter we will give an overview of the data collected, focusing on drawing a general picture of the users and their ratings, in order to explore the charateristics of pedagogical paper recommendation. We will start by describing the goal and hypothesis of our experimental study in subsection 3.1, followed by the experiment setup in subsection 3.2, design and general analysis of learner ratings in subsection 3.3, correlation analysis among learner ratings in subsection 3.4. In the next chapter, we will describe our recommendation algorithms and the particular experiments that have been conducted.

3.1 Goals and Hypotheses

The goals of our experimental study are (i) to explore the characteristics of pedagogical paper recommendation, and (ii) to test various recommendation techniques in a pedagogical context. The first goal differentiates our study from other paper recommendation system research in non-learning environments, while the second goal compares traditional recommendation

techniques with our proposed ones. We will focus on the first goal in this chapter, and leave the second to Chapter 4.

With the first goal in mind, we make the following conjectures:

Conjecture 1. *In the context of education, some learners are willing to learn topics that are not very interesting from their perspective.*

Conjecture 2. *The overall rating given by learners to a paper may not only depend on the interestingness of the paper from their perspective, but also on the richness of knowledge that has been gained by them from reading the paper and/or the usefulness of the paper in helping them to understand the course subject.*

Conjecture 3. *The intent of learners in recommending a paper to others may not only depend on the interestingness of the paper from their perspective, but also on the richness of knowledge that has been gained by them from reading the paper and/or the usefulness of the paper in helping them understand the course subject.*

Conjecture 4. *The closeness of a learner's job to the paper topics may also affect their overall ratings on that paper and their intent of recommending it to others.*

Conjectures 2, 3, and 4 are an extension of Conjecture 1 with respect to paper recommendation.

3.2 Experiment Set Up: Data Collection and Preprocessing

The experimental study was conducted in a fall course on software engineering, COMP 5211, which is an introductory software engineering course for Masters-level students. The course was offered in the Hong Kong Polytechnic University, in academic year 2005/2006, from September 2005 to December 2005. The following subsections explain the experimental process in detail.

3.2.1 Paper Collection and Paper Modeling

Matching a pre-determined course syllabus, the tutor carefully selected 26 papers, and among them, 21 papers were suggested to the students[1].

Table 3-1. List of papers' topics and their publications

Paper	Related topics	Journal/magazine name
P1	Requirements Engineering	IEEE Software
P2	Project Management; Software Quality Management	Communication of the ACM
P3	Requirements Engineering	IEEE Software
P4	Requirements Engineering; Software Quality Management	IEEE Software
P5	Software Engineering and Business	IEEE Software
P6	Requirements Engineering; Agile Programming; Project Management	IEEE Software
P7	Project Management	IEEE Software
P8	User Interface Design; Software Testing; Web Engineering	ACM Interactions
P9	Search Engine; Recommender Systems	IEEE Internet Computing
P10	Web Engineering; User Interface Design	IEEE Software
P11	Web Engineering; User Interface Design; Software Testing	ACM Interactions
P12	Web Engineering; User Interface Design	ACM Interactions
P13	Web Engineering; User Interface Design; Software Testing	ACM Interactions
P14	Web Engineering; User Interface Design; Software Testing	ACM CHI (Conference)
P15	User Interface Design; Software Engineering	ACM Interactions
P16	Web Engineering; User Interface Design; Software Testing; Case Study	ACM CHI (Conference)
P17	Web Engineering; Software Testing	IEEE Computer
P18	Software Testing	IEEE Software
P19	Project Management; Quality Management; Case Study	IEEE Software
P20	Web Engineering	ACM Interactions
P21	Project Management; Case Study	ACM Interactions

[1] The papers were not selected for this study; in fact, it was selected before this study begun and was based on the course syllabus. In order to avoid data contamination, students were not informed that their feedbacks would be used for this study. They were told so when they signed the consent forms.

The remaining five papers were left out due to the time constraints. These papers were selected not for our experimental purpose, but rather as useful for teaching and learning in this course. In fact, 20 out of the 21 selected papers were also used as reading materials for the same course in year 2004/2005, a year before this experiment was conducted. Appendix A lists the papers used in our experiments, where Table 3-1 summarize the papers in terms of the main topic and publication place.

3.2.2 Subjects/Participants and Learner Model

Twenty eight students signed the consent forms allowing us to use their data for experiments. At the beginning of the course, they were required to fill in a questionnaire (see Appendix B), in order to obtain their user models (profiles). Three students did not fill out the questionnaire, so altogether there were 25 valid user models for our experiments.

Figure 3.1 tabulates the average learner interest on each of the 13 major course topics where the maximum rating 5 represents that the learner is very interested in that topic, and the lowest rating 1 represents that s/he is not interested in that topic at all.

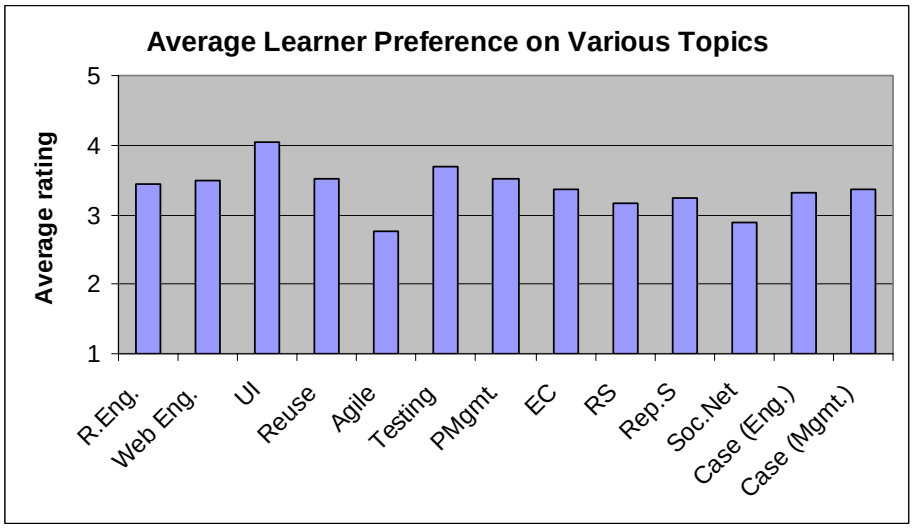


Figure 3-1 Average learner preferences on various topics (quantified in ratings from 1 to 5)

Along the x-axis are the major topics: software requirement engineering (R. Eng.), Web design and Web engineering (WebEng.), user interface design (UI), software reuse (Reuse), agile programming (Agile), software testing and quality management (Testing), project management and managing people (PMgmt), e-commerce/e-banking/online shopping (EC), search engines and recommender systems (RS), trust and reputation systems on the Internet (RepS), social networking (including chat, collaborative work) (Soc. Net.), a case study in engineering issues (Case (Eng.)), and a case study in management issues (Case (Mgmt)). It shows that the most popular topic was user interface design (UI), with an average rating of almost 4.04; then comes software testing and quality management (Testing), with an average rating of 3.68. The least interesting topic was agile programming (Agile), with an average rating of 2.8.

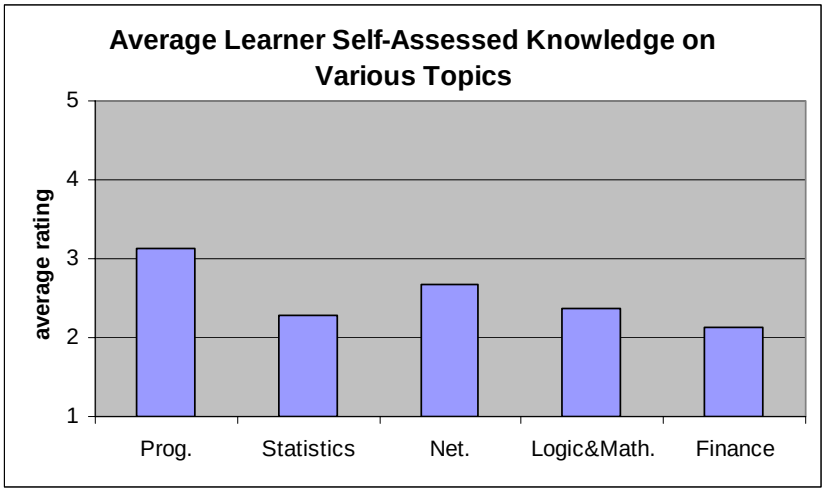


Figure 3-2 Average learner self-assessed knowledge on various topics (quantified in ratings from 1 to 5)

Figure 3-2 shows the average ratings of learner knowledge background. Here, the maximum rating 5 represents that the learner felt that s/he is very familiar with that topic and the lowest rating 1 represents that s/he felt that s/he is not familiar at all with that topic. The topics listed in the horizontal lines are: programming (Prog), statistics (Statistics), networks and the

Internet (Net.), logic and mathematics (Logic & Math.), and finance and management (Finance). Most learners felt that they have a moderately strong background in programming, with an average rating of 3.12.

Figure 3-3 shows the job categories of the learners, with most working in the area of programming/coding (Prog.). Software development (Soft.Dev.) is the second largest catgegory. The fewest number of people work in academics/teaching (Academic) and in the engineering (Eng.) area. The type of learner job may relate to their preference on various topics as well as their background knowledge. As stated in Conjecture 3, we believe that a learner's job type may affect their preferences in assessing a paper.

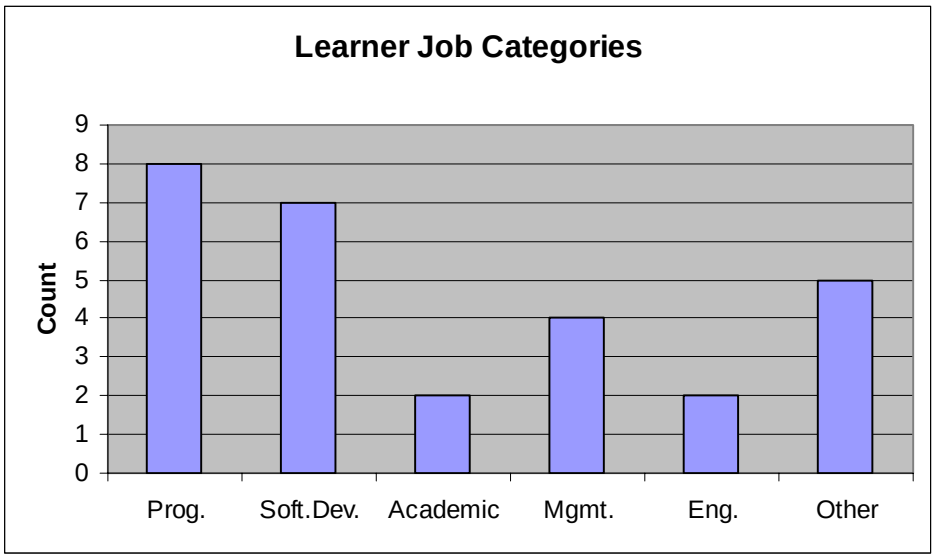


Figure 3-3 The number of learners engaging in various jobs, from left to right: programming (coding), software development (testing, design, etc.), academic (teaching), management (including marketing and finance), engineering (electrical, mechanical, etc.), and other

Finally, Figure 3-4 shows the areas in which the learners want to learn more, i.e. their goals/expectations in taking this class. They may choose more than one area in answering this part (see the questionnaire in Appendix B). Sometimes learner interests and goals are apparently

contradictory, for example rating a subject highly but not wanting to learn more about it. Intuitively, an interesting topic may not necessarily be part of the learning goals if the learner believes that s/he already knows a lot on that topic. Instead, s/he may want to learn more on some areas related to her/his job even if s/he may not be all that interested in them.

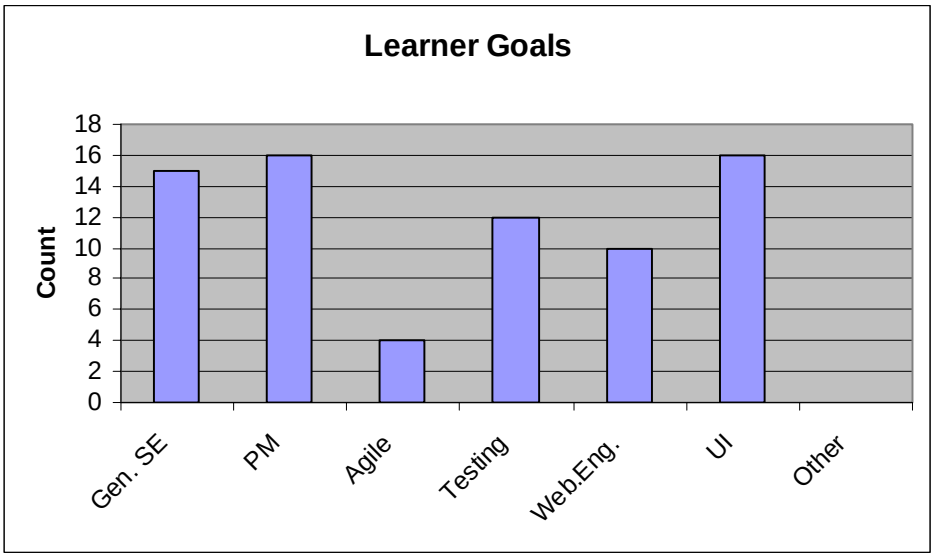


Figure 3-4 The number of learners who are interested in learning specific areas (they may choose more than one), from left to right: general topics on software engineering, software project management, agile programming, software testing, web engineering, user interface design, and others (specified by learners)

The result of our consistency check is shown in Figure 3-5. 16 out of 25 learners show various degree of inconsistency, which means those learners claim that they like a topic which is not their learning goal, or they want to learn a topic which is not an interesting one. The degree of inconsistency is measured as the number of inconsistent topics divided by 5 (i.e. the total number of specified topics: agile programming, Web engineering, software project management, software testing, and user interface design). A category “other” is used to count a learner who filled in “moderate” for all topics.

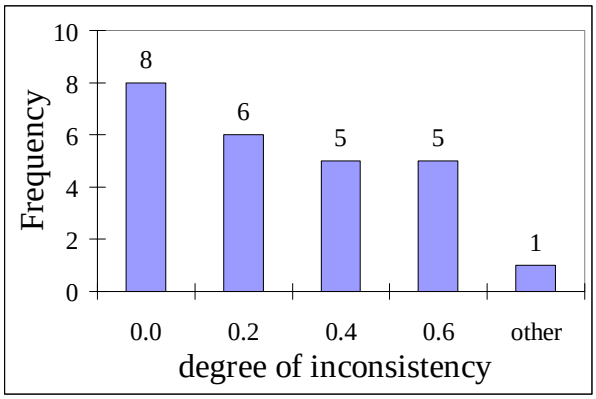


Figure 3-5 The number of learners who are inconsistent in their interests and goals

If inconsistent learners are those with degree of inconsistency > 0, then we have 16 inconsistent and 8 consistent learners (excluding 1 in the category "other"). Thus, given the actual ratio of inconsistent and consistent learners is 16:8 and *df* = 1 (we only have 2 categories: inconsistent and consistent), the goodness-of-fit chi-squared test gives *p*-value = 0.05 when the expected ratio of inconsistent and consistent learners equals to 11.21:12.79. In other words, our data cannot reject a null hypothesis that 44.84% (=11.21/24) learners are inconsistent at significant level α = 0.05. Certainly, the number (44.84%) in this null hypothesis is high, because we include all degree of inconsistency (0.2, 0.4, and 0.6). If we restrict our criteria of inconsistency to only those with degree = 0.6 (only 5 learners), then our data cannot reject a null hypothesis that 8.88% learners are inconsistent, also at significant level α = 0.05. In either case, the results support our Conjecture 1; hence, with respect to the population of our sample, we may safely claim

Conclusion 1. *In the population of graduate students, some learners are willing to learn topics that are less than interesting from their perspective.*

However, it is hard to say that whether or not this conclusion can be generalized to other graduate courses or similar software engineering course in other universities. A further empirical study is needed in the future.

3.2.3 Learner Ratings

During the course, after reading each paper, learners filled in a feedback form (see Appendix C). Basically, the feedback form consists of three parts. The first part asks the learner to describe the topics of the paper read by him/her so we can assess his/her understanding or seriousness in reading it. The second part, the main part of this questionnaire, asks the learner to rate the paper in various aspects (interesting, difficult, helpful, etc.). Finally, the third part is a comment part where learners can write down their critical comments after reading the paper. This part is the basis for evaluating the learners' reading assignments.

We use a 4 point Likert scale in all questions in the second part, except for the last question (question 7) on the peer-recommendation issue. The reason for using an even-number scale is to prevent a mid-choice by the learner, that is a choice that provides essentially no information. The following subsections describe each question and its general results.

3.3 Analysis of Learner Ratings

3.3.1 Paper Difficulty

When we ask learners whether a paper is difficult or not, the available options are (4) "very difficult", (3) "difficult", (2) "easy", and (1) "very easy". We do not provide a middle option "moderately difficult" between "difficult" and "easy" in order to obtain a clear decision, either difficult or easy. Figure 3-6 shows the frequency of ratings given by 25 learners to 21 papers. Overall, there are 13 missing ratings; thus, totally only 512 ratings are recorded. It is shown here that the majority rating is option (2) "easy" and approximately 20% is (3) "difficult".

However, this only shows the general composition of learner ratings on paper difficulty. Detailed information on each paper itself is shown in Figure 3-7. Note here that learner identities have been anoymized to maintain their privacy.

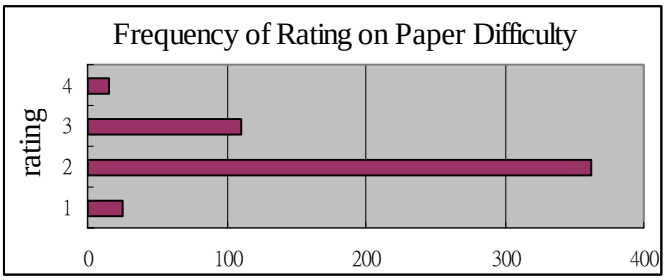
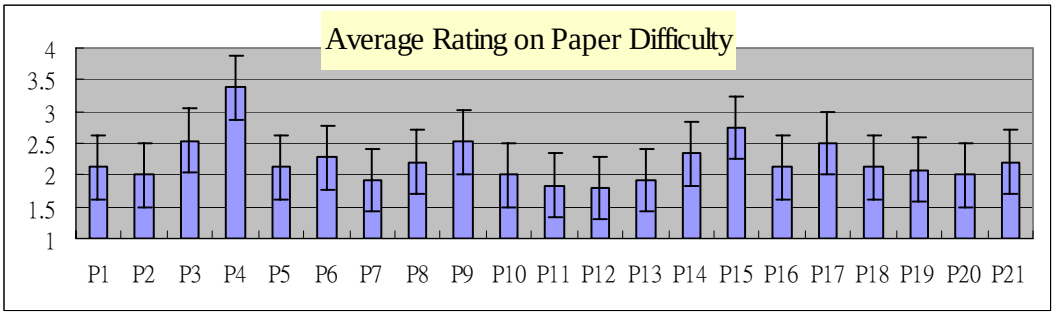
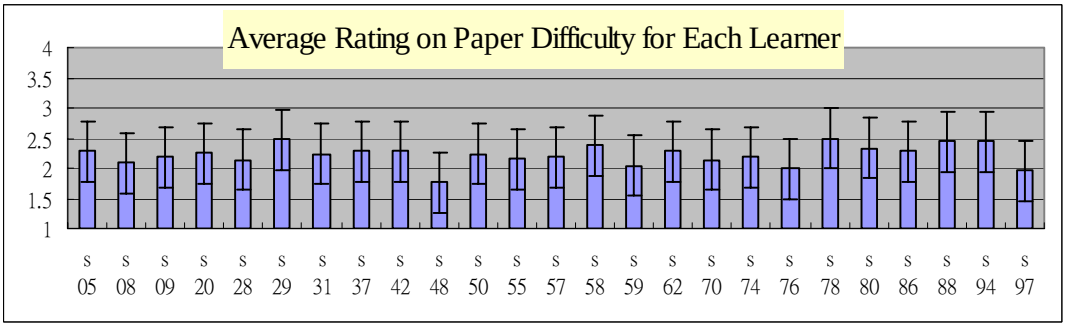


Figure 3-6 The frequency of ratings on paper difficulty given by 25 learners to 21 papers



(a)



(b)

Figure 3-7 The average ratings on paper difficulty for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

Figure 3-7(a) shows the average ratings on each paper by all learners, and Figure 3-7(b) shows the average ratings over all papers given by each learner. The average ratings are taken by averaging the numbers for each option, i.e. (4), (3), (2), and (1). These average values do not represent the absolute difficulty of each paper to the learners because each learner has only four options to choose, with no in-between options. However, they can be used for statistical comparison across papers, with a higher average rating meaning that the paper is more *likely* be a difficult one for average learners, assuming a linear scaling (an assumption we apply thoughout our analysis).

From Figure 3-7(a), if we assume that the average rating of difficult papers is 3 or above and a relatively difficult one receives ratings between 2.5 (a midpoint between 1 and 4) and 3, then only the fourth paper (P4)[2] is considered as a difficult one (average rating = 3.375) and four others (P3, P9, P15, and P17)[3] are considered as relatively difficult (average rating between 2.5 and 2.75) (see Figure 3-7(a)). The rest are considered as relatively easy or very easy (average ratings below 2.5).

Moreover, it is shown from Figure 3-7(b) that *on average*, the learners give ratings lower than 2.5, which means *on average* they can understand the papers easily. Unlike the average ratings on each paper which varied between 1.79 and 3.38 (standard deviation = 0.362), the average ratings given by each learner are less varied (standard deviation = 0.171) and within the range [2, 2.5] except for learner "s48" (= 1.76) and learner "s97" (= 1.95). This suggests that most learners felt largely the same about their understanding of papers.

[2] P4 is a paper related to requirements engineering for ERP (Enterprise Resource Planning) systems. The majority of learners stated in the rating forms that this paper is 'very boring, and…difficult to understand'.

[3] P3 is a paper discussing the state of the art of requirements engineering; P9 discusses the design of Amazon.com's recommendation engine; P15 focuses on human centered design and skills for software engineers; P17 is the seminal article by Jakob Nielsen on usability engineering. Interestingly, P17 is very much well received by learners despite its difficulty to understand, while P4 receives the lowest average rating from learners (refer to the section 3.3.6 discussion on the overall ratings).

3.3.2 Relation to Learner Job

When we ask learners whether a paper is related to his/her job or not, the available options are (4) “very much”, (3) “relatively”, (2) “not really”, and (1) “not at all”. Because the relation of a paper to the learner’s job is vaguer than its difficulty, here we provide a vague option (3) “relatively” for learners who found that the paper is related to their job but cannot decide the level of its relationship. However, more than half of ratings (50.78%) fall in option (2) “not really” and another 12.11% fall in option (1) “not at all” as shown in Figure 3-8. This suggests that on average near 2/3 of papers are self-assessed as not really related to the learners’ jobs. In fact, only 5 papers (<25%) received an average rating greater than 2.5 (a value between “relatively” and “not really”) (see Figure 3-9(a)).

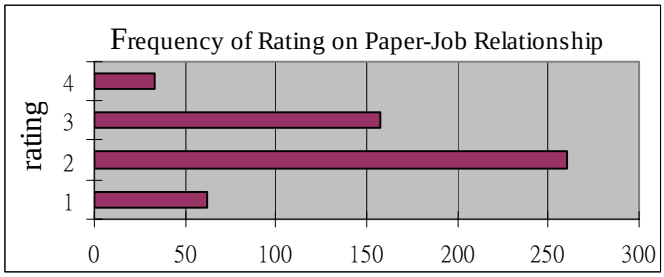
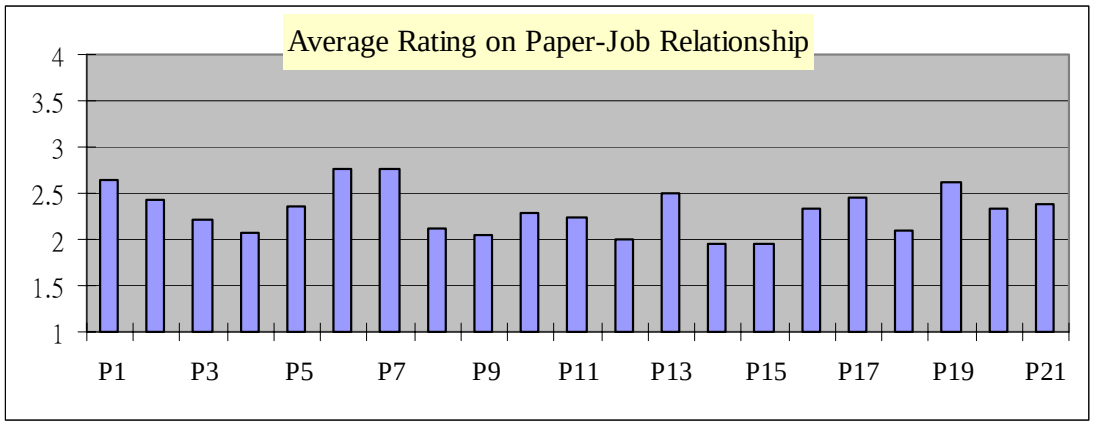
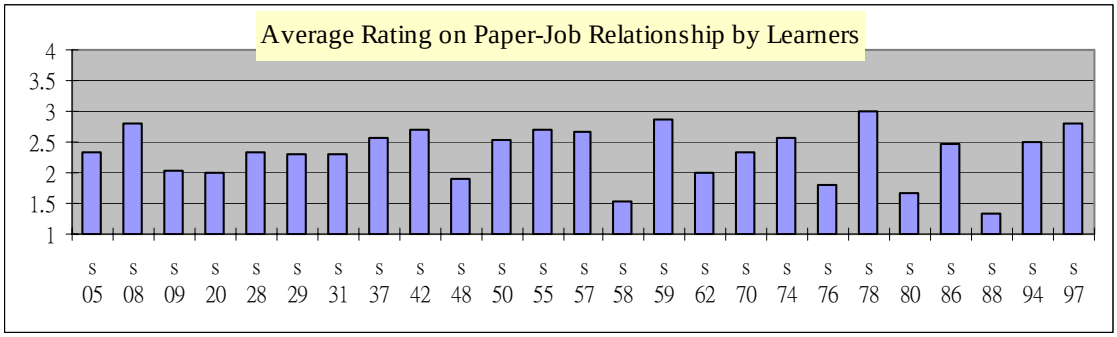


Figure 3-8 The frequency of ratings on job-relatedness given by 25 learners to 21 papers

Figure 3-9 shows the average ratings on job-relatedness for (a) each of the 21 papers and (b) each learner. Again, the average ratings in Figure 3-9 do not represent the absolute degree of the relation between the paper topic(s) and the learners’ job. Moreover, the pattern shown should be interpreted under assumption of a linear scale (i.e. 1 to 4) of its options.



(a)



(b)

Figure 3-9 The average ratings on job-relatedness for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

The standard deviations of data shown in Figure 3-9(a) and (b) are 0.252 and 0.439, respectively. A high variation of average ratings with respect to individual learners (Figure 3-9(b)) is due to some learners who work in the academic sector (e.g. s88) or managerial positions (e.g. s80) who have rated almost all papers as relevant to their job, while others who work as programmers (e.g. s08, s59, and s78) found that only very few papers were related to their job (notably P6 on the importance of prototyping and P13 on the 'myths' of software usability).

Throughout the thesis, we will use "Job_related" with a capital letter to represent the rating on this given by the learners.

3.3.3 Interestingness of Papers

The third question is to ask whether the paper is interesting or not. The available options provided to learners are similar to those in job-relatedness, i.e. (4) “very much”, (3) “relatively”, (2) “not really”, and (1) “not at all”. 60.94% of learner ratings fall in option (3) “relatively” and another 11.13% falls in option (4) “very much” as shown in Figure 3-10; thus, fewer than 28% were evaluated as not (really) interesting.

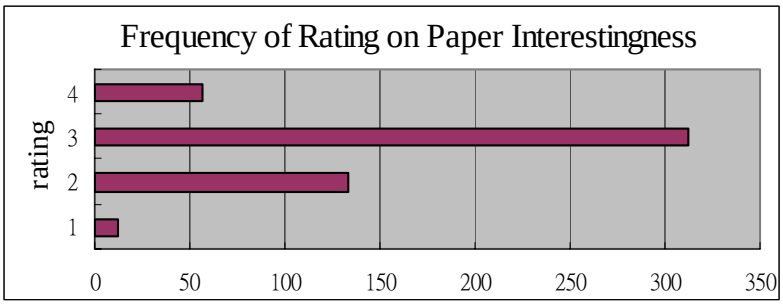
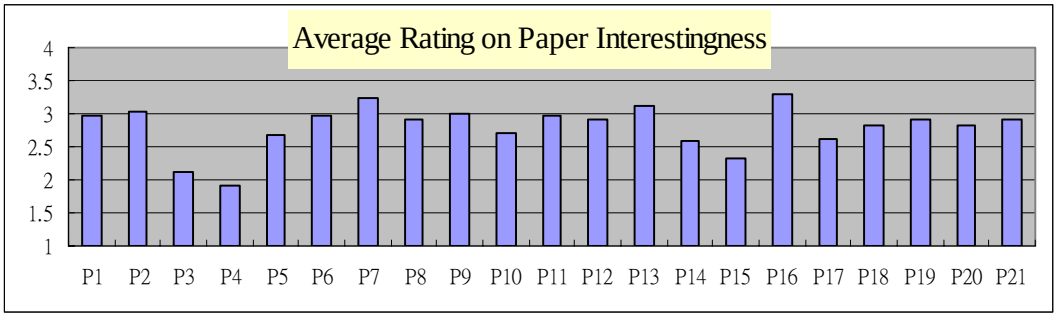
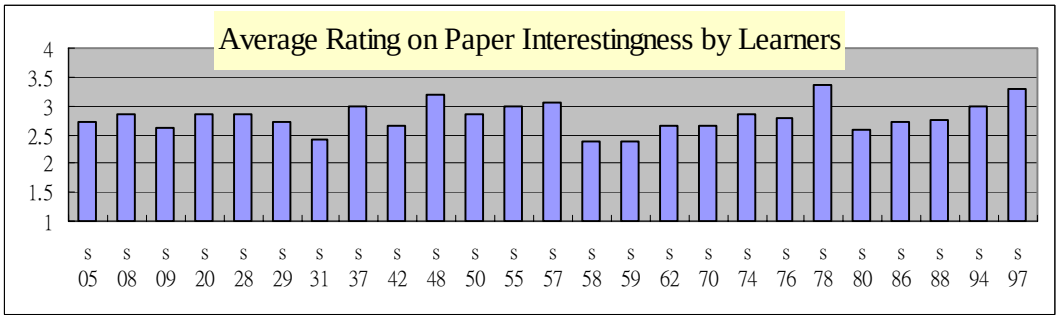


Figure 3-10 The frequency of ratings on paper interestingness given by 25 learners to 21 papers

Figure 3-11(a) and (b) show the average ratings on paper interestingness for all 21 papers and all 25 learners, respectively. From this figure we can see that on average only three papers have average ratings less than 2.5 (a value between “relatively” and “not really”) and also only three learners have given average ratings less than 2.5. The fourth paper (P4) on requirements engineering and software quality management (for ERP systems) received the lowest average rating (= 1.92). The standard deviations of average ratings for each paper and by each learner are 0.34 and 0.253, respectively. We have performed a correlation analysis between the learners’ average ratings (data in Figure 3-11(b)) and the learner self-assessed interests on various topics (namely *the estimated average paper interestingness by topics*), which will be explained below.



(a)



(b)

Figure 3-11 The average ratings on paper interestingness for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

The estimated average paper interestingness by topics

From the learners' self-assessment on each topic (see section 3.2.2), we obtained the learners' preferences on 13 different topics. Since each paper is categorized by its topic(s), we can estimate whether a paper should be interesting or not from a learner's perspective. This estimated value for each learner can be calculated as the weighted sum of his/her interests:

Interest = $\sum$ *user_model(i) * paper_model(i)* for all topics *i*

where *paper_model(i)* are the values set by tutors, and consistent with paper topics.

Applying the same computation for all papers, we can obtain the *estimated average paper interestingness by topics* for all papers for a particular learner. We use "by topics" because this estimation is calculated according to the matching of paper topics and user preference on each topic. Figure 3-12 shows the estimated average paper interestingness normalized to values between 2 and 5 for each learner.

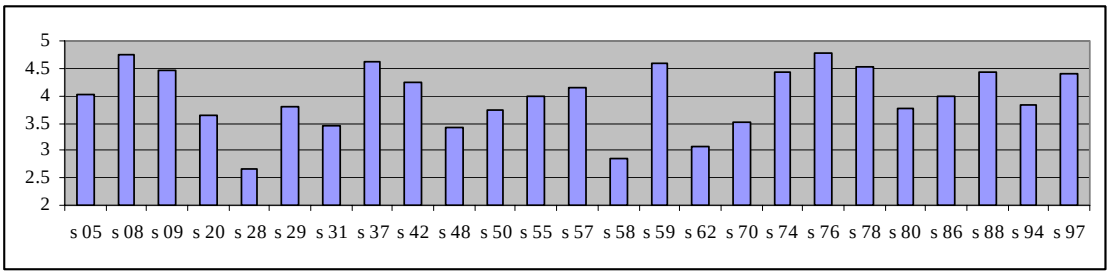


Figure 3-12 The estimated average ratings on paper interestingness for the first learner (S05) to the 25th learner (S97)

Intuitively, if the learner's preferences on various topics greatly affects his/her rating of the interestingness of each paper, then this estimated value should be correlated to his/her rating of the interestingness of each paper (i.e. Figure 3-11(b)). However, the calculated Pearson correlation between data in Figure 3-11(b) and Figure 3-12 is 0.262, which is a weak positive correlation. Three factors may cause this discrepancy:

- A learner's rating on paper interestingness is not merely affected by the topics covered in each paper, but also the paper's contents, writing style/presentation, difficulty, etc.
- The learner's self-assessment of their preference is biased, e.g. drifted according to their learning experience.

In this thesis, we will not further pursue which factor(s) are actually affecting this correlation. Rather, we leave it for future work. Throughout the thesis, we will use "Interest" with a capital letter to represent the rating on this given by the learners.

3.3.4 Aiding Learners in Understanding Course Subject

The fourth question is to ask learners is whether the paper is helpful or not in understanding the Software Engineering concepts and techniques they have learned in the class. The available options provided to learners are similar to those in paper-job relationship and interestingness, i.e. (4) “very much”, (3) “relatively”, (2) “not really”, and (1) “not at all”. From Figure 3-13 we can see that 76.56% of learner ratings fall in option (3) “relatively” and less than 18% as “not really helpful” or “not helpful at all”. This is not surprising because the papers have been selected carefully to help students in learning Software Engineering.

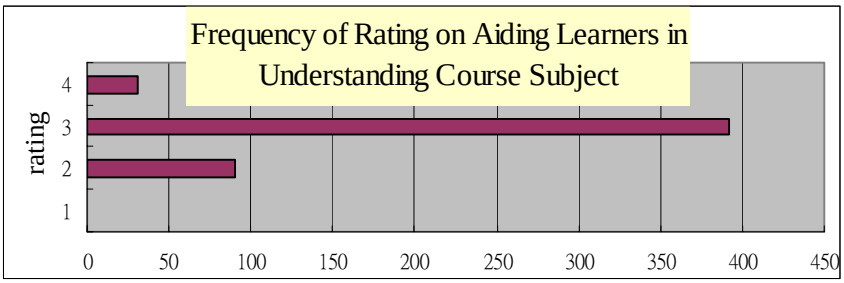
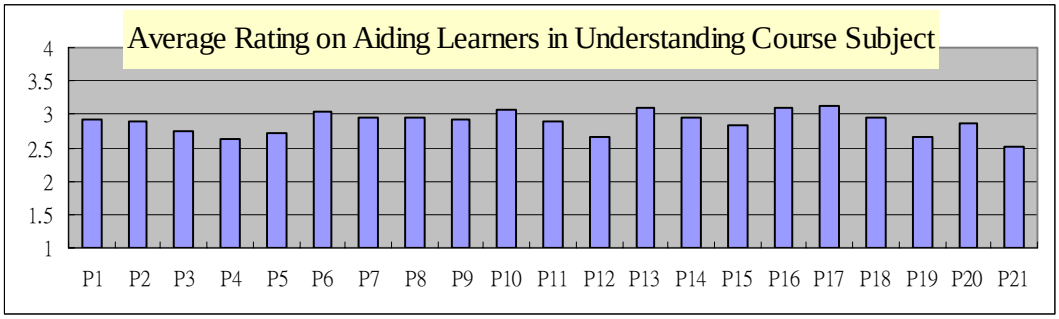
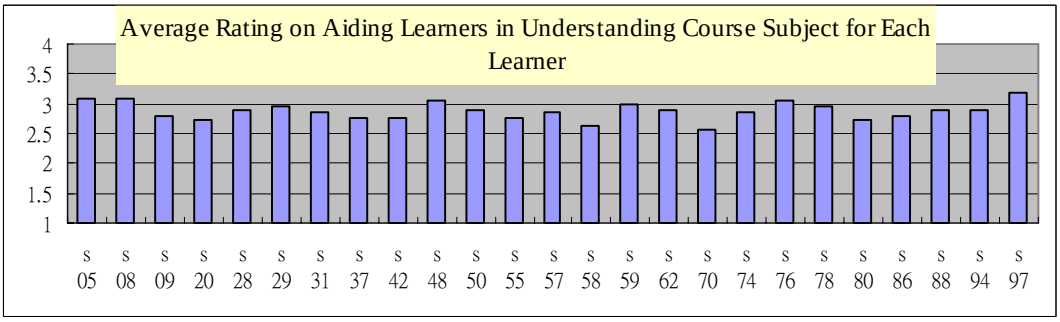


Figure 3-13 The frequency of ratings on paper contents in aiding learners to understand course subject (given by 25 learners to 21 papers)

Figure 3-14(a) and (b) show the corresponding average ratings for all 21 papers and all 25 learners, respectively. From this figure we can see that on average none of the papers or learners has an average rating less than 2.5 (a value between “relatively” and “not really”). If we interpret 2.5 as a cutoff for relatively helpful, then we conclude that on average those papers are relatively helpful for learners or they are relatively advantaged from reading those papers.



(a)



(b)

Figure 3-14 The average ratings on paper contents in aiding learners to understand course subject for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

The standard deviations of data in Figure 3-14(a) and (b) are rather small, 0.167 and 0.15 respectively, which means most papers are relatively useful and learners consistently agree on that.

Throughout the thesis, we will use “Aid_learning” with a capital letter to represent the rating on this given by the learners.

3.3.5 Learn Something “New”

The fifth question is to ask learners whether they have learned something new from the paper. The available options are (4) “absolutely”, (3) “relatively”, (2) “not really”, and (1) “not at all”. We use the term “absolutely” to accommodate learners who are confident that they have gained a lot of new knowledge/ information from reading a paper. However, since the options

are ordered in the same way as those in the second to the fourth questions, learners may interpret "absolutely" in this aspect in the same way as "very much" in other aspects (job-paper relatedness, interestingness, and helpfulness).

Figure 3-15 shows a similar pattern of learners' ratings on this aspect and a paper interestingness and "helpful". That is the majority of learners (roughly 79%) feel that they have learned something new them after reading the paper. However, approximately 21% of the ratings are for "not at all" and "not really", which suggests that some learners found nothing new in the paper.

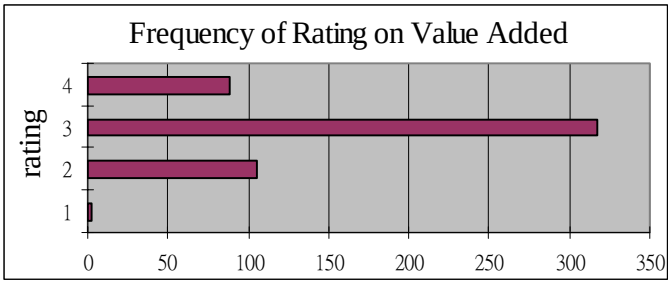


Figure 3-15 The frequency of ratings on paper value added (given by 25 learners to 21 papers)

Figure 3-16(a) shows that all papers actually get a relatively high average rating (higher than 2.5 with standard deviation = 0.202), and only the fifth paper (P5) with marginal rating (= 2.52). This paper (P5), focusing on the business perspective of developing a piece of software, is a short 3 page long guest editors' introduction to a special issue in IEEE Software, which has been chosen because it is relevant to the course (rank 9th among all 21 papers on job-paper relatedness, 16th on interestingness, and 17th on helping the learner understanding).

Figure 3-16(b) shows that two learners gave average ratings lower than 2.5 while ten learners gave average ratings greater than or equal to 3, which means they gained relatively different amounts of new knowledge/information from reading these papers. A high variation

(standard deviation = 0.322) is not surprising because some learners in this class may have worked in the software development industry for several years and have gained related knowledge, while others have never heard of such things.

Throughout the thesis, we will use “Value_added” with a capital letter to represent the rating on this given by the learners.

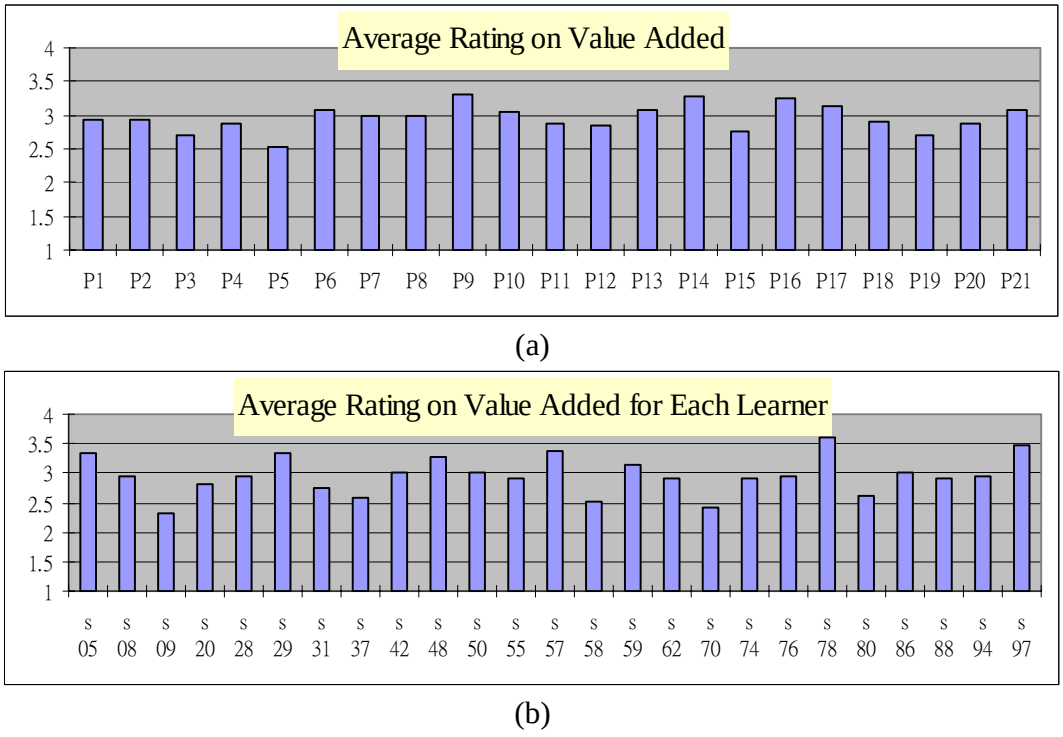


Figure 3-16 The average ratings on paper value added for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

3.3.6 Overall Ratings

The sixth question is to ask learners their overall rating toward a paper, which means their general impression after reading the paper. The available options are (4) “very good”, (3) “good”, (2) “relatively”, and (1) “bad”. We did not include a “very bad” option because we

predicted that extremely few papers would be rated “very bad”. We use “relatively” as the second lowest option, which can be interpreted as both marginal (relatively bad) and relatively good option at the same time. Thus, we can provide finer positive scales for some learners who tend to rate in a positive region. Figure 3-17 shows that the distribution of ratings is actually similar to those in “value added” or “paper interestingness”. Here, most ratings (near 93%) fall in (3) “good” and (2) “relatively” categories, i.e. 58.2% and 37.11% respectively, and only 3 ratings (0.586%) are “bad”.

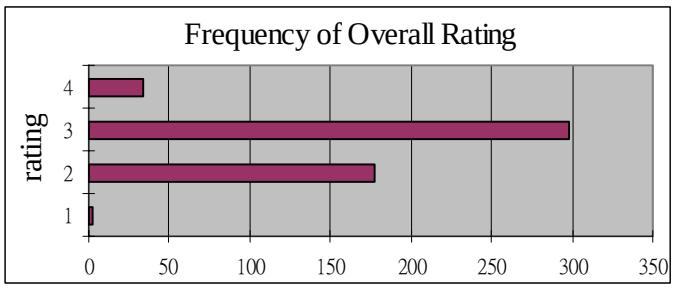


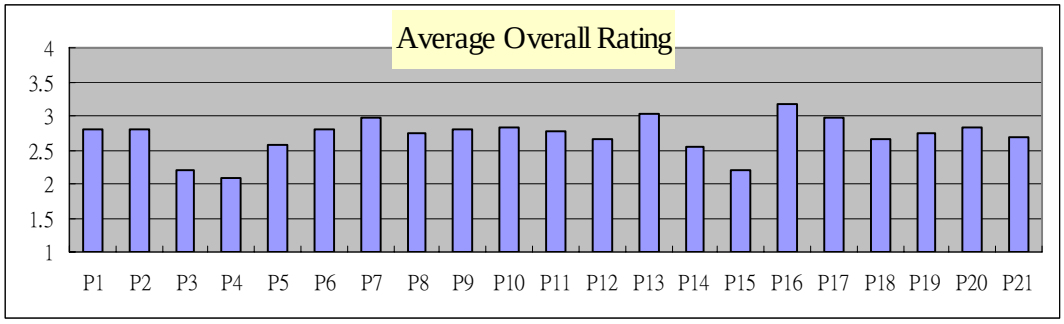
Figure 3-17 The frequency of ratings on paper overall value (given by 25 learners to 21 papers)

However, there are three papers (P3, P4, and P15) that receive an average rating below 2.5 as shown in Figure 3-18(a). Thus, if we choose the average rating 2.5 as a midpoint between a relatively good and a relatively bad overall rating, then we may conclude that 18 out of 21 papers received a relatively good overall rating, where two of them received on average a “good” (>3) rating. Among all the overall ratings, only P4 received three “bad” ratings (= 1); none of the other papers received this “bad” rating. In fact, P4 is also the most difficult and the least interesting paper.

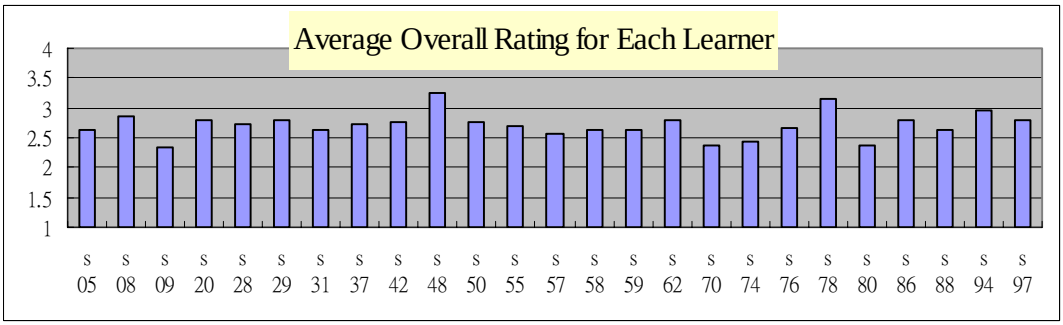
On the learner side, four learners give an average rating lower than 2.5. And two learners give an average rating higher than 3, while the rest give average ratings between 2.5 and 3.

Again, if we consider 2.5 as the midpoint about which there is a relatively good overall rating, then we may conclude that most recommended papers receive relatively good ratings from the majority of learners (i.e. 21 out of 25). In general, the standard deviation across papers is 0.271, which is higher than the standard deviation across learners, 0.215.

Throughout the thesis, we will use "Overall" with a capital letter to represent the overall rating given by the learners.



(a)



(b)

Figure 3-18 The average ratings on paper overall ratings for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

3.3.7 Peer Recommendation

The seventh question is to ask learners whether or not they will recommend the paper to fellow classmates, which means their own assessment as to whether the paper is worthy for others or not. For this question, the available options are only three, i.e. (3) "absolutely yes", (2)

"maybe", and (1) "no", because the nature of the question does not need a finer answer for our analysis. A binary choice ("yes" or "no") is our major concern, and a mid-choice "maybe" is enough to serve as the buffer for learners who do not want to rank so absolutely.

Figure 3-19 shows the distribution of ratings. Here, most ratings (92%) fall in either (3) "absolutely yes" or (2) "maybe" (i.e. 23.8%, 68.6%), and only 41 ratings (8%) are "no".

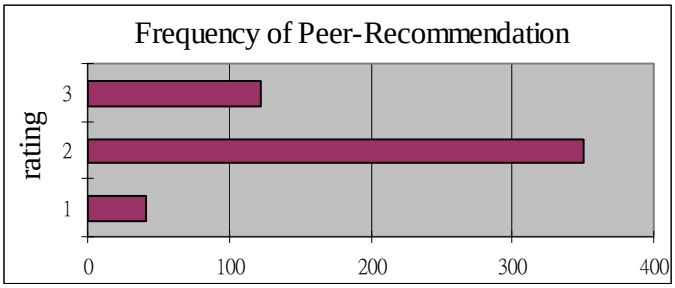


Figure 3-19 The frequency of ratings on whether to recommend the paper to other learners (given by 25 learners to 21 papers)

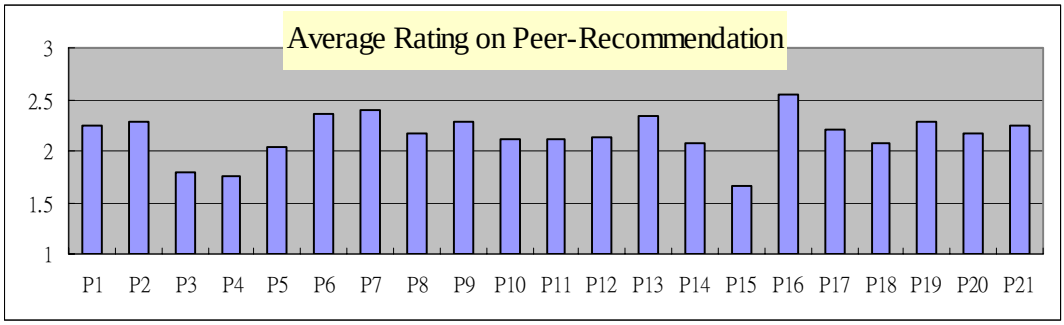
There are three papers (P3, P4, and P15) that receive an average rating below 2 as shown in Figure 3-20(a). These papers are those with average overall ratings lower than 2.5 (see Figure 3-19(a)). Similarly, the average overall rating of the sixteenth paper (P16[4]) is the highest, which is also the most recommended paper. A high correlation between average overall ratings and peer-recommendation across papers, i.e. 0.927, confirm the consistency of average overall ratings and peer-recommendations across papers.

On the learner side for each individual learner, the correlation between average overall learner rating of a paper and their willingness to recommend to peers is moderate only, 0.541. It is not surprising, because there are many other criteria or factors for a learner in recommending or not recommending a paper to others. For example, a learner's selfishness could be a factor in

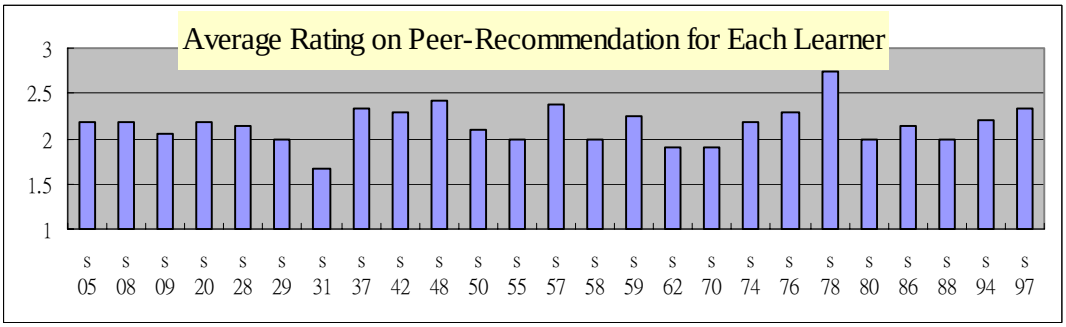
[4] This paper is a case study of usability engineering on Windows user interface design, which contains rich information on how the various Windows user interface designs can support usability. Due to the nature of this paper, it is not surprising that it is very well received.

reducing the learner's intention of recommending a good paper. Or, a learner may not recommend a paper which is less useful to others from his/her perspective; even though he/she gives a high overall rating. In fact, the correlation between average ratings on interestingness and peer-recommendation is 0.598, and the correlation between average ratings on value-added and peer-recommendation is 0.614; both are higher than the correlation of the average ratings on peer-recommendation and overall-rating.

Throughout the thesis, we will use "Peer_rec" with a capital letter to represent the rating on 'peer recommendation' given by the learners.



(a)



(b)

Figure 3-20 The average ratings on peer-recommendation aspect for (a) the first paper (P1) to the 21st one (P21), and (b) the first learner (S05) to the 25th learner (S97)

3.4 The Pedagogical Factors in the Recommendation: Correlation Analysis

This subsection is to verify Conjecture 2, 3, and 4. Conjecture 2 states that the overall rating given by learners to a paper is affected by the interestingness of the paper and other factors such as its value added and/or its usefulness in aiding the understanding of the topic(s) in the course. Conjecture 3 states that the intent of learners to recommend a paper to others is affected by the interestingness of the paper, plus its value-added and/or aiding-learning factors. Similarly, conjecture 4 states that the closeness of the paper's topics to the learners' job may affect their peer-recommendation and overall ratings.

Table 3-2, 3-3, and 3-4 respectively show the Pearson, Spearman and Kendall correlation coefficient matrices. The Pearson correlation coefficient corresponds to the classical linear correlation coefficients and is well suited for continuous data. Both Spearman and Kendall correlation coefficient are commonly used for ordinal data; hence, their coefficients are interpreted in terms of the variability of the ranks (Spearman) or the probability of the difference in ranking directions (Kendall). It is shown in all tables that both Aid_learning and Value_added are positively correlated to Overall and Peer_rec ratings, with correlation coefficient between 0.347 and 0.48 (see bold elements in Tables 3-2, 3-3 and 3-4). This may lead to the inference that Aid_learning and Value_added affect Overall and Peer_rec. However, there is a positive correlation between Interest and Value_added/Aid_learning, so one may argue that the correlation of Value_added and Overall rating is due to other causes, for example that Interest affects both Value_added and Overall rating at the same time [Blalock 1961]. In order to validate our conjectures, it is necessary to show that ratings on Value_added or Aid_learning are indeed affecting Overall or Peer_rec ratings independently from Interest. Four analyses are performed here: partial correlation, structural equation modeling (SEM), principal components and partial least squares regression (PLS), and manual comparison.

Table 3-2 Pearson correlation coefficient matrix

Variables	Difficulty	Job_ Related	Interest	Aid_ Learning	Value_ Added	Overall	Peer_ Rec
Difficulty	1	-0.148	-0.369	-0.146	-0.005	-0.330	-0.338
Job_related	-0.148	1	0.288	0.177	0.167	0.248	0.348
Interest	-0.369	0.288	1	0.323	0.358	0.605	0.557
Aid_learning	-0.146	0.177	0.323	1	0.405	**0.424**	**0.374**
Value_added	-0.005	0.167	0.358	0.405	1	**0.480**	**0.434**
Overall	-0.330	0.248	0.605	**0.424**	**0.480**	1	0.571
Peer_rec	-0.338	0.348	0.557	**0.374**	**0.434**	0.571	1

Table 3-3 Spearman correlation coefficient matrix

Variables	Difficulty	Job_ Related	Interest	Aid_ Learning	Value_ Added	Overall	Peer_ Rec
Difficulty	1	-0.154	-0.342	-0.129	-0.006	-0.304	-0.320
Job_related	-0.154	1	0.258	0.154	0.151	0.223	0.339
Interest	-0.342	0.258	1	0.308	0.348	0.593	0.548
Aid_learning	-0.129	0.154	0.308	1	0.397	**0.412**	**0.360**
Value_added	-0.006	0.151	0.348	0.397	1	**0.474**	**0.434**
Overall	-0.304	0.223	0.593	**0.412**	**0.474**	1	0.557
Peer_rec	-0.320	0.339	0.548	**0.360**	**0.434**	0.557	1

Table 3-4 Kendall correlation coefficient matrix

Variables	Difficulty	Job_ Related	Interest	Aid_ Learning	Value_ Added	Overall	Peer_ Rec
Difficulty	1	-0.140	-0.322	-0.122	-0.006	-0.289	-0.304
Job_related	-0.140	1	0.236	0.143	0.138	0.206	0.313
Interest	-0.322	0.236	1	0.291	0.323	0.569	0.521
Aid_learning	-0.122	0.143	0.291	1	0.378	**0.396**	**0.347**
Value_added	-0.006	0.138	0.323	0.378	1	**0.449**	**0.409**
Overall	-0.289	0.206	0.569	**0.396**	**0.449**	1	0.534
Peer_rec	-0.304	0.313	0.521	**0.347**	**0.409**	0.534	1

3.4.1 Partial Correlation

Partial correlation is commonly used in modeling causality of models with 3 or 4 variables. Let $r_{AB.C}$ be the Pearson correlation of variables A and B, controlling for variable C,

and r_{AB} be the Pearson correlation of variables A and B. If $r_{AB.C} = r_{AB}$, the inference is that the control variable C has no effect. If $r_{AB.C}$ approaches 0, then r_{AB} is spurious (the correlation is spurious), i.e. there is no direct causal link between A and B (see Figure 3-21(a)). It is either C affects A and B (anteceding), or A affects C which affects B (intervening). If $r_{AB} > r_{AB.C} > 0$, then we have a partial explanation (see Figure 3-21(b)). In this case, A partially affects B regardless of the fact that it affects (or is affected by) C. Our computations on partial correlation are shown in Table 3-5.

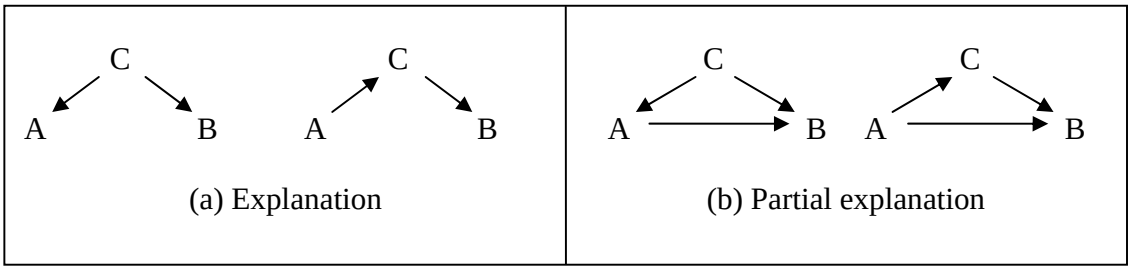


Figure 3-21 Causal inference with partial correlation when (a) $r_{AB.C} = 0$, and (b) $r_{AB} > r_{AB.C} > 0$

Table 3-5 Results of partial correlation

			Pearson partial correlation ($\mathbf{r_{AB.C}}$)	
Variables:	**C** (control)	**A**	**B**: Overall	**B**: Peer_rec
Group I	-	Value_added	0.4798	0.4335
	Interest	Value-added	0.3539	0.3017
Group II	-	Aid_learning	0.4242	0.3740
	Interest	Aid_learning	0.3038	0.2469
Group III	-	Interest	0.6046	0.5574
	Value_added	Interest	0.5282	0.4780
	Aid_learning	Interest	0.5456	0.4975

Note: when control variable C is nil, we have $\mathbf{r_{AB}}$ instead of $\mathbf{r_{AB.C}}$

Three groups (I, II, and III) are used as the comparison. In Group I, we check r_{AB} and $r_{AB.C}$ for C=Interest, A=Value_added, and B is either Overall or Peer-rec. The results of r_{AB} are 0.4798 and 0.4335 for B is Overall and Peer-rec, respectively. After introducing Interest as a control, the correlations decrease to 0.3539 and 0.3017, respectively; hence, $r_{AB} > r_{AB.C} > 0$ in this group. In Group II, we check r_{AB} and $r_{AB.C}$ for A=Aid_learning. In Group III, we check r_{AB} and $r_{AB.C}$ for the reverse causality, i.e. Interest is affected by Value_added or Aid_learning. In fact, $r_{AB} > r_{AB.C} > 0$ for all groups; that is the results favor a partial explanation model. In other words, to some degree Value_added and Aid_learning affect Overall and Peer_rec ratings independently from Interest. However, the presence of multicollinearity among variables in partial correlation analysis may diminish the validity of the claim. In addition, it is not clear whether the model is still valid in the presence of other variables (e.g. Difficulty or Job_related). We perform structural equation modeling (SEM) to verify the fitness of our model.

3.4.2 Structural Equation Modeling (SEM)

Structural Equation Modeling (SEM) works similarly to multiple regression, but in a more powerful way taking into account the nonlinearities, multiple latent dependent as well as independent variables each measured by multiple indicators, etc. SEM may be used as a more powerful alternative to multiple regression, correlation analysis and factor analysis. LISREL (LInear Structural RELations) [Kelloway 1998] is one of the statistical package to conduct SEM, and is used in our modeling. The analysis consists of two parts. In the first part, we test partial-explanation models in which (i) Interest affects Value_added before both affect Overall ratings or Peer_rec and (ii) Value_added affects Interest before both affect Overall or Peer_rec ratings. Figure 3-22 shows the results for Overall. Here, all parameters are freely adjusted until they fit a criterion (i.e. maximum likelihood estimation). Figure 3-22 shows the path diagram constructed for both partial-explanation model (i) and (ii), which consists of error variances (values beside

each variable in Figure 3-22) and regression parameters (values at the arrow connecting two variables in Figure 3-22). Both models are exact fit, measured in Root Mean Square Error of Approximation (RMSEA = 0.0). Note, a good-fit model does not imply the model is true; but rather that it is capable of representing the data in a structural model. But from both models, we are able to find out that Value_added affects Overall with regression parameter = 0.29 (approximately 2/3 of the parameter of Interest (= 0.45)), which once again justifies our conjecture that Overall ratings are affected by factors other than Interest, in this case Value_added. For Peer_rec, the results are similar as for Overall except with lower regression parameters.

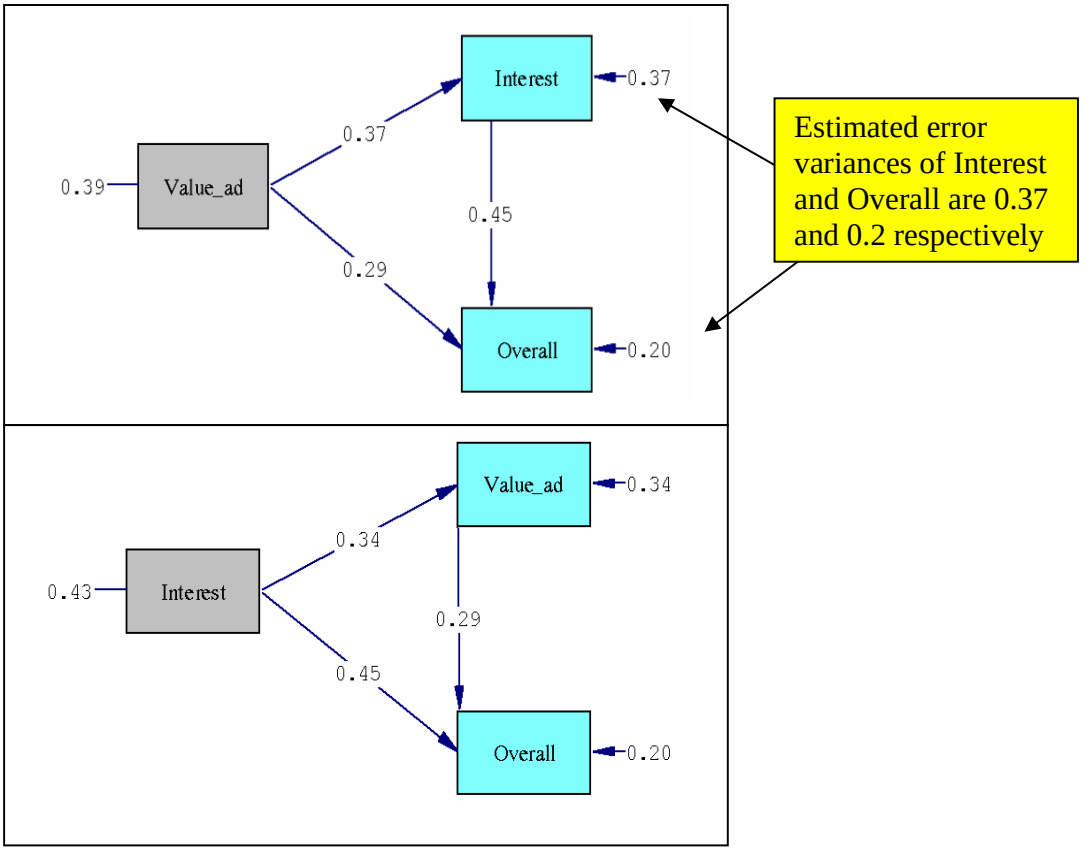


Figure 3-22 Path diagrams where Interest is affected by Value_added (top) and Value_added is affected by Interest (bottom)

Next, we build a more complete model and check whether or not adopting more explanatory variables is better in terms of model-fitness. Figure 3-23 presents these more complete models incorporating all variables except for Peer_rec.

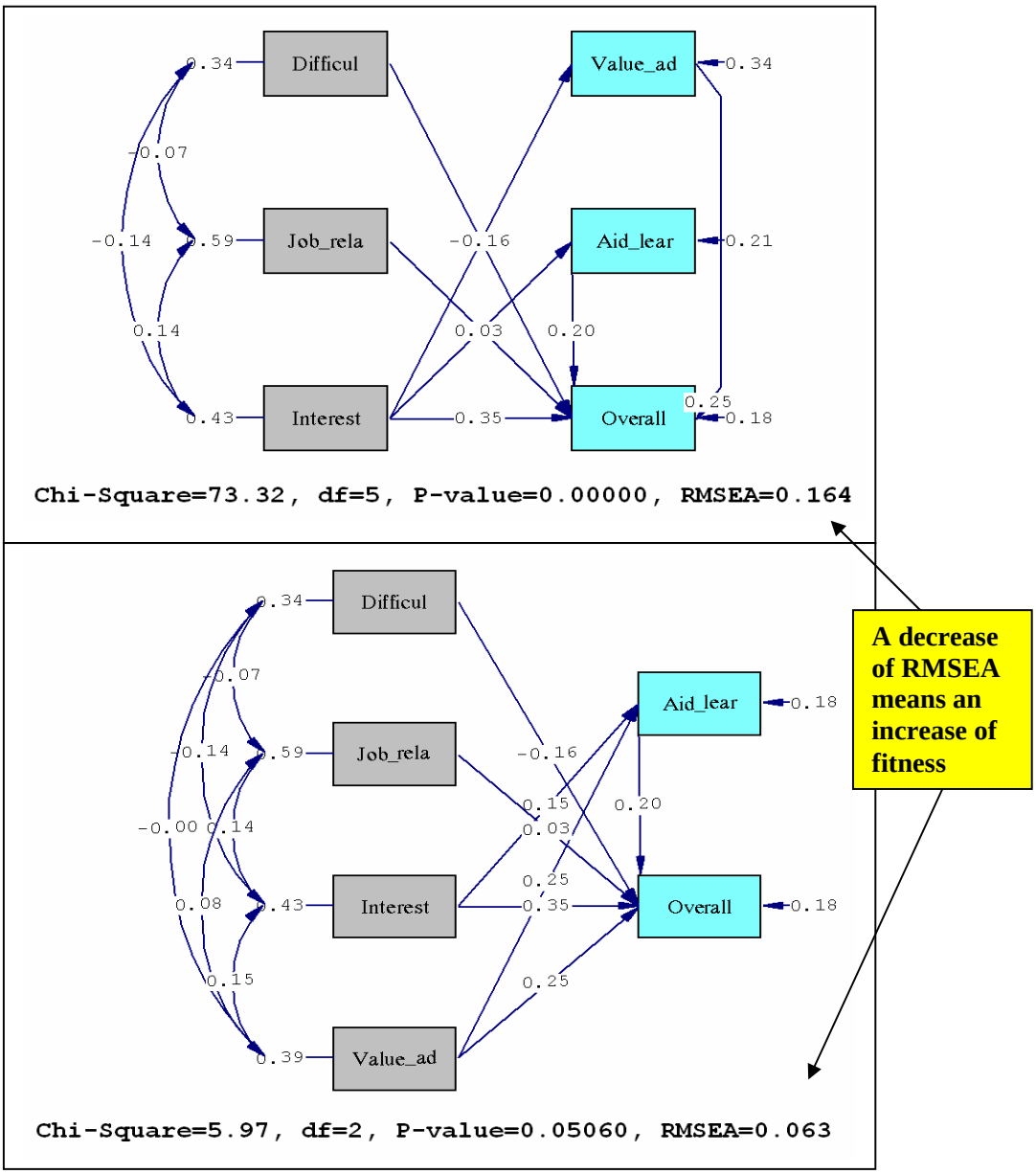


Figure 3-23 Path diagrams with Value_added as a dependent variable (top) and as an explanatory variable (bottom)

In the top figure, we include Value_added as a dependent variable affected by Interest, while in the bottom figure Value_added is an explanatory variable. The goodness-fit measure, RMSEA, is shown here. Note, a RMSEA statistic greater than 0.08 signifies a less reasonable approximation [Browne and Cudeck, 1993]; thus, yielding a less reasonable fit for the model in the top figure (RMSEA = 0.164) and more reasonable fit for the bottom one (RMSEA = 0.063). Therefore, leaving Value_added as an explanatory variable, as shown in the bottom figure, can increase the fitness of the model to a reasonable one (i.e. one with RMSEA < .08).

Further, the reduced-form regression equations along with their coefficient of determination (R^2) in the top figure are

(M1a) Value_added = 0.34 Interest ($R^2 = 0.13$)

(M1b) Aid_learning = 0.24 Interest ($R^2 = 0.10$)

(M1c) Overall = - 0.16 Difficult + 0.03 Job_related + 0.49 Interest ($R^2 = 0.40$)

And those of the bottom figure are

(M2a) Aid_learning = 0.15 Interest + 0.25 Value_added ($R^2 = 0.20$)

(M2b) Overall = - 0.16 Difficult + 0.03 Job_related +

0.38 Interest + 0.31 Value_added ($R^2 = 0.47$)

It is shown here that the coefficient of determinations (R^2) of regression equation between Interest and Value_added (M1a) and between Interest and Aid_learning (M1b) are very low, i.e. $R^2 = 0.13$ and $R^2 = 0.10$ respectively, which means these linear equations cannot accurately explain the variance of dependent variables; in other words, Interest weakly affects Value_added and Aid_learning. And if we add Value_added into the regression equation of Overall (M2b), then we get a higher coefficient determination compared to without Value_added (M1c), i.e. $R^2 =$

0.47 in M2b and $R^2 = 0.40$ in M1c. Thus, it is better to consider Value_added as an explanatory variable rather than using it as a dependent one.

A similar evaluation is performed on Peer_rec, and similar results are obtained.

In conclusion, we can derive a better model when Value_added and Aid_learning are not directly affected by Interest. The correlation of Value_added and Interest may be explained by other latent variables (unobserved factors) that may affect both Value_added and Interest. In the next subsection we show another approach to verifying our conjectures. We perform linear regression analysis after transforming the explanatory variables.

3.4.3 Principal Components Regression and Partial Least Squares Regression

Principal components regression (PCR) combines principal components analysis (PCA) and linear regression. PCA transforms observations from a p-dimensional space to a q-dimensional space, $q \leq p$, while conserving as much information as possible (in terms of the total variance) from the original dimensions. The resulting dimensions are non-correlated weighted components which are linear combinations of the original variables. The weights are usually represented by eigenvalues produced during transformation. High-eigenvalue components are principal components which contain most information in the original data [Jolliffe 2002]; hence, they provide windows of opptunity to analyze the correlations between the original variables. Meanwhile, by removing low-eigenvalue components, we can also reduce the dimensionality of the original data and simplify the regression model. If an explanatory variable is redundant (e.g. collinear with other variables), then it will vanish during dimensional reduction by PCR. In our test, we will check if Value_added and/or Aid_learning will vanish when we reduce the dimensionality of explanatory variables to two components only. In other words, whether these two variables have some impact on the Overall ratings or Peer-rec or both.

Partial least squares regression (PLS) also uses PCA in building non-correlated components but differs from PCR in the sense it considers the accuracy of regression during the selection of components in regression. The components selected are not necessarily those with the highest eigenvalues, but those that explain as many as possible of the independent variables. As such, PLS performs a simultaneous decomposition of explanatory variables (components) and dependent variables with the constraint that these components explain as much as possible of the covariance between explanatory and dependent variables. In our test here, we set the stopping criteria for both PCR and PLS to be when they find at most two components. Thus, other components, if any, will be excluded from the regression. We used XLSTAT 2007 to perform both PCR and PLS, with Difficulty, Job_related, Interest, Aid_learning, and Value_added as explanatory variables, and Overall and Peer_rec as dependent variables. The following regression equations were obtained automatically from both PCR and PLS:

PCR:

$$\begin{aligned}\text{Overall} = {} & 0.496 - 0.145\ \text{Difficulty} + 0.138\ \text{Job\_related} + \\ & 0.245\ \text{Interest} + 0.297\ \text{Aid\_learning} + \\ & 0.222\ \text{Value\_added} \qquad (R^2 = 0.448) \end{aligned} \tag{3.1}$$

$$\begin{aligned}\text{Peer\_rec} = {} & 0.396 - 0.184\ \text{Difficulty} + 0.136\ \text{Job\_related} + \\ & 0.235\ \text{Interest} + 0.244\ \text{Aid\_learning} + \\ & 0.170\ \text{Value\_added} \qquad (R^2 = 0.450) \end{aligned} \tag{3.2}$$

PLS:

$$\begin{aligned}\text{Overall} = {} & 0.618 - 0.170\ \text{Difficulty} + 0.039\ \text{Job\_related} + \\ & 0.314\ \text{Interest} + 0.267\ \text{Aid\_learning} + \\ & 0.248\ \text{Value\_added} \qquad (R^2 = 0.509) \end{aligned} \tag{3.3}$$

$$\begin{aligned}\text{Peer\_rec} = {} & 0.309 - 0.156\ \text{Difficulty} + 0.1407\ \text{Job\_related} + \\ & 0.237\ \text{Interest} + 0.225\ \text{Aid\_learning} + \\ & 0.189\ \text{Value\_added} \qquad (R^2 = 0.457) \end{aligned} \tag{3.4}$$

The PCR model uses a system generated value of 61.1% variability of the original explanatory data, i.e. the amount of information retained by the first two components of PCA. This value is quite low, because the suggested variability in PCR is at least 80% (i.e. the default setting of XLSTAT). We restricted our model to a low variability in order to verify the "survivability" of Value_added and Aid_learning as explanatory variables in the model. From all the equations above, we found that the regression parameters of Value_added and Aid_learning are between 0.170 and 0.297, while the parameters of Interest are between 0.235 and 0.314. Here, the regression parameters of Value_added and Aid_learning are relatively big with respect to that of Interest; hence, the result supports the survivability of these two variables in explaining Overall and Peer_rec ratings. In fact, the Variable-Importance-in-the-Projection (VIPs) index which measures the importance of an explanatory variable of both Value_added and Aid_learning from PLS are above the critical value 0.8, which leads us to strongly believe that they contribute significantly to the model [Wold 1995] (see Figure 3-24).

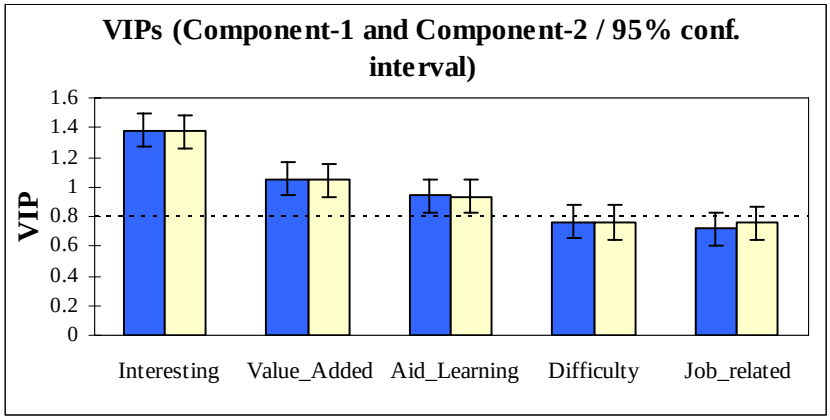


Figure 3-24 The variable-importance-in-the-projection index (VIPs) of both component 1 and component 2 from PLS

3.4.4 Manual Comparison and Conclusion

The previous three verification methods are applied to all data without subjective selection. Here we manually compare the following five pair groups:

I. "high Interest and low Value_added" versus "high Interest and high Value_added"

II. "low Interest and low Value_added" versus "low Interest and high Value_added"

III. "high Interest and low Aid_learning" versus "high Interest and high Aid_learning"

IV. "low Interest and low Aid_learning" versus "low Interest and high Aid_learning"

V. "high Job_related" versus "low Job_related"

We consider "high" to be a rating of 3 or 4, and "low" if the rating a rating of 1 or 2. Table 3-6 shows the comparison results: average Overall and Peer_rec ratings and the p-values after performing a t-test on each pair of a group. The number of observations (N) that satisfy the criteria of each group is also provided.

The p-values show a very significant difference between the mean values in each group, where a high Value-added results in higher average Overall and Peer_rec ratings for both high and low Interest groups. Similarly, a high Aid_learning also results in higher average Overall and Peer_rec ratings. Given previous study results using partial correlation and SEM, which show substantial contributions of Value_added and Aid_learning to Overall and Peer_rec, and the survival of them after dimensionality reduction in PCR and PLS, we derive the following conclusions:

Conclusion 2. *In the population of graduate students, the overall rating given by learners to a paper may not only depend on the interestingness of the paper from their perspective, but also on the richness of knowledge that has been gained by them from reading the paper and/or the usefulness of the paper in helping them to understand the course subject.*

Conclusion 3. *In the population of graduate students, the intent of learners in recommending a paper to others may not only depend on the interestingness of the paper from their perspective, but also on the richness of knowledge that has been gained by them from reading the paper and/or the usefulness of the paper in helping them to understand the course subject.*

Table 3-6 Results of manual comparison

	Groups	Overall		Peer_rec	
		Mean	value	Mean	value
I	Interest, low Value_added (N=55)	2.455	$3 \cdot 10^{-10}$	2.073	$2 \cdot 10^{-5}$
	Interest, high Value_added (N=311)	2.974		2.354	
II	Interest, low Value_added (N=50)	2.060	0.0005	1.640	0.012
	Interest, high Value_added (N=93)	2.312		1.828	
III	Interest, low Aid_learning (N=43)	2.535	$1 \cdot 10^{-5}$	2.140	0.0032
	Interest, high Aid_learning (323)	2.944		2.334	
IV	Interest, low Aid_learning (N=47)	2.043	0.0004	1.574	0.001
	Interest, high Aid_learning (N=96)	2.313		1.854	
V	Job_related (N=190)	2.842	$3 \cdot 10^{-5}$	2.374	$2 \cdot 10^{-12}$
	Job_related (N=319)	2.627		2.028	

From group V of Table 3-6 we also get significant differences between the average Overall ratings of high Job_related and low Job_related groups. A significant difference is also observed on average Peer_rec ratings whereas a relatedness of a paper to learner's job contribute to higher average ratings. However, from equations (3.1) to (3.4) the contribution of Job_related in the linear regression model is less than the contributions of Interest, Value_added, and Aid_learning. To further test the contribution of Job_related in predicting Overall and Peer_rec

ratings, we perform a linear regression without dimensional reduction. The following linear equations are the results:

$$\begin{aligned} \text{Overall} = {} & 0.673 - 0.160\ \text{Difficulty} + 0.031\ \text{Job\_related} + \\ & 0.352\ \text{Interest} + 0.199\ \text{Aid\_learning} + \\ & 0.255\ \text{Value\_added} \qquad (R^2 = 0.492) \end{aligned} \tag{3.5}$$

$$\begin{aligned} \text{Peer\_pec} = {} & 0.501 - 0.166\ \text{Difficulty} + 0.120\ \text{Job\_related} + \\ & 0.264\ \text{Interest} + 0.132\ \text{Aid\_learning} + \\ & 0.210\ \text{Value\_added} \qquad (R^2 = 0.445) \end{aligned} \tag{3.6}$$

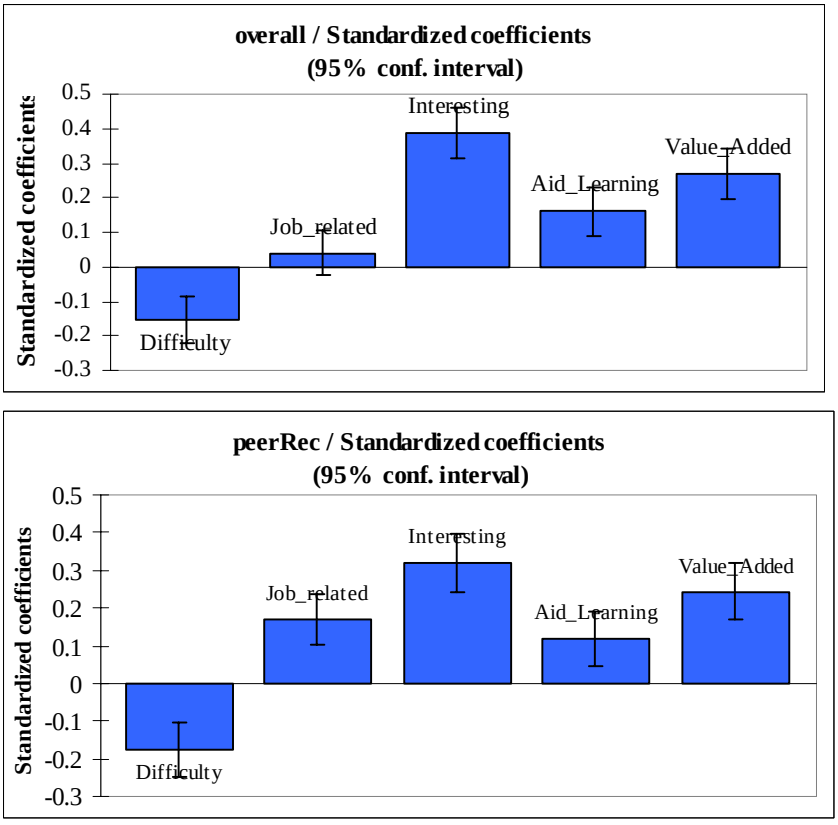


Figure 3-25 The standardized coefficients of linear regression models in predicting Overall and Peer_rec ratings

Figure 3-25 shows the standardized coefficients of the corresponding regression equations. Standardized coefficients are commonly used to compare the relative weights of the explanatory variables, where a higher absolute value of a coefficient means a more important

variable. From the top diagram in Figure 3-25, we conclude that Job_related does not contribute significantly in predicting Overall ratings (its 95% confidence interval includes the zero value). However, it can be used to predict Peer_rec as shown in the bottom diagram of Figure 3-25. From this, we can derive:

Conclusion 4. *In the population of part-time graduate students, the closeness of learners' jobs to the paper topics significantly affects their overall ratings but does not significantly predict the overall ratings. However, the closeness of learners' jobs to the paper topics is significantly affecting their intent of recommending it to others and can significantly predict the degree of willingness to make peer recommendation to others.*

3.5 Conclusions

In this chapter, we described the charateristics of pedagogical paper recommendation based on the collected data. Our results are able to support our four conjectures that highlight the importance of several features in making paper recommendation in this domain. In particular:

1. learner interest is not that important and not the only dimension for making recommendations;
2. other contextual information-seeking goals such as task- and course-related goals are related to learners' perceived value of the papers;
3. learners' willingness of further making peer recommendation on a paper depends largely on the closeness of its content topic to their job nature.

These observations can help tutors support the learner in making a decision as to which items to select, and highlights the uniquessness of pedagogical recommendation when compared to other types of recommendation.

In the next chapter, we will go a step further to propose a set of recommendation algorithms in this domain by considering a range of recommendation techniques and how well they would have performed if informed by data about learners and papers collected in our experimental study.

CHAPTER 4
PEDAGOGICAL PAPER RECOMMENDATION TECHNIQUES AND EXPERIMENTS

As we pointed out in the previous chapter our experiments mainly serve two needs (i) to explore the characteristics of pedagogical paper recommendation, and (ii) to test various recommendation techniques in a pedagogical context. The first goal (discussed in Chapter 3) differentiates our study from other paper recommendation system research in non-learning environments, while the second goal exploring a vareity of recommendation algorithms. This chapter introduces a set of recommendation techniques and describes how the data gathered in the experimental study presented in Chapter 3 can be used to explore the appropriatness of these algorithms.

Generally, these techniques can be grouped into five categories:

- non-personalized recommendation,
- content-based filtering,
- collaborative filtering,
- hybrid filtering, and
- manual recommendation.

A detailed description of each of these techniques is presented in section 4.1. Since all techniques involve a number of combinations of parameters, section 4.2 describes the parameter tuning. In section 4.3, we make comparison among these

techniques using the real data obtained from our students in the software engineering course.

4.1 Pedagogical Paper Recommendation Techniques

Since recommendation in a pedagogical context differs from those in other domains, we will explore recommendation techniques beyond the traditional ones, that is by modifying the traditional techniques according to the characteristics of the learning domain. Broadly speaking, these characteristics are due to (i) the limited number of users, (ii) the large number of unrated or new items, (iii) the likelihood of the learners' difficulty in understanding the items, and (iv) the numerous purposes of the recommendation.

With respect to the limited number of users and the large number of unrated/new items, our recommender system shall take cold-start problem seriously, hence cannot rely solely on rating-based CF. For this reason, we will consider content-based filtering, user-model-based CF, or other techniques that do not need many learner ratings. In fact, even for rating-based CF we may not have enough co-rated items, hence we have to consider multidimensional ratings on each item in order to boost the number of co-ratings so that to find more accurate neighbors for a target user. In addition, we also take into consideration paper popularity in an attempt to start up the recommendation when there are not many ratings in the system. Factors considered in our multi-dimensional CF will mainly be used to correlate one user with another; that is to find the similarity between users and make recommendations accordingly. These factors include:

- a paper's ratings on Overall, Value-addedness, and Peer recommendations,
- features about a learner including learner interest and background knowledge, and

- the average of a paper's Overall ratings (paper popularity)

The possibility that our learners may encounter difficulty to understand the papers has pushed us to consider their background knowledge; hence to adopt content-based and hybrid filtering techniques that incorporate this factor.

With respect to the numerous purposes of recommendation, a tutor may aim at the overall learners' satisfaction (the highest possible Overall ratings), or to stimulate learners' interest only (the highest possible Interest ratings), or to help the learners to gain new information only (the highest possible Value-added ratings), etc. Given the fact of numerous recommendation purposes, it is imperative and appealing to collect multidimensional ratings and to study multidimensional CF that can utilize the ratings.

Table 4-1 organizes and categorizes the various recommendation techniques that are discussed in this section. Generally, they fall into four main categories: content-based, CF-based, hybrid recommendation, and other techniques.

Table 4-1 A summarization of the various recommendation techniques.

Category	**Name**	**Remarks**
Content-based	ContentF	Content-based (User-item) filtering
CF-based	1D-CF	Uni-dimensional rating-based CF
	3D-CF	Multi-dimensional rating-based CF
	UM-CF (2D-CF)	User-model based CF
	5D-CF	Rating- and user-model-based CF
Hybrid	Pop3D	Non-personalized and rating-based CF
	Pop5D	Non-personalized and 5D-CF
	Pop2D	Non-personalized and User-model based CF
	PopCon2D	A combination of Non-personalized , user-item content filtering and 2D-CF
	AL	Artificial learner-based
Other types	Manual rec.	
	Pop1D	Non-personalized method

Note here that we regard the injection of paper popularity (the average Overall ratings of a paper) as a non-personalized method; but under certain circumstances, it is very useful to start up the recommendation, therefore, suitable for cold-start problems. The wider range of the recommendation methods proposed in this thesis can better serve a wide variety of possible learning scenarios and contexts where there are limited papers or learners in the system.

4.1.1 Non-Personalized Recommendation (Benchmark)

Non-personalized recommendation is the simplest technique used in many recommendation systems, such as in *Amazon.com*^{TM}, *YouTube*^{TM}, etc. After users consume an item (e.g. reading a book, listening to a song, etc.) they may provide a simple rating to the item (e.g. from 1 star to 5 stars). Then, the system makes recommendations based on the average ratings that an item has received before. That is, those items consistently liked by average users tend to be selected. These average ratings are commonly tagged to the items as metadata.

The non-personalized recommendation technique explored in this thesis generates items based on a group of users' tastes, i.e. a group of graduate students registered to take the software engineering course. As pointed out in both [O'Connor et al. 2001; Martin and Torrens 2006], users within a group gather because of a 'single activity' [O'Connor et al. 2001]. In this thesis, registering and taking the software engineering course at the same level is the 'single big activity'. In our domain, each paper is populated by the users' ratings, that is, those well received papers will be pushed up; otherwise, they will be downgraded. From now on, the average rating of each paper k among a group of learners, denoted by $\tilde{r}_k$, will be labeled as the paper's *popularity*.

4.1.2 User-Item Content-Based Filtering (ContentF)

In the ContentF method, we recommend paper(s) to a user by comparing the user model and the content of each paper. When we add a paper into our system, we will categorize it according to its topics and the minimal knowledge needed for understanding it. For example, some papers may require some advanced statistics or mathematical background for understanding them, while others may only require simple statistics. Intuitively, a highly technical paper (e.g. one with a complex mathematical model or UML diagram) may get a lower rating from novice learners compared to a less technical one, because novices may not understand it clearly. However, it may get a higher rating from expert learners due to the richness or accuracy of its content; for instance, it may present an algorithm that is proven to be of polynomial complexity, or it may present a novel experimental methodology, which is supported by a good statistical analysis, etc.

Table 4-2 shows an example of paper models that were given manually and used in our experiment. In this table we have 13 different features (T1 to T13) to represent the topics of each paper, and 5 different other features (K1 to K5) to represent the minimal knowledge to understand the paper (see Figure 4-1 for a mapping of these symbols). If a paper does not relate to a topic, then we indicate this with "1" (irrelevant to that topic); otherwise we use another number up to "5" (very relevant). In most cases, a paper may relate to more than one topic at the same time. For example, paper 11 [Bederson et al. 2004] is related to web design and application (T2) and user interface design (T3) at the same time, while its focus is more on the latter, thus receiving a higher relevance score.

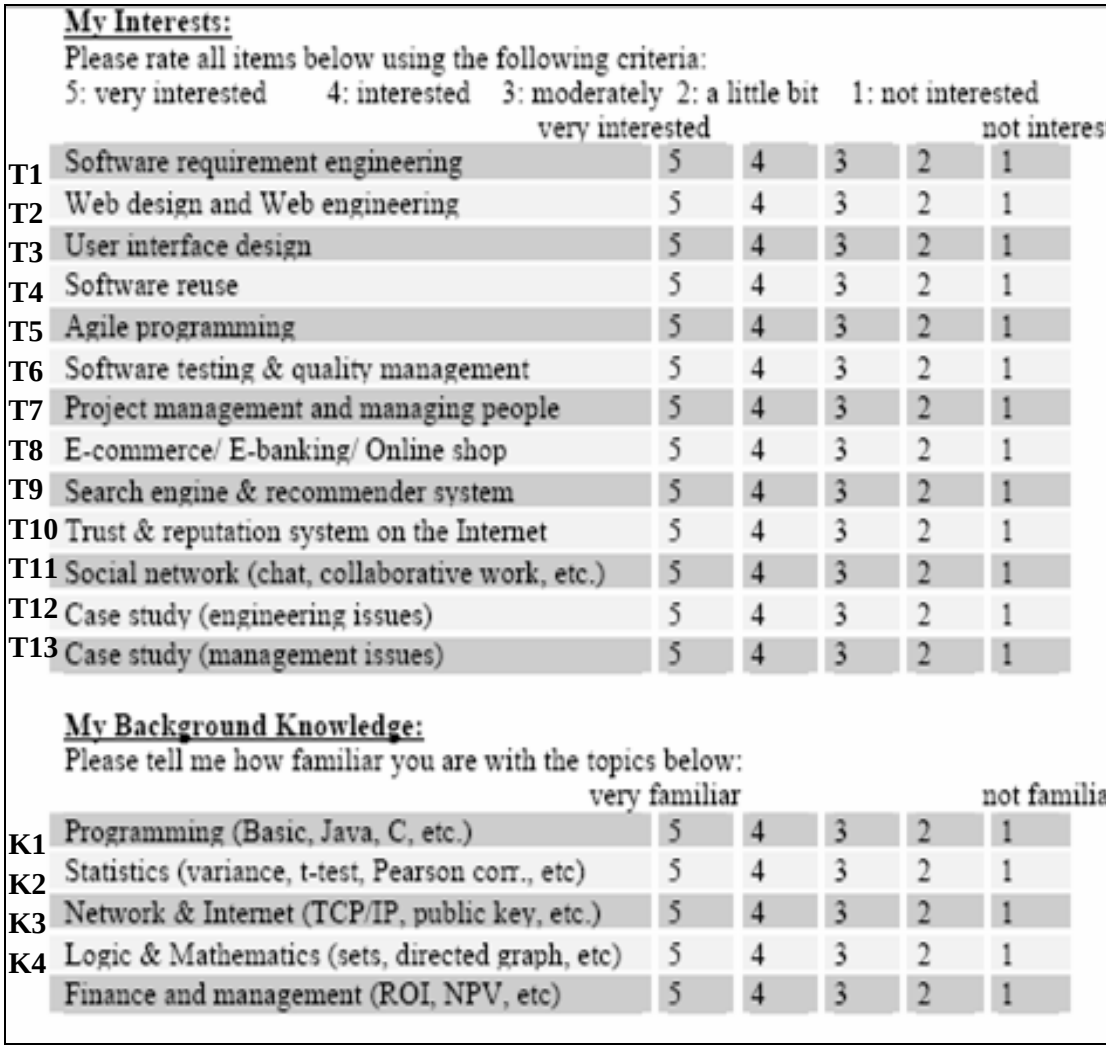
Finally, column K1 to K5 are the required knowledge to understand a paper ("1" for not required at all and "5" for highly required). In fact, most papers do not require any technical knowledge to be understood. These numbers are given manually and

subjectively by a tutor. Indeed, they can be adjusted later, either by other tutor(s) or by learners, for example, by averaging the values given by all tutors.

Table 4-2 Paper models and their descriptions

Paper	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	K1	K2	K3	K4	K5
1	5	1	1	2	1	4	1	1	1	1	1	1	3	1	5	1	1	1
2	1	1	1	1	1	3	5	1	1	1	1	1	4	1	1	1	1	1
3	5	1	1	1	1	4	3	1	1	1	1	1	3	3	4	1	1	1
4	5	1	1	4	1	1	1	1	1	1	1	3	1	1	1	1	1	1
5	1	1	1	1	1	1	5	1	1	1	1	1	4	1	1	1	1	1
6	5	1	3	1	3	1	4	1	1	1	1	1	3	1	1	1	1	1
7	1	1	1	1	1	2	5	1	1	1	1	1	4	3	1	1	1	1
8	1	5	5	1	1	4	1	1	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	3	1	4	5	1	3	1	1	3	1	1	5	1
10	1	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	1	3	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	1	4	4	1	1	4	1	1	1	1	1	1	1	1	1	1	1	1
13	3	5	5	1	1	5	1	1	1	1	1	1	1	1	1	1	1	1
14	1	5	5	1	1	4	1	1	1	1	1	1	1	3	1	1	1	1
15	1	4	4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16	3	4	5	1	1	5	1	1	1	1	1	5	1	4	1	1	1	1
17	1	5	5	1	1	5	1	1	1	1	1	1	1	1	1	1	1	1
18	1	1	1	1	1	5	1	1	1	1	1	1	5	1	1	1	1	1
19	1	1	1	1	1	5	5	1	1	1	1	1	4	1	1	1	1	1
20	1	5	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
21	1	1	3	1	1	1	5	1	1	1	1	1	4	1	1	1	1	1

Figure 4-1 shows an example of the questionnaire for obtaining features used in the user model for our experiment. In this example, the first two groups of our questionnaire ("My Interest" and "My Background Knowledge") are matches to the paper features (T1 to T13 and K1 to K5 respectively). Thus, we consider two factors as a basis to measure the *closeness of a user to a paper*, i.e. matches in the topics of *interest* and *minimal background knowledge* needed to understand the paper.



My Interests:

Please rate all items below using the following criteria:

5: very interested 4: interested 3: moderately 2: a little bit 1: not interested

		very interested				not interes
T1	Software requirement engineering	5	4	3	2	1
T2	Web design and Web engineering	5	4	3	2	1
T3	User interface design	5	4	3	2	1
T4	Software reuse	5	4	3	2	1
T5	Agile programming	5	4	3	2	1
T6	Software testing & quality management	5	4	3	2	1
T7	Project management and managing people	5	4	3	2	1
T8	E-commerce/ E-banking/ Online shop	5	4	3	2	1
T9	Search engine & recommender system	5	4	3	2	1
T10	Trust & reputation system on the Internet	5	4	3	2	1
T11	Social network (chat, collaborative work, etc.)	5	4	3	2	1
T12	Case study (engineering issues)	5	4	3	2	1
T13	Case study (management issues)	5	4	3	2	1

My Background Knowledge:

Please tell me how familiar you are with the topics below:

		very familiar				not familia
K1	Programming (Basic, Java, C, etc.)	5	4	3	2	1
K2	Statistics (variance, t-test, Pearson corr., etc)	5	4	3	2	1
K3	Network & Internet (TCP/IP, public key, etc.)	5	4	3	2	1
K4	Logic & Mathematics (sets, directed graph, etc)	5	4	3	2	1
	Finance and management (ROI, NPV, etc)	5	4	3	2	1

Figure 4-1. Questionnaire for obtaining learner interests and knowledge (Appendix B)

We cannot simply use correlation to measure the closeness in our case because topics covered by a paper may only be a subset of a user's interests, and vice versa. Instead, we compute various measurements as shown in Figure 4-2.

```
FUNCTION Closeness_1( )

        FOR EACH topic i
                // match paper topic with learner interest
                IF user_model(i) ≥ 3 AND paper_model(i) ≥ 3 THEN
                        interest = interest + 1
                ELSE IF user_model(i) ≤ 2 AND paper_model(i) ≥ 3 THEN
                        interest = interest – 1
                 END IF

        FOR EACH knowledge k
                //match knowledge pieces required to understand the paper with learner
         knowledge background
                 IF user_model(k) ≥ 3 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr + 1
                ELSE IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – 1
                END IF

                //the final degree of closeness between the user-paper match is determined by both
         learner interest and knowledge background.
        RETURN Closeness_1 = 4 * interest + weight * backgr

END FUNCTION
```

(a)

```
FUNCTION Closeness_2( )
        FOR EACH topic i
                 IF user_model(i) ≥ 2 AND paper_model(i) ≥ 3 THEN
                        interest = interest + 1
        FOR EACH knowledge k
                 IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – 1
        RETURN Closeness_2 = 4 * interest + weight * backgr
END FUNCTION
```

(b)

```
FUNCTION Closeness_3( )

        FOR EACH topic i
                IF user_model(i) ≥ 2 AND paper_model(i) ≥ 2 THEN
                        interest = interest + user_model(i)

        FOR EACH knowledge k
                IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – paper_model(k)

        RETURN Closeness_3 = 4 * interest + weight * backgr

END FUNCTION
```

(c)

```
FUNCTION Closeness_4( )

        FOR EACH topic i
                IF user_model(i) ≥ 3 AND paper_model(i) ≥ 2 THEN
                        interest = interest + user_model(i)

        FOR EACH knowledge k
                IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – paper_model(k)

        RETURN Closeness_4 = 4 * interest + weight * backgr

END FUNCTION
```

(d)

```
FUNCTION Closeness_5( )

        FOR EACH topic i
                IF user_model(i) ≥ 2 AND paper_model(i) ≥ 2 THEN
                        interest = interest + user_model(i) ^ 2

        FOR EACH knowledge k
                IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – paper_model(k) ^ 2

        RETURN Closeness_5 = 4 * interest + weight * backgr

END FUNCTION
```

(e)

```
FUNCTION Closeness_6( )

        FOR EACH topic i
                IF paper_model(i) ≥ 2 THEN
                        interest = interest + user_model(i) * paper_model(i)

        FOR EACH knowledge k
                IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – paper_model(k) ^ 2

        RETURN Closeness_6 = 4 * interest + weight * backgr

END FUNCTION
```

(f)

```
FUNCTION Closeness_7( )

        FOR EACH topic i
                interest = interest + user_model(i) * (paper_model(i) – 1)

        FOR EACH knowledge k
                IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        backgr = backgr – (paper_model(k) – 1) ^ 2

        RETURN Closeness_7 = 4 * interest + weight * backgr

END FUNCTION
```

(g)

Figure 4-2. Seven algorithms used in calculating the closeness between a user and a paper

The variable *user_model(i)* and *user_model(k)* are the ratings representing the user *interest* in a specific topic and his/her self-assessment about his/her own background *knowledge* as shown in Figure 4-1 (both on a Likert scale 1 to 5). The variable *paper_model(i)* and *paper_model(k)* are the ratings given by tutors to a paper (also on a scale 1 to 5). For instance, if a paper talks about a case study in software requirements engineering, then the rating on both "software requirements engineering" and "case study

(engineering issues)" will be high (e.g. 4 or 5) while its rating on other topics will be low (e.g. 1). Moreover, *interest*, *weight* and *backgr* are variables used to calculate the *closeness*, where *weight* is a control variable in our experiments.

In *Closeness_1( )*, we respectively impose a fixed reward (+1) and penalty (–1) for either matched or unmatched interest-topics and background-minimal knowledge. In other words, when we compare the features of the paper with those of the users, for the un-matched background knowledge, a penalty is imposed to reduce the possibility of recommending the paper. This approach aligns with the major goal of our pedagogic paper recommender in that even if a paper matches a user's interest, if it is not pedagogically appropriate for the user to understand, the rating it will receive will be low. The larger value of *Closeness_1( )* (and also in the other algorithms in Figure 4-2), the closer or better the paper is as a candidate for recommendation.

In *Closeness_2( )*, we only impose a reward (+1) for matched interest-topics and a penalty (–1) for unmatched background-minimal knowledge. In *Closeness_3( )*, we use a similar method to that in *Closeness_2( )* except that the reward/penalty is not fixed ±1, but the ratings given by the user/tutor, i.e. *user_model(i)* or *paper_model(k)*. In *Closeness_4( )*, we use a similar method as in *Closeness_3( )* except that the condition of the first matching is changed from *user_model*(i) ≥ 2 to *user_model*(i) ≥ 3 for a sensitivity test. In *Closeness_5( )*, we further increase the penalty/reward to $user\_model(i)^2$ or $paper\_model(k)^2$ in order to see the effect of magnifying penalty/reward during the recommendation process. In *Closeness_6( )*, we relax the condition on interest-topics and change the reward to *user_model(i)*paper_model(i)* for studying the effect of multiplying user and tutor ratings. Finally, in *Closeness_7( )*, we

relax the condition on interest-topics and change the reward to *user_model(i)*paper_model(i)* for studying the effect of multiplying user and tutor ratings, and also reduce the penalty slightly into $(paper\_model(k) - 1)^2$.

Based on the closeness between a user and papers, we can then determine which paper(s) to recommend to the user, i.e. the closest paper(s) are recommended. We may also combine these results with those from collaborative filtering, which will be explained in the following section.

4.1.3 Collaborative Filtering (CF)

In the collaborative filtering (CF) method we recommend paper(s) to a user according to the highest rated papers of his/her neighbors. According to the method in determining his/her neighbors, we divide our CF methods into *rating-based CF*, *user-model-based CF*, and *a combination of them*. In rating-based CF, the neighbors of a target user are determined according to the correlation between ratings given by the target user and those given by other users. In user-model-based CF, the neighbors are determined according to the correlation between the user models of the target user with those of others. As in section 4.3.2 in our experiment, these user models are represented by learner features collected using the questionnaire (shown in Figure 4-1 and Appendix B). Moreover, according to the dimensionality of ratings of user models used, we can further sub-categorize them as uni-dimensional or multi-dimensional CF.

4.1.3.1 Uni-Dimensional Rating-Based Collaborative Filtering (1D-CF)

Uni-dimensional rating-based[1] CF is in fact the traditional CF that has been discussed in the literature. A target user's neighbors are determined according to the Pearson correlation of ratings given to a set of co-rated papers *K* by the user and each of the other users, where only a single dimensional rating of those papers, e.g. their *overall rating*, is counted.

However, this may not apply in paper recommendation for novice learners, because most of them may not have read any paper at all (e.g. first year graduate students). Besides, not many papers are commonly assigned as part of a learning activity in a course. Thus, our research focus is for the case with a limited number of co-rated papers, i.e. |*K*| is more than 1 but up to 15 only, with emphasis on |*K*| less than or equal to 5. Indeed, a higher number |*K*| is desirable as long as it is available from the learners, because it can increase the accuracy of recommendation which will be shown later.

The number of neighbors |*B*| used to is set to be from 2 to 15. Note that a higher number of |*B*| may not always be helpful in increasing the accuracy of recommended papers. The reason is that an excessive number of neighbors may include those with a very low Pearson correlation, or even negative correlation to the target user, which in turn may reduce the accuracy of recommendation.

4.1.3.2 Multi-Dimensional Rating-Based Collaborative Filtering (3D-CF)

Typically, users only provide a single rating for each item; for instance, whether a user likes/dislikes a movie, song, book, etc. Therefore, the recommended items to a target user can be found by calculating the estimated rating of target user *a* on item *j*, denoted

by $r^e_{a,j}$ (the superscript indicates the rating is an estimation) for all *j*, where a higher estimated rating means that the target user may like the item more than those with a lower estimated rating. However, this is not the case in the evaluation of learning material, as we may not only want to know whether students like/dislike a recommended paper but also whether or not they have learned something from reading the paper, and/or whether or not they have difficulty in understanding the paper, etc. Thus, we need to consider multiple ratings (dimensions) on various aspects in recommending a new paper.

Therefore, we consider three dimensional CF here where the dimensions include the *overall ratings*, *value-addedness* and *peer recommendation*. The following are example of questions used to extract those ratings from students after they read a paper (see Appendix C for a complete questionnaire):

5. (value-addedness) Do you learn something "new" after reading this paper?
   4. absolutely 3. relatively 2. not really 1. not at all

6. (overall rating) What is your overall rating towards this paper?
   4. very good 3. good 2. relatively 1. bad

7. (peer recommendation) Will you recommend this paper to your fellow classmates?
   3. absolutely yes 2. maybe 1. no

Recall that in CF, the estimated rating for a target user *a* on paper *j* $r^e_{a,j}$ is computed through their neighbors, denoted as *B*, using the formula.

$$r^e_{a,j} = \frac{\sum_B P(a,b) \times r_{b,j}}{\sum_B P(a,b)} \qquad (4\text{-}1)$$

[1] Some RSs such as knowledge-based RSs rely on critiques instead of pure ratings to make recommendations; hence, in order to differentiate rating-based CF from non-rating-based CF, we clearly specify this in the naming of this method.

where *P (a, b)* denotes the Pearson correlation of ratings given by *a* and *b*. Note here that we have two variables which may affect the value of $r^e_{a,j}$: number of co-rated papers in calculating the Pearson correlation, i.e. |*K*|, and the number of neighbors used in making the recommendation, i.e. |*B*|. Without lost of generality, we assume that the target user is *a*. Therefore, for the sake of brevity, we drop subscript *a*, unless otherwise specified.

The weighted sum Pearson correlation is used to calculate the distance between two users. Suppose $P_d(a, b)$ is the Pearson correlation based on the rating *r* on dimension *d*:

$$P_d(a,b) = \frac{\sum_K (r_{a,k} - \bar{r}_a)(r_{b,k} - \bar{r}_b)}{\sqrt{\sum_K (r_{a,k} - \bar{r}_a)^2 \sum_K (r_{b,k} - \bar{r}_b)^2}} \qquad (4\text{-}2)$$

where $r_{i,k}$ is the rating by user *i* on co-rated item *k* on dimension *d*. Others are the same as those discussed in Chapter 2.

The weighted sum Pearson correlation for our 3-D CF can be formulated as:

$$P_{3D}(a, b) = w_{overall} P_{overall}(a, b) + w_{valueadd} P_{valueadd}(a, b) + w_{peer\_rec} P_{peer\_rec}(a, b) \quad (4\text{-}3)$$

where $w_{overall} + w_{valueadd} + w_{peer\_rec} = 1$, and the values are determined by the tutor manually or the system automatically. In our experiment, these weights are tuned manually following a series of trials, in which those weights reported in this thesis are representative ones. However, automatic tuning should be performed in a fully operated system.

In our experiments, the number of neighbors used by this method is also set to be from 2 to 15, where they are selected according to the weighted sum Pearson correlation $P_{3D}(a, b)$ in Equation (4-3). After we identify the closest neighbors for a target user *a*, we

then calculate the *aggregate rating* of each paper to make recommendations. The selection method is by calculating the sum of weighted ratings by all closest neighbors:

$$r_j^{3D} = \sum_B P_{3D}(a,b) r_{b,j} \qquad (4\text{-}4)$$

Here r_j^{3D} is the aggregate rating to paper j from all neighbors in set B, $P_{3D}(a, b)$ is the weighted Pearson correlation between target user a and his/her neighbor b, and $r_{b,j}$ is the rating given to paper j by neighbor b. After we calculate r_j^{3D} for all papers, we can find the best paper(s) for recommendation, i.e. paper(s) with the highest r_j^{3D}.

4.1.3.3 User-Model-Based Collaborative Filtering (UM-CF)

In rating-based CF, a target user needs to have rated a few papers before we can find his/her neighbors, which is a major drawback, especially at the beginning of each semester where no ratings have been placed by students. This issue has been long known as one type of cold-start problem, i.e. the new user problem [Schein et al. 2002]. Fortunately, user-model-based CF can be used for users who previously have not rated any paper. The idea is as follows. Suppose we have a target user a, who has not rated any item, and many other users who have rated some items, e.g. on a Likert scale 1 to 5. Our RS will find an ordered list of neighbors $B = \{b_1, b_2, \ldots b_h\}$, such that $P(a, b_1) \geq P(a, b_2) \geq \ldots \geq P(a, b_h)$, where $P(a, b_i)$ denotes the *Pearson correlation* of user model a to its neighbor b_i (see Figure 4-1 for the attributes of this user model and possible values). In this case, the *Pearson correlation* is given by:

$$P(a,b) = \frac{\sum_K (r_{a,k} - \bar{r}_a)(r_{b,k} - \bar{r}_b)}{\sqrt{\sum_K (r_{a,k} - \bar{r}_a)^2 \sum_K (r_{b,k} - \bar{r}_b)^2}} \qquad (4\text{-}5)$$

where K is the set of *user model features* co-rated by both a and b; $r_{a,k}$ or $r_{b,k}$ is the rating given by user a or b to feature k; and $\bar{r}_a$ and $\bar{r}_b$ are their average rating to features in K. When two user model ratings are similar, their Pearson correlation is close to 1. Hypothetically, given their similar attitudes and preferences, both of them should also have a similar taste in rating a paper.

After computing the Pearson correlation between the target user and each other user, the system will select the neighbors of each target user as usual. However, in our experiment, we consider a user's interests and background knowledge separately (cf. Figure 4-1). Therefore, we get two Pearson correlations and need to combine them:

$$P_{2D}(a, b) = w_{interest}\, P_{interest}(a, b) + w_{backgrd\_knowledge}\, P_{backgrd\_knowledge}(a, b) \qquad (4\text{-}6)$$

where $w_{interest}$ and $w_{backgrd\_knowledge}$ are the weights used in our linear combination. After that, we can then calculate the estimated rating of target user a on item j, denoted by $r^{e}_{a,j}$, using a similar method to Equation (4-1) but only for 2D:

$$r_j^{2D} = \sum_B P_{2D}(a,b) r_{b,j} \qquad (4\text{-}7)$$

4.1.3.4 Combination of Rating-based and User-Model-Based Collaborative Filtering (5D-CF)

In previous chapters, we have provided motivation for incorporating student models in making pedagogical paper recommendation. Thus, we want to check whether or not the incorporation of learners' knowledge background would have either negative or positive effects on the performance of the recommender. As we argued earlier in traditional CF-based RSs, the user profile (such as the demographic data, life style [Lekakos and Giaglis 2005], etc.) is not considered, yet the profile contains a lot of

valuable information for a RS to improve its performance. Therefore, a combination of rating-based CF and user-model based CF may result in a more powerful recommendation method. In our experiment, for the sake of simplicity, we combine Pearson correlations of 3D rating-based CF and user-model-based CF in a linear form. Recall that from rating-based CF, we obtain 3D-Pearson correlation $P_{3D}(a, b)$, and we also have the 2D-Pearson correlation between learners based on their student models, $P_{2D}(a, b)$. Given these, we compute the aggregated Pearson correlation as follows:

$$P_{5D}(a, b) = P_{3D}(a, b) + w_{2D}\ P_{2DstdModel}\ (a, b) \qquad (4\text{-}8)$$

where $w_{2D} = \{0.5, 1, 3, 5, 10\}$. From $P_{5D}(a, b)$ we can identify the best neighbors for a target user. After that, we use a similar formula to calculate the aggregate rating of each paper:

$$r_j^{5D} = \sum_B P_{5D}(a,b) r_{b,j} \qquad (4\text{-}9)$$

After we calculate r_j^{5D} for all papers, we can find the best paper(s) for recommendation, i.e. paper(s) with the highest r_j^{5D}.

Note: since we explore $(w_{interest}, w_{bkgrKnowledge}) = (1, 0)$ in our experiments with various version of P_{2D}, we thus effectively have a version of (4-8) that is P_{4D}. Later, we will show that this composition is better than 5D-CF in terms of the accuracy of recommendation.

4.1.4 Hybrid Filtering

In later section we combined rating-based CF and user-model-based CF. Since both are based on collaborative recommendation, we do not categorize the combined method as a hybrid method. Rather, a hybrid method here is defined as a method which

combines non-personalized recommendation, user-item content-based filtering and collaborative filtering.

4.1.4.1 Combinations of Non-Personalized Recommendation and Rating-Based Collaborative Filtering (Pop1D and Pop3D)

In section 4.1.1 we have discussed non-personalized group recommendation where we only count the *popularity* of a paper within a group of learners, which in its simplest form is the average rating of the paper. In sections 4.1.3.1 and 4.1.3.2 we have also shown techniques for uni-dimensional rating-based CF (hereafter 1D CF) and multi-dimensional rating-based CF (hereafter 3D-CF). If the resulting estimated rating from a CF method is combined with the average rating (popularity) given by all users in a group, then we will have hybrid recommendations (hereafter Pop1D and Pop3D, respectively). In our experiment, the recommendation is determined by linear combination of the average rating given by all users in a group to a paper *j*, i.e. $\tilde{r}_j$, and a rating calculated from 1D- and 3D-CF, i.e. r_j^{1D} and r_j^{3D} respectively, which results in the following:

$$r_j^{Pop1D} = r_j^{1D} + w_r n\tilde{r}_j \qquad (4\text{-}10)$$

$$r_j^{Pop3D} = r_j^{3D} + w_r n\tilde{r}_j \qquad (4\text{-}11)$$

where $w_{\tilde{r}}$ is the weight of linear combination and is the control variable in our experiment; n is the number of neighbors in CF, i.e. $n = |B|$; and both r_j^{Pop1D} and r_j^{Pop3D} are the resulting estimated ratings to determine the recommended papers.

The reason for us to combine CF with the non-personalized method is to remedy the low accuracy of CF especially when the number of co-rated papers is low, i.e. by considering the 'authority' of a given paper according to people who have rated it: if the

paper has been well-received, then the 'authority' level of it is high. That is, if a given paper is popular among all the users, then the target user might also be interested in it. Although this popular paper cannot be used to differentiate the different tastes of users, it is worth recommending. In our experiment, we vary the weight to look at the effect it might have on the general outcome of the recommendations.

4.1.4.2 Combinations of Non-Personalized Recommendation and the Combined Rating-Based and User-Model-Based Collaborative Filtering (Pop5D)

In the previous section we have shown how to combine non-personalized recommendation with rating-based CF, which can be used to remedy the poor performance of rating-based CF when the number of co-rated papers is low. In section 4.1.3.4 we have also shown the technique to combine rating-based and user-model-based CF, resulting in a 5D-CF. In this section we will show how to combine non-personalized recommendation and the combination of rating-based and user-model-based CF, herein a hybrid 5D recommendation or Pop5D for short.

Remember from Equation (4-9) we have an estimated rating of the combined rating-based and user-model-based CF, i.e. $\boldsymbol{r_j}^{\boldsymbol{5D}}$. Similar to our approach in section 4.1.3.4, we can combine this rating with the average rating of each paper $\tilde{\boldsymbol{r}}_{\boldsymbol{j}}$ to obtain a hybrid 5D recommendation (Pop5D) for the papers:

$$\boldsymbol{r_j^{Pop5D} = r_j^{5D} + w_r n\tilde{r}_j} \qquad (4\text{-}12)$$

where n is the number of neighbors ($n = |B|$). Note that the Pop5D actually upgrades Pop3D, by injecting two aspects of student models into the recommendation process.

Although the computation in Pop5D is more complex than the other CF-based recommendation approaches, we speculate, under certain circumstances, it might help

inform the recommender and therefore improve the recommendations. We performed a series of experiments to investigate the necessity and effectiveness of Pop5D. In our experiment, we will shown that Pop4D (a reduction of Pop5D) where we only consider user interest or ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0)) generates more accurate recommendation than Pop5D. Due to a large number of possible combinational values of our parameters, our experiments consist of a series of reduced treatments summarized in Table 4-3, where only a few combinational representative weights are considered. The result is 3x5x2x2 = 60 different treatments.

Table 4-3 Treatment used in Pop5D

Student models		*Paper's popularity* $\tilde{\boldsymbol{r}}_{\boldsymbol{j}}$	
($w_{interest}$, $w_{bkgrKnowledge}$)	(1, 0), (1, 0.5), (1, 1)	$w_{\tilde{r}}$ (weights in $\tilde{\boldsymbol{r}}_{\boldsymbol{j}}$)	1, 5
$w_{2DStdModel}$	0.5, 1, 3, 5, 10	$\|B\|$ (number of neighbors)	5, 10

We do not combine non-personalized recommendation with solely user-model-based recommendation. However, as shown in Table 4-3, we may have $w_{2DStdModel}$ = 10 where a very high weight put on user-model-based ratings gives us a case where user-model-based CF has a small proportion of rating-based CF.

4.1.4.3 Combinations of Non-Personalized Recommendation and User-Model-Based Collaborative Filtering (PopUM-CF)

As mentioned previously, one of the drawbacks of rating-based CF is its reliance on co-rated papers which may not exist for new users. To tackle this, we may use either user-model-based CF or non-personalized recommendation. A direct result is we may also want to combine both methods to find a possibly more accurate recommendation method. This combination is nothing but a reduction of the combined approach explained

in section 4.1.4.2, where we ignore the rating-based CF composition from Equation (4-12). Thus, what we have is:

$$r_j^{Pop2D} = r_j^{2D} + w_r n\tilde{r}_j \qquad (4\text{-}13)$$

where r_j^{2D} is obtained from Equation (4-7) in section 4.1.3.3, n is the number of neighbors = $|B|$.

4.1.4.4 Combinations of Content-Based Filtering, Non-Personalized Recommendation, and User-Model-Based Collaborative Filtering (PopCon2D)

Another hybrid method is to combine content-based filtering discussed in section 4.1.2 with non-personalized recommendation and user-model-based CF, namely PopCon2D (for **Pop**ularity + **Con**tent-based filtering + **2D** user-model-based CF). However, we shall normalize the closeness value generated by each algorithm in Figure 4-2 before we can combine them. The easiest normalization is to divide each value with max (|*closeness*|) so that our closeness value is always between [-1, 1].

Suppose $\tilde{r}_j$ is the average rating of paper j (where a paper's *popularity* is on a 4 point Likert scale and r_j^{2D} is the estimated ratings of neighbors in user-model-based CF, i.e. Equation 4-7. The recommended paper(s) can be calculated by the following formula on each paper j:

$$r_j^{PopCon2D} = r_j^{2D} + n\left(w_r\tilde{r}_j + w_c Closeness\right) \qquad (4\text{-}14)$$

where w_r is the weight assigned to paper popularity and w_c is a weight on the closeness value calculated from content-based filtering. Again, paper(s) with a higher rank will be picked up. We may also reduce the PopCon2D method into PopCon only, by not considering user-model-based CF.

In the paper recommendation research community, this type of hybrid recommendation algorithm is rarely seen due to the fact that existing research in paper recommendation is still restricted to treat papers as similar to items in other domains where only users' interests are considered.

4.1.4.5 The Artificial-Learner-based Approach

As discussed in the previous chapter, one of the most challenging problems in RSs is the "*cold-start*" problem. Fortunately, so far, quite a lot of research has attempted to tackle this problem, including our non-personalized recommendation and user-model-based CF (assuming we have a new user but the papers have been rated by others), and content-based filtering (assuming we have a new user and/or a new paper), or a combination among these methods. However, one may wonder whether we can have a user-model-based CF for a new user and a new paper case. To realize this, we propose an artificial-learner-based approach.

In an educational domain, the cold-start problem can become worse since there may be many more items than users [Herlocker et al. 2004]. Indeed, the cold-start problem may appear frequently in paper recommendation, because there are many new papers published and often a relatively small number of students enrolled in a class each year, which increases the difficulty to find 'nearest neighbors' of a target learner, whether using CF or non-CF approaches. Moreover, we cannot use random assignment for two reasons: it may degrade teaching quality, and learners do not have the time or motivation to read too many irrelevant papers. One might suspect that since the candidate papers used in the experimental analysis in this thesis are well-selected, therefore the credibility and reliability of the approaches and the results are questionable. However, like those

well-selected ‘high quality’ recipes in *Kalas* [Svensson et al. 2005], we believe the result is still convincing.

In an artificial-learner-based approach, first we randomly generate learners’ features in terms of their interest in different topics and their background knowledge. Then, we use similar algorithms to those used in content-based filtering to generate artificial ratings for each paper. After this process, our artificial learner recommendation method is ready to be deployed: when a target new user comes, we use user-model-based CF to find his/her artificial neighbors, which can recommend a set of papers to him/her as if they are real human neighbors. Certainly, the accuracy of recommendation in this approach relies on the “correctness” of artificial ratings generated by artificial learners and the accuracy of finding “similar” neighbors. Since the uncertainty factor in this approach is higher than from content-based filtering and user-model-based CF, a lower accuracy may be expected. However, this approach may have some benefits: firstly, we may generate as many artificial learners as we want, which may get closer to the taste of the real target learner; thus, we can remedy the lack of similar human learners in user-model-based CF [Tang and McCalla 2004]. Secondly, we may use machine learning techniques to update the prior artificial ratings of an artificial learner according to the feedback (ratings) by its human neighbors of the same paper, thus, creating many prototypical learner models. Artificial learners are also useful if the rating database is sparse, as such, these pseudo ratings can be generated convenient and injected to database on which CF-based techniques can then be performed.

4.1.5 Manual Recommendation (Tutor Recommendation)

Manual recommendations are where the tutor examines both the user and paper features in order to determine the set of the recommended papers. Let us first look at an example. Table 4-4 lists out the characteristics of Learner 78[2].

Table 4-4 A user model example to elaborate tutor recommendation

Learner Interest												
T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**	**T11**	**T12**	**T13**
3	5	5	**4**	**4**	**3**	**3**	**3**	**3**	**3**	**4**	5	5
Knowledge Background												
K1	**K2**	**K3**	**K4**	**K5**								
4	**2**	**2**	**3**	**2**								
Learning Goals												
General SE		**PM**		**Agile**		**Testing**		**WebEng.**		**UI**		**Other**
1		**0**		1		**0**		1		1		**0**

From Table 4-4 (see interpretation of T*i* and K*i* in Figure 4-1) and Figure 4-1 we can see that learner 78 is interested in web applications (T2) and user interfaces (T3) (preferably those articles embedded with case discussions, T12, T13), while he/she is proficient in programming (K1), and is not weak in math (K4). His/her job is software development, and his/her major learning goals[3] are aligned with his/her interests, i.e. web application and user interface, as well as a general understanding of agile software development and software development as a whole. In manual recommendation, first the tutor selects papers that closely match learner 78's interests (refer to the candidate papers

[2] Note that since the data collection procedure is anonymous and not carried out by the author of this thesis, as necessitated by ethics requirements, all the learners in this study are identified by numbers.

and their associated features, listed in Table 4-2): papers 8, 10, 15, 16, 17. Then, the tutor considers the learner's other pedagogical features, such as knowledge background, learning goal, etc. Since the learner states that he/she is eager to learn something on agile programming, therefore paper 6 focusing on agile software development is added to the list. Meanwhile, the learner is proficient in programming; therefore, papers 8 and 16 are pushed up high in the list due to their emphasis on associating programming terms with the content. In addition, paper 16 is a case study which matches the learner's interest well. Now the list becomes: 6, 8, 16, 17, 10. Paper 17, focusing on usability engineering, is a classical tutorial paper on this topic and thus has a high reputation; hence, it is recommended. Paper 15 discusses issues the importance of teaching key user interface design rules in software engineering education. Although the paper matches the learner's interest, but since the paper's popularity among users is low, therefore it is removed from the list. In summary, learner 78 receives the following recommended papers: 6, 8, 10, 16, and 17.

Figure 4-3 illustrates the recommendation flow. Clearly, when tutors make paper recommendations, they should consider the many feature dimensions of the learner in order to tailor the papers.

[3] Learner 78's learning goal is to gain a general understanding of agile software development and software development as a whole. But he/she wants to know more about web application and user interface, two topics that he/she is interested in.

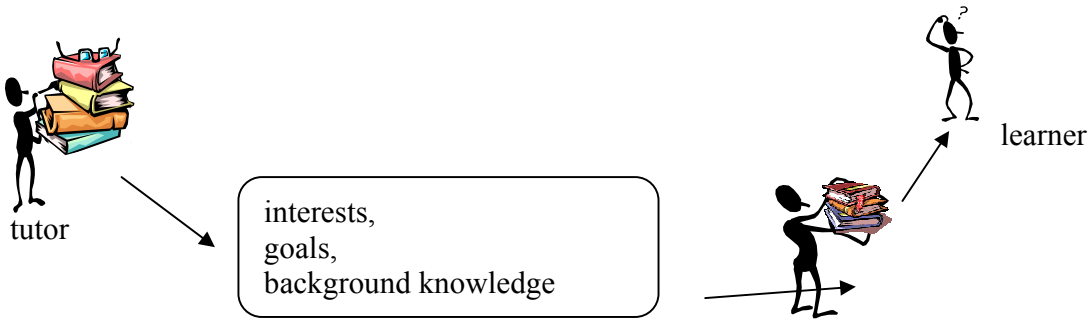


Figure 4-3. Flow of the manual tutor recommendation process

4.1.6 Discussion: Pedagogical Paper Recommendation and Its Relationship with Multi-dimensional Recommendation

Recall that in Adomavicius et al. 's approach [2005][3], the recommendation space consists of not only item and user dimensions, but also many other contextual dimensions, such as location, time and so on. Although the approach takes into account some additional contextual information in making the recommendations, the computation methods are still similar to those in traditional one-dimensional RSs. For example, one of the computational approaches is to reduce the original dimensions to only considering the relevant recommendation space; say, we can remove the *time* dimension by limiting recommendations to weekdays only. In [Lekakos and Giaglis 2006], the closeness between users is measured in terms of their *lifestyle* instead of *pure ratings* in traditional CF: the chance that users with the same lifestyle tend to have similar tastes will be higher. Similar to the approach in [Lekakos and Giaglis 2006], learners' pedagogical features are used in our approach to measure the similarity between them. However, we also consider paper features in the recommendation process which is different from the existing

[3] In fact, as early as in 1999, Pazzanni [1999] pointed out that contextual information is valuable for making recommendations. Over the years, there have been some studies discussing the use of contextual information to make recommendations. However, it is not until 2005 that the term multi-dimensional CF came into being, coined by Adomavicius et al. [2005].

approaches that only consider users' contextual information in making recommendations [Lekakos and Giaglis 2006, Pazzanni 1999, Adomavicius et al. 2005]. For instance, the *overall authority* of each paper (i.e. *popularity* denoted by $\tilde{r}_j$) is used to factor out papers that are not well received.

In the 3D-CF approach, we consider papers' overall-ratings, value-addedness and peer recommendations. As for the learners' pedagogical features, in order to study how important they are in the recommendation process, we assign different weights, and tune the values to search for satisfying results.

To summarize, our approaches make use of both paper and user features to construct a learner profile for making recommendations. In addition, we also take into consideration overall paper popularity in an attempt to start up the recommendation.

Table 4-5 below summarizes those factors considered in our multi-dimensional CF to correlate one user with another.

Table 4-5 Factors that are considered in our Multi-Dimensional CF.

Dimension	**Factors**
3D	Overall rating, Value-addedness, Peer recommendations
5D	Overall rating, Value-addedness, Peer recommendations, Learner interest, Learner background knowledge
Pop3D	Overall rating, Value-addedness, Peer recommendations, $\tilde{r}$
Pop5D	Overall rating, Value-addedness, Peer recommendations, Learner interest, Learner background knowledge, $\tilde{r}$

Obviously, many of the approaches summarized in Table 4-6, when it comes down to the computation, still rely on the overall rating an item has received to find neighbors for a target user. This is fundamentally different from our proposed methods described in this chapter. In fact, our proposed methods, to some degree are more

complicated than other techniques in which tens of thousands of evaluations can easily be performed. In our domain, both the number of candidate papers and learners are limited, which forces us to develop recommendation mechanisms which do not so heavily rely on the numerical ratings.

Table 4-6 Composition of our approach with other multi-dimensional CF approaches

	Type of additional information	Method of finding neighbors for a target user
Adomavicius et al. 2005	• users' demographic data, • item information, • consuming information	• users' overall rating toward each item
Pazzani 1999	• users' demographic data • content information	• learned user profile based on content information • user's overall rating toward each item
Lekakos and Giaglis 2006	• users' demographic data • lifestyle data	• learned user profile based on content information • user's overall rating toward each item
Our approach	• user models • paper features	• learned user profile based on content information, • user's overall rating toward each item, • the popularity of each paper

The following table, Table 4-7, organizes and categorizes the various recommendation techniques discussed in this section.

Table 4-7 A summarization of the various recommendation techniques.

		Rec. type	Type of object	Type of data	Useful for cold-start?
	Manual rec.		User model Paper model		Y
Pop1D	Non-personalized	Non-personalized	$\tilde{r}_k$	$\tilde{r}_k$	N
ContentF	User-item	Content-based	*user_model(i)* *paper_model(k)*	Interest Knowledge background	Y
1D-CF	Uni-dimensional	CF	Paper rating	Overall rating	N
3D-CF	Multi-dimensional	CF	Paper model	Overall rating; Value-addedness; Peer recommendation	N
2D-CF	User-model based	CF	User model	Interest and background know.	Y
5D-CF	Rating- and user model based	CF	User model Paper model	Overall rating; Value-addedness; Peer recommendation Interest and background know.	N
Pop3D	Non-personalized and rating-based CF	Hybrid	$\tilde{r}_k$ Paper model	$\tilde{r}_k$ Overall rating; Value-addedness; Peer recommendation	Y
Pop5D	Non-personalized and 5D-CF	Hybrid	$\tilde{r}_k$ User model Paper model	$\tilde{r}_k$ Overall rating; Value-addedness; Peer recommendation Interest and background know.	Y
Pop2D	Non-personalized and User-model based CF	Hybrid	$\tilde{r}_k$ User model	$\tilde{r}_k$ Interest and background know.	Y
PopCon2D	A combination of Non-personalized , user-item content filtering and 2D-CF	Hybrid	$\tilde{r}_k$ *user_model(i)* *paper_model(k)*	$\tilde{r}_k$ Interest and background know.	Y
AL	Artificial learner-based	Hybrid	User model	Interest and background know.	Y

In the next section, we will discuss our experimental analysis comparing these various recommendation strategies, and make suggestions accordingly on their appropriateness under various conditions in our domain.

4.2 Parameter Tuning

Due to the large amount of parameter combinations, we assume the independency among parameters used in the recommendation techniques. This assumption is not entirely correct, because we observed inter-dependency among some parameters in terms of the average Overall ratings. However, this assumption speeds up the tuning process because it allows us to tune each parameter separately. Nonetheless, in some cases, especially when several parameters are foreseen as dependant ones, we have relaxed the assumption and adopt a simultaneous-tuning method that largely increases our search space. In most cases, we picked the parameter values through manual adjustment: we pick and test a set of values within a larger interval, and then pin down to a smaller interval that results in a better performance, and so forth until a further adjustment does not significantly increase the performance. In general, the following parameters need to be tuned during experimentation:

1. the number of neighbors and co-rated papers used in CF-based recommendation
2. the weights of learner model (learners' interest and background knowledge) used in content-based filtering
3. the weights of learner ratings on the Peer Recommendation and Value-addedness used in CF-based recommendation
4. the weights of the predicted ratings obtained from the content-based filtering, CF-based recommendation, and paper popularity in a hybrid recommendation

We are particularly interested in parameter settings that can work effectively when the number of co-rated papers and neighbors are relatively small (since in our domain, these conditions are more common). In this thesis, those parameter settings with less significant results were discarded, and not included in the discussions afterward. However, we will describe each setting used in our comparison and explain why we discard some settings, if any.

4.2.1 Non-personalized recommendation (Benchmark)

The averages Overall rating (*popularity*) of our 21 papers from the highest to the lowest are {3.167, 3.042, 2.96, 2.958, 2.84, 2.826, 2.8, 2.8, 2.8, 2.8, 2.76, 2.75, 2.75, 2.68, 2.667, 2.667, 2.56, 2.542, 2.208, 2.208, 2.08}. A system using a non-personalized recommendation technique will always recommend the paper(s) of the highest average rating; hence, the expected average rating obtained from recommending the most popular (top 1) paper is 3.167. Similarly, the expected ratings of recommending the top 3 and the top 5 most popular papers using this technique are 3.055 (the average of the top 3), and 2.992 (the average of the top 5), respectively. These values are adopted as *the best-case benchmark* for other recommendation techniques. We call it the best case because in this case we assume all ratings are available prior to recommendation.

In another extreme situation, no rating is available for any paper. In this case, a non-personalized recommendation method will randomly assign those papers to learners. The expected result of recommending the top 1, the top 3, and the top 5 papers is just the expected ratings for all papers, which is the average rating, 2.708. This method can be regarded *as the worst-case benchmark* method. It can be used as an alternative benchmark for the case when no prior rating is available for a paper.

4.2.2 User-item content-based filtering (ContentF)

Altogether, seven different functions have been evaluated in this method, i.e. *Closeness_1(), …, Closeness_7()*. For each function, we assign two different *weights*, 0 and 1, which is used to tune the importance of knowledge background in determining the appropriateness of a paper to be recommended to the learner. Since all functions are deterministic, we only performed analyses of $7 \times 2 = 14$ combined methods, where each of them consists of 25 data points taken from 25 learners. The results (the average Overall ratings of recommended papers) are shown in Table 4-8. Column "Top 1" refers to the average rating when the system recommends only one (the best) paper. Column "Top 3" and "Top 5" refers to the average rating when the system recommends the best 3 and 5 papers to the learners respectively. On average, closeness_7 with weight = 0 has the highest average Overall rating; thus, it is used in the comparison with other recommendation methods.

Table 4-8 Average Overall ratings obtained from the user-item content-based filtering recommendation

	Top 1		Top 3		Top 5	
Weight	0	1	0	1	0	1
closeness_1	2.880	3.000	2.707	2.813	2.752	2.760
closeness_2	2.840	2.840	2.720	2.933	2.800	2.784
closeness_3	3.000	3.000	2.853	2.907	2.792	2.800
closeness_4	3.080	3.040	2.827	2.867	2.784	2.768
closeness_5	2.960	2.960	2.880	2.880	2.792	2.784
closeness_6	3.120	3.120	2.973	3.000	2.840	2.840
closeness_7	3.160	3.120	3.013	3.027	2.880	2.896

4.2.3 One-dimensional rating-based collaborative filtering (1D-CF)

In this method, we randomly select a set of co-rated papers for calculating the Pearson correlation between a target learner and the other 24 learners to obtain his/her 10 neighbors. The number of suggested neighbors (= 10) is not arbitrarily chosen, but was selected after our analysis using various number from 2 to 15 (the results will be explained later in the discussion part). Fourteen different numbers of co-rated papers, i.e. $|K| = \{2, 3, \ldots, 15\}$, have been tested in this method. For statistical purposes, we repeat the random selection of co-rated papers 32 times, resulting in 32 trials/learner × 25 learners = 800 trials done for each treatment.

Since a co-rated paper may be recommended back to learners, using a large number of co-rated papers may taint the predictive power of the recommendation system (this will be explained later in the discussion part). In addition, a large number of co-rated papers are not available in most courses. Hence, for the purpose of our comparison with other recommendation methods, we present the results of the cases when the number of co-rated papers equal to 2, 4, or 8 as typical for small, medium and large co-rated sets of papers, respectively. These values will be applied to all other recommendation techniques which use rating-based CF.

4.2.4 Multi-Dimensional Rating-Based Collaborative Filtering (3D-CF)

In multi-dimensional rating-based collaborative filtering we consider ratings on Overall, Value_added, and Peer_rec ratings, in both finding neighbors and calculating the predicted recommended papers. The weighted sum Pearson correlation is:

$$P_{3D}(a, b) = w_{overall}\, P_{overall}(a, b) + w_{valueadd}\ P_{valueadd}(a, b) + w_{peer\_rec}\, P_{peer\_rec}(a, b) \quad (4\text{-}3)$$

The weights assigned to each dimension, i.e. ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}), are set to be: (0.8, 0.1, 0.1), (0.9, 0.05, 0.05), (0.91, 0.05, 0.04), (0.91, 0.04, 0.05), (0.92, 0.04,

0.04), (0.92, 0.05, 0.03), (0.93, 0.03, 0.04), (0.93, 0.04, 0.03), (0.94, 0.04, 0.02), (0.94, 0.03, 0.03), (0.96, 0.02, 0.02). The combinations of the weights here are not chosen randomly; in fact, a lot other sets have been tested, for instance, (0.25, 0.25, 0.5) and (1/3, 1/3, 1/3). However, we found out that only when the second and third weights are less than 0.1 can the benefits of the multi-dimensional CF manifest. Hence, in our comparison with other methods we use (0.8, 0.1, 0.1). Other parameters (number of neighbors and co-rated papers) and the number of trials for each treatment are the same as those of one-dimensional CF.

4.2.5 User-Model-Based Collaborative Filtering (UM-CF)

$$P_{2D}(a, b) = w_{interest}\, P_{interest}(a, b) + w_{backgrd\_knowledge}\, P_{backgrd\_knowledge}(a, b) \qquad (4\text{-}6)$$

Where $P(a, b)$ denotes the Pearson correlation between the target user and the candidate user: it is used to measure the closeness between the two users, in terms of their interest and background knowledge. After that, neighbors can be found.

Unlike rating-based CF, the UM-CF is a deterministic method; thus, no repetition is needed. In total, only 25 data points (25 users) are collected from each treatment where we did vary the values of those control parameters.

Besides the number of neighbors used in CF we have a pair of control parameters in this recommendation technique, ($w_{interest}$, $w_{bkgrKnowledge}$), which are set to some representative values: {(.5, .5), (.8, .2), (.9, .1), (1, 0)}. Our experimental results show that the combination (1, 0) is the best among these four in terms of the average Overall rating given to the top-five-recommended papers. Also, this result is achieved when we use 5 neighbors in CF. Table 4-9 shows the average ratings given by 25 learners after reading the top-five-recommended papers from their neighbors.

In our experiment, we only checked the cases where the number of neighbors = {5, 6, 7, …, 15}. The bold values shown in Table 4-9 represent the highest average rating in each column, where 2.968 is the highest among all ratings (column (1, 0) with number of neighbors = 5). Thus, this combination is used to represent this recommendation method in our comparison with other methods. Also, we will use 5 neighbors in hybrid methods which consist of user-model-based CF, unless otherwise specified.

Table 4-9 Average Overall ratings obtained from the user-model-based CF

Neighbor	(0.5, 0.5)	(0.8, 0.2)	(0.9, 0.1)	(1, 0)
5	2.832	2.88	2.936	**2.968**
6	2.84	2.88	2.88	2.936
7	2.856	2.896	2.888	2.912
8	2.848	2.936	**2.96**	2.928
9	2.8	**2.944**	2.928	2.952
10	2.856	2.928	2.936	2.936
11	2.864	2.912	2.904	2.928
12	2.872	2.88	2.944	2.912
13	**2.896**	2.904	2.928	2.944
14	2.888	2.912	2.888	2.912
15	2.888	2.888	2.888	2.92

4.2.6 Combination of Rating-Based and User-Model-Based Collaborative Filtering (5D-CF)

Since we have rating-based CF (3D-CF) in this combination, each treatment will be repeated 800 times as in other cases involving rating-based CF. In addition to the control parameters used in both the rating-based CF and the user-model-based CF described earlier, another control parameter is w_{2D} = {0.5, 1, 3, 5, 10}, which is a linear weight used to combine the results from rating-based and user-model-based CF, as in Equation 4-8 and shown here:

$$P_{5D}(a, b) = P_{3D}(a, b) + w_{2D}\ P_{2DstdModel}(a, b) \qquad (4\text{-}8)$$

The number of neighbors used in this method is set to 10 for both rating-based and user-model-based CF, a value chosen without further evaluation due to the impracticality of exhaustive testing on a large number of combinations. According to what we obtained in the previous section, 4.2.5, adopting ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0) is the best in terms of the average Overall rating. Table 4-10 shows partial results that justify this adoption, where two sets of ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) values and three different values for the number of co-rated papers are used, and the system recommends the top paper.

Table 4-10 Average Overall ratings obtained from the combination of rating-based and user-model-based CF where w_{2D} = 1

($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) =		**(0.8, 0.1, 0.1)**		**(1, 0, 0)**	
($w_{interest}$, $w_{bkgrKnowledge}$) =		**(1, 0)**	**(0.5, 0.5)**	**(1, 0)**	**(0.5, 0.5)**
No. of co-rated paper, $\|K\|$	**2**	2.965	2.864	2.971	2.873
	4	3.078	2.943	3.071	2.966
	8	3.131	2.955	3.146	2.971

In all settings, adopting ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0) is better than adopting (0.5, 0.5). It is also true for top-3 and top-5 recommendations, and for other numbers of co-rated papers. Hence, we use ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0) for the comparison with other recommendation methods. For the sake of consistency when we compare this method to other rating-based CF methods, the number of co-rated papers in rating-based CF, $|K|$, is set to 2, 4, and 8.

4.2.7 Combinations of Non-Personalized Recommendation and Rating-Based Collaborative Filtering (Pop1D and Pop3D)

Both 1D- and 3D-CF are combined with non-personalized recommendation (best-case benchmark), namely Pop1D and Pop3D respectively, as shown in Equation 4-10 and 4-11:

$$r_j^{Pop1D} = r_j^{1D} + w_r n\tilde{r}_j \qquad (4\text{-}10)$$

$$r_j^{Pop3D} = r_j^{3D} + w_r n\tilde{r}_j \qquad (4\text{-}11)$$

The setting of this method is similar to that in 1D- and 3D-CF, except with additional weights $w_r = \{0.05, 0.1, 0.2, 0.4, 0.5, 1, 2, 3, 4, 5\}$ for combining them with the benchmark. Furthermore, in our experiment, we only consider the number of neighbors $n = \{5, 10\}$. The selection of these weights is rather arbitrary, with the higher weight resulting in higher overall rating when the number of co-rated papers is low, as shown in Table 4-11.

Table 4-11 Average Overall ratings obtained from the combination of non-personalized recommendation and rating-based CF for the recommendation of top-five papers

Pop1D										
w_r =		0.1	0.2	0.4	0.5	1	2	3	4	5
No. of co-rated papers, $\|K\|$	2	2.895	2.897	2.905	2.917	2.953	2.974	2.981	**2.983**	**2.983**
	4	2.952	2.958	2.965	2.968	2.988	2.995	2.992	2.994	**2.995**
	8	3.023	**3.024**	3.022	3.022	3.018	3.009	3.007	3.001	2.995
Pop3D with ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) = (0.8, 0.1, 0.1)										
w_r =		0.1	0.2	0.4	0.5	1	2	3	4	5
No. of co-rated papers, $\|K\|$	2	2.903	2.911	2.923	2.927	2.961	2.986	**2.997**	2.995	2.995
	4	2.950	2.957	2.967	2.969	2.989	2.993	2.991	**2.995**	**2.995**
	8	3.018	**3.022**	3.020	3.019	3.014	3.004	3.001	2.995	2.995

The highest average ratings are achieved when w_r = 3, 4, or 5 for $|K|$ = 2 or 4. We choose w_r = 5 for comparison with other recommendation methods.

4.2.8 Combinations of Non-Personalized Recommendation and the Combined Rating-Based and User-Model-Based Collaborative Filtering (Pop5D)

We collected 800 data points for each treatment in this method. We performed a series of experiments. Due to a large number of possible combinational values of our parameters, our experiments consist of a series of reduced treatments summarized in Table 4-12, where only very few combinational values are considered. The result will be 3x5x2 = 30 different treatments.

Here, we combine the best-case benchmark method with 5D-CF. Table 4-12 shows some experimental results on this method for the case of top-five paper recommendation when ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) = (0.8, 0.1, 0.1).

Table 4-12 Average Overall ratings obtained from the combination of non-personalized recommendation and the combined rating-based and user-model-based CF for the recommendation of top-five papers

Pop5D with ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) = (0.8, 0.1, 0.1)											
w_r =		**1**					**5**				
W_{2D} =		**0.5**	**1**	**3**	**5**	**10**	**0.5**	**1**	**3**	**5**	**10**
$\|K\|$	**2**	2.969	2.966	2.959	2.960	2.962	**2.988**	2.988	2.979	2.978	2.965
	4	**2.997**	2.990	2.983	2.969	2.968	2.995	2.993	2.984	2.976	2.962
	8	**3.019**	3.016	2.988	2.980	2.975	2.994	2.992	2.975	2.964	2.968

It is shown here for each block where w_r = 1 or w_r = 5 that higher average ratings are obtained on each row when w_{2D} = 0.5. Hence, we pick w_r = 1 for the comparison with other recommendation methods, and when w_{2D} = 0.5.

4.2.9 Combinations of Non-Personalized Recommendation and User-Model-Based Collaborative Filtering (PopUM-CF)

The setting in the Pop UM-CF is the same as that in user-model-based CF with the addition of w_r = 5. The number of neighbors $|B|$ used in CF is also 5.

4.2.10 Combinations of Content-Based Filtering, Non-Personalized Recommendation, and User-Model-Based Collaborative Filtering (PopCon2D)

Since both content-based filtering and user-model-based CF are deterministic methods, we do not need to repeatedly apply this combination for each treatment. In our experiment we pick w_r = {0, 0.1, 0.2, 0.3, ..., 1.3}, w_c = {0, 0.05, 0.1, 0.15, 0.2, 1}. When w_c = 0.15 and w_r = 0.1, and the system yields relatively better performances and the results are reported.

4.2.11 The Artificial-Learner-Based Approach

Six different methods are used to generate artificial ratings from artificial learners. 20000 artificial learners with randomly generated "interest" and "knowledge background" are created in each method. After that, we use user-model-based CF to find artificial neighbors of a target human learner. Then, those neighbors will recommend a paper to the learner who rated the paper. Since 20000 artificial learners are created, we do not need to perform repeated recommendations to each learner; thus, only 25 data points are collected from each method.

Table 4-13 shows the average Overall rating given by learners after the recommendation by six different types of artificial learners. The recommendation by AL4 results in the highest average ratings (shown in bold in Table 4-13). The method to generate artificial ratings in AL4 is shown in Figure 4-4.

Table 4-13 Average Overall ratings obtained from the recommendation by six different methods in artificial learners

	AL1	AL2	AL3	AL4	AL5	AL6
Top-1	2.880	2.880	2.840	**2.880**	2.840	2.880
Top-3	2.747	2.773	2.787	**2.787**	2.760	2.773
Top-5	2.736	2.776	2.752	**2.808**	2.720	2.792

```
FUNCTION Rating( )
        FOR EACH topic i
                IF user_model(i) ≥ 4 AND paper_model(i) ≥ 4 THEN
                        interest = interest + 1

        FOR EACH knowledge k
                IF user_model(k) ≤ 2 AND paper_model(k) ≥ 3 THEN
                        understanding = understanding + 1

        Score = 4 * interest – understanding
        IF Score > 8 THEN RETURN 4
        ELSE IF Score > 4 THEN RETURN 3
        ELSE RETURN 2
END FUNCTION
```

Figure 4-4 The function used to generate artificial ratings with artificial learners

4.2.12 Manual/Tutor Recommendation

Here, one human tutor assigns papers to each of the 25 learners; thus, only 25 data points are collected. The selection method is explained in section 3.5. Since this is not an automatic method, it is included for reference only.

4.2.13 Summary

In the experimental results, we will also show ceiling referred to as the highest possible ratings obtained from recommending the top-N papers to learners. Ceiling is calculated by averaging the highest-N ratings from each learner. The ceiling of our data for top 1, top 3, and top 5 is 3.560, 3.347, and 3.248, respectively.

Some techniques involve several combinations of parameters, which have been tested in our experiments. However, we will only use some parameter settings in our analysis, those that we have left after discarding those with less significant results. The following sections describe each setting used in our comparison and explain why we discard some settings, if any.

4.3 Experimental Results and Analysis

4.3.1 Data and Evaluation Protocol

Based on the collected data, we performed a series of experimental studies to tune the various recommendation methods discussed in section 4.1 in order to determine which weights and other variables work best in which circumstances and make comparisons on recommendation performance.

Different from that in classical literature, we do not use the "all-but-one" protocol for the evaluation of our recommendation techniques. In the "all-but-one" protocol (mainly in the CF-based recommendation), all but one of the ratings given by a target user are used to find neighbors who eventually used to predict the rating of the single item that is held out. Instead, our CF-based methods allow neighbors to pick and recommend the best items (papers), regardless of whether the user has rated them or not. Then based on the recommended paper list(s), we then examine the rating(s) the target user provided. If more than one paper is returned, an average of paper ratings is reported. The first reason of using this protocol is for the cross-comparison between CF-based, content-based filtering, and other methods. The second reason is that our recommender system is *not* intended to be used for accurately predicting user ratings.

4.3.2 Experimental Results on Pedagogical CF-based Recommendation

As stated previously, the major goal of pedagogical paper recommendation is not to accurately predict the ratings given by learners, but rather to find out the best paper(s) to serve various pedagogical purposes such as helping a learner in understanding the subject, providing "new" information, etc. Hence, the effectiveness of the recommender system depends on how we are going to present the recommended items to the learners, and in how the recommended items help in achieving appropriate tasks and learning goals.

In this section we use one-dimensional CF to determine the accuracy of recommender algorithms. As discussed in Chapter 2, traditional CF utilizes the overall rating of items by users in finding a user's neighbors, deciding on the recommended papers, and assessing the success of the recommendation. However, overall rating may not be the only metric to assess the success of pedagogical recommendation, because the goal of recommendation may not be to gain high overall ratings but instead to achieve pedagogical goals such as stimulating the learning of knowledge. Specifically, the main goal of this section is to verify whether or not using overall ratings to find neighbors in CF is still appropriate when the purpose of recommendation is to help a learner in understanding the subject (to achieve high Aid_learning ratings) or providing new information (to achieve high Value_added ratings). In other words, we want to see how well the Overall rating predicts the Aid_learning + Value_added ratings, using CF and the experimental data collected in the software engineering course.

4.3.2.1 Procedure

First, we randomly select a set of co-rated papers and calculate the Pearson correlation between a target learner and the other 24 learners to obtain his/her 10

neighbors (the number is not arbitrarily chosen, but was selected after our analysis of one-dimensional CF: we found out that when the number of neighbors equals 10, the performance of recommendation is relatively satisfying). Then, from these neighbors we obtain the weighted estimated rating $r^e_{a,j}$ (cf. Equation (4-1)) for all papers. Here, the calculation of the weighted estimated rating uses the same dimension (Aid_learning, Value_added, Overall) as the assessment metric, but, may not be the same as the dimension used in calculating the Pearson correlation. Finally, the highest-estimated-rated paper is recommended to the target learner, who returns his/her true rating on this paper. Note that this recommended paper may be one of the co-rated papers used in the process of finding neighbors.

In our experimental design, the number of co-rated papers varies from 2 to 10. Intuitively, the larger the number of co-rated papers used, the more accurately neighbors can be found, which increases the accuracy of the recommendation. We randomly select the particular co-rated papers. For statistical purposes, we repeat the random selection of co-rated papers 32 times, resulting in 1 observed rating/trial × 32 trials/learner × 25 learners = 800 data points for each treatment.

4.3.2.2 Results and Analysis

Table 4-14 shows the results of our experiment in this part. Three assessment dimensions are tested, i.e. Overall, Aid_learning, and Value_added, shown as three different blocks in the table. For each assessment dimension, we use three different dimensions for finding neighbors, shown in the second, third, and fourth columns. The second column always shows the results of the dimension being assessed.

Table 4-14 Results of one-dimensional CF recommendation with various assessment dimensions (Top 1)

No. of co-rated papers, $\|K\|$	**Assessed dimension: Overall**		
	Overall	**Value_added** *(p-value)*	**Aid_learning** *(p-value)*
2	2.924	**2.861** *(0.0097)*	**2.851** *(0.0037)*
3	3.040	**2.973** *(0.0072)*	**2.893** *($3\cdot10^{-8}$)*
4	3.111	**2.996** *($1\cdot10^{-5}$)*	**2.971** *($1\cdot10^{-7}$)*
5	3.141	**3.010** *($7\cdot10^{-7}$)*	**2.986** *($8\cdot10^{-9}$)*
6	3.161	**2.986** *($1\cdot10^{-10}$)*	**3.008** *($1\cdot10^{-8}$)*
7	3.168	**2.971** *($2\cdot10^{-13}$)*	**3.024** *($7\cdot10^{-8}$)*
8	3.214	**2.994** *($2\cdot10^{-17}$)*	**3.023** *($2\cdot10^{-12}$)*
9	3.226	**2.960** *($2\cdot10^{-24}$)*	**3.019** *($1\cdot10^{-14}$)*
10	3.236	**2.953** *($1\cdot10^{-27}$)*	**3.019** *($2\cdot10^{-16}$)*
	Assessed dimension: Aid_learning		
	Aid_learning	**Overall** *(p-value)*	**Value_added** *(p-value)*
2	2.971	2.948 *(0.1317)*	**2.911** *(0.0030)*
3	3.026	2.991 *(0.0547)*	**2.951** *(0.0002)*
4	3.081	**3.033** *(0.0156)*	**2.975** *($9\cdot10^{-7}$)*
5	3.113	**3.074** *(0.0438)*	**2.983** *($2\cdot10^{-9}$)*
6	3.126	**3.088** *(0.0418)*	**2.993** *($7\cdot10^{-10}$)*
7	3.154	**3.099** *(0.0071)*	**3.006** *($4\cdot10^{-11}$)*
8	3.174	**3.111** *(0.0024)*	**3.026** *($2\cdot10^{-11}$)*
9	3.168	3.146 *(0.1762)*	**3.026** *($6\cdot10^{-10}$)*
10	3.184	3.165 *(0.2062)*	**3.056** *($1\cdot10^{-8}$)*
	Assessed dimension: Value_added		
	Value_added	**Overall** *(p-value)*	**Aid_learning** *(p-value)*
2	3.300	3.296 *(0.4463)*	**3.241** *(0.0198)*
3	3.294	**3.235** *(0.0203)*	**3.190** *(0.0002)*
4	3.315	**3.229** *(0.0015)*	**3.143** *($2\cdot10^{-9}$)*
5	3.341	**3.226** *($3\cdot10^{-5}$)*	**3.110** *($5\cdot10^{-16}$)*
6	3.345	**3.220** *($5\cdot10^{-6}$)*	**3.114** *($6\cdot10^{-16}$)*
7	3.353	**3.223** *($2\cdot10^{-6}$)*	**3.120** *($4\cdot10^{-16}$)*
8	3.359	**3.236** *($6\cdot10^{-6}$)*	**3.133** *($1\cdot10^{-15}$)*
9	3.363	**3.245** *($9\cdot10^{-6}$)*	**3.134** *($4\cdot10^{-16}$)*
10	3.354	**3.230** *($5\cdot10^{-6}$)*	**3.140** *($2\cdot10^{-14}$)*

A bold font is used for a value when it is significantly different from the corresponding value in the second column (in the same row), i.e. their p-value inside

parentheses is less than 0.05. Here, the p-values are obtained from a one-tail t-test for means assuming equal variance (homoscedastic).

As shown in Table 4-14, all values in the third and fourth column are less than their corresponding values in the second column and their differences are mostly significant. The results show that we cannot use overall ratings in finding a learner's neighbors in CF when the recommendation goal is to find high Value_added or Aid_learning ratings. Instead, we should use the same dimension.

Conclusion 1. *In the population of graduate students, a one-dimensional collaborative-filtering-based paper recommendation system should use the same dimension for finding neighbors and calculating the recommended paper as the goal of recommendation.*

This result is important because the goal of pedagogical paper recommendation, which is determined by the tutor, may not be to attain as high satisfaction (overall rating) as possible. Hence, obtaining a single (overall) rating in a pedagogical recommendation system is not enough to serve various pedagogical purposes. Instead, the system must assess ratings which relate to the goal of the recommendation. This shows the difference between pedagogical recommendation and other recommendation systems (e.g. movie, CD, etc.).

4.3.3 Performance Comparison of Recommendation Techniques

In this section we provide a general comparison of various recommendation techniques. For the sake of simplicity, we will only use average Overall ratings as the metric for comparison. We leave the comparison of other metrics (Value_added, Aid_learning, and Peer_rec) for future study. We will use the data from Chapter 3 to explore the tunings of the various algorithms described in section 4.2, varying parameters

and weights in a principled way to discuss the settings that most accurately predict learners' own preferences.

Some techniques involve several combinations of parameters, which have been tested in our experiments. However, we will only use some parameter settings in our analysis, those that we have left after discarding those with less significant results.

Table 4-15 lists the results of our comparison. The values higher than their corresponding best-case benchmark value are highlighted in bold. Some of them are accompanied by *p*-value (obtained after doing a one-tail t-test assuming homoscedastic). We should be careful in comparing these results with the best-case benchmark, because the best-case benchmark assumes that all papers are completely rated, while some methods in our comparison are not. For instance, the performances of ContentF and Artificial Learners should be compared to the worst-case benchmark rather than to the best-case benchmark.

From this table, we observe that most results are worse than the best-case benchmark (second row), but all results are better than the worst-case benchmark (3rd row). Moreover, those that are better than the best-case benchmark are not statistically significant (p-value ≥ 0.22), but most results better than the worst-case benchmark are significantly better (p-value < 0.05, not shown here).

Table 4-15 Average Overall ratings obtained from various recommendation methods

Methods	**Average Overall ratings when top-n papers are recommended, n = {1, 3, 5}**		
	Top 1	**Top 3**	**Top 5**
Best-case benchmark (*Popularity* only)	3.167	3.055	2.992
Worst-case benchmark (random)	2.708	2.708	2.708
ContentF (*closeness_7*)	3.160	3.013	2.880
1D-CF (number of co-rated papers $\|K\| = 2$)	2.924	2.877	2.847
1D-CF (number of co-rated papers $\|K\| = 4$)	3.111	2.990	2.937
1D-CF (number of co-rated papers $\|K\| = 8$)	**3.214** ($p = .33$)	**3.090** ($p = .29$)	**3.010** ($p = .35$)
3D-CF ($\|K\| = 2$)	2.946	2.884	2.851
3D-CF ($\|K\| = 4$)	3.105	2.984	2.934
3D-CF ($\|K\| = 8$)	**3.210** ($p = .34$)	**3.085** ($p = .31$)	**3.007** ($p = .38$)
UM-CF ($(1_{interest}, 0_{bkgrKnowledge})$)	3.000	2.960	2.968
5D-CF ($(0.8_{overall}, 0.1_{valueadd}, 0.1_{peer\_rec})$, $w_{2D} = 1$, $\|K\| = 2$)	2.965	2.950	2.919
5D-CF ($(1_{overall}, 0_{valueadd}, 0_{peer\_rec})$, $w_{2D} = 1$, $\|K\| = 2$)	2.971	2.945	2.917
5D-CF ($(0.8_{overall}, 0.1_{valueadd}, 0.1_{peer\_rec})$, $w_{2D} = 1$, $\|K\| = 4$)	3.078	3.002	2.965
5D-CF ($(1_{overall}, 0_{valueadd}, 0_{peer\_rec})$, $w_{2D} = 1$, $\|K\| = 4$)	3.071	3.003	2.957
5D-CF ($(0.8_{overall}, 0.1_{valueadd}, 0.1_{peer\_rec})$, $w_{2D} = 1$, $\|K\| = 8$)	3.131	**3.064**	**3.011** ($p = .35$)
5D-CF ($(1_{overall}, 0_{valueadd}, 0_{peer\_rec})$, $w_{2D} = 1$, $\|K\| = 8$)	3.146	**3.068**	**3.015** ($p = .32$)
5D-CF ($(0.8_{overall}, 0.1_{valueadd}, 0.1_{peer\_rec})$, $w_{2D} = 10$, $\|K\| = 2$)	2.998	2.975	2.954
5D-CF ($(1_{overall}, 0_{valueadd}, 0_{peer\_rec})$, $w_{2D} = 10$, $\|K\| = 2$)	3.004	2.979	2.950
5D-CF ($(0.8_{overall}, 0.1_{valueadd}, 0.1_{peer\_rec})$, $w_{2D} = 10$, $\|K\| = 4$)	3.008	2.977	2.959
5D-CF ($(1_{overall}, 0_{valueadd}, 0_{peer\_rec})$, $w_{2D} = 10$, $\|K\| = 4$)	3.013	2.980	2.963
5D-CF ($(0.8_{overall}, 0.1_{valueadd}, 0.1_{peer\_rec})$, $w_{2D} = 10$, $\|K\| = 8$)	3.008	2.977	2.970
5D-CF ($(1_{overall}, 0_{valueadd}, 0_{peer\_rec})$, $w_{2D} = 10$, $\|K\| = 8$)	3.011	2.980	2.969
Pop1D ($\|K\| = 2$)	3.160	3.048	2.983
Pop1D ($\|K\| = 4$)	3.160	**3.070** ($p = .40$)	**2.995**
Pop1D ($\|K\| = 8$)	3.160	**3.086** ($p = .30$)	**2.995**
Pop3D ($\|K\| = 2$)	3.160	3.047	**2.995**
Pop3D ($\|K\| = 4$)	3.160	**3.071** ($p = .40$)	**2.995**
Pop3D ($\|K\| = 8$)	3.160	**3.088** ($p = .29$)	**2.995**
Pop5D ($\|K\| = 2$)	3.081	3.035	2.969
Pop5D ($\|K\| = 4$)	3.129	**3.067**	**2.997**
Pop5D ($\|K\| = 8$)	3.158	**3.099**	**3.019**

		(p = .23)	(p = .29)
PopUM-CF	3.160	**3.080** (p = .39)	2.976
PopCon2D (w_c = 0.15, w_r = 0.1, ($0.9_{interest}$, $0.1_{bkgrKnowledge}$))	**3.280** (p = .22)	3.040	2.930
Artificial Learners	2.880	2.787	2.808
Tutor Recommendation	2.880	2.919	-
Ceiling	3.560	3.347	3.248

4.3.4 Performance Comparison of Content-Based Filtering and Artificial-Learner-Based Recommendation

ContentF, Artificial Learners, and Tutor Recommendation rely on the matching of user models and paper models; hence do not need paper ratings. Thus, all of them can be used to recommend papers which have not been rated before. However, Tutor Recommendation is not an automatic recommendation method, so is used for reference in our comparison.

From Table 4-15, the results of Artificial Learners are better than the worst-case benchmark, but not all of them are significantly different, i.e. the *p*-values of the one-tail t-test for means are 0.079 (Top-1), 0.142 (Top-3), and 0.044 (Top-5) (not shown in the table). Also, the results of ContentF are better than the results of Artificial Learners where their corresponding *p*-values are 0.046 (Top-1), 0.010 (Top-3), and 0.165 (Top-5). But the results of ContentF are significantly better than the worst-case benchmark (*p*-values < 0.002).

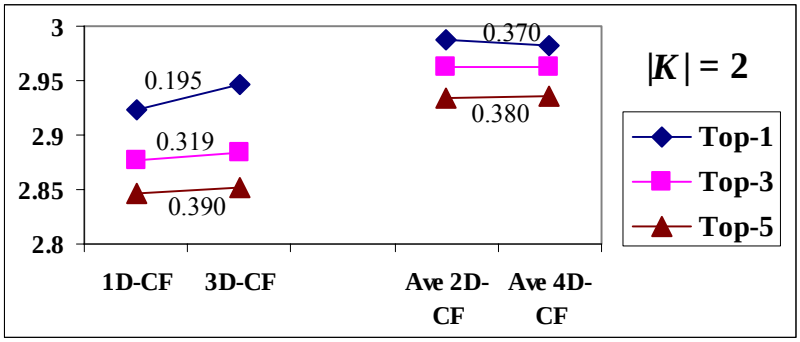
Conclusion 2. *In terms of overall ratings given by graduate students to recommended papers, the content-based filtering method, Closeness_7( ), is significantly better than random recommendation.*

4.3.5 Performance Comparison of Multi-Dimensional Collaborative Filtering

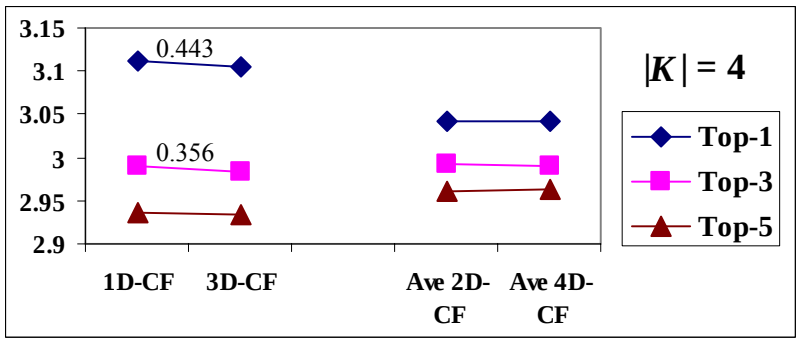
The effect of dimensionality on CF varies according to the number of co-rated papers $|K|$ and the number of recommended papers (i.e. Top-1, -3, or -5). Figure 4-5 plots the average results shown in Table 4-15, where Ave2D-CF is the average value of 5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), w_{2D} = 1) and 5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), w_{2D} = 10). And, Ave4D-CF is the average value of 5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), w_{2D} = 10) and (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), w_{2D} = 1).

We use the term 2D instead of 5D in Ave2D-CF because we actually have ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0) and ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) = (1, 0, 0), which altogether only count for two dimensions (Interest and Overall only). Similarly, we have ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0) and ($w_{overall}$, $w_{valueadd}$, w_{peer_rec}) = (0.8, 0.1, 0.1) for 4D case. Since Ave2D-CF and Ave4D-CF are the combination of rating-based and user-model-based CF, we separate them from pure rating-based CF (i.e. 1D-CF and 3D-CF) in Figure 4-5. Some *p*-values of the one-tail t-test for equal means are placed adjacent to the line connecting the pair of data.

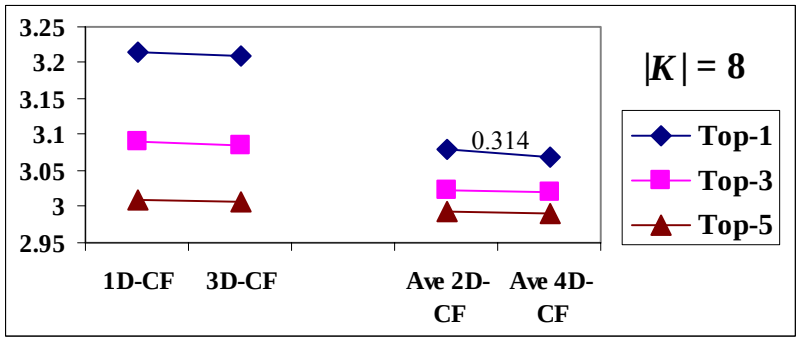
When the number of co-rated papers $|K|$ = 2, the average ratings increase as more dimensions are used in rating-based CF, i.e. from 1D-CF to 3D-CF, as shown in the left part of Figure 4-5(a). But as $|K|$ increases to 4 or 8, the effect disappears (the left part of Figure 4-5(b) and (c)). In Ave2D-CF and Ave4D-CF, the effects of dimensionality are ambiguous and insignificant. In fact, no changes are statistically significant. For example, the smallest *p*-value is 0.195 for the Top-1 data from 1D-CF and 3D-CF when $|K|$ = 2.



(a)



(b)



(c)

Figure 4-5 The average Overall ratings in 3D-CF methods

The ambiguous effects on dimensionality raise skepticism that a higher dimension may significantly affect the recommendation when a small number of co-rated papers are used in a rating-based CF recommendation system. To verify this, we have performed additional analysis on various weights and dimensions. Among them, we use a

combination of various weights on Value_added and Peer_rec as shown in Table 4-16, and various combinations of Overall, Interest, Value_added, Aid_learning, Difficulty, and Job_related ratings as shown in Table 4-17.

We will first look at the effect of various weights of value-added ness and peer recommendation on overall ratings. Table 4-16 shows the average Overall ratings of target users after 1D- and 3D-CF in the experiments. A bold-font is used to highlight an increase after the inclusion, and an underline is used for a significant increase ($p < .05$). The results suggest that *incorporating ratings from value-added ness and peer recommendation can slightly improve the performance of CF-based RSs when the number of co-rated papers is small.* However, only the results at $|K| = 2$ and ($w_{Overall}$, w_{Value_added}, w_{Peer_rec}) = (0.9, 0.05, 0.05) are statistically significant.

Table 4-16 The effects of weights ($w_{Overall}$, w_{Value_added}, w_{Peer_rec}) on average Overall ratings in 3D-CF methods

$\|K\|$		**1D-CF**	**($w_{Overall}$, w_{Value_added}, w_{Peer_rec})**			
			(.9, .05, .05)	**(.8, .1, .1)**	**(.6, .2, .2)**	**(.4, .3, .3)**
2	**Top-1**	**2.924**	<u>2.970</u>	2.946	2.948	2.941
	Top-3	**2.877**	<u>2.918</u>	2.884	2.885	2.880
	Top-5	**2.847**	<u>2.875</u>	2.851	2.852	2.851
4	**Top-1**	**3.111**	**3.106**	**3.105**	**3.085**	**3.100**
	Top-3	**2.990**	**2.985**	**2.984**	2.991	**2.985**
	Top-5	**2.937**	**2.936**	**2.934**	2.940	**2.936**
6	**Top-1**	**3.161**	3.165	3.165	**3.144**	3.163
	Top-3	**3.049**	**3.041**	**3.040**	**3.033**	**3.043**
	Top-5	**2.987**	**2.984**	**2.985**	**2.975**	**2.986**
8	**Top-1**	**3.214**	**3.208**	**3.210**	**3.188**	**3.205**
	Top-3	**3.090**	**3.087**	**3.085**	**3.072**	**3.085**
	Top-5	**3.010**	3.012	**3.007**	**2.995**	**3.010**

The results of adopting other dimensions show that 3D-CF has a better performance (in terms of average Overall rating) also when the number of co-rated papers is small, as shown by the majority of bold-font values when $|K| = 2$ in Table 4-17. However, only three of these results are statistically significant (p-value < 0.05). Most are barely significant or not significant with p-values between 0.106 and 0.481.

Table 4-17 The effects of dimensionality on average Overall ratings in 3D-CF methods

Parameters of 3D-CF methods		**3D-CF**	**1D-CF**	***p*-value**
($0.8_{Overall}$, $0.1_{ValueAdd}$, $0.1_{Interest}$) $\|K\| = 2$	Top-1	**2.969**	2.924	0.041
	Top-3	**2.880**	2.877	0.407
	Top-5	**2.848**	2.847	0.468
($0.8_{Overall}$, $0.1_{AidLearning}$, $0.1_{Interest}$) $\|K\| = 2$	Top-1	**2.953**	2.924	0.133
	Top-3	**2.885**	2.877	0.291
	Top-5	2.847	2.847	-
($0.8_{Overall}$, $0.1_{AidLearning}$, $0.1_{ValueAdd}$) $\|K\| = 2$	Top-1	**2.956**	2.924	0.106
	Top-3	**2.888**	2.877	0.232
	Top-5	**2.855**	2.847	0.282
($0.8_{Overall}$, $0.1_{Difficulty}$, $0.1_{JobRelated}$) $\|K\| = 2$	Top-1	**2.934**	2.924	0.348
	Top-3	**2.906**	2.877	0.031
	Top-5	**2.872**	2.847	0.024
($0.8_{Overall}$, $0.1_{ValueAdd}$, $0.1_{Interest}$) $\|K\| = 4$	Top-1	3.106	3.111	-
	Top-3	2.982	2.990	-
	Top-5	2.931	2.937	-
($0.8_{Overall}$, $0.1_{AidLearning}$, $0.1_{Interest}$) $\|K\| = 4$	Top-1	3.104	3.111	-
	Top-3	**2.998**	2.990	0.312
	Top-5	**2.940**	2.937	0.418
($0.8_{Overall}$, $0.1_{AidLearning}$, $0.1_{ValueAdd}$) $\|K\| = 4$	Top-1	**3.113**	3.111	0.481
	Top-3	**2.998**	2.990	0.315
	Top-5	2.936	2.937	-
($0.8_{Overall}$, $0.1_{Difficulty}$, $0.1_{JobRelated}$) $\|K\| = 4$	Top-1	3.081	3.111	-
	Top-3	2.985	2.990	-
	Top-5	2.937	2.937	-

Conclusion 3. *In terms of overall ratings given by graduate students to recommended papers, the multi-dimensional collaborative filtering method may be better than a uni-*

dimensional collaborative filtering method when the number of co-rated papers is small, e.g. two papers only.

The cause of this phenomenon is unclear. We conjecture that when the number of co-rated papers is small, one-dimensional ratings are not enough to find close neighbors. If we add other dimensional ratings, these ratings can provide more information in finding close neighbors, resulting in better results. However, when the number of co-rated papers is larger, e.g. $|K| = 4$, this information is less useful or even becomes noise which may reduce the effectiveness of finding close neighbors. A justification of this conjecture is left to the future work.

4.4 Chapter Discussion

In this section we will discuss some issues arising from our experiment. Explanations about the number of co-rated papers and neighbors used in CF are given. Threats to the experimental results and our methodology are also discussed.

4.4.1 The Effect of Paper Popularity in Recommendation

Incorporating paper popularity can increase the average Overall ratings in both user-model-based and rating-based CF when the number of co-rated papers is small. Table 4-18 shows partial experimental results related to this issue. Each cell consists of a pair of average Overall ratings obtained from methods with/without incorporating paper popularity. The higher value is highlighted in bold-font, and an underline is used when their difference is significant (p-value < 0.05).

Incorporating popularity is only inferior when the number of co-rated papers is large, e.g. $|K| = 8$ in Table 4-18. Intuitively, putting excessive weight on paper popularity in CF will result in the best-case benchmark method. Hence, paper popularity should be integrated into CF when the number of co-rated papers is small.

Table 4-18 Effect of paper popularity in various recommendation methods

Methods	**Average Overall ratings when top-n papers are recommended, n = {1, 3, 5}**		
	Top 1	**Top 3**	**Top 5**
Best-case benchmark	3.167	3.055	2.992
Pop1D / 1D (|*K*| = 2)	**3.160** / 2.924	**3.048** / 2.877	**2.983** / 2.847
Pop1D / 1D (|*K*| = 4)	**3.160** / 3.111	**3.070** / 2.990	**2.995** / 2.937
Pop1D / 1D (|*K*| = 8)	3.160 / **3.214**	3.086 / **3.090**	2.995 / **3.010**
Pop3D / 3D (|*K*| = 2)	**3.160** / 2.946	**3.047** / 2.884	**2.995** / 2.851
Pop3D / 3D (|*K*| = 4)	**3.160** / 3.105	**3.071** / 2.984	**2.995** / 2.934
Pop3D / 3D (|*K*| = 8)	3.160 / **3.210**	**3.088** / 3.085	2.995 / **3.007**
PopUM-CF / UM-CF	**3.160** / 3.000	**3.080** / 2.960	**2.980** / 2.968

Conclusion 4. *Paper popularity should be integrated into rating-based collaborative filtering methods when the number of co-rated papers is small, such as two or four papers only.*

4.4.2 Number of Co-rated Papers in CF

In our experiment, a co-rated paper in rating-based CF methods may be recommended back to learners. Hence, using a large number of co-rated papers increases the chance of recommending a highly rated paper included in the co-rated papers. The effects are twofold. First, this approach may reduce the chance of recommending "new" papers, i.e. those have not been read by learners; hence, the experimental results do not necessarily capture the effectiveness of recommendation methods in practice. Second, this approach may make us believe that the recommendation methods perform well when we increase the number of co-rated papers. In other words, a high average rating given to recommended papers does not mean a good recommendation technique when we use a large number of co-rated papers, since when a relatively larger number of co-rated papers

are considered, the chance that these papers will be recommended is higher due to the fact that the numbers of learners and papers are limited. As such, the recommendation will work strongly as these co-rated papers can push up the average ratings.

4.4.3 Number of Neighbors in CF

In some experiments involving collaborative filtering methods, we choose the number of neighbors equal to 10. This number is not arbitrarily chosen, but was selected after our analysis of one-dimensional CF. Figure 4-6 shows some of the average Overall ratings in 1D-CF against the number of neighbors, N, for the number of co-rated papers $|K|$ equal to 2, 4, and 8. It also shows the average of average Overall ratings for all combinations of |K| = {2, 3, …, 15} and {Top-1, Top-3, Top-5}, labeled as "Total Average".

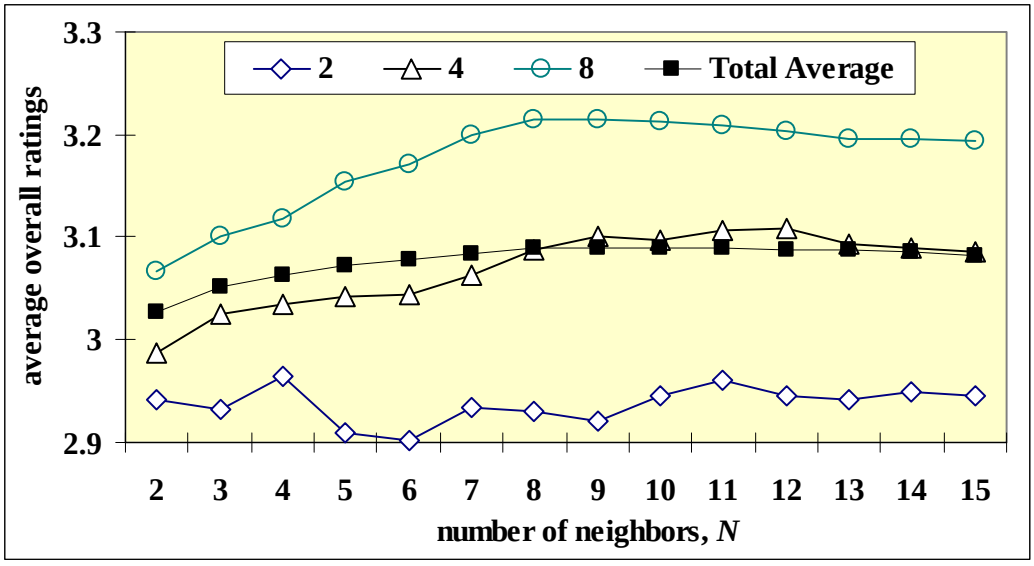


Figure 4-6 The average Overall ratings against the number of neighbors used in 1D-CF

As shown in Figure 4-6, the number of neighbors affects the average ratings when $|K|$ = 4 and 8, where the peak is around N = 12 and 8, respectively. When the number of

co-rated papers is two, the performance is very unstable, with a few ups and downs (the lowest data series on Figure 4-6). After an initial steady rise, the average ratings slightly decrease when we use a higher number of neighbors. This phenomenon is not difficult to explain. The quality of pure CF-based methods relies on the quality of the neighbors and the denseness of the rating database. The latter condition, though, is not easy to reach in our domain. Hence, the success of recommendations tends to rely on the quality of neighbors. In other words, when the quality neighbors for the target user are identified, the recommendations made will also be better. Recommendation systems must rely on the ratings candidate users give to find the closest neighbors: the closer the candidate user and the target user have agreed on a paper, the higher chance the candidate user is the neighbor. Therefore, the most important is not the *number* of co-rated papers per se, but the similarity between the ratings. In our data, the "total average" reaches its maximum value when $N = 10$, which is used as the value in our rating-based CF methods unless otherwise specified.

4.4.4 Cold-Start Recommendation

Three types of cold-start problems are considered in our experiment: the new-learner-old-paper problem, the old-learner-new-paper problem, and the new-learner-new-paper problem.

The new-learner-new-paper problem refers to situation when we have a set of unrated papers and a target learner who has never rated any paper. In this case, if we do not have a target learner model, which is unlikely in a pedagogical environment, so only a random assignment method (which would yield accuracy equivalent to the worst-case benchmark in Table 4-15) can be used. If we have a target learner model, then we can use content-based filtering (ContentF), Artificial Learners, and/or Tutor Recommendation, in

addition to the random assignment method. Both Artificial Learners and Tutor Recommendation are inferior compared to ContentF, but they are significantly better than the random method. In previous work, we have also shown that Artificial Learners is better than a random method in paper recommendation for undergraduate students [Tang and McCalla 2005a].

The new-learner-old-paper problem refers to the situation where we have some rated papers in the system (previously rated by other learners), and we want to recommend paper(s) to a new target learner who has never rated any paper before. In our discussion here, we assume that the learner model is available. In this case, all methods for the new-learner-new-paper problem can be used. In addition, user-model-based CF (UM-CF), the best-case benchmark, and their combinations, i.e. PopUM-CF and PopCon2D, can be used.

The old-learner-new-paper problem refers to the situation where we have a set of learners who have rated some papers in the system but some new papers are added to the system. To handle this situation, we can use methods used for the new-learner-new-paper problem. Other paper-paper corresponding methods, such as co-citation, are left for future study, since it is not clear whether or not these methods apply well in a pedagogical environment.

The last case (the only case which is not considered to be a cold-start problem) is the old-learner-old-paper situation, where there is a set of rated papers and a learner who has rated some papers. In this case rating-based CF and its combination with others can also be used.

Given an initial state of a recommendation system, we may start by Tutor Recommendation, Artificial Learners, and/or ContentF. As more papers are recommended, read, and rated by learners, we may use other more effective methods. The integration of these methods and when to use which of these approaches will be described in the next chapter.

CHAPTER 5
THE DESIGN AND TESTING OF THE PAPER RECOMMENDER

In this chapter we will show a prototypical recommender system. The prototype contains all the proposed algorithms and is intended to show the recommendation results only. In the fully implemented system, this prototype may not be shown to learner but the tutors or system administrators only.

In the fully implemented system and implementation, for each learner we may recommend 5 to 25 papers according to the syllabus, which may later be rated by learners. More ratings may be filled into the system when we combine ratings from several different related courses. For example a learner may take a “Software Engineering” course in the first semester and rate 20 papers and then an “Advanced Software Engineering” course in the second semester and rate 10 papers, etc. But this may raise various issues, such as the consistency of learner models in those courses (e.g. the improvement of his/her background knowledge or the change of his/her interest), whose solution is beyond the scope of this thesis and left for future work. Hence, in this thesis we assume that all papers and learners belong to the same course. In the next subsection we will show the overall architecture of the prototype, followed by the evaluation of the prototype using synthetic learners.

5.1 The Overall Architecture Description of the System Prototype

As pointed out in previous chapters, the proposed paper recommendation is achieved through a careful assessment and comparison of both learner and paper characteristics. In other

words, each individual learner model will first be analyzed, in terms of not only learner interest, but also pedagogical features, that is, the learner's background knowledge in specific topics. Paper models will also be analyzed based on the topic, degree of peer recommendation, etc. The recommendation is carried out by matching learner interest with the paper topics where the goal is also that the technical level of the paper should not impede the learner in understanding it. Therefore, the suitability of a paper toward a learner is calculated as the appropriateness of paper to help the learner in general.

When the tutor first initiates the system, s/he will be requested to tell the system briefly about learner model, including learning goals, interest, knowledge background, etc, similar to the questionnaire in Appendix B. The tutor will then be looking at the user interface that lists an initial set of recommended articles that matches the learner's profile.

5.1.1 The Recommendation Mechanism

Table 4-6 summarizes the recommendation approaches that have been implemented in the prototype. In Figure 5-1, we take a closer look at the entities in the system.

Figure 5-2(a) below shows the starting screenshot of the prototype. Broadly, there are four main areas in the prototype:

1. Area 1 is the user model sheet which lists out the user models, with newly added models at the top;
2. Area 2 is the paper model sheet which stores the information on the papers and is mainly reserved for tutors. Currently, the maximum number of papers is 100.
3. Area 3 is the rating sheet, which is used to enter the numerical ratings on the paper, in terms of seven factors, excluding the textual comments in the paper feedback form.

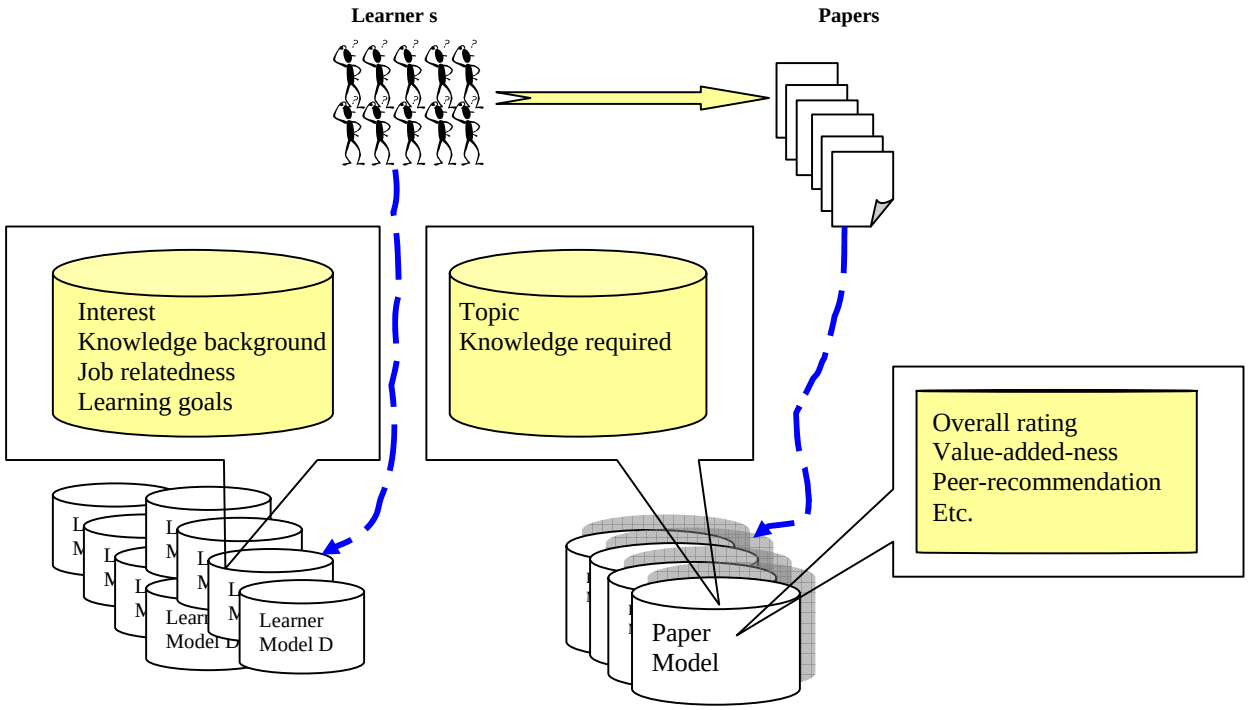
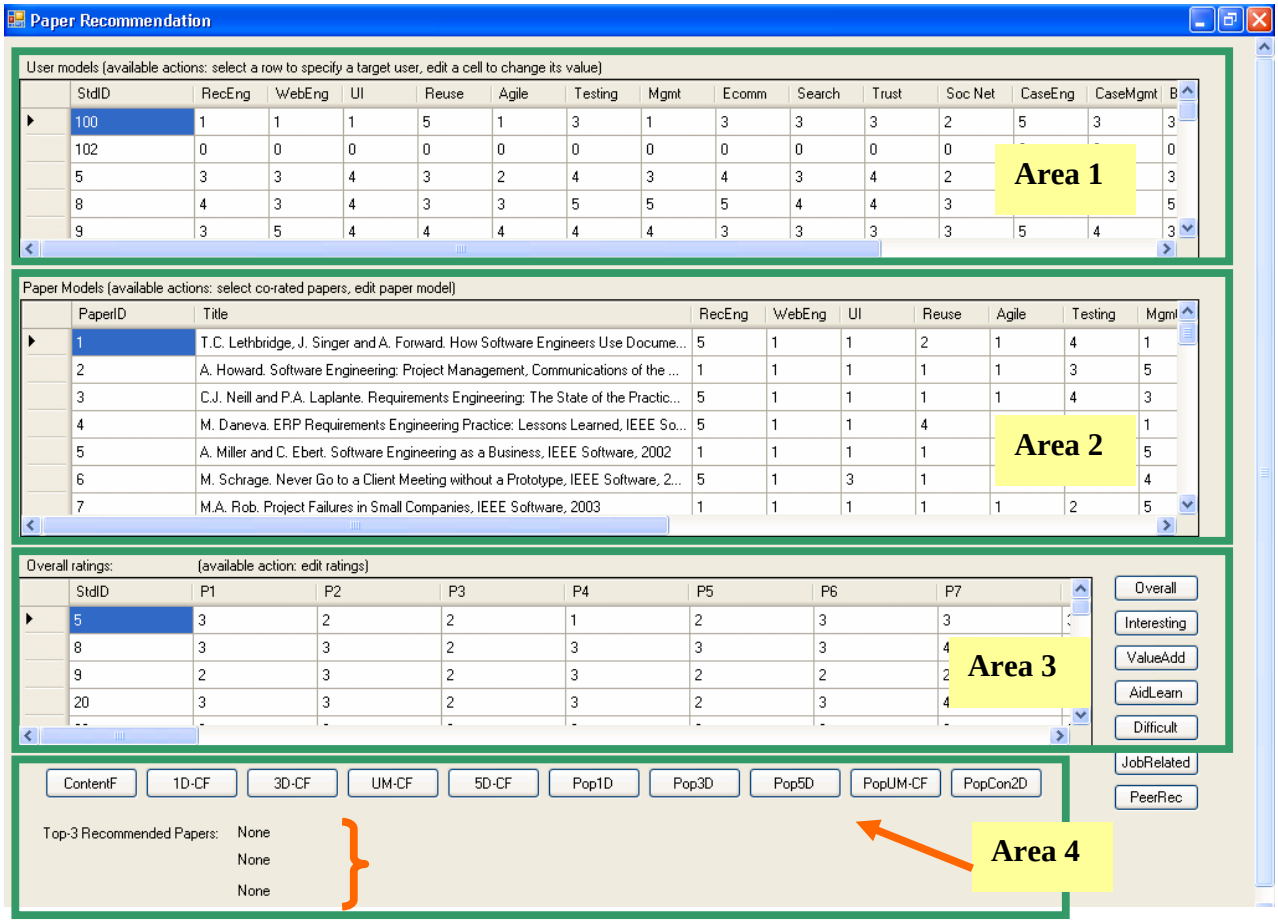
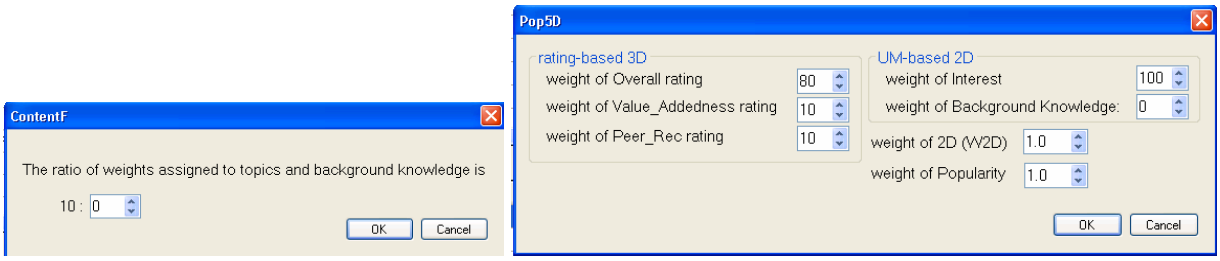


Figure 5-1 A closer look at the recommender system

4. Area 4 is the recommendation panel, where tutors can choose among 10 recommendation methods to make recommendations. The recommendation result includes the top 3 papers listed in order. When a recommendation method is chosen, a dialog box consisting of the relevant parameters will pop-up (e.g. Figure 5-2(b) and (c)).



(a)



(b) (c)

Figure 5-2 The screenshot of the system when it gets started

5.1.2 A Walk through the Prototype

In order to see how the system generally flows, let us look at a typical scenario, where a new student joins an already functioning class. Here, we assume that learners are required to read 10 papers only among the pool of 21 papers.

Bob, a newly enrolled part time student, is now a Project Manager in a Hong Kong-based software company. He has rich hands-on knowledge and experience in Programming, but his Mathematical Knowledge is poor. His interest in registering in the course is to gain some knowledge about User Interface Design, as shown in the user model sheet in Figures 5-3 and 5-4.

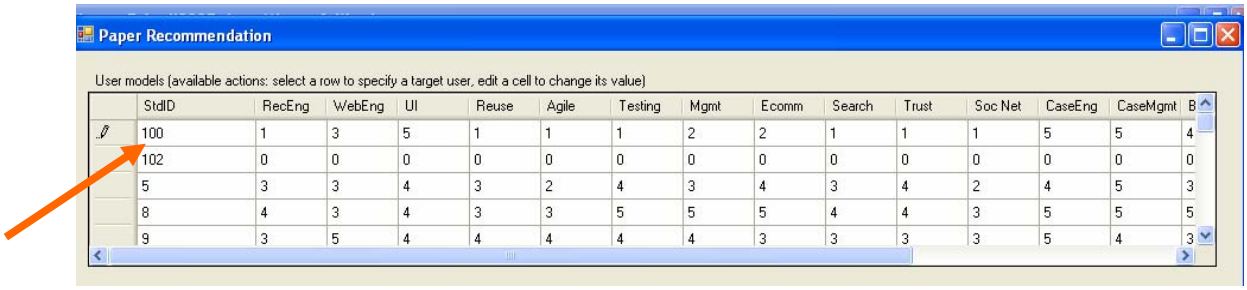


Figure 5-3 Bob enters his learner model (1) (Identified by StdID 100)

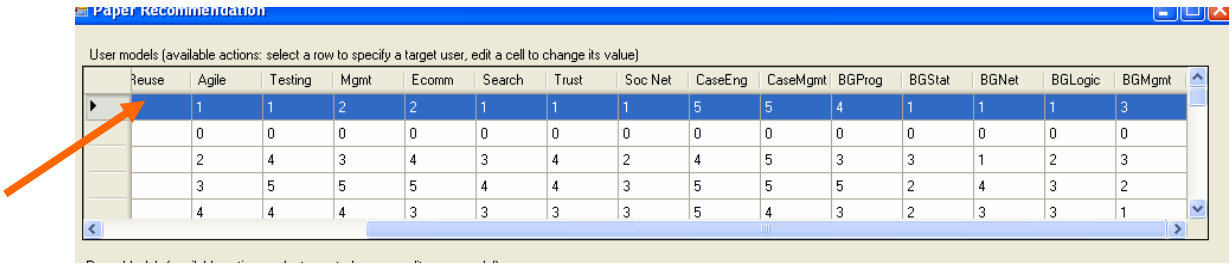


Figure 5-4 Bob enters his learner model (2) (Identified by StdID 100)

As Bob is new to the system and has not rated any papers before, the system can only use the content-based technique (ContentF) to pick up some papers for him. Suppose the tutor uses the default parameters as shown in Figure 5-2(b); then, Figure 5-5 shows the three recommended papers.

The first paper recommended, 'The Windows® UI: A Case Study in Usability Engineering', (Paper 16), uses the Windows User Interface design, as an example to conceptualize the design rules for a user interface with high usability. Since most learners work with Microsoft software almost every day in their work and life, a paper on how the user interface engineers in this software giant have integrated the '*human factors*' into the design would certainly be considered positively by learners such as Bob.

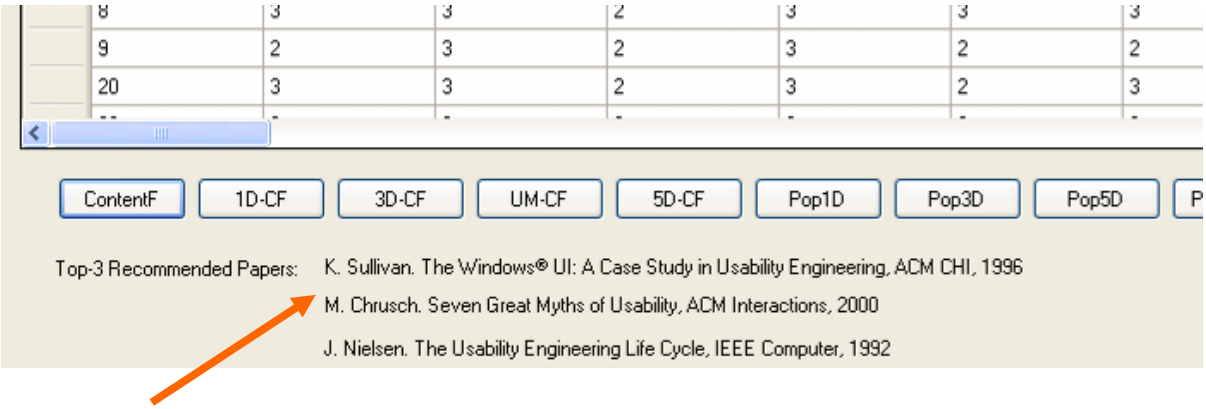


Figure 5-5 Recommended papers for Bob

In addition, the paper's discussions are closely coupled with corresponding screenshots, and the paper is not technically written, and therefore, it will not require readers to have substantial knowledge of usability engineering, mathematics, etc that fits Bob's background knowledge. Further, Bob works as a project manager himself, and shows interest in articles related to user interface design, especially those with real cases that fit the topics of this paper. Furthermore, the paper has been receiving high ratings from other learners in terms of almost all categories; hence, it is expected to receive a good rating from Bob too.

The second paper, 'Seven Great Myths of Usability' (Paper 13), focuses on some conceptual discussions on usability. Although it is not accompanied by case studies, it is

worthwhile reading. Like the first paper, the language used in this paper is plain and easy to understand. The last recommended paper, 'The Usability Engineering Life Cycle'(Paper 17), is a classical one on usability engineering, authored by the guru Jakob Nielsen. Although the paper is long, and not so easy to understand, due to its technical language, it is well received by almost all existing users.

The previous scenario shows how the prototype recommends reasonable papers to Bob, a new learner, using content-based filtering. However, in some situations the tutors may use other recommendation methods, which will be explained in the next section.

5.2 The Adoption of Recommendation Mechanism

In this section, we will show how the system makes decisions on choosing appropriate recommendation mechanisms (and why), based on a series of learning scenarios that cover the various pedagogical possibilities. Assume that in a course of 14 weeks, a total of 10 papers will be recommended to a target learner by the end of the course. Then, we define a new learner as one who has not read any paper, and a learner half-way in the course as somebody who may have read 2 or 4 papers, and a learner near the end of the course as one who may have read 8 papers. In these three typical situations, our understanding of a learner increases, which can be used to refine the recommendations.

Our discussion here is further classified into two main cases: a case with complete ratings (with 21 papers) and a case with partial ratings (11 papers)[1]. In the former, it is assumed that the system has been used by learners who have read 21 papers (among them) and consists of enough data to start-up the recommendation process; while in the latter, the insufficient amount of data

[1] Even if we assume that a target learner will only read up to 10 papers, the system may have learners who have read more than 10 papers (e.g. 21 or 11 papers). These numbers are picked to illustrate some typical scenarios to test the recommender system only.

(papers) poses challenges to the recommendation mechanism, in terms of the number of co-rated papers and the number of ratings in total. However, both situations are equally important in this domain. In order to select appropriate recommendation methods, we will verify them using our experimental data. However, the verification results reported her are domain dependant; thus should not be applied directly to recommendation in other courses.

5.2.1 Case One: Evaluation with Enough Data to Start-up the Recommendations

In this case, we consider the following typical learning scenarios:

1. A new learner starting a course
2. A learner half-way through the course
3. A learner near the end of the course

Under these scenarios where the course has been offered in previous semesters, the system is already full of user-ratings, papers and user models obtained from previous learners taking the course. In other words, there is enough user-rating data to make recommendations (in our case, with 21 papers and 25 learners, and more than 400 ratings). Meanwhile, the sufficiency of data poses some challenges as well to determine, for example, the number of co-rated papers, the number of neighbors, etc.

As for the new learner joining the course (scenario 1, above) when the system knows little about him/her, either the content-aspect or the match between user model and paper model has to be considered; hence, ContentF, UM-CF, PopUM-CF, PopCon2D, and artificial learners can be considered. Table 5-1 compares these approaches in terms of the average overall ratings when the top 1 and the top 3 papers are returned based on our experimental data. PopCon2D performs better for recommending top 1 and PopUM-CF is better in recommending top 3 papers.

Note that the value that higher than its best-case benchmark is denoted by bold font with a corresponding *p*-value to represent its statistical difference.

Table 5-1 A performance comparison of methods for a new learner (with enough data)

Methods	**Average Overall ratings**	
	Top 1	**Top 3**
Best-case benchmark (*Popularity* only)	3.167	3.055
ContentF (*closeness_7*)	3.160	3.013
UM-CF (($1_{interest}$, $0_{bkgrKnowledge}$))	3.000	2.960
PopUM-CF	3.160	**3.080** (p = .39)
PopCon2D (w_c= 0.15, w_r= 0.1, ($0.9_{interest}$, $0.1_{bkgrKnowledge}$))	**3.280** (p = .22)	3.040
Artificial Learners	2.880	2.787

Observing from Table 5-1, PopCon2D's performance is satisfying when w_c = 0.15, w_r = 0.1, and the weight of interest and background knowledge in content-based filtering is ($0.9_{interest}$, $0.1_{bkgrKnowledge}$). The result indicates that when the system knows little about a new learner, it can adopt PopCon2D and PopUM-CF to make recommendations. However, care should be taken when adopting PopCon2D or PopUM-CF, since in both a paper's popularity $\tilde{r}_j$ is incorporated in the computation; therefore, when the number of papers is too few (in our case, less than 10) ContentF should be used.

Recall that ContentF has the benefits of not relying on prior ratings to make recommendations, therefore, it is most suitable when there are cold-start problems, or when the system does not have enough data to start up. It works by comparing the user model and the content of each paper, where the matching is performed on multiple features including paper topic and content appropriateness. The 7 variations of ContentF differ from each other in the treatment of paper topic and content appropriateness. Figure 5-6 compares the performance of these ContentF approaches using our experimental data. In each variation, we calculate the

average ratings of four categories, viz. the results of recommending top 1 and top 3 papers while considering the background knowledge (i.e. Top 1(1) and Top 3(1)) or not considering it (i.e. Top 1(0) and Top 3(0)). The results revealed that closeness_7 performs the best in each category; thus, it is recommended when a new learner engages in a functioning course where the ratings are sparse.

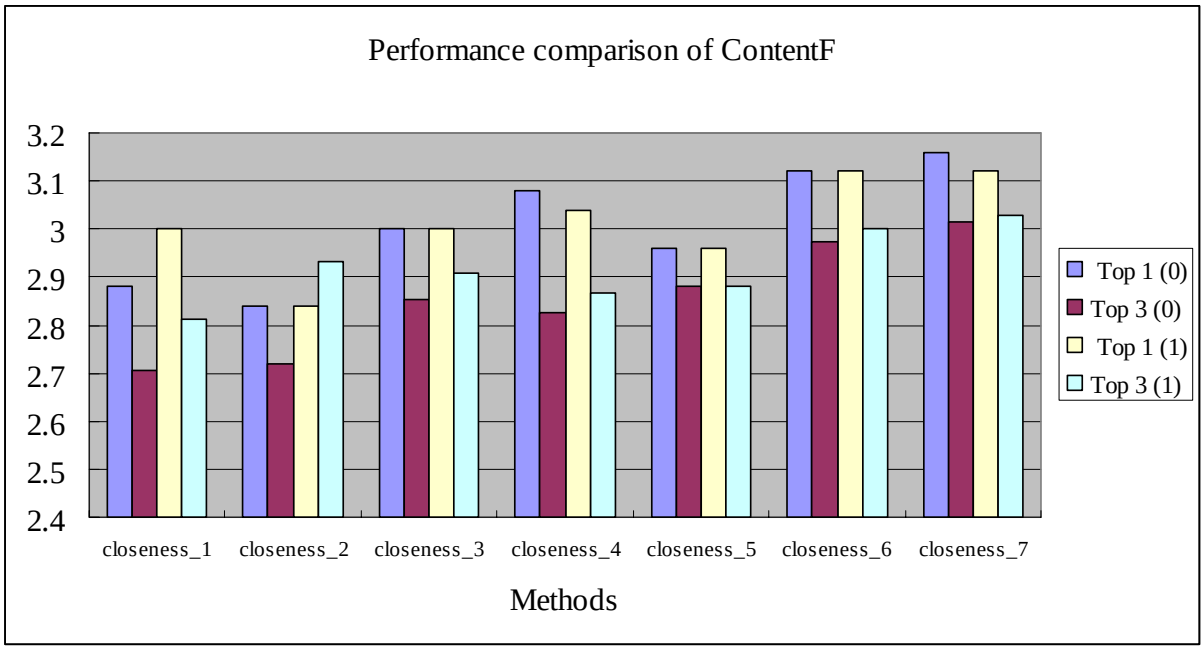


Figure 5-6 Performance comparison of ContentF (with enough data)

As for the learner half-way in the course (scenario 2) when the system knows more about him/her (he/she has rated some papers), either CF or the match between user model and paper model can be considered. Table 5-2 compares these approaches in terms of the average overall ratings when top 1 and top 3 papers are returned. Again, PopCon2D performs better for recommending top 1 and PopUM-CF is better in recommending top 3 papers.

Table 5-2 A performance comparison of methods for a learner half way in the course (with enough data)

Methods	**Average Overall ratings**	
	Top 1	**Top 3**
Best-case benchmark (*Popularity* only)	3.167	3.055
1D-CF (number of co-rated papers $\|K\| = 2$)	2.924	2.877
1D-CF (number of co-rated papers $\|K\| = 4$)	3.111	2.990
3D-CF ($\|K\| = 2$)	2.946	2.884
3D-CF ($\|K\| = 4$)	3.105	2.984
UM-CF (($1_{interest}$, $0_{bkgrKnowledge}$))	3.000	2.960
5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), $w_{2D} = 1$, $\|K\| = 2$)	2.965	2.950
5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), $w_{2D} = 1$, $\|K\| = 2$)	2.971	2.945
5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), $w_{2D} = 1$, $\|K\| = 4$)	3.078	3.002
5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), $w_{2D} = 1$, $\|K\| = 4$)	3.071	3.003
5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), $w_{2D} = 10$, $\|K\| = 2$)	2.998	2.975
5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), $w_{2D} = 10$, $\|K\| = 2$)	3.004	2.979
5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), $w_{2D} = 10$, $\|K\| = 4$)	3.008	2.977
5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), $w_{2D} = 10$, $\|K\| = 4$)	3.013	2.980
Pop1D ($\|K\| = 2$)	3.160	3.048
Pop1D ($\|K\| = 4$)	3.160	**3.070** ($p = .40$)
Pop3D ($\|K\| = 2$)	3.160	3.047
Pop3D ($\|K\| = 4$)	3.160	**3.071** ($p = .40$)
Pop5D ($\|K\| = 2$)	3.081	3.035
Pop5D ($\|K\| = 4$)	3.129	**3.067**
PopUM-CF	3.160	**3.080** ($p = .39$)
PopCon2D ($w_c = 0.15$, $w_r = 0.1$, ($0.9_{interest}$, $0.1_{bkgrKnowledge}$))	**3.280** ($p = .22$)	3.040

When it comes for the learner approaching the end of the course (scenario 3), both CF and the match between user model and paper model can be considered. Table 5-3 compares these approaches in terms of the average overall ratings when top 1 and top 3 papers are returned. In this scenario, PopCon2D performs better for recommending top 1 and Pop5D performs well in recommending top 3 papers.

Table 5-3 A performance comparison of methods for a learner near the end of the course (with enough data)

Methods	**Average Overall ratings**	
	Top 1	**Top 3**
Best-case benchmark (*Popularity* only)	3.167	3.055
1D-CF (number of co-rated papers $\|K\| = 8$)	**3.214** ($p = .33$)	**3.090** ($p = .29$)
3D-CF ($\|K\| = 8$)	**3.210** ($p = .34$)	**3.085** ($p = .31$)
5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), $w_{2D} = 1$, $\|K\| = 8$)	3.131	**3.064**
5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), $w_{2D} = 1$, $\|K\| = 8$)	3.146	**3.068**
5D-CF (($0.8_{overall}$, $0.1_{valueadd}$, 0.1_{peer_rec}), $w_{2D} = 10$, $\|K\| = 8$)	3.008	2.977
5D-CF (($1_{overall}$, $0_{valueadd}$, 0_{peer_rec}), $w_{2D} = 10$, $\|K\| = 8$)	3.011	2.980
Pop1D ($\|K\| = 8$)	3.160	**3.086** ($p = .30$)
Pop3D ($\|K\| = 8$)	3.160	**3.088** ($p = .29$)
Pop5D ($\|K\| = 8$)	3.158	**3.099** ($p = .23$)
PopUM-CF	3.160	**3.080** ($p = .39$)
PopCon2D ($w_c = 0.15$, $w_r = 0.1$, ($0.9_{interest}$, $0.1_{bkgrKnowledge}$))	**3.280** ($p = .22$)	3.040

Note that again, in Tables 5-1, 5-2 and 5-3, bold values mean they are higher than the referential best-case benchmark (with associated p-value); the same convention is applied throughout this thesis.

In general, PopCon2D performs well in all cases. This result is interesting, yet not surprising if we look closely at the method. In particular, it is a combination of ContentF, non-personalized and UM-CF (by considering both learners' interest and background knowledge) to make recommendation, therefore, it has the potential to complement one another's performance. It is notable though, the PopCon2D that works best in these two scenarios has been tuned through experiments and the one reported in both cases occurs when $w_c = 0.15$ (w_c is a weight on the closeness value calculated from content-based filtering; it is related to learner knowledge background and is used to determine the importance of considering learner knowledge background when picking a paper that the learner will find easy to understand) with $w_r = 0.1$ (the weight on $\tilde{r}_j$) and the composition of ($w_{interest}$, $w_{bkgrKnowledge}$) = (.9, .1). In other words, although

the PopCon2D with this set of the variables still treats learner interest as much more important than knowledge background, it will also consider the paper's popularity among the learners. This method in fact characterizes the features of the recommendation and highlights the importance of appropriately incorporating them into the recommendation process.

5.2.2 Case Two: Evaluation with Insufficient Papers

When the system is being used initially, the number of available papers and their ratings may not be as many as in the ideal situation where the system can find a set of suitable papers for each learner. This situation also corresponds to the scalability issue with respect to the accuracy of each recommendation method discussed in Chapter 4. To simulate the case of having insufficient papers, we assume only 11 papers are available in the system. In this case, we also consider the following typical learning scenarios:

1. A new learner starting the course
2. A learner half-way through the course

We do not consider learners near the end of the course because by the end of the course, the number of available paper is only 11 while the total number of recommended paper is 10, or only one paper will not be recommended to each learner. In such a case, the evaluation of recommendation results is less convincing. Our empirical study assumes the availability of 11 papers, which are taken without replacement for 32 times from the pool of 21 papers. Table 5-4 compares the recommendation performance by showing the average ratings of the recommended items between methods applied to 11 papers and those applied to 21 papers.

Table 5-4 The comparison of average overall ratings obtained from various recommendation methods when the database contains 11 and 21 papers

Methods	Average Overall ratings			
	11 papers		21 papers	
	Top 1	Top 3	Top 1	Top 3
Best-case benchmark (*Popularity*)	3.076	2.962	3.167	3.055
ContentF (*closeness_7*)	3.030 (–14%)	2.859 (–40%)	3.160 (–2%)	3.013 (–14%)
1D-CF (|*K*| = 2)	2.901 (–53%)	2.846 (–45%)	2.924 (–62%)	2.877 (–61%)
1D-CF (|*K*| = 4)	**3.144** (+21%) ($p < .01$)	**2.974** (+5%) ($p = .21$)	3.111 (–14%)	2.990 (–22%)
1D-CF (|*K*| = 8)			**3.214** (+12%) ($p = .33$)	**3.090** (+12%) ($p = .29$)
3D-CF (|*K*| = 2)	2.940 (–41%)	2.831 (–51%)	2.946 (–56%)	2.884 (–59%)
3D-CF (|*K*| = 4)	**3.123** (+14%) ($p = .03$)	**2.968** (+2%) ($p = .34$)	3.105 (–16%)	2.984 (–24%)
3D-CF (|*K*| = 8)			**3.210** (+11%) ($p = .34$)	**3.085** (+10%) ($p = .31$)
UM-CF ($(1_{interest}, 0_{bkgrKnowledge})$)	2.945 (–40%)	2.900 (–24%)	3.000 (–42%)	2.960 (–33%)
5D-CF ($(.8_o,.1_v,.1_p)$, w_{2D} =1, |*K*|=2)	2.958 (–36%)	2.885 (–30%)	2.965 (–51%)	2.950 (–36%)
5D-CF ($(1_o, 0_v, 0_p)$, w_{2D} =1, |*K*|=2)	2.974 (–31%)	2.883 (–31%)	2.971 (–50%)	2.945 (–38%)
5D-CF ($(.8_o, .1_v, .1_p)$, w_{2D} =1, |*K*|=4)	**3.099** (+7%) ($p = .18$)	**2.970** (+3%) ($p = .28$)	3.078 (–23%)	3.002 (–18%)
5D-CF ($(1_o, 0_v, 0_p)$, w_{2D} =1, |*K*|=4)	**3.099** (+7%) ($p = .18$)	**2.971** (+3%) ($p = .28$)	3.071 (–24%)	3.003 (–18%)
5D-CF ($(.8_o, .1_v, .1_p)$, w_{2D} =1, |*K*|=8)			3.131 (–9%)	**3.064** (+3%)
5D-CF ($(1_o, 0_v, 0_p)$, w_{2D} =1, |*K*|=8)			3.146 (–5%)	**3.068** (+4%)
Pop1D (|*K*| = 2)	3.070 (–2%)	2.950 (–5%)	3.160 (–2%)	3.048 (–2%)
Pop1D (|*K*| = 4)	3.071 (–2%)	**2.986** (+9%) ($p = .06$)	3.160 (–2%)	**3.070** (+5%) ($p = .40$)
Pop1D (|*K*| = 8)			3.160 (–2%)	**3.086** (+11%) ($p = .30$)
Pop3D (|*K*| = 2)	3.069 (–2%)	2.956 (–2%)	3.160 (–2%)	3.047 (–3%)
Pop3D (|*K*| = 4)	3.071 (–2%)	**2.983** (+8%) ($p = .08$)	3.160 (–2%)	**3.071** (+5%) ($p = .40$)
Pop3D (|*K*| = 8)			3.160 (–2%)	**3.088** (+11%) ($p = .29$)
Pop5D (|*K*| = 2)	3.021 (–17%)	2.939 (–9%)	3.081 (–22%)	3.035 (–7%)
Pop5D (|*K*| = 4)	**3.086** (+3%)	**2.999** (+14%) ($p < .01$)	3.129 (–10%)	**3.067** (+4%)
Pop5D (|*K*| = 8)			3.158 (–2%)	**3.099** (+15%) ($p = .23$)
PopUM-CF	3.065 (–3%)	**2.972** (+4%) ($p = .16$)	3.160 (–2%)	**3.080** (+9%) ($p = .39$)
PopCon2D	**3.110** (+10%) ($p = .22$)	2.928 (–13%)	**3.280** (+29%) ($p = .22$)	3.040 (–5%)

Since the sampling of only 11 papers affects the size of overall ratings, the comparison cannot be made using absolute values. Rather, we provide the percentage gain/loss made by each method relative to the best-case benchmark and the ceiling inside the parentheses in Table 5-4, i.e. 0% gain means the average rating of the method is the best-case benchmark and +100% gain means the average rating of the method attains the ceiling. A loss (negative value) means the average rating of the method is lower than the best-case benchmark. For statistical analysis, the *p*-values of some gains are provided.

Note that the comparison of CF-based results involving co-rated papers cannot be applied within the same row. From the table, the results of $|K| = 2$ or $|K| = 4$ in the 11-paper cases are more comparable to those of $|K| = 4$ or $|K| = 8$ in the 21-paper cases, respectively, rather than with the results in the same row.

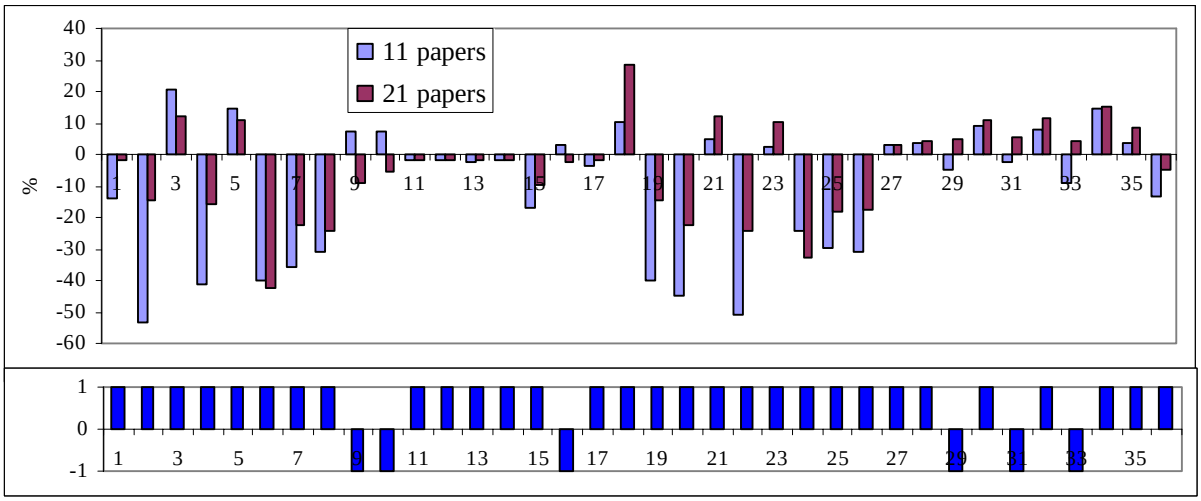


Figure 5-7 The pair-wise of percentage gain/lose with 11 and 21 papers respectively

The upper diagram of Figure 5-7 shows the pair-wise percentage gain/loss in the 36 11-paper cases with their counterpart in the 21-paper cases, i.e. from ContentF to PopCon2D for both Top-1 (18 pairs) and Top-3 (18 pairs) recommendations (see Table 5-4). When the pair is in the same direction, i.e. both positive/negative, we consider it to be consistent (denoted by +1).

But when the pair is in different directions, we consider it to be inconsistent (denoted by -1). The lower diagram of Figure 5-7 shows the number of (in)consistent cases, that is, cases where the 11-paper and 21-paper tendencies disagree. As shown here, the consistent cases appear in 83% (30 out of 36) of the total recommendation methods we used. This means that most recommendation methods are scalable (with respect to the gain/loss relative to the best-case benchmark) at least from the 11-paper to the 21-paper cases. Whether or not they are also consistent when the number of involved papers is larger is an open question. We conjecture that the methods are consistent to up to 50 papers in terms of their gain/loss relative to the best-case benchmark.

Given the above data, when a new learner joins the system assuming the ratings by others exist, the best method is to use PopCon2D to recommend 1 paper or PopUM-CF for the recommendation of 3 papers. This result is consistent with that when there are 21 papers. However, with PopCon2D in the 11-paper case, we use w_c = 0.2 with w_r = 0.5 and the composition of ($w_{interest}$, $w_{bkgrKnowledge}$) = (.8, .2), which differs from that in the 21-paper case, where we use w_c = 0.15 with w_r = 0.1 and the composition of ($w_{interest}$, $w_{bkgrKnowledge}$) = (.9, .1). In other words, if we tuned the value of the weights in the computation, we might find out that a slight change can quantitatively improve the pedagogical recommendation performance. Hence, it is suggested that a set of the most appropriate parameters be selected by trying various sets of possible parameters according to available ratings in an offline fashion occasionally by the system.

When a learner is half-way through the course, i.e. after rating at least four papers, 1D-CF for recommending a single paper and Pop5D CF for recommending three papers are the best

methods. This result suggests that multidimensional CF may not necessarily be better than uni-dimensional CF.

Experimental results on the case when the learner is near the end of the course are similar to those when the learner is half-way through the course.

5.2.3 Summary

Table 5-5 summarizes our recommendations on adopting appropriate mechanisms based on the system context.

Table 5-5 A summary of recommended recommendation methods

Context		**Appropriate recommendation method(s)**	
		Top one	**Top three**
When there is enough ratings and papers	The learner is new to the course	**PopCon2D**	**PopUM-CF**
	The learner is half–way in the course	**PopCon2D**	**PopUM-CF**
	The learner is near the end of the course	**PopCon2D**	**Pop5D**
When there is not enough ratings and papers	The learner is new to the course	**PopCon2D**	**PopUM-CF**
	The learner is half–way in the course	**1D-CF**	**Pop5D**

From the above experiments we found out PopCon2D performs very well under four typical learning scenarios for picking the best one paper, and the more complex CF algorithms including Pop5D, PopUM-CF work well for making the best three recommendations. Due to its characteristics, PopCon2D can not only be used to start up the recommendation but also to inform the recommendation (since it contains information such as paper popularity, paper

content, user model on learner interest and knowledge background which can be used to generate recommendation without using paper ratings). In dimensions such as this, with a limited number of both papers and learners (and other constraints such as course syllabus), we can conclude that considering other features than just overall rating and user interest can help inform the recommendation.

When the system does not have enough data on paper and user models, a content-based filtering method is appropriate by matching the new user model and existing user and paper models. However, when there are not enough papers to perform the matching, some other features such as popularity $\tilde{r}_j$ need to be injected to inform the RS, as in PopCon2D and PopUM-CF. These methods characterize the features of the pedagogical paper recommendation and reflect that human judgments of scientific articles are influenced by a variety of factors including a paper's topical content, its content appropriateness and its value in helping users achieve their task [Custard and Sumner 2005]. It also highlights the importance of appropriately incorporating such factors into the recommendation process.

It is observed that most of the recommendation methods are scalable (in our case, from 11 to 21 papers) with respect to the gain/loss relative to the best-case benchmark, although it is still unclear the degree of this scalability.

The experiments conducted so far and suggestions made on the adoption of recommendation methods are based on the data we have collected during a one semester course. Therefore, it is unclear whether these methods are still suitable when user ratings keep increasing. If this is the case, the system should perform maintenances (e.g. re-evaluation) occasionally in an offline fashion, in order to adjust its policy on the adoption of the recommendation approaches. In the next section, we will show the usage guideline of this prototype under various conditions,

including a re-evaluation strategy and an ensemble method when more than one recommendation method is available.

5.3 Case Studies

Previous sections mainly show examples that are not intended to cover a larger possible use of this prototype. In this section we will describe in more detail other scenarios and the guideline of using this prototype. For the sake of clarity, we assume a similar class in software engineering is offered; thus, using the same papers but not necessarily with the same user models or ratings. Indeed, this prototype can be easily modified to other classes, e.g. AI or data mining, by changing the related subtopics and the required background knowledge to understand the papers.

In this stage, these case studies reflect the typical usage of the system without considering the usability of it; that is, tutors do not need to choose these algorithms manually, instead, an automatic module will choose the appropriate recommendation mechanism(s) under each condition.

5.3.1 Scenario One: New Learners with Unrated Papers during the First Use of the System

Suppose a tutor will use the system at the beginning of the class consisting of 30 students. Suppose the system has not been used previously and no paper or learner models are available (see Figure 5-8). Suppose the tutor wants to recommend one to three papers from five candidate papers related to the topics taught in the first few weeks with the goal of attaining as high Overall rating as possible.

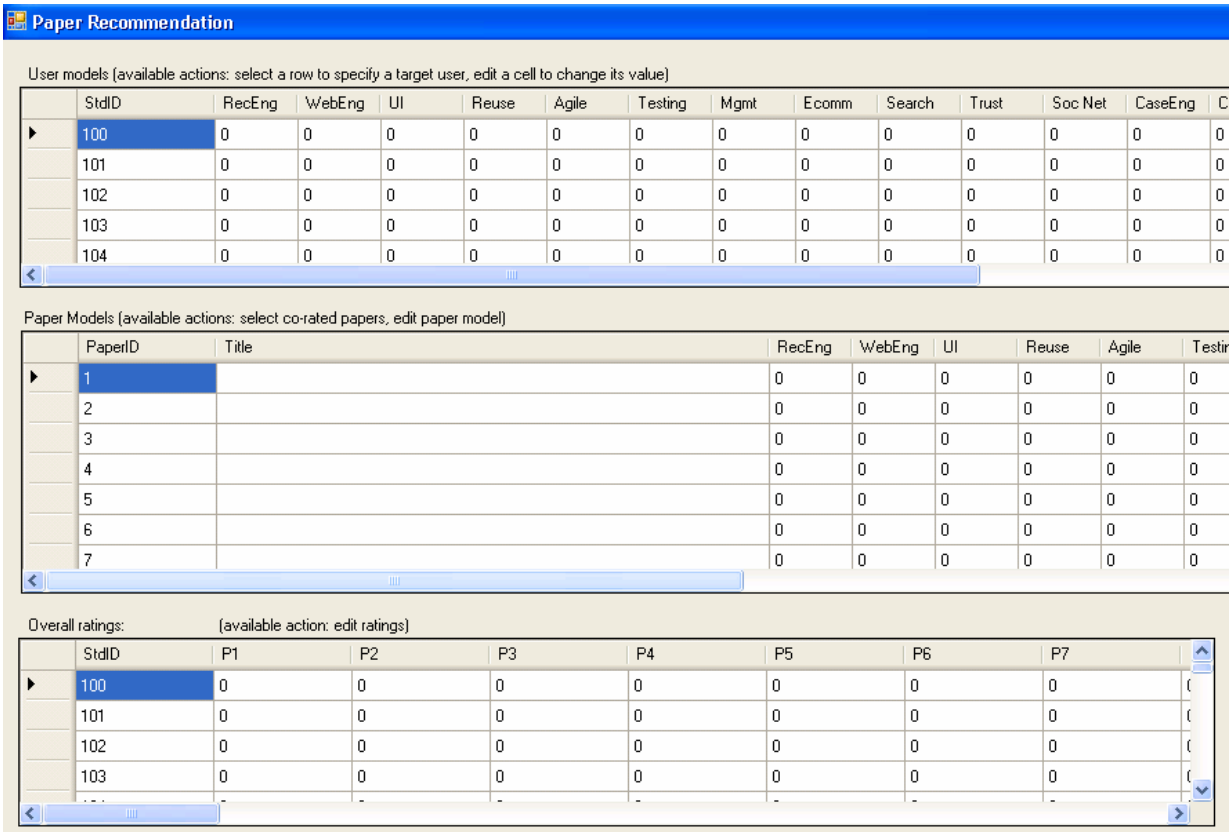


Figure 5-8 The initial system with empty user and paper models

In this case the following steps have to be taken by the tutor. First, the tutor will fill the five paper models by subjectively rating them according to their subtopics and the required background knowledge to read them. Then, s/he will ask learners to fill in a questionnaire about their learner model, i.e. their interest in the relevant subtopics and their self-assessed knowledge on the required background knowledge (see user model sheet and paper model sheet in Figure 5-9). After all paper models and learner models are filled, the tutor can start to use the system to find recommended papers. However, since the system does not have any rating, the only option is to use content-based filtering (ContentF). Figure 5-9 shows an example of the recommendation result using default parameter setting ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0).

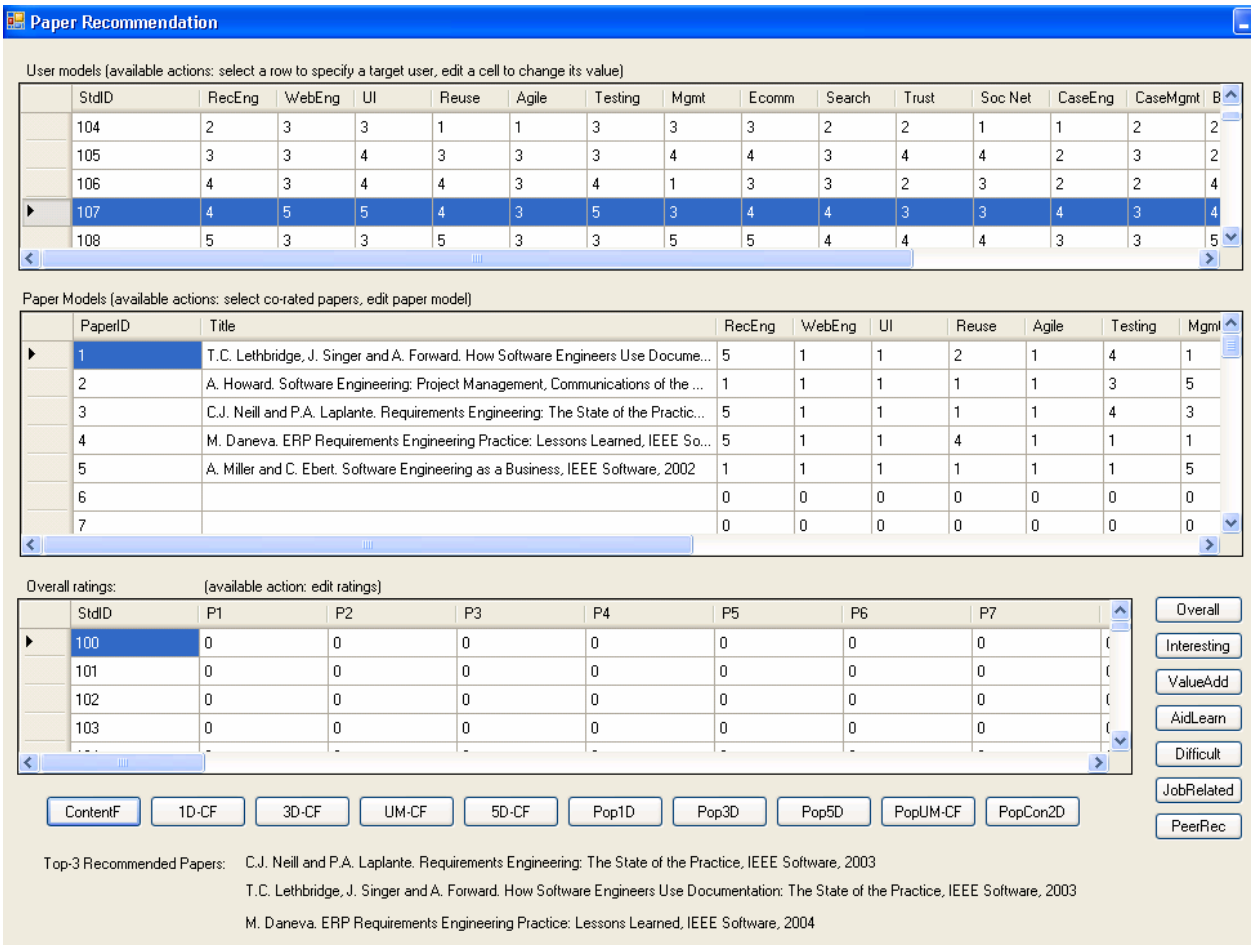


Figure 5-9 The recommendation results using content-based filtering at the beginning of the class

It is shown at the bottom of Figure 5-9 that the three papers recommended to learner #107 belong to subtopic requirements engineering, software reuse, and software testing. For example, the first recommended paper written by Neil and Laplante is rated by the tutor with “5” in the category requirements engineering (RecEng) and “4” in software testing (Testing) as shown in the third row of the paper model sheet in Figure 5-9. Similarly, the second recommended paper written by Lethbridge et al. . also receives the same ratings in these subtopics as shown in the first row of the paper model sheet in Figure 5-9. Finally, the third paper by Daneva is rated “5” on requirements engineering and “4” on software reuse (the fourth row of the paper model sheet).

If we look at the user model sheet, the recommendation results conform to the learner #107 (the fourth row of user model sheet), who has rated “4” for both requirements engineering and software reuse, and “5” for software testing. The other two candidate papers by Howard and Miller and Ebert are related to software project management, which both are rated “5” in this subtopic as shown in the second and fifth row of paper model sheet in Figure 5-9. Since learner #107 is less interested in project management, as s/he rated it “3” only (see the fourth row and the eighth column of the user model sheet with column heading “Mgmt” in Figure 5-9), these papers will not be recommended. These results are consistent with what showed be expected with content-based filtering.

Suppose the learners have rated the recommended papers, and the tutor has added more candidate papers related to UI and usability engineering into the system, say after several weeks of the class. Since the new papers have not been rated by the learners and the old five papers have only been rated sparsely, the reliability of using paper popularity is low; hence, it is not appropriate for the tutor to use recommendation methods that include paper popularity. In addition, due to sparse rating on existing papers; it is also not appropriate to use rating-based CF, because the chance of finding enough co-rated papers for running rating-based CF is low. Also, since we have new papers without prior ratings in our system, user-model-based CF will not be able to recommend these papers. Thus, under this condition, the tutor has to use content-based filtering again. Figure 5-10 shows the results of content-based filtering in this situation with default parameter setting, where three UI-related papers are recommended to learner #107. Note, learner #107 is indeed interested in the UI topic (rated “5” in his/her user model, see the fourth column of the fourth row in the user model sheet of Figure 5-10). Again, this result conforms to expectations for content-based filtering.

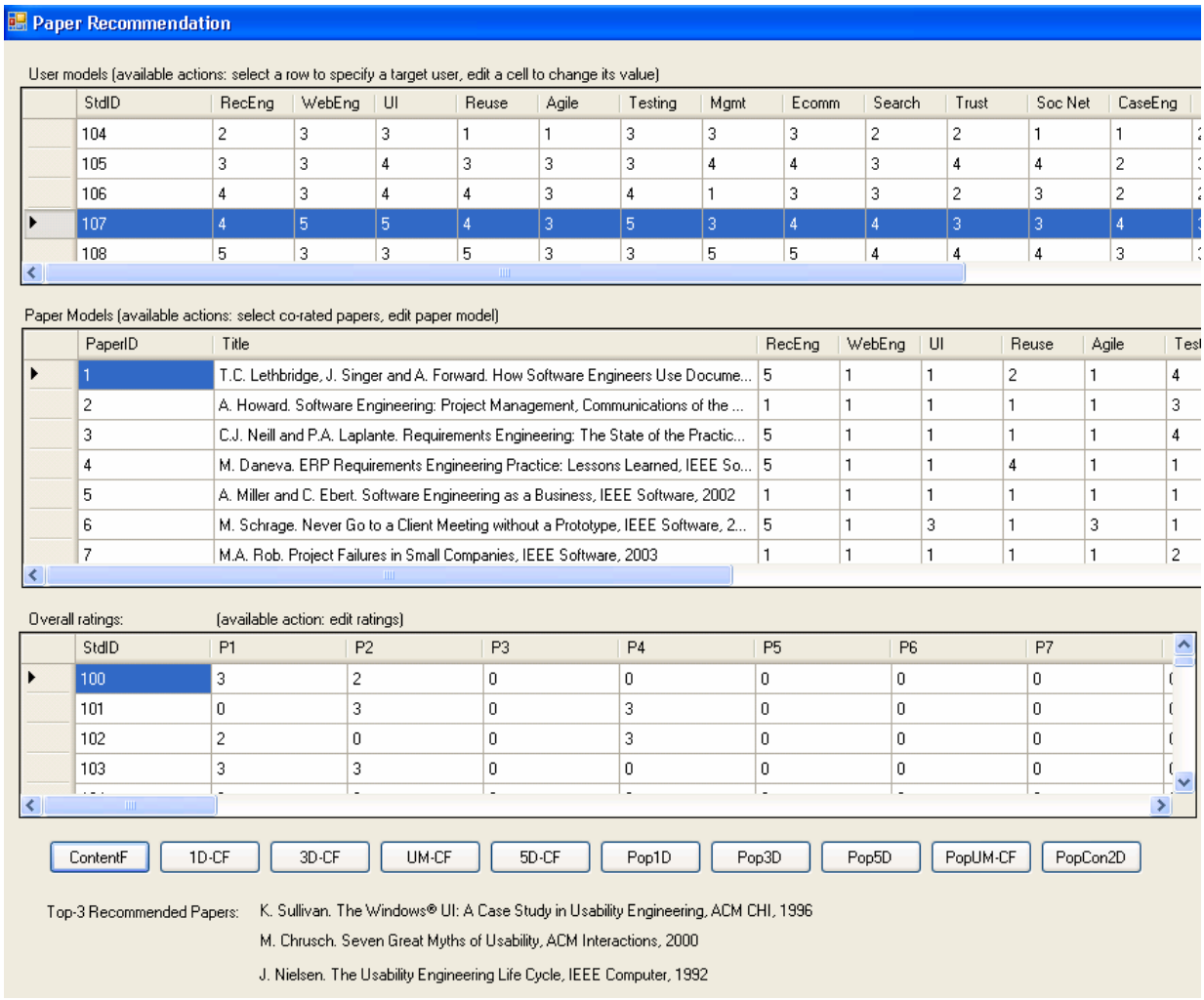


Figure 5-10 The recommendation after new papers are added half-way through the class

Now, suppose several more weeks have passed and more papers have been recommended and rated, e.g. at the end of the class, where each paper has been rated by at least *N* learners, say 15 learners,[2] and no more papers have been added. In this case, the tutor may choose methods that incorporate paper popularity. However, since the ratings by learners are sparse, rating-based

[2] Since the range of the rating is {1, 2, 3, 4}, 15 data points are enough to justify the frequency distribution of the ratings to a paper. This number is also enough to verify the difference between two average ratings (the popularity of two papers), e.g. by performing a t-test. Indeed, using a larger number, such as 30, is better but may not be practical in many situations, especially when the ratings are sparse and the number of learners is few.

CF is not appropriate. Hence, the tutor may choose ContentF, UM-CF, PopUM-CF, or PopCon2D. According to our previous empirical analysis, PopCon2D and PopUM-CF are better than ContentF and UM-CF, where PopCon2D may be the best method to recommend the top one paper in most cases (cf. Table 5-5). Figure 5-11 shows the recommended papers after the tutor has used PopCon2D with parameters ($w_{interest}$, $w_{bkgrKnowledge}$) = (1, 0) and w_r = 5.0. Note that these results may not be the best ones. A better recommendation strategy may be obtained using other methods (e.g. by an ensemble method).

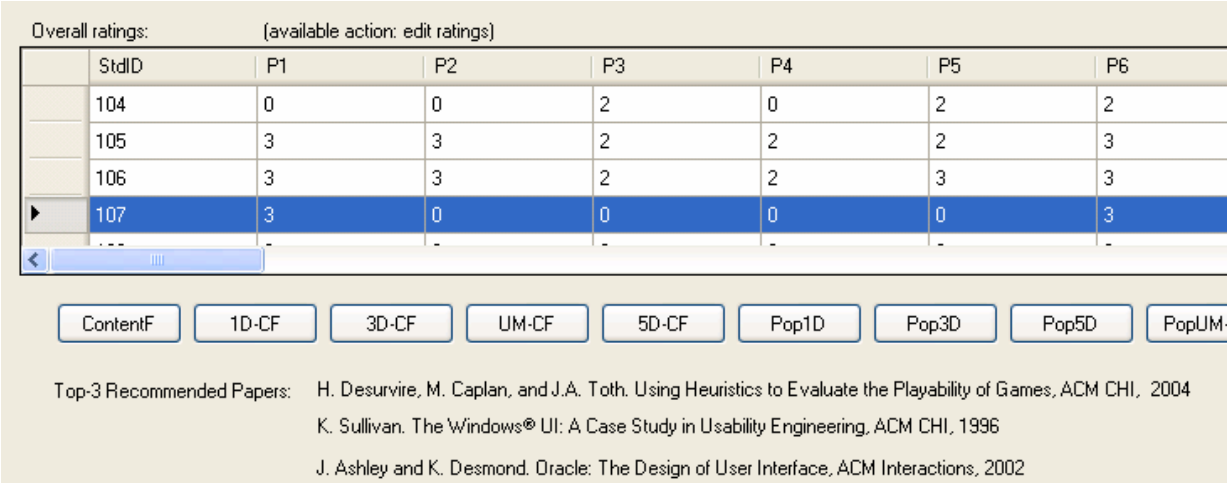


Figure 5-11 The recommendation at the end of the class

5.3.2 Scenario Two: New Learners with Existing Ratings on Partial Papers

In this scenario, we also assume that a tutor will use the system at the beginning of the class consisting of 30 students. However, we assume the tutor has the Overall ratings provided by the previous year's learners, say 30 learners, on 10 mandatory papers. Since these learners did not use the system, the system does not have their learner model. Suppose the tutor has thought of adding 10 more candidate papers (newly added papers) in this semester and wants to recommend one to three papers from all candidate papers in the first few weeks. The initial data inside the system is as shown in Figure 5-12. Note that if the tutor wants to recommend papers

within a subset of candidate papers, s/he may exclude irrelevant paper models and ratings at the initial stage.

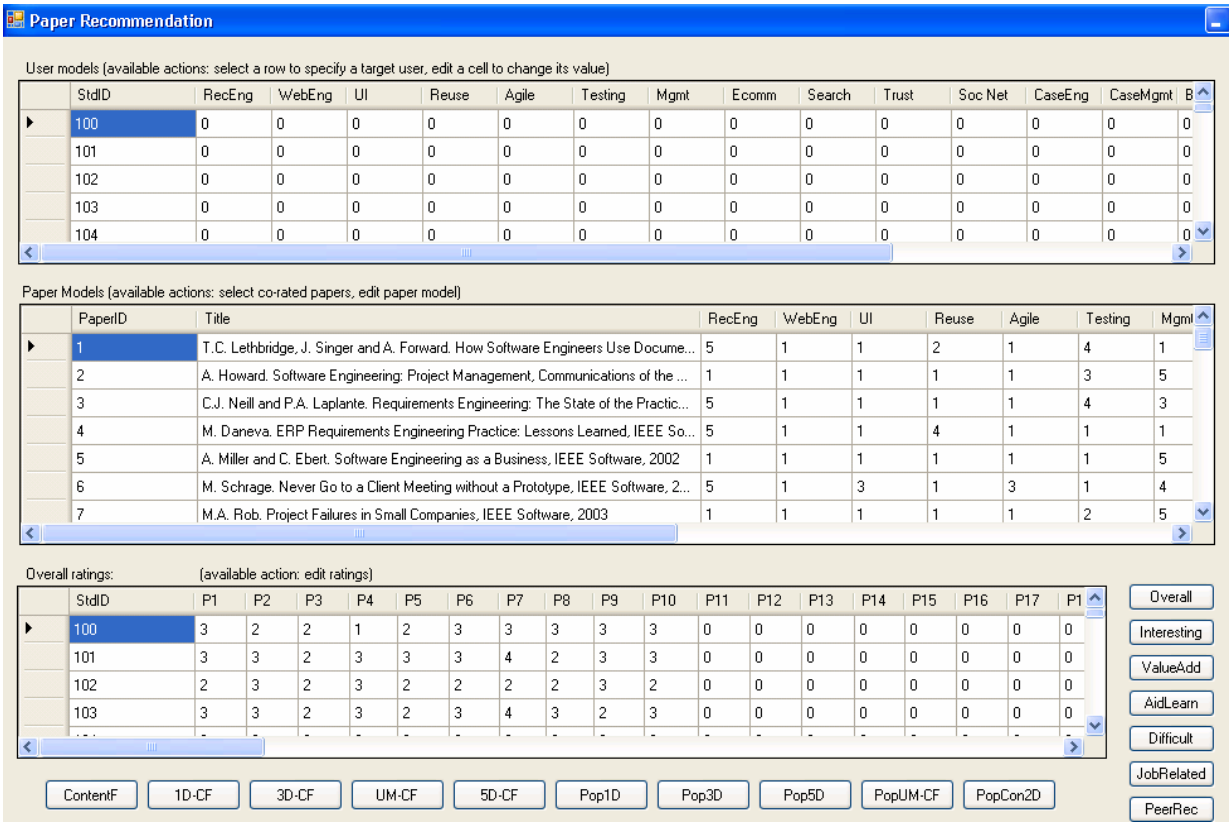


Figure 5-12 The initial system with ratings from previous users and paper models

As the system has some ratings but not the previous learners' user model, the feasible recommendation options at the beginning class are ContentF and PopCon2D. However, PopCon2D is biased toward the rated papers only, hence, shall be avoided until other newly added papers are rated by at least $N \geq 15$ learners. Thus, again, ContentF will be chosen for the recommendation made to new learners.

As the class proceeds, these learners have provided enough ratings, say more than six. At this moment, the tutor may choose 1D-CF and UM-CF in addition to ContentF. Again, there is a disadvantage in using 1D-CF, because the old learners, who may more plausibly be the

neighbors of a target learner, have rated only old papers; thus, 1D-CF is incapable of recommending new papers. Similarly, the new learners are less likely to be neighbors of a target user due to the lower chance of finding co-rated papers among them. But, as we do not have user models of old learners, UM-CF is restricted to new learners only. Hence, we suggest ContentF here.

Now, when the class is near its end, each learner may have provided more ratings; assume each paper has been rated by more than 15 learners. Hence, the tutor may use PopCon2D. The other popularity-based recommendations such as PopUM-CF may also be used.

The recommendation methods in this scenario are similar to those in the previous scenario. It may seem like the additional Overall ratings provided by old learners do not affect the recommendation, because rating-based CF is not suggested here. However, this is not true. When we use PopCon2D or PopUM-CF we have incorporated ratings by old learners in the calculation of paper popularity, information that we did not have in the first scenario. Old ratings are more useful when the newly added papers will not be recommended, e.g. in the first few weeks. In such a case, both PopCon2D and PopUM-CF can be used. Again, PopCon2D may be better than PopUM-CF in recommending the best paper (cf. Table 5-5).

5.3.3 Scenario Three: New Learners with Existing Ratings (Revisited) --- Ensemble Methods

In this subsection, we will show a method of choosing a recommendation paper when more than one method is applicable. Suppose a tutor has used the system for years, where many ratings exist in the system with the same candidate papers. To use the system at the beginning of the class, the tutor has the following options: ContentF, UM-CF, PopUM-CF, and PopCon2D. In order to recommend a paper to the user, the tutor may use an ensemble method, which is commonly used in classifier systems for obtaining a robust result. The underlying principle is as

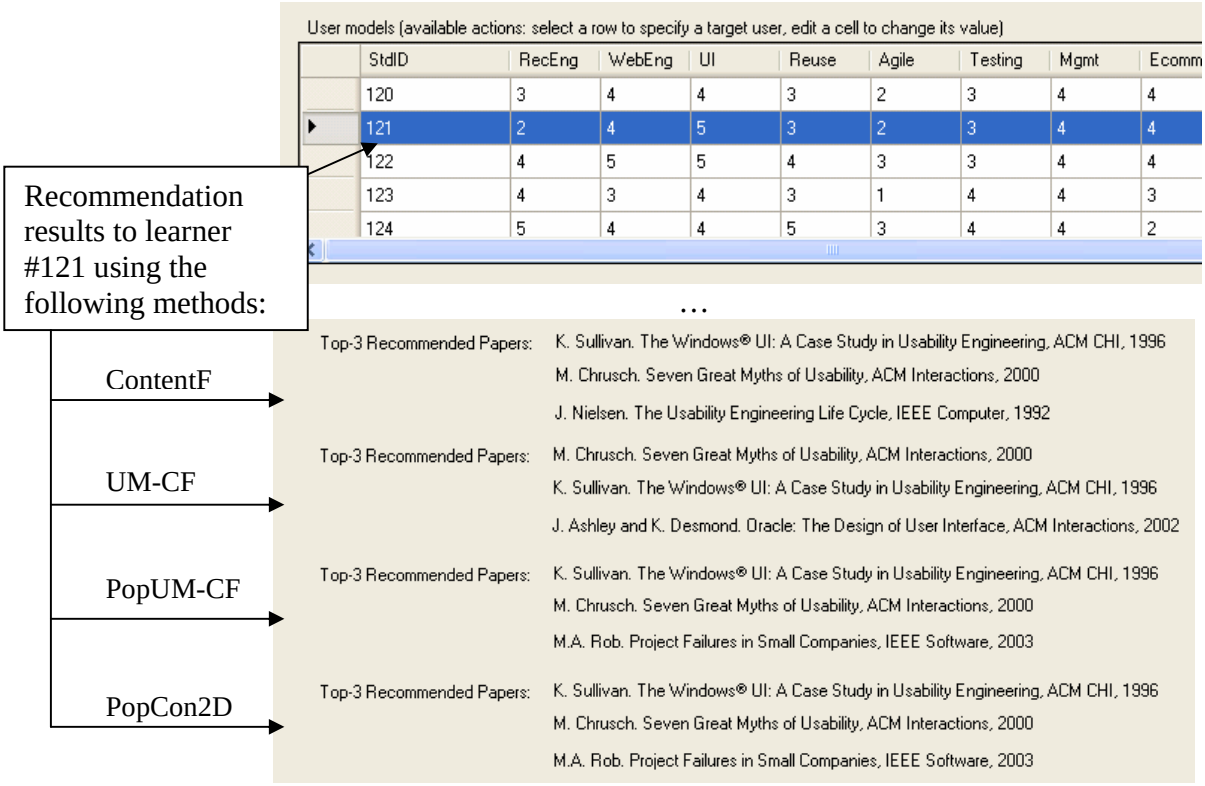
follows: if by using various recommendation methods we get the same paper, then this paper is believable as the best one. However, if the recommended papers by these methods are not consistent, then we *may* get the best one through a voting mechanism.

Figure 5-13 illustrates the results of an ensemble method, where we apply a weighted voting mechanism. The weights are 3, 2, and 1 for the best, second best, and third best recommended papers, respectively. The best three recommended papers from each applicable method (ContentF, UM-CF, PopUM-CF, and PopCon2D) are shown at the top of Figure 5-13, while the bottom shows the calculation of their weighted voting. We use default parameter setting (based on experimental results) in these methods, i.e. $w_{interest}$ = 1 in ContentF and PopCon2D, 5 neighbors in UM-CF and PopUM-CF, and w_r = 5.0 in PopUM-CF and PopCon2D. For example, since the top three recommended papers using ContentF are the Sullivan, Chrusch, and Nielsen's papers, Sullivan's paper is voted 3, Chrusch's paper is voted 2, and Nielsen's paper is voted 1. Similarly, by using UM-CF, Chrusch's paper is voted 3, Sullivan's is voted 2, and Ashley's is voted 1, etc.

As shown at the bottom of Figure 5-13, Sullivan paper "The Windows UI: A Case Study in Usability Engineering", received the highest total vote, i.e. 11 points. In fact, it appears as the most recommended paper in three out of four methods, which shows a consistent recommendation outcome. As more papers are rated by the learners, e.g. in the middle or at the end of the class, the tutor may apply other methods, such as 1D-CF, 3D-CF, Pop5D, etc.

However, an ensemble method is not always a good choice, especially with regard to computational complexity. Suppose we have 10 different methods, where each of them has 100 variants through parameter adjustment. Then, we have 1000 different methods that may all be included in the ensemble process, which pose an inefficient computation even when we run them

automatically. In fact, most of these variants are likely to generate low-rating results. Hence, before we can apply an ensemble method, we have to perform parameters adjustment (tuning) and method selection so as we only combine results from 3 to 5 methods.



Paper by:	**ContentF**	**UM-CF**	**PopUM-CF**	**PopCon2D**	**Total Vote**
Sullivan	3	2	3	3	**11**
Chrusch	2	3	2	2	9
Nielsen	1	0	0	0	1
Ashley & Desmond	0	1	0	0	1
Rob	0	0	1	1	2

Figure 5-13 An illustration of an ensemble method where the Sullivan paper is recommended to learner #121 after obtaining the highest votes from four applicable recommendation methods

Since the ensemble method poses many unexplored problems and is not the objective of this thesis, we will leave it for our future work. Nevertheless, the tutor may use it as an optional

strategy while using this prototype, especially when we cannot find the best recommendation method from the available ones.

5.3.4 Scenario Four: Online Re-evaluation

In the previous subsection, we have shown an ensemble method to unify results from various applicable recommendation methods using this prototype. In fact, as we have previously suggested, a tutor may perform an online re-evaluation of applicable methods in order to choose the "best" recommendation method. The re-evaluation is by using a strategy similar to the all-but-one by hiding the ratings of a target user on each round through running all applicable methods on existing data (known ratings). The idea is as follows: suppose we have some ratings from the previous year's learners and we want to recommend papers to new learners. Suppose we have several applicable methods but we do not know which one is the best one. This situation may arise when we have collected enough data (ratings) in the middle or near the end of the class, or when we want to use this prototype for classes other than software engineering. Then we may pick a learner from the previous year (i.e. an old learner), whose ratings are known, to test these methods. If any of the methods is superior to the rest in making recommendation, in terms of the ratings given by the old learner, then the method may also be the best for making a recommendation to new learner. Figure 5-14 shows an example where learner #110 is used as the tester using ContentF. The result shows that the best recommended paper is rated "3", which is not a high one.

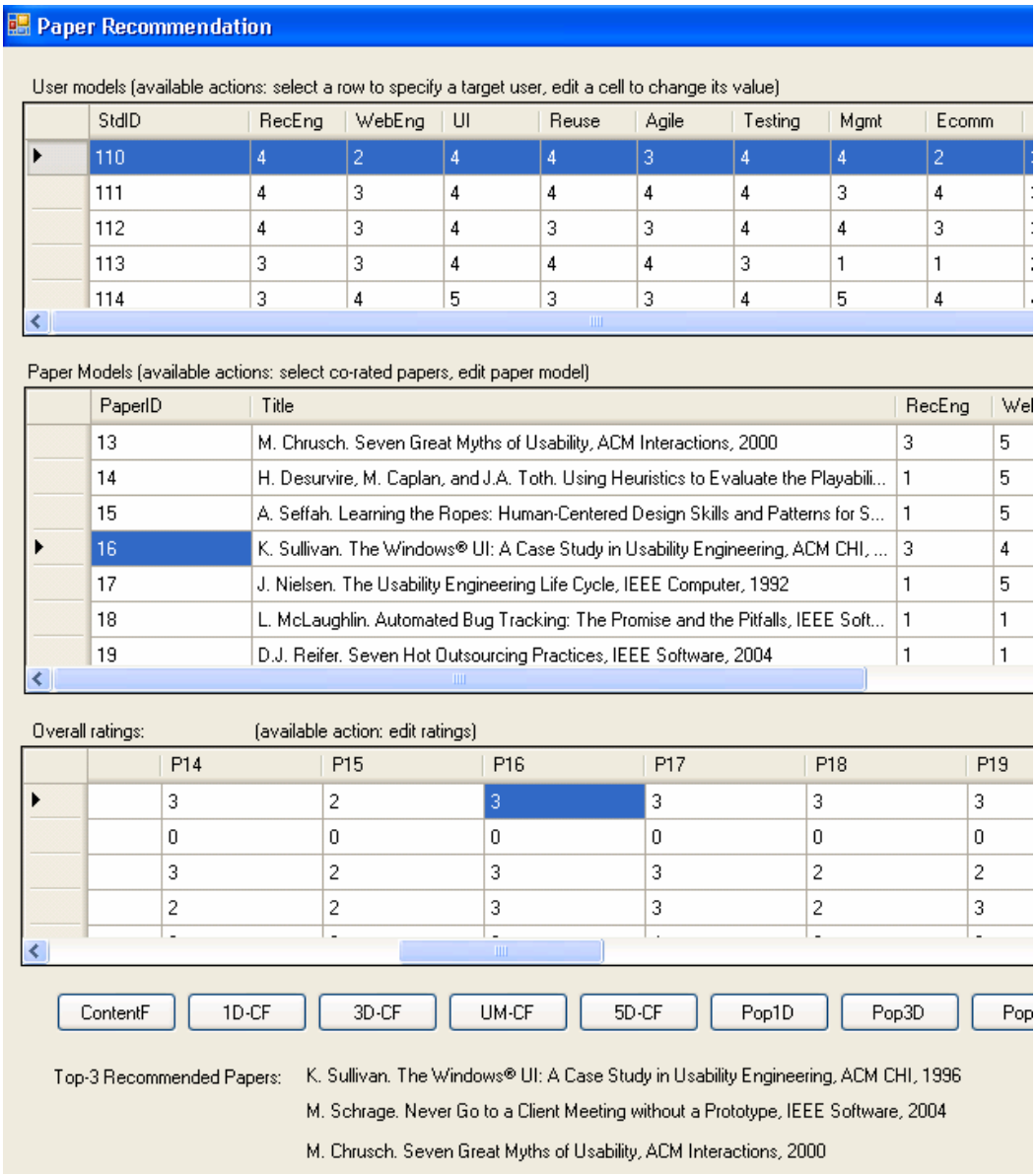


Figure 5-14 An illustration of a manual re-evaluation of applicable methods

To increase the confidence in our re-evaluation, we may test these methods on more than one old learner. In current prototype, the re-evaluation can only be performed manually; hence, is not efficient and recommended. An automatic re-evaluation method is desirable to help tutor in deciding the best recommendation method. In the next section, we will show the algorithms that can be used to perform this re-evaluation.

5.4 How to Choose Appropriate Recommendation Methods

This section shows some algorithms that can be used to implement automatic re-evaluation. The purpose is to demonstrate the feasibility of our approach in building a better prototype. However, a further detailed elaboration may be required before it is implemented.

Notations:

$\boldsymbol{P}$	: the set of n candidate papers with indices {1, 2, 3, …, n}
$\boldsymbol{L}$	: the set of m learners with indices {1, 2, 3, …, m}
m	: index of the target learner (by default it is the last component of $\boldsymbol{L}$)
$\boldsymbol{L}_{-m}$	: the set of $m - 1$ learner, excluding the target learner
$\mathcal{R}_{m \times n}{}^{d}$	: the matrix consisting of ratings by all learners (m rows) to all papers (n columns) at dimension d = {Overall, Val_Add, Peer_Rec}
$r_{lp}{}^{d}$	: the entry of the matrix $\mathcal{R}^{d}$, i.e. the rating by learner l to paper p in dimension d; if no rating exists, then $r_{lp}{}^{d} = 0$
$N(l)$	: the number of ratings given by learner l (nonzero entries in the l-th row of $\mathcal{R}^{Overall}$)
$N(p)$	: the number of ratings given to paper p (nonzero entries in the p-th column of $\mathcal{R}^{Overall}$)
$\mathcal{U}_{m \times k}$	: the user-model matrix consisting of m-learner ratings to k user-model entries
$\mathcal{P}_{n \times k}$	: the paper-model matrix consisting of n-paper ratings given by the tutor to k paper-model entries
t_1	: the minimum number of ratings by each learner; this value is determined by the tutor for rating-based CF or user-model-based CF
t_2	: the minimum number of co-rated papers determined by the tutor
MObj	: the array of x recommendation-method objects consisting of method names and their properties (value, parameters, etc.)

The following algorithm is used to find recommendation method(s) to target learner m. For simplicity, we arrange the target learner's ratings on the last row of $\boldsymbol{R}^{d}$ and $\boldsymbol{U}$.

****

Algorithm RECOMMENDATION

Input value: $\mathcal{R}^{Overall}$, $\mathcal{R}^{Val\_Add}$, $\mathcal{R}^{Peer\_Rec}$, $\mathcal{U}$, $\mathcal{P}$

Output value: recommendation method(s) to target learner m.

//calculate the average co-rated papers between the target user and the existing users
//round the value to integer
$AveCor = \lfloor \sum_{l \in \boldsymbol{L}_{-m}} \boldsymbol{CORATE}(l, m) / (m - 1) \rfloor$

//check whether or not all learners in training dataset have enough ratings
//and whether or not all papers have been rated (new-paper problem)
If $\forall l \in \boldsymbol{L}_{-m}\ N(l) < t_1 \vee \exists p \in \boldsymbol{P}\ N(p) = 0$ **Then**
    //since we have sparse training dataset or new paper problem, choose *ContentF*
  Return *ContentF*

//when there is neither a sparsity problem nor a new-paper problem, check whether the co-rated papers are less
//than t_2 or not
Else If $\forall l \in \boldsymbol{L}_{-m}\ \boldsymbol{CORATE}(l, m) < t_2$ **Then**
    //since we do not have enough co-rated papers, we exclude *rating-based CF*
    //check whether we have enough ratings to consider paper popularity or not

If $\forall p \in \boldsymbol{P}\ N(p) < 5$ **Then**
//we do not have enough ratings to consider popularity-based
//recommendation methods
//then check whether we have complete user models or not
If $\forall u \in \mathcal{U}\ u \neq 0$ **Then**
//since we have complete user models, we should consider *UMCF*
Return *SELECT*(*ContentF*, *UMCF*)
Else
//since we have incomplete user models, we exclude *UMCF*
Return *ContentF*
Else
//since we have enough ratings to consider paper popularity
//check whether we have complete user models or not
If $\forall u \in \mathcal{U}\ u \neq 0$ **Then**
//since we have complete user models, we will consider *PopUMCF*
Return *SELECT*(*PopCon2D*, *PopUMCF*)
Else
//since we have incomplete user models, we exclude *PopUMCF*
Return *PopCon2D*

//check whether or not the co-rated papers are greater than or equal to t_2
Else If $\exists l \in \boldsymbol{L}_{-m}$ ***CORATE***$(l, m) \geq t_2$ **Then**
//check whether we have enough ratings to consider paper popularity or not
If $\forall p \in \boldsymbol{P}\ N(p) < 5$ **Then**
//we will not consider *popularity-based recommendation methods*
//check whether we have complete user models or not
If $\forall u \in \mathcal{U}\ u \neq 0$ **Then**
//since we have complete user models
//check whether all 3D ratings are complete or not
If $\forall l, p\ (r_{lp}^{Overall} \neq 0 \Rightarrow r_{lp}^{Value\_Add} \neq 0 \wedge r_{lp}^{Value\_Add} \neq 0)$ **Then**
//since we have complete 3D ratings, consider *3DCF and 5DCF*
Return *SELECT*(*ContentF*, *UMCF*, *1DCF*(*AveCor*),
3DCF(*AveCor*), *5DCF*(*AveCor*))
Else
//since we have incomplete 3D ratings, we exclude *3DCF and 5DCF*
Return *SELECT*(*ContentF*, *UMCF*, *1DCF*(*AveCor*))
Else
//since we have incomplete user models
//check whether all 3D ratings are complete or not
If $\forall l, p\ (r_{lp}^{Overall} \neq 0 \Rightarrow r_{lp}^{Value\_Add} \neq 0 \wedge r_{lp}^{Value\_Add} \neq 0)$ **Then**
//since we have complete 3D ratings, we consider *3DCF*
Return *SELECT*(*ContentF*, *1DCF*(*AveCor*), *3DCF*(*AveCor*))
Else
//since we have incomplete 3D ratings, we exclude *3DCF*
Return *SELECT*(*ContentF*, *1DCF*(*AveCor*))
Else
//we will consider *popularity-based recommendation methods*
//check whether we have complete user models or not
If $\forall u \in \mathcal{U}\ u \neq 0$ **Then**
//since we have complete user models
//check whether all 3D ratings are complete or not
If $\forall l, p\ (r_{lp}^{Overall} \neq 0 \Rightarrow r_{lp}^{Value\_Add} \neq 0 \wedge r_{lp}^{Value\_Add} \neq 0)$ **Then**
//since we have complete 3D ratings, we can consider *PopCon2D*,
//Pop3D, and Pop5D
Return *SELECT*(*PopCon2D*, *PopUMCF*, *Pop1D*(*AveCor*),

Pop3D(*AveCor*), *Pop5D*(*AveCor*))
Else
//exclude *Pop3D*, *Pop5D*, and replace *PopCon2D* with *ContentF*
Return *SELECT*(*ContentF*, *PopUMCF*, *Pop1D*(*AveCor*))
Else
//since we have incomplete user models, we exclude *PopUMCF* and
//*Pop5D*, and we replace *PopCon2D* with *ContentF*
//check whether all 3D ratings are complete or not
If $\forall l, p\ (r_{lp}^{Overall} \neq 0 \Rightarrow r_{lp}^{Value\_Add} \neq 0 \wedge r_{lp}^{Value\_Add} \neq 0)$ **Then**
//since we have complete 3D ratings, we consider *Pop3D*
Return *SELECT*(*ContentF*, *Pop1D*(*AveCor*), *Pop3D*(*AveCor*))
Else
//since we have incomplete 3D ratings, we exclude *Pop3D*
Return *SELECT*(*ContentF*, *Pop1D*(*AveCor*))

****

The notation (*AveCor*) in *1DCF*(*AveCor*), *3DCF*(*AveCor*), etc. represents the average number of co-rated papers that a target learner may have with existing learners. In Chapter 4, we have shown that the number of co-rated papers affects the accuracy of recommendation. Hence, we have to consider it in selecting the 'best' recommendation method for a target learner.

Here, we will not discuss the recommendation functions *ContentF*, *UMCF*, *PopCon2D*, *PopUMCF*, *1DCF*(*AveCor*), *3DCF*(*AveCor*), *5DCF*(*AveCor*), *Pop1D*(*AveCor*), *Pop3D*(*AveCor*), and *Pop5D*(*AveCor*), because to this point our concern is to pick recommendation method(s) from available ones. The function ***CORATE***(*l*, *m*) is simply to calculate the number of co-rated papers between two learners' rating; it is given below.

Algorithm CORATE(*l*, *m*)
Input value: indices of the learner *l* and target learner *m*, and $\mathcal{R}^{Overall}$.
Output value: the number of co-rated papers given by learner *l* and target learner *m*.
Corated = 0
For *p* = 1 **to** *n*
If $r_{lp}^{Overall} \neq 0 \wedge r_{mp}^{Overall} \neq 0$ **Then** *Corated*++
Return *Corated*

Another important function is to choose the 'best' recommendation method(s) which is realized using function ***SELECT***(*method*(*1*).*name*, …, *method*(*N*).*name*), whose input parameters are recommendation-method names. To select the best method(s), we only need to compare a set

of real values {*method*(*1*).*value*, …, *method*(*N*).*value*}, that represent the accuracy of the corresponding methods in recommending papers. These values are obtained from offline maintenance using the available training dataset. The values will be stored in an object array ***MObj*** which consists of *x* recommendation-method objects. In general, each object has a name, a value, and some parameters used for recommendation:

MObj[*j*].*name*
MObj[*j*].*value*
MObj[*j*].*parameter1*
MObj[*j*].*parameter2*
….
MObj[*j*].*parameterN*

For the sake of simplicity, we use the average Overall ratings received by each method to represent the object's value.

****

Algorithm SELECT(*method*(*1*).*name*, *method*(*2*).*name*, …, *method*(*N*).*name*)
Input value: A list of methods name <*method*(*1*).*name*, …, *method*(*N*).*name*>
The array of *x* recommendation objects <*MObj*[*1*], …, *MObj*[*x*]>
Output value: recommended method(s)

//retrieve *method*(*1*), …, *method*(*N*) from the existing method database
//and store them in an object array *Temp*[]
Temp[] = Null
For *i* = 1 **to** *N*
For *j* = 1 **to** *x*
// retrieve the recommendation object when it matches the input parameter
If *MObj*[*j*].*name* = *method*(*i*).*name* **Then**
//store the recommendation object in the array *Temp*[]
Temp[*i*] = *MObj*[*j*]
Exit For

//use ***MAX***() function to find the largest *Temp*[].*value*
max = ***MAX*** (*Temp*[].*value*)
//return a set of method(s) whose value is the highest,
Return {*Temp*[*i*].*name* | $\forall i \in \{1, \ldots, N\}$ *Temp*[*i*].*value* = *max*}

****

As mentioned previously, *MObj*[*1*].*value*, …, *MObj*[*N*].*value* are calculated off-line according to available ratings on a training dataset. To reduce computational cost, those are

calculated periodically by a maintenance module (similar to those that will be discussed in section 5.5.1). This module will perform experiments on the training dataset using all recommendation methods. The returned value from each method is the average rating of its recommended papers. In other words, we want to choose the method whose average ratings are highest once a set of methods are deemed by the algorithm to be appropriate in a certain circumstance. The process is similar to what we have done in our experiment in Chapter 4. Since each method needs different parameters and inputs, we will only show algorithm VAL_ContentF and VAL_Pop5D(10) here.

****

Algorithm VAL_ContentF

//the inputs of this algorithm are Overall ratings by training dataset, user models,
//paper models, and a set of pre-defined weight $w_{bgKnldg}$ stored in $\boldsymbol{W}$.
Input value: $\mathcal{R}^{Overall}$, $\mathcal{U}$, $\mathcal{P}$, $\boldsymbol{W}$
Output value: value (i.e. average Overall ratings) and other parameters (i.e. *Closeness_Y*, $w_{bgKnldg}$)

//test all seven available algorithms to calculate closeness value between a learner
//model and paper model
For $y = 1$ **to** 7
    //test various weights of background knowledge in calculating the
    //closeness value, for example we test $\boldsymbol{W}$ = {0.01, 0.02, 0.03, …, 0.99, 1.00}
    For $\forall w_{bgKnldg} \in \boldsymbol{W}$
        //calculate the average ratings given by all learners in the training dataset
        For $l = 1$ **to** $m - 1$
            p = ***ContentF***(y, $w_{bgKnldg}$, l, $\mathcal{U}$, $\mathcal{P}$)
            TotalRating = *TotalRating* + $r_{lp}^{Overall}$

        AveRating[y][$w_{bgKnldg}$] = *TotalRating* / ($m - 1$)

//store the value and indices of the maximum AveRating[][] in *MaxAveRating*
MaxAveRating ← ***MAX***(*AveRating*[][])

//assign the best ContentF and its parameters to ***MObj***
For $j = 1$ **to** x
    // retrieve the recommendation object for ContentF
    If *MObj*[j].*name* = *ContentF* **Then**
        MObj[j].*value* = *MaxAveRating.value*
        MObj[j].*parameter1* = *MaxAveRating.y*
        MObj[j].*parameter2* = *MaxAveRating.*$w_{bgKnldg}$

****

Note: We skip the detail of ***ContentF***( ) and ***MAX***( ) here. The algorithm ***ContentF***( ) can be developed easily with the main component *Closeness_y( )*.

****

Algorithm VAL_Pop5D(10)

Input value: $\mathscr{R}^{Overall}$, $\mathscr{R}^{Val\_Add}$, $\mathscr{R}^{Peer\_Rec}$, $\mathscr{U}$, $\mathscr{P}$, $\boldsymbol{W_{Int}}$, $\boldsymbol{W_{OVP}}$, $\boldsymbol{W_r}$, $\boldsymbol{W_{2D}}$, $\boldsymbol{N}$

Output value: value (i.e. average Overall ratings) and other parameters (i.e. $w_{Interest}$, $w_{Overall}$, w_{Value_add}, w_{Peer_Rec}, w_r, w_{2D}, *neighbors*)

//test 11 different number of neighbors $\boldsymbol{N}$ = {5, 6, 7, …, 15}
For n = 5 **to** 15
  //test 6 weights of interest in UMCF {0.5, 0.6, …, 1.0}
  For $\forall w_{Int} \in \boldsymbol{W_{Int}}$
    //test 11 different 3D rating-based weights (**O**verall, **V**alue_Add, **P**eer_Rec) =
    //(0.7, 0.15, 0.15), (0.72, 0.14, 0.14), …, (0.9, 0.05, 0.05)
    For $\forall w_{OVP} \in \boldsymbol{W_{OVP}}$
      //test 5 different weights of 2D effect $\boldsymbol{W_{2D}}$ = {0.5, 1, 3, 5, 10}
      For $\forall w_{2D} \in \boldsymbol{W_{2D}}$
        //test 4 different weights of paper popularity w_r = = {1, 2, 5, 10}.
        For $\forall w_r \in \boldsymbol{W_r}$
          //calculate the average ratings given by all learners
          For l = 1 **to** m – 1
          //find the best recommended paper, assuming the rest of the learners are the training
          //dataset and the number of co-rated papers are controlled to10 only (the last parameter of
          //Pop5D( ))
            p = ***Pop5D***(n, w_{Int}, w_{OVP}, l, $\mathscr{R}^{Overall}$, $\mathscr{R}^{Val\_Add}$, $\mathscr{R}^{Peer\_Rec}$, $\mathscr{U}$, $\mathscr{P}$, 10)
            TotalRating = *TotalRating* + $r_{lp}^{Overall}$

          AveRating[n][w_{Int}][w_{OVP}][w_{2D}][w_r] = *TotalRating* / (m – 2)

//store the value and indices of the maximum AveRating[][][][][] in *MaxAveRating*
MaxAveRating ← ***MAX***(*AveRating*[][][][][])

//assign the best Pop5D(*10*) and its parameters to ***MObj***
For j = 1 **to** x
  // retrieve the recommendation object for Pop5D(*10*)
  If *MObj*[j].*name* = *Pop5D*(*10*) **Then**
    MObj[j].*value* = *MaxAveRating.value*
    MObj[j].*parameter1* = *MaxAveRating.neighbor*
    MObj[j].*parameter2* = *MaxAveRating.*$w_{Interest}$
    MObj[j].*parameter3* = *MaxAveRating.*$w_{Overall}$
    MObj[j].*parameter4* = *MaxAveRating.*w_{Value_add}
    MObj[j].*parameter5* = *MaxAveRating.*w_{Peer_Rec}
    MObj[j].*parameter6* = *MaxAveRating.* w_r
    MObj[j].*parameter7* = *MaxAveRating.*w_{2D}

****

5.5 Discussions

5.5.1 The Issue of Over-fitting and Solution

The issue of over-fitting the data remains a danger in any kind of data-centric approach. Therefore, a question arises as to whether the choice of the variables (say, the number of co-rated papers and neighbors, the value of the weights) still apply in the same way in another situation (e.g. when more ratings are available). To address this issue, we believe that the system should occasionally re-run the various recommendation methods under different conditions in order to determine which algorithm(s) would yield reasonably satisfying results. And this maintenance should be conducted in an offline fashion. Figure 5-15 draws a typical flow of the maintenance.

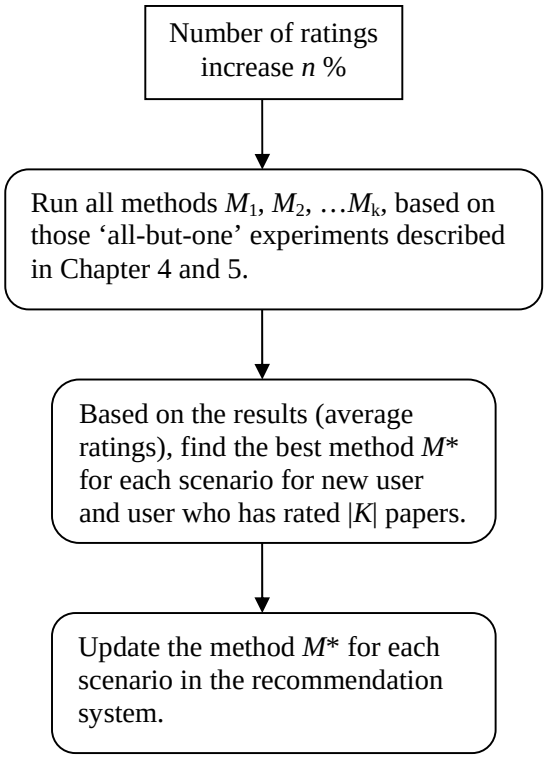


Figure 5-15 The flow of offline system maintenance

One suggestion to trigger the system to begin the process is when the number of ratings has increased *n* percent. For example, in the testing discussed previously, when the number of papers increases from 11 to 21, an (approximate) 90% increase, the effectiveness of each recommendation method does not change much, with a consistency of around 83%. However, it is only a special case of rating increase due to the addition of more papers. To be safe, we suggest re-parameterization of each recommendation method when *n* equals to 10% - 50%, i.e. by running with all the methods, and ranking those methods that can be used in the new situation. After that, a "best" method, if different from a previous one, can replace the existing one to be used when making recommendation.

However, when the amount of data keeps increasing, the computational cost incurred from system maintenance will be huge and therefore cannot be neglected. As such, the system should undergo some 'garbage collection' periods: deleting papers that are not welcome or, keeping more recent data (say, for the past 3 to 5 years). The latter has been studied in [Tang et al. 2005] for movie recommendation, where old movies (and therefore old ratings) are removed from the database, and traditional CF is then run on the remaining partial data. Through experiments on the benchmark MovieLens data sets, they demonstrated that the approach can not only be used to scale down the candidate sets, but also increase the accuracy of the recommender system [Tang et al. 2005].

5.5.2 Constraints of the Testing

The full space of user evaluation is certainly complex and not all issues have surfaced. The experiments and evaluation studies discussed in this and previous chapters are based on the data collected in a course offered in one semester only; therefore, the generality of results needs to undergo further validation before more certain conclusions can be drawn.

These follow up studies should be performed (ideally) in two semesters (on the same course or related courses) with more newly added papers. Then, some open issues can be further investigated. For instance, what is the appropriate number of co-rated papers in our domain? Theoretically, the more we know about a target learner, the better we can suggest the papers since we can use a larger number of co-rated papers to find his/her neighbors. However, as we argued in 4.4.2, specifying a larger number of co-rated papers may reduce the chance of recommending "new" papers, i.e. those have not been read by learners; hence, the experimental results do not necessarily capture the effectiveness of recommendation methods in practice. In addition, using a larger number of co-rated papers may make us believe that the recommendation methods perform well as we increase the number of co-rated papers. In other words, a high average rating given to recommended papers does not mean a good recommendation technique when we use a large number of co-rated papers, since when a relatively larger number of co-rated papers is considered, the chance that these papers will be recommended is higher due to the fact that the numbers of learners and papers are limited. Therefore the suggested number of co-rated papers can be better determined if more follow-up experiments can be performed.

Another interesting and important issue should also be studied: how learner ratings on aspects other than the single overall rating of each paper can influence user choice? In the experiments discussed in this thesis, we have been able to establish some interesting insights into this issue, for example, that besides learner interest, contextual information-seeking goals such as task- and course-related goals are related to learners' perceived value of the papers and learners' willingness to recommend a paper to peers depends largely on the closeness of its content topic to their job nature. However, these patterns were established based on correlations in the data we obtained. Hence, a more extensive study is needed to pin down how new learners' perception of

the paper quality has been affected by their knowledge of the paper's previous rating in these dimensions. This issue is important, since it measures the quality of a RS in terms of how it has been able to support learners' tasks when they engage in the course (for example, to like the paper because it is related to their team project). And the measurement will be more appropriate because it might reveal how a paper recommender can help users complete their tasks.

In spite of these weaknesses, we believe that our study has been able to reach the research goals we set when we proposed the pedagogical paper recommender which unfortunately has been largely ignored in the research community. In particular, the evaluations that have been conducted lead us to believe that the recommender system can support learners' decisions reasonably soundly. In fact, learners' feedbacks toward these reading articles are very positive. Furthermore, senior-level undergraduate students[3] may perceive quite differently from graduate students (especially those with working experience); therefore, their ratings of these papers could be quite different. It would be interesting to look into the performance of the system with undergraduate students and compare the degree of user satisfaction between the two user groups. In spite of this, the system prototype and the evaluation studies described in Chapters 4 and 5 suggest that learner satisfaction is a complicated function of learner characteristics, rather than just matching a paper against learner interest.

Obviously, finding a 'good' paper is not trivial: it is not about the simple fact that the user will either accept the recommended items, or not; rather, it is a multiple-step process that typically entails the users navigating the paper collection, understanding the recommended items, seeing what others like/dislike, and making decisions. Therefore, a future research goal to

[3] According to the syllabus for undergraduate students in Hong Kong Polytechnic University, where the human-subject study was taken, the software engineering course is offered as a core subject for second-year students (but note that in Hong Kong, it only takes three years for students to graduate).

proceed from the study here is to design for different kinds of social navigation in order to study the impact on user behavior, and how over the time, user behavior feeds back to influence the system performance.

5.5.3 Efficiency of Experimental Replications

The purpose of our study is not only for graduate-level courses in software engineering. In the future we may generalize our approach to other courses and also for professionals, for instance, those working in the research department of a company. As more data are collected from various learners and papers, the scalability of the experimentation becomes an interesting issue, and some related questions have to be answered. For example, if we need to replicate the experiment in other courses or on a larger dataset from various courses, how complex or difficult is it to replicate the experiment? With a growing number of papers or learners, how can the experiment be scaled up? Also, if we want to perform automatic tuning of the suggested parameters, how is it possible to realize such automation? In this subsection, we will discuss some answers to these questions.

Firstly, when we have a large number of learners, then we may obtain groups of learners with similar backgrounds and rating patterns. In this case, we may first apply data clustering to group all learners; then we could treat each group (or stereotypes) as an individual user, so that the computation on CF-related methods could be reduced effectively. As a result, the recommendation made is actually group-oriented [O'Connor et al. 2001]. In this approach, two types of methods are layered: in the first layer, we use data clustering technique to pre-process the data by identifying learner groups; then in the second layer, our CF-related methods could be applied. It is interesting to note that in the case, those factors that we might need to consider, including paper popularity, paper difficulty, learner interest etc, will be computed within each

group. The computation could be a simple mathematical average of these ratings (i.e. mean), or more sophisticated selection by finding the most representative user(s).

Secondly, when more rounds of experiments have been done, some recommendation methods may be more frequently effective than others; hence we suggest to dispose of those weak methods that may less frequently attain good outcomes, say, Pop1D, pop3D, and 5D-CF. Also, if a computationally costly method cannot show a significant improvement over less costly ones, then we may discard it as well. So, the efforts of replication on a larger dataset in the future might be reduced. Specifically, based on our study, we suggest that in the next round of experiments, only 1D-CF, PopCon2D, PopUM-CF and Pop3D should be retained.

Finally, variable tuning is achieved both semi-automatically and manually (Section 4.2) based on our pre-experiment studies documented in Chapter 3; in future experiments, it is desirable to automate the tunings by applying rule-based mechanisms based on our suggestions described in Chapter 4 and this chapter according to various possible learning scenarios. We predict that after performing more experiments, we might also be able to make suggestions as to the variable values and the recommendation methods in this domain, just as those suggested by Herlocker et al. [1999] in traditional RSs and have been used ever since they were proposed.

CHAPTER 6
CONCLUSIONS

Because of the uniqueness of the learning domain, especially social learning environments, paper recommendation in e-learning environments should be achieved through a careful assessment and comparison of both learner and paper characteristics. In other words, each individual learner model will first be created, in terms of learner interests and other pedagogical features, such as learners' background knowledge in specific topics. Paper models will also be created based on the topic, degree of peer recommendation, etc. Some techniques are carried out by matching the learner interest with the paper topics where the technical level of the paper should not impede the learner in understanding it. Therefore, the suitability of a paper toward a learner is calculated as the combination of the learner interest toward the paper and the appropriateness of it to help the learner in general, that is, its pedagogical impact.

6.1 Lessons Learned

6.1.1 General Comments on the Experimental Results

The experimental results are encouraging, especially in confirming that making recommendations to learners in social learning environments is not the same as making recommendations to users in commercial environments such as Amazon.com. In such learning environments, learners are willing to accept items that are not interesting, yet meet their learning goals in some way or another. For instance, our experimental results suggest that user-model based CF works well with content-based filtering and non-personalized methods (such as paper

popularity) as in PopCon2D and Pop5D. Although the computation in Pop5D is more sophisticated than the other CF-based approaches, we speculate that under certain circumstances it helps inform the recommender and therefore, improve the recommendations. The results also indicate that incorporating ratings from value-added-ness and peer recommendation can slightly improve the performance of CF-based RSs when the number of co-rated papers is small. Among the proposed multi-dimensional methods, 3D-CF demonstrated the most potential in terms of average Overall rating when the number of co-rated papers is small.

In order to further our understanding of these factors, we used statistical analysis methods to uncover which factors are correlated with successful recommendation (as described in section 3.4). These objective evaluation methodologies are appropriate for our domain and essential to support our claims. The evaluation framework is based on the task the system is intended to support (as emphasized in [Herlocker et al. 2004, McNee et al. 2006a., Terveen and McDonald 2005]), that is, to help learners pedagogically. Our findings illuminate learner satisfaction as a complicated function of learner characteristics, rather than the single issue of whether the paper topics matched learner interests.

In general, PopCon2D performs very well under almost half of typical learning scenarios for picking the best one paper, and the more complex CF algorithms including Pop5D, PopUM-CF work well for making the best three recommendations. Due to its characteristics, PopCon2D not only can be used to start up the recommendation but also to inform the recommendation. The results lead us to speculate that if in the domain that there are limited number of both papers and learners, considering other features rather than relying on Overall rating and user interest can help inform the recommendation.

In addition, it is observed that most of the recommendation methods are scalable with respect to the gain/loss relative to the best-case benchmark, at least as we move from 11 to 21 papers. Although it is still unclear whether the scalability continues as we move to bigger paper repositories, we conjecture that the methods may be consistent to up to 50 papers in terms of recommendation accuracy. Of course, the number of learners, the testing period, and the number of papers were limited; and the full space of user evaluation is certainly complex. When there are not enough user-ratings in the system, PopCon2D is suggested to recommend 1 paper. A set of the most appropriate parameters can be selected by trying various sets of possible parameters given available ratings in the system maintenance stage.

6.1.2 Issues and Challenges in the Design of Pedagogical Recommender Systems

Due to the limited number of students, papers and other learning restrictions, a tutor cannot require students to read too many papers in order to stock the database. As such, the majority of typical recommender systems might not work well in the pedagogical domain. This thesis attempts to bridge the gap by proposing a set of recommendation mechanisms that work well in the domain. Through experimental studies and prototypical analysis, we draw a number of important conclusions regarding the design and evaluation of these techniques in our domain. In spite of it, there need more works to further our understanding of it.

For instance, we realized that one of the biggest challenges is the difficulty to test the effectiveness or appropriateness of a recommendation method due to a low number of available ratings. Testing the method with more students, say, in two or three more semesters, may not be helpful, because the results are still not enough to draw conclusions as strong as those from other domains where the ratings can be as many as millions.

Hence, we are eager to see the collaborations from different institutions in using the system in a more distributed and larger-scale fashion (as it is very difficult to achieve it in using

one class each time and in one institution). Through this broader collaboration, our future work includes the design of a MovieLens-like benchmark database as a test-bed on which more algorithms can be tested (including ours).

Meanwhile, in those environments where a wider range of learning scenarios exist (such as those simulated in Chapter 5), evaluations can be performed in a more systematic way; for instance, to continuously ask students to engage in 'on-line re-evaluation' as described in Section 5.3.4. Or, we can also perform more task-oriented evaluations, such as to evaluate the appropriateness of recommendations through a group-oriented task where learners will be asked design and implement an automatic tool for, say, a family calendar. In this case, learners would be required to write a research-paper[1] documenting their experiences as well as lessons learned from the task. Both after-project-questionnaire and analysis of the report could help to evaluate the actual 'uses' of the recommended papers for learning. Another task can be asking learners to conduct a research study on a topic of their choice in three weeks, say, the adoption of a CASE tool for software testing, and present their work in front of the class. In another example where multiple institutions are involved, we may divide students into groups from different institutions (or even different countries), and each group is then required to undertake a distributed software project, where a number of CASE tools can be used to coordinate their actions. In this case, reference articles can be supplied weekly as in our settings, though, more papers can be included. Then, by the end of their project, learners will be asked to not only turn in a workable system, but also document what they have learned in a research paper.

[1] This type of the project is typical for senior-level and graduate level courses. In fact, in the semester where the data for this thesis was collected, similar group projects were adopted, and learners self-reported that they made use of the papers recommended to them either as direct references or indirect ones to find more papers. For instance, several learners used Papers 16 and 17 as guidelines to conduct heuristic evaluation on the chosen system and to document their findings in their reports.

Then, in each of these tasks, learners will be asked to evaluate the pedagogical usefulness of the recommended papers as to, among other things:

- how the knowledge they learned can be used to guide their joint software development effort,
- how the papers can be used as 'seed' ones to help them 'Google' more.

We believe that the task-oriented evaluation framework is more appropriate than purely presenting the MAE, ROC, precision, recall, etc., measures in directly assessing user satisfaction with and acceptance of the recommended items. In spite of this, we think that the factors we have considered so far (e.g. interestingness, value-addedness, etc.) represent the most typical factors that need to be taken into consideration when making recommendations in the domain.

In addition, as shown from the analysis with the prototype, the papers are related to software engineering (including user interface design and usability engineering); hence, it is not appropriate to generalize the results to make recommendation to students in other classes. Since in some subjects, papers may exhibit more technical difficulties due to their inherent features (e.g. in Artificial Intelligence or Data Mining), so are students who may also be different when they begin to take this course, which in turn affect those pedagogical factors considered in the performance of the recommender system. For instance, in user-model based CF, we intend to match user interest and background knowledge to a paper's topics and its difficulty in making recommendations to ensure that students have enough background knowledge to understand the paper. If, say, the recommendation is made to students taking an AI course, due to the nature of papers in this area, tutors should consider the technical difficulty of each paper carefully by setting a more significant weight on paper difficulty (in the SE course and due to the overall

nature of the papers, the weight is suggested to be 0 after performance comparisons) to reflect the importance of considering paper difficulty.

When the RS is to make a recommendation for users other than in a formal learning environment, say, for research users with different backgrounds in a company, we think that most of the proposed techniques will still be effective. That is, for those junior users, who might not have much research experience, similar pedagogical factors still need to be considered; however, for experienced users with rich research experiences, the typical content-based, CF-based techniques, and user-model-based CF should work well. When there are limited numbers of ratings or papers, other factors such as paper popularity and peer-recommendation would still be useful to inform the RS to make decisions. As such, the underlying features of our proposed algorithms still stand.

The issues and conclusions we suggest can enlighten future studies on this topic, and further our understanding in making recommendations in this domain.

6.2 Implication of the Pedagogical Paper Recommender

In this thesis, we only investigate the recommendation of research articles. Nevertheless, we believe that the study can be extended for other educational resources, such as learning objects, chapters with different related topics in a digital book, tutorial materials, etc. In fact, almost all educational resources can be regarded as learning objects with different granularity, situated environments and purposes. Hence, the various recommendation mechanisms can be extended to make personalized recommendation of learning objects to individual learners of different needs.

However, when considering, especially, the reusability of learning objects, some other interesting yet challenging issues arise. When both the number of learners and learning objects grow, data pre-processing should be performed, for example, to pre-cluster the learners as well

as the learning objects based on their corresponding models, so as to make 'purposed' recommendations. While learners are represented by their pedagogical features, learning objects are annotated by multiple dimensional ratings including overall rating, degree of peer-recommendation, etc. [Tang and McCalla 2005a]. It is obvious that as more and more learners have read and rated an item, the amount of user feedback with respect to the item will accumulate. These data can be processed by intelligently mining patterns in the cross-product of page usage and user models (see Figure 6-1), and making recommendations accordingly [McCalla 2004]. For instance, patterns could be found such as learning objects that are highly rated by learners with a deep understanding of some knowledge, by learners with unusual tastes, etc. The sequence of a learner's interactions with various learning objects forms a "foot print" of that learner's activity in the system [Tang and McCalla 2006, 2007b].

The research paper recommender system has been one of the core inspirations for the ecological approach, the learner centered, adaptive, reactive and collaborative learning framework proposed in [McCalla 2004]. Specifically, in the ecological approach, it is assumed that there are a large number of learning objects (research papers, web learning resources, on-line quiz banks, etc.), and a number of different applications that support learners including learning object recommenders (similar to the paper recommender), collaborative activities such as reading, editing, expert and expert finders, and so on. When a learner interacts with a learning object, the object is 'annotated' with an instance of the learning model.

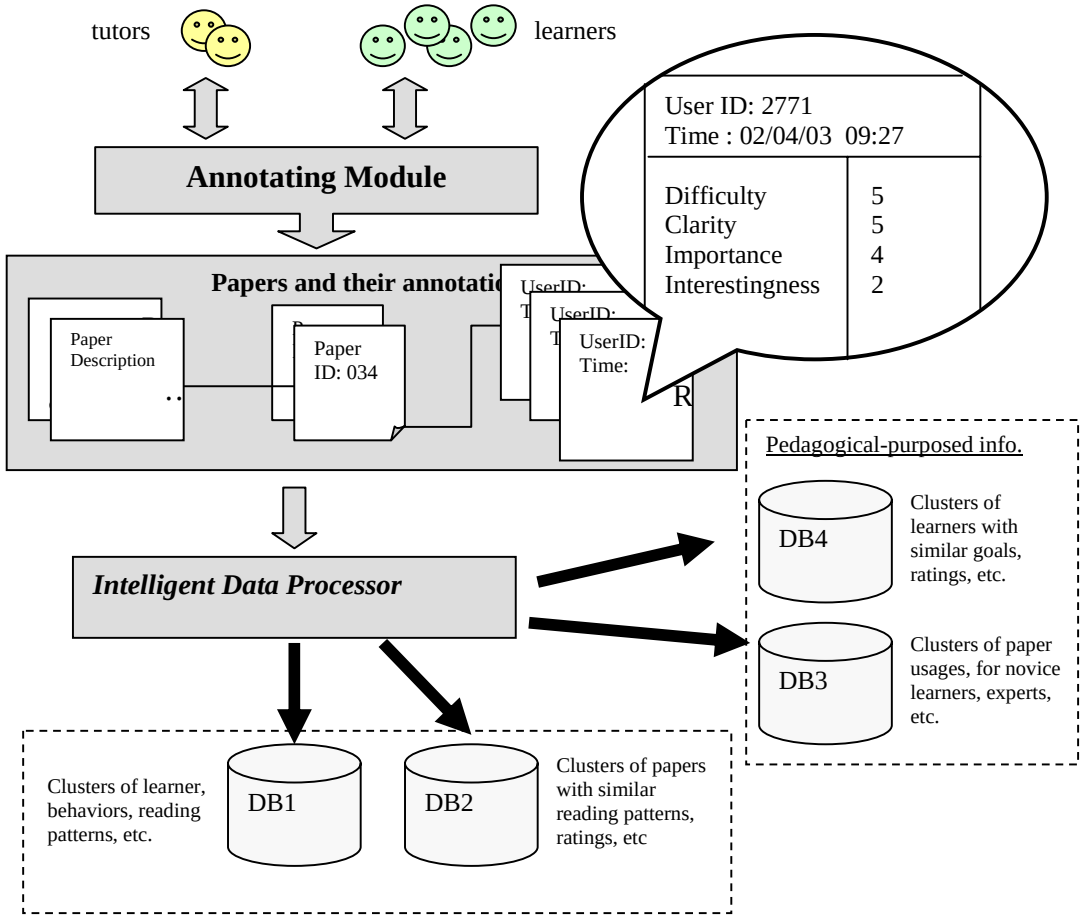


Figure 6-1 Paper annotations with temporal sequences of user models

After a learner has socialized with other learners, corresponding information can also be tagged. These tagged pieces of information embedded in a learner model instance can include:

- features about the learner, including cognitive, and social characteristics and most importantly their goal(s) in making the encounters;
- learner feedback on the information content of the learning object, including its information and cognitive quality. For instance, the information quality such as the content accuracy, and up-to-date-ness; the cognitive quality such as the appropriateness of the object for learners

'like' him/her and the efficacy of the object with respect to their goal(s) in accessing the object;

- information about how the learner interacted with the content, including observed metrics such as dwell time, patterns of access, etc.;
- information about other learners with whom the learner has interacted, including their characteristics such as interests, goals, etc.

Over time, each learning object slowly accumulates learner model instances that collectively form a record of the experiences of all sorts of learners as they have interacted with the learning object as well as their various interactions.

In sum, then the ecological approach highlights the vision that information gradually accumulates about learning objects, the interactions, and the users. These pieces of information capture 'on-the-fly' information about various activities in the system and therefore should be interpreted in the context of end-use during the active learning modeling [Vassileva et al. 2003]. Gradually, through mechanisms like natural selection, the system will determine what information is useful and what is not, for which purpose.

6.3 Future Directions

Compared to music/book/movie recommendations, paper recommendation is a relatively new, interesting, yet potentially more challenging area due to many subtle human factors that go into making a pedagogically sound recommendation.

6.3.1 Further Experimental Studies of the Paper Recommender

The most immediate future study following this thesis is to refine and implement the prototype discussed in Chapter 5 and deploy it in more courses used by learners/tutors. Since the experiments conducted so far and suggestions made on the adoption of recommendation methods are based on the data we have collected during a one semester course, it is unclear whether these

methods are still suitable when user ratings keep increasing. Therefore, follow up studies should be performed (ideally) in two semesters (on the same course or related courses) with more newly added papers. The system will then make recommendations based on existing paper and user models. Interesting issues that could be studied include:

- *The weights with respect to some key pedagogical features in CF and hybrid approaches*

  In the multi-dimensional CF, the weighted sum Pearson correlation is formulated as:

  $$P_{3D}(a, b) = w_{overall}\, P_{overall}(a, b) + w_{valueadd}\ P_{valueadd}(a, b) + w_{peer\_rec}\, P_{peer\_rec}(a, b) \qquad (4\text{-}3)$$

where $w_{overall} + w_{valueadd} + w_{peer\_rec} = 1$. The weights assigned to the three features $w_{overall}$, $w_{valueadd}$, w_{peer_rec} reflect the importance of incorporating the corresponding feature in the computation. For example, when $w_{valueadd}$ and w_{peer_rec} are set 0, the multiple dimensional CF becomes the typical CF; in other words, we only consider a paper's overall rating to make recommendations. In our experimental study, these weights are tuned manually following a series of trials, in which those weights reported in this thesis are representative ones: a paper's overall rating has higher impact than the other two factors since no matter what kind of pedagogical goals there are, keeping learners interested in what they are learning remains a most important goal in the education domain. When there are more learner ratings, it would be necessary to make adjustments to the values, initially through manual tuning; over time, automatic tuning is desirable.

- *The number of co-rated papers in CF-based approaches*

Using existing limited data, we found out that a co-rated paper in rating-based CF methods may be recommended back to learners. Hence, using a large number of co-rated papers increases the chance of recommending a highly rated paper included in the co-rated papers, which could lead us to wrongly believe that the recommender system yields satisfying results. Therefore, in our experiment, we suggest the number of co-rated papers be small, i.e. 2 or 4.

Since CF relies on co-rated papers to identify neighbors for a target user, therefore, using such a small number of co-rated papers could reduce the quality of recommendations when there are much more data. However, it is unlikely for the system to collect a large amount of data, since it is not feasible to assign learners to read many papers in one semester; and the class size is usually restricted to tens or hundreds only. As such, it would be interesting to tune this value to see its effect on recommendation under this condition.

- *The number of neighbors in CF-based approaches*

Recommendation systems must rely on the ratings candidate users give to find the closest neighbors: the closer the candidate user and the target user have agreed on a paper, the higher the chance that the candidate user is the neighbor. Therefore, the key step is neighborhood forming; that is, identifying quality neighbors for the target user. Through experimental trials, we found out that when the number of neighbors N is 10, the recommender generates relatively satisfying results; therefore, it is suggested and used in our rating-based CF methods. For recommendations in other domains such as movies, Herlocker et al. suggest that the number of neighbors be 30 [Herlocker et al. 1999, Herlocker and Konstan 2002] which is adopted in the community. Note here though, in the movie domain, the benchmark database is MovieLens, a much denser database consisting of almost 1 million ratings. When the user-ratings grow larger, and denser, it is not known whether CF-based approaches with N=10 would still lead to satisfying results.

- *The effect of different groups of learners*

From the experimental study, one conclusion that can be drawn is that graduate students are willing to learn something that is less interesting to them, which is quite different from the users in other domains, such as music, book, movies etc. However, since the experiments were conducted in one graduate-level software engineering course, we are not sure whether or not this

conclusion can be generalized to other graduate courses, or graduate students in other universities. Further, senior-level undergraduate students without working experience may be different from graduate students with working experience, and thus, their feedback about these papers could be different. Hence, it would be interesting to look into the performance of the system using undergraduate students and compare the degree of user satisfaction between the two groups, which can in turn help us further our understanding of two different yet closely related communities.

- *The effect of paper popularity in content-based and hybrid approaches*

Paper popularity is the average rating of a paper by a group of learners. In our experimental study, since there are only 25 learners, we treat them as one group even though there are some subtle differences among these learners. If we break these learners down into different groups, the data would be far from enough to execute group-oriented recommendations generated from applying data clustering methods. Therefore, the paper popularity used in both content-based and hybrid approaches in this thesis reflects the views from a specific group of learners; it is unknown how the recommendation algorithms should be adjusted when the system is deployed to undergraduate students largely without working experience.

Another interesting thing we might see is that over time, some algorithms might not be needed at all if their performance is not satisfactory, or they are computationally too costly. In particular, we suggest that in the next round of experiments, only 1D-CF, PopCon2D, PopUM-CF and Pop3D should be retained.

6.3.2 Study the 'Social' Aspect of the Paper Recommender

6.3.2.1 Social Affordance

Since the first automated rating-based recommender system (RS) was proposed in 1994 [Resnick et al. 1994], the accuracy of RSs remained the ultimate goal in the research literature

until early 2004 when several researchers began to explore other goals for RSs [Herlocker 2004; McNee et al. 2006a; Riedl and Dourish 2005; Adomavicius and Tuzhilin 2005], among them:

- to design a human-recommender framework and methodology to understand users, their tasks, and their interactions with the system [McNee et al. 2006a]
- to find more interactive ways to understand user needs and make the recommendations transparent to the users [Adomavicius and Tuzhilin 2005]
- to prevent RSs from user manipulation [Lam et al. 2006, Lam and Riedl 2004]

In the e-learning domain, one of the foci is to provide exciting new capabilities to the users of the RSs to interact with the recommended items. In particular, it is important in learning environments to not only allow learners to determine the reason those recommended items were recommended, but also to allow them to interact with these items. One possibility is to allow learners to browse the annotated learner models attached to each item (paper annotation with learner models [Tang and McCalla 2005a]) so they can see, for example, what type of users tend to like particular items and therefore determine whether or not they will accept the recommended items, or choose other items on their own.

This puts the focus more on the *social affordance* of a paper in the social learning environment and highlights the paper's social end-use context [McCalla 2004] rather than returning the most accurate paper recommendations. According to [Kreijins et al. 2002], *social affordance* in a CSCL environment refers to any elements of the learning design and environment that would help facilitate students' learning [Kreijins et al. 2002]. They further argue that '*a good set of social affordances will establish the desired sound social space that is characterized by an effective structure, trust and belonging*…" (page 7, [Kreijins et al. 2002]). Social affordance of an item in a CSCL environment can allow users to perceive their

environments (which include the items, other users, as well as the system), formulate their actions, and navigate in the information space accordingly. One useful way of enhancing social affordance is by making social interaction visible. For instance, [Mazza and Dimitrova 2004] have focused on monitoring activities in discussion boards, performance on quizzes as well as content access patterns. [Reffay and Chanier 2002] have implemented a Distance Learning Management System (DLMS) that can identify clusters and cliques among the interactions. They focus on the activity of the class as a whole, not on the individual student. [Brooks et al. 2007] have used socio-diagrams to visualize users' activities in order to capture their socialization and interactions within an asynchronous forum. These visualization approaches can help learners spot chances to start communication and collaboration, and allow instructors and other teaching staff to scaffold the learning content.

6.3.2.2 Social Affordance in a Pedagogical Paper Recommender

Determining how the multi-dimensions of an educational resource can be made visually and interactively available to its learners can help us answer more questions raised in this thesis. In the Pedagogical Paper Recommender, a paper is recommended based on a variety of factors that a learner provides in terms of not only the paper's overall rating as to topic appropriateness, but also pedagogical values (situated factors) such as the usefulness of the paper to the learner's current task, the appropriateness of the information content etc. When the system allows users to visualize and understand these aspects of a paper, it, in fact, creates a rich space for learners to interact with the system, and each other. For instance, it can help raise the awareness of a learner towards the candidate papers or provide an opportunity for the learner to socialize with others through initiating discussions. It can also allow the users to spot trends in popular topics, as in

PaperLens [Lee et al. 2005]. To further illustrate our ideas, Figure 6-2 illustrates some possible ways to manifest a paper's social affordance.

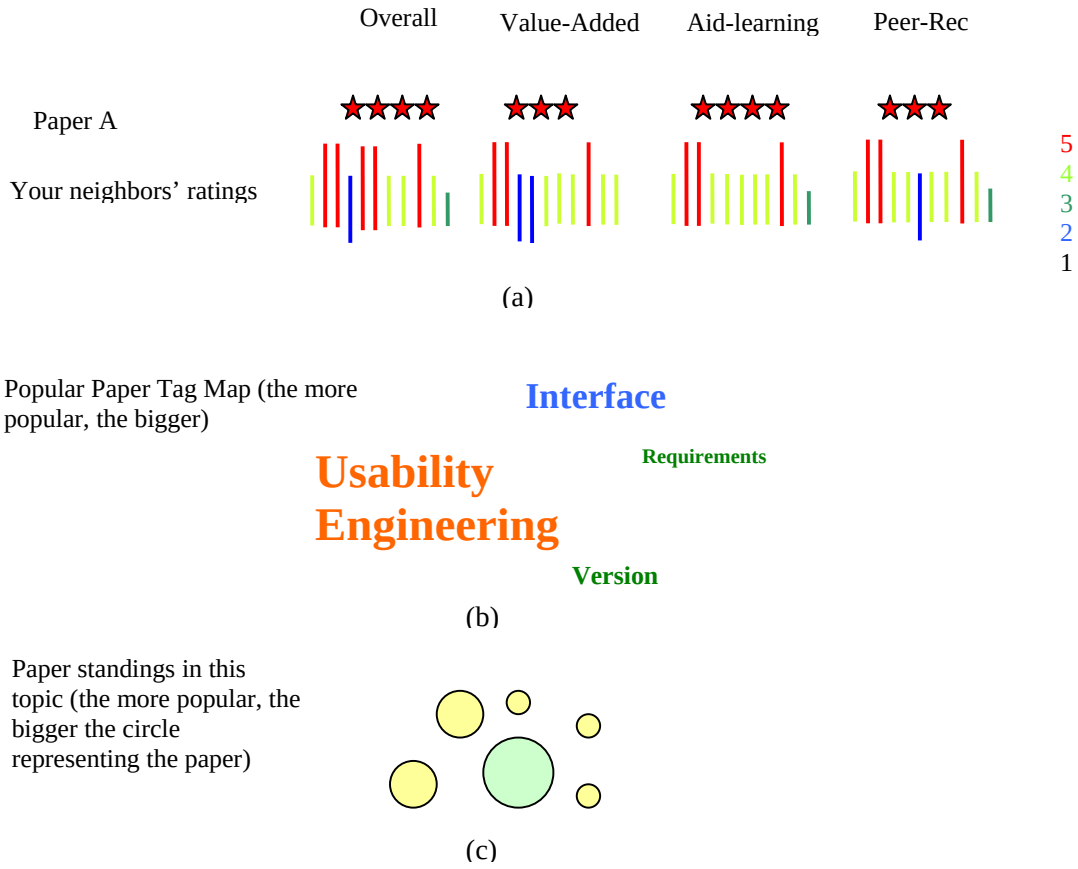


Figure 6-2 The social affordances of a paper [Tang and McCalla 2007b]

These methods can be designed to create a social space for learners to interact with the system (Figure 6-2):

- paper A's ratings (given by the learner) not only include the commonly used overall rating, but also its ratings along other dimensions such as value-added, aid-learning and peer-recommendation as well as the learner's neighbors' ratings along these dimensions (Figure 6-2 (a));

- a popular paper tag map is capable of making the most well-accepted topic (s) visible (Figure 6-2 (b)) in a group;
- the paper's position among the pool of similar papers measured on the dimensions that users have rated (Figure 6-2 (c)), so, say, learners in a group can understand which paper is the most useful.

These visualization alternatives can clearly reveal the social features of a paper in relation to both learners and other papers. The main goal is to investigate how one user's behavior and tastes can be influenced by watching other users' likes and dislikes. It is known that people like to watch what others are doing. We hypothesize that a user's taste might be changed when looking over another's shoulder [Barry 1994]; hence, it might not be surprising that there are gaps between users' pick-up and system-recommended items. Through visualizing each paper's social standing, the system can raise the awareness among learners by offering a highly flexible social affordance on each paper (for instance, a learner can see how the paper has been perceived in this group, or among different groups since in another group the same paper might not be so popular), and meanwhile, allow the users to perceive their own affordances when working with the document. These can also help the system to draw all manner of inferences about the papers, the learners and their relationships.

6.4 Concluding Remarks

In conclusion, the focus of current RS research has shifted from the algorithmic design of the recommendation mechanisms to the social aspects of the system (thus the term 'human-recommender interaction' [McNee et al. 2006b]), including preserving the privacy and security of the recommendations [Lam and Riedl 2004, Lam et al. 2006, Mobasher et al. 2007], preventing the RS from user manipulation [Mobasher et al. 2007, Lam and Riedl 2004, 2005],

the HCI aspect of RS, such as the usability of the system and transparency of the recommendations made [Puolamaki et al. 2005, Sinha and Swearingen 2002, McNee et al. 2006a, 2006b], a cross-domain recommender system making computations based on user preferences across domains such as movies, books, TV shows [Winoto and Tang 2008] etc.

Finding a 'good' paper is not trivial: it is not about the simple fact that the user will either accept the paper(s) or not. Rather, it is a multiple-step process that typically entails the users navigating the paper collections, understanding the recommended items, seeing what others like/dislike, and making decisions.

In addition, the pre-requisite structure underlying most courses, if combined with the pedagogical paper recommendation approaches might better help in predicting which papers would be useful and suitable to the current topic. For instance, in a syllabus, it is stated that the students have taken an introductory software engineering course and a programming language course as a pre-requisite; in other words, the knowledge background of students is known.

It is our hope that the studies we initiated here can open up opportunities for researchers to probe into the use of automated social tools to support active learning and teaching in future networked learning environments [McCalla 2000] where '*the flow of knowledge will be governed by the speed of human to human interaction*' (p. 179) as well as human to system interaction.

REFERENCES

1. Adomavicius, G., Sankarayanan, R., Sen, S. and Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems*, 23 (1): 103-145. January 2005.
2. Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extension. *IEEE Transactions on Knowledge and Data Engineering*, 17(6): 734-749, June 2005.
3. Aimeur, E., Onana, F. S. M. and Saleman, A. (2007). HELP: a recommender system to locate expertise in organizational memories. In *Proc. of IEEE/ACS International Conference on Computer Systems and Applications (AICCSA '07)*, Amman, Jordan , 866-874.
4. Baeza-Yates, R. and Ribeiro-Neto, B. 1999. *Modern information retrieval*. Addison-Wesley, 1999.
5. Balabanovic, M. and Shoham, Y. (1997). Fab: content-based collaborative recommendation. *Communications of the ACM*, 40(3): 66-72. 1997.
6. Baldi, P., Frasconi, P., Smyth, P. (2003). *Modeling the Internet and the Web: probabilistic methods and algorithms*. John Wiley & Sons. 2003.
7. Barry, C. (1994). User-defined relevance criteria: an exploratory study. *Journal of the American Society for Information Science (JASIS)*, 149-159, April 1994.
8. Basu, C., Hirsh, H. and Cohen, W.W. (1998). Recommendation as classification: using social and content-based information in recommendation. In *Proc. of the 15th National Conference on Artificial Intelligence (AAAI'98)*, Madison, Wisconsin, USA, 714-720.
9. Basu, C., Hirsh, H., Cohen, W. and Nevill-Manning, C. (2001). Technical paper recommendations: a study in combining multiple information sources. *Journal of Artificial Intelligence Research* (*JAIR*), 1: 231-252.
10. Belkin, N. and Croft, B. (1992). Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12): 29-37, 1992.
11. Bederson, B., Clamage, A., Czerwinski, M. P. and Robertson, G. G. (2004). DateLens: a fisheye calendar interface for PDAs. *ACM Interactions*, July+August 2004, 90-119.
12. Bollacker, K., Lawrence, S. and C. Lee Giles, C. L. 1999). A system for automatic personalized tracking of scientific literature on the web. In *Proc. Of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'1999)*, Berkeley, CA, USA, 105-113.
13. Blalock, H. (1961). *Causal Inferences in Nonexperimental Research*. Chapel Hill, NC: UNC Press.
14. Bradshaw, S., Scheinkman, a. and Hammond, K. (2000). Guiding people to information: providing an interface to a digital library using reference as a basis for indexing. In *Proc. of 4th ACM International Conference on Intelligent User Interface* (*IUI'00)*, New Orleans, LA, USA, 37-43.

15. Breese, J.S., Heckerman, D., Kadie, C., (1998). Empirical analysis of predictive algorithms for collaborative filtering. *Proc. of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI'98)*, Madison, Wisconsin, USA, 43–52.

16. Brooks, C., Liu, W.G., Hansen, C., McCalla, G. and Greer, J. (2007) Making sense of complex learner data. In *Workshop of Assessment of Group and Individual Learning Through. Intelligent Visualization (AGILEeViz), 13th International Conference on Artificial Intelligence in Education (AIED 2007).* Marina Del Rey, LA, USA, 28-33.
http://aied.inf.ed.ac.uk/AIED2007/AgileVizWorkshopAIED2007.pdf

17. Browne, M. W. and Cudeck, R. (1993). Alternative ways of assessing model fit. In K. A. Bollen and J. S. Long (Eds.) *Testing Structural Equation Models*. Sage: Newbury Park, CA.

18. Brusilovsky, P., Farzan, R. and Ahn, J. (2005). Comprehensive personalized information access in an educational digital library. In *Proc. Of IEEE/ACM Joint Conference on Digital Libraries (ACM DL'2005)*, Denver, CA, USA, 9-18.

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

20. Cadiz, J., Gupta, A., and Grudin, J. (2000). Using web annotations for asynchronous collaborative around documents. In *Proc. Of ACM Conference on Computer Supported Collaborative Work* (*CSCW'00)*, Philadelphia, PA, USA, 309-318.

21. Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. *ACM SIGIR Workshop on Recommender Systems*, Berkeley, CA, USA, 1999.

22. Claypool, M., Le, P., Waseda, M., and Brown, D. (2001). Implicit Interest Indicators, In *Proceedings of ACM International Conference on Intelligent User Interfaces (IUI'2001,* Santa Fe, New Mexico USA, 33-40.

23. Cosley, D., Lam, S., Albert, I., Konstan, J. and John Riedl. (2003). Is seeing believing?: how recommender system interfaces affect users' opinions. In *Proc. of ACM Conference on Human Factors in Computing Systems (CHI'2003),* Fort Lauderdale, Florida, USA, 585-592.

24. Cosley, D., Lawrence, S., and Pennock, D. (2002). REFEREE: An Open Framework for Practical Testing of Recommender Systems using ResearchIndex. In *Proc.* of *International Conference on Very Large Data Bases (VLDB'2002)*, Hong Kong, China, 35-46.

25. Crestani, F, Lalmas, M., van Rijsbergen, C. and Campbell, I. (1998). "Is this document relevant?...probably": a survey of probabilistic models in information retrieval. *ACM Computing Surveys*, 30(4): 529-552, December 1998.

26. Cox, D. and Greenberg, S. (2000). Supporting collaborative interpretation in distributed groupware. (2000). In *Proc. Of ACM Conference on Computer Supported Collaborative Work* (*CSCW'00)*, Philadelphia, PA, USA, 289-298.

27. Custard, M., and Sumner, T. (2005). Using machine learning to support quality judgment. *D-Lib Magazine*, Oct. 2005, 11(10). available at: http://www.dlib.org/dlib/october05/custard/10custard.html

28. Dahlen, B.J., Konstan, J., Herlocker, J., Good, N., Borchers, A., and Riedl, J. (1998). Jump-starting MovieLens: user benefits of starting a collaborative filtering system with 'dead data', *Technical Report, TR 98-017*. Department of Computer Science, University of Minnesota.

29. Davis, J. and Huttenlocher, D. (1995). Shared annotation for cooperative learning. In *Proc. Of Computer Supported Collaborative Learning (CSCL'95)*. Bloomington, Indiana, USA, 84-88.

30. Deshpande, M and Karypis, G. (2004). Item-based Top-n recommendation algorithms. *ACM Trans. Information Systems*, 22(1): 143-177.

31. Esmaili, K.S., Neshati, M., Jamali, M., Abolhassani, H. and Habibi, J. (2006). Comparing performance of recommendation techniques in the blogsphere. *European Conference on Artificial Intelligence' 2006 Workshop on Recommender Systems*. Riva del Garda, Italy.

32. Garfield E., (1995). Citation indexes for science: A new dimension in documentation through association of ideas. *Science*, 122: 108-111.

33. Good, N., Schafer, J.B., Konstan, J.A., Borchers, A., Sarwar, B., Herlocker, J. and Riedl, J. (1999). Combining collaborative filtering with personal agents for better recommendations. In *Proc. of the 16th National Conference on Artificial Intelligence (AAAI'99)*, Orlando, Florida, USA, 439-446.

34. Grant, S. and McCalla, G. I. (2001) A hybrid approach to making recommendations and its application to the movie domain. In *Proceedings of 2001 Canadian Artificial Intelligence Conference*. Stroulia, E. and Matwin, S.(eds) LNAI 2056. 257-266

35. Han, H., Giles, C.L., Manavoglu, E. and Zha, H. (2003). Automatic document metadata extraction using support vector machines. In *Proc. of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'03)*, Houston, Texas, USA, 37-48.

36. Hanley, J. and Mcneil, B. (1982). The meaning and use of the area under a receiver operating characteristics (ROC) curve. *Radiology*, 143: 29-36.

37. Herlocker, J. (2000). *Understanding and improving automated collaborative filtering systems*. PhD thesis, University of Minnesota.

38. Herlocker, J. and Konstan, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, 5, 287–310, 2002.

39. Herlocker, J., Konstan, J, Terveen, L. And Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*. 22(1): 5-53, January 2004.

40. Herlocker, J., Konstan, J., and Riedl, J. (2000). Explaining collaborative filtering recommendations. In *Proc. of the 2000 Conference on Computer Supported Coperative Work* (*CSCW'2000*), Philadelphia, PA, USA, 241-250.

41. Herlocker, J., Konstan, J., Borchers, A., and Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proc. of the 22nd annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, Berkeley, CA, USA, 230-237.

42. Hill, W., Stead, L., Rosenstein, M. and Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use. In *Proc. of ACM Conference on Human Factors in Computing Systems (ACM CHI'95)*, Denver, Colorado, 194-201.
43. Hofmann, T. and Puzicha, J. (1999). Latent class models for collaborative filtering. In *Proc. of International Joint Conference on Artificial Intelligence* (*IJCAI'99)*, Stockholm, Sweden, 688-693.
44. Hofmann, T., Puzicha, J. and Jordan, M.I. (1999). Learning from dyadic data. In *Proc. of the 1998 Conference on Advances in Neural Information Processing Systems II(NIPS'99)*, Denver, CO, USA, 466-472.
45. Hook, K, Rudstrom, A and Waern, A. (1997) Edited adaptive hypermedia: combining human and machine intelligence to achieve filtered information. In *Workshop on Flexible Hypertext,* Apr. 1997.
46. Huang, Z., Chung, W., Ong, T. and Chen, H. (2002). A graph-based recommender system for digital library. In *Proc. of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'*2002), Portland, Oregon, USA, 65-73.
47. Jansen, B.J., Spink, A., Bateman, J., Saracevic, T. (1998). Searhers, the subjects they search and sufficiency: a study of a large sample of Excite searches. In *Proc. of World Conference of the WWW, Internet and Intranet* (*Webnet'98)*. Orlando, Florida, USA, 472-477.
48. Jones, S., Cunningham, S.J., and McNab, R. (1998). An analysis of usage of a digital library. In *Proc. of European Conference on Digital Library* (*ECDL'98)*. Heraklion, Crete, Greece, vol. 1513, 261-277.
49. Kelloway, E. K. (1998). *Using LISREL for Structural Equation Modeling: A Researcher's Guide*, Sage.
50. Khoo, N.K. and Denise J. J. Chen, D. (1995).The evolution of intelligent agent and game theory: towards the future of intelligent automation. Surprise 95 Intelligent Agents and Games. Final Report, 1995.
51. Kobyashi, M. and Takeda, K. (2000). Information retrieval on the web. *ACM Computing Surveys*, 32(2): 144-173, June 2000.
52. Konstan, J., Miller, B.N., Maltz, D., Herlocker, J., Gordon, L.R. and Riedl, J. (1997). GroupLens: applying collaborative filtering to Usenet news. *Communications of the ACM,* 40(3): 77-87, March 1997.
53. Klobas, J. E. (1995). Beyond information quality: fitness for purpose and electronic information resource use. *Journal of Information Science*, 21(2): 95-114.
54. Kreijns, K., Kirschner, P.A. and Jochems, W. (2002). The sociability of computer-supported collaborative learning environments. *Journal of Education Technology and Society*, 5(1): 8-22.
55. Krulwich, B. (1997). LIFESTYLE FINDER: Intelligent user profiling using large-scale demographic data. *AI Magazine*, 18(2): 37-45.
56. Lam, S.K. and Riedl, J. (2004). Shilling recommender systems for fun and profit. In *Proc. of the 13th International Conference on World Wide Web (WWW2004),* New York, NY, USA, 393-402.

57. Lam, S.K., Frankowski, D., and Riedl, J. (2006). Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In *Proceedings of the 2006 International Conference on Emerging Trends in Information and Communication Security (ETRICS)*, Freiburg, Germany, 2006.

58. Lawrence, S., Giles, C. L., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6): 67-71, 1999.

59. Lee, B., Czerwinski, M., Robertson, G. and Bederson, B. (2005). Understanding research trends in conferences using PaperLens. Proc. of *ACM Conference on Human Factors in Computing Systems (CHI 200*5), Portland, Oregon, USA, 1969-1972.

60. Lekakos, G. and Giaglis, G. (2004). A Lifestyle-based approach for delivering personalized advertisements in digital interactive television. *Journal of Computer-Mediated Communication*, 9(2).
Available at: http://jcmc.indiana.edu/vol9/issue2/giaglis.html

61. Lekakos, G. and Giaglis, G. (2006). Improving the prediction accuracy of recommendation algorithms: approaches anchored on human factors. *Interacting with Computers*, 18(3): 410-431, 2006.

62. Lilien, G. L., Kotler, P., and Moorthy, S. K. (1992). *Marketing Models*. Prentice Hall, 22-23.

63. Ling, K., Beenen, G., Ludford, P., Wang, X., Chang, K., Li, X., Cosley, D., Frankowski, D., Terveen, L., Rashid, A. M., Resnick, P., & Kraut, R. (2005). Using social psychology to motivate contributions to online communities. *Journal of Computer-Mediated Communication*, 10(4).
Available at http://jcmc.indiana.edu/vol10/issue4/ling.html

64. Jolliffe, I. T. (2002). Principal Component Analysis, Second Edition. Springer, New York.

65. Marshall, C. (1997). Annotation: from paper books to the digital library. In *Proc. of IEEE/ACM Joint Conference on Digital Libraries* (*ACM/IEEE JCDL'97)*, Philadelphia, PA, USA, 131-140.

66. Martin, F.J. and Torrens, M. (2006). Talk notes at *Recommenders*'06, Bilbao, USA.

67. McCalla, G.I. (2000). The fragmentation of culture, learning, teaching and technology: implications for the artificial intelligence in education research agenda in 2010. *Special Millennium Issue on AIED in 2010, Int. J. of Artificial Intelligence in Education*, 11: 177-196.

68. McCalla, G.I, Vassileva, J., Greer, J. and Bull, S. (2000). Active learner modeling. In *Proceedings of Intelligent Tutoring Systems (ITS'2000)*, Montreal, Canada. 53-62.

69. McCalla, G.I. (2004). The ecological approach to the design of e-learning environments: purpose-based capture and use of information about learners. *Journal of Interactive Media in Education (JIME)*, 1, 2004.

70. McCarthy, K., Reilly, J., McGinty, L., and Smyth, B. (2004). On the dynamic generation of compound critiques in conversational recommender systems. In *Proc. of the 3rd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems* (*AH'2004*), Eindhoven, the Netherlands, 176-184.

71. McCarthy, K., McGinty, L., Smyth, B., and Reilly, J. (2005). On the evaluation of dynamic critiquing: a large-scale user study. In *Proc. of the 20th National Conference on Artificial Intelligence* (*AAAI'05*), Pittsburgh, PA, USA, 535-540.

72. McGinty, L., and Smyth, B. (2003). On the role of diversity in conversational recommender systems. In *Proc. of the 5th International Conference on Case-Based Reasoning* (*ICCBR 2003*), Trondheim, Norway, 276-290.

73. McLaughlin, M. and Herlocker J. (2004). A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proc. of the 27th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'2004)*, Sheffield, UK, 329-336.

74. McNee, S., Lam, S.K., Konstan, J. and Riedl, J. (2001). Interfaces for eliciting new user preferences in recommender system. In *Proc. of the 9th International Conference on User Modeling* (*UM'2001)*, Sonthofen, Germany, 178-187.

75. McNee, S, Albert, I., Cosley, D., Gopalkrishnan, P., Lam, S., Rashid, A., Konstan, J. and Riedl, J. (2002). On the recommending of citations for research papers. In *Proc. Of ACM Conference on Computer Supported Collaborative Work* (*CSCW'02)*, New Orleans, Louisiana, USA, 116-125.

76. McNee, S., Riedl, J. and J.A. Konstan. (2006a). Being accurate is not enough: how accuracy metrics have hurt recommender systems. *In the Extended Abstracts of the 2006 ACM Conference on Human Factors in Computing Systems (CHI 200*6), Montreal, Canada, 1097-1101.

77. McNee, S., Riedl, J. and J.A. Konstan. (2006b). Making recommendations better: an analytic model for human-recommender interaction. In the *Extended Abstracts of the 2006 ACM Conference on Human Factors in Computing Systems (CHI 2006)*, Montreal, Canada, 1103-1108.

78. Melville, P., Mooney, R.J. and Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. In *Proc. of National Conference on Artificial Intelligence (AAAI'2002)*, Edmonton, Alberta, Canada, 187-192.

79. Middleton, S. E., Shadbolt, N. R. and De Roure, D. C. (2004). Ontological user profiling in recommender systems. *ACM Transactions on Information Systems*, 22(1): 54-88, January 2004.

80. Miller, B., Konstan, J. and Riedl, J. (2004). PocketLens: toward a personal recommender system. *ACM Transactions on Information Systems*, 22(3): 437-476, July 2004.

81. Mobasher, B., Burke, R., Bhaumik, R. and Williams, C. (2007). Towards trustworthy recommender systems: an analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology (ACM TOIT)*, 7(2): 1-41, May 2007.

82. Mooney, R.J. and Roy, L. (2000). Content-based book recommending using learning for text categorization. In *Proc. of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'2000)*, San Antonio, TX, USA, 195-204.

83. O'Connor, M., Cosley, D., Konstan, J. A. and Riedl, J. (2001). PolyLens: a recommender system for groups of users. In *Proc. of European Conference on Computer Supported Collaborative Work* (*ECSCW*'01), Bonn, Germany, 199-218.

84. O'Mahony, M. P. and Smyth, B. (2007). A recommender system for on-line course enrolment: an initial study. In *Proc. of ACM conference on Recommender Systems (RecSys'07)*, Minneapolis, Minnesota, USA, 133-136.

85. Paepcke, A., Garcia-Molina, H., Rodriguez-Mula, G., and Cho, J. (2000). Beyond document similarity: understanding value-based search and browsing technologies. *ACM SIGMOD Record*, 29(1): 80-92, March 2000.

86. Pazzani, M. (1999). A Framework for collaborative, content-based, and demographic filtering. *Artificial Intelligence Rev*., 393-408, December 1999.

87. Pazzani, M. and Billsus, D. (1997). Learning and revising user profiles: the identification of interesting web sites. *Machine Learning*, 27: 313-331.

88. Pazzani, M., Muramatsu, J. and Billsus, D. (1996). Syskill and Webert: Identifying interesting web sites. In *Proc. of National Conference on Artificial Intelligence (AAAI'96)*, Portland, Oregon, USA, 54-61.

89. Puolamaki, K., Salojarvi, J., Savia, E., Simola, J. And Kaski, S. (2005). Combining eye movements and collaborative filtering for proactive information retrieval. *Proc. of the 28th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'2005)*, Salvador, Bahia, Brazil, 146-153.

90. Rashid, A.M., Albert, I., Cosley, D., Lam, S.K., McNee, S., Konstan, J.A., & Riedl, J. (2002). Getting to know you: learning new user preferences in recommender systems. In *Proc. of the 2002 International Conference on Intelligent User Interfaces (IUI'02)*, Miami, Florida, USA, 127-134.

91. Rashid, A.M., Ling, K., Tassone, R.D., Resnick, P., Kraut, R. and Riedl, J. (2006). Motivating participation by displaying the value of contribution. In *ACM Conference on Human-Computer Interactions* (*CHI'2006*), Montréal, Québec, Canada, 955-958.

92. Rashid, A.M., Lam, T., Karypis, G. and Riedl, G. (2006). ClustKNN: A highly scalable hybrid model- and memory-based CF algorithm. In *Workshop on Web Mining and Web Usage Analysis, the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*WEBKDD'2006)*, Philadelphia, PA, USA.

93. Recker, M., Walker, A. and Lawless K. (2003). What do you recommend? Implementation and analyses of collaborative information filtering of web resources for education. *Instructional Science*, 31: 299-316.

94. Riedl, J., and Dourish, P. (2005). Introduction to the special section on recommender systems. *ACM Transactions on Computer-Human Interaction* (*ACM TOCHI*) 12(3): 371-373, September 2005.

95. Rieh, S. Y. and Belkin, N. J. (2000). Interaction on the web: scholars' judgment of information quality and cognitive authority. In *Proc. of the 63rd Annual Meeting of the American Society for Information Science*, Chicago, IL, USA, vol. 37, 25-38.

96. Rieh, S.Y. (2002). Judgment of information quality and cognitive authority in the web. *Journal of the American Society for Information Science and Technology*, 53(2): 145-161. January 2002.

97. Rucker, J. and Polanco, M.J. (1997). SiteSeer: personalized navigation of the web. *Communications of the ACM*, 40(3): 73-75.

98. Salton, G. (1963). Associative document retrieval techniques using bibliographic information. *Journal of the ACM*, 10(4): 440-457.

99. Salton, G. (1989). *Automatic text processing*. Addison-Wesley, 1989.

100. Sarwar, B., Konstan, J., Borchers, A., Herlocker, J., Miller, B. and Riedl, J. (1998). Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proc. of ACM Conference on Computer Supported Collaborative Work (ACM CSCW'98)*, Seattle, WA, USA, 345-354.

101. Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. In *Proc. of the 2nd ACM Conference on Electronic Commerce (EC'00)*, Minneapolis, Minnesota, USA, 285-295.

102. Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2001). Item-based collaborative filtering recommender algorithms. In *Proc. of ACM International World Wide Web Conference (WWW'01)*, Hong Kong, China, 285-295.

103. Schafer, J., Konstan, J. and Riedl, J. 2001. Electronic Commerce Recommender Applications. *Data Mining and Knowledge Discovery*, 5, (1/2, 2001), 115-152.

104. Schein, A., Popescul, A., Ungar, L.H. and Pennock, D. (2002). Methods and metrics for cold-start recommendations. In *Proc. of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02)*, Tampere, Finland, 253-260.

105. Schein, A., Popescul, A., Ungar, L.H. and Pennock, D. (2005). CROC: A new evaluation criterion for recommender systems. *Electronic Commerce Research, Special Issue: World Wide Web Electronic Commerce, Security and Privacy*, 5(1): 51-74, January 2005.

106. Shardanand, U., and Maes, P. (1995). Social information filtering: algorithms for automating 'word of mouth'. In *Proc. of ACM Conference on Human Factors in Computing Systems (ACM CHI'95)*, Denver, Colorado, USA, 210-217.

107. Shugan, S. M. (1980). The cost of thinking. *Journal of Consumer Research*. 7(2): 99-111.

108. Sinha, R, and Swearingen, K. (2002). The role of transparency in recommender systems. In *Proc. of ACM CHI 02 Conference on Human Factors in Computing Systems Conference Companion*. 830-831.

109. Small, H. (1973). Co-citation in the scientific literature: a new measure of the relationship between two documents. *Journal of the American Society for Information Science (JASIS)*. 24(4): 265-269.

110. Smyth, B., McGinty, L, Reilly, J. and McCarthy, K. Compound critiques for conversational recommender systems. (2004). In *Proc. of 2004 IEEE/WIC/ACM International Conference on Web Intelligence* (*WI' 2004*), Beijing, China, 141-151.

111. Smyth, B., McCarthy, K., Reilly, J., O'Sullivan, D., McGinty, L., and Wilson, D.C. (2005). Case studies in association rule mining for recommender systems. In *Proc. of the 2005 International Conference on Artificial Intelligence* (*ICAI-05*), Las Vegas, Nevada, USA, 27-30.

112. Spink, A., Jansen, B.J., Bateman, J. (1998). Users' searching behavior on the Excite web search engine. In *Proc. of World Conference of the WWW, Internet and Intranet (Webnet'98)*, Orlando, Florida, USA, 828-833.

113. Sumner, T., Khoo, M., Recker, M. And Marlino, M. (2003). Understanding educator's perceptions of "quality" in digital libraries. In *Proc. of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'2003)*, Houston, Texas, USA, 269-279.

114. Svensson, M., Hook, K., and Coster, R. (2005). Designing and evaluating Kalas: a social navigation system for food recipes. *ACM Transactions on Computer-human Interaction*, 12(3): 374-400. September 2005.

115. Swearingen, K. and Sinha, R. (2001). Beyond algorithms, an HCI perspective on recommender systems. *Workshop on Recommender Systems* at *the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*.

116. Swets, J.A. (1963). Information retrieval systems. *Science*, 141: 245-250.

117. Swets, J. A. (1969). Effectiveness of information retrieval methods. *American Doc*., 20: 72-89.

118. Swets, J.A. (1988). Measuring the accuracy of diagnostic systems. *Science*, 240: 1285-1289.

119. Tang, C, Lau, R.W.H., Li, Q., Yin, H., Li, T. and Kilis, D. (2000). Personalized courseware construction based on web data mining". In *Proc. of the 1st International Conference on Web Information Systems Engineering* (*WISE 2000*), Hong Kong, China, vol.2, 204-211.

120. Tang, T, Winoto, W and Chan, K. C. C. (2005). Scaling down candidate sets based on the temporal feature of items for improved hybrid recommendations. In B. Mobasher and S. S. Anand (Eds.) *Intelligent Techniques in Web Personalization* 2003, LNAI 3169, 169-185, Springer.

121. Tang, T. Y., and McCalla, G. I. (2004). Utilizing artificial learners to help overcome the cold-start problem in a pedagogically-oriented paper recommendation system. In *Proc. of the 3rd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH' 2004*), Eindhoven, The Netherlands, 245-254.

122. Tang, T. Y., and McCalla, G.I. (2005a). Paper annotations with learner models. In *Proc. of the 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, Amsterdam, The Netherlands, 654-661.

123. Tang, T. Y., and McCalla, G.I. (2005b). Smart Recommendation for an Evolving E-Learning System: Architecture and Experiment. *International Journal on E-Learning* 4 (1): 105-129.

124. Tang, T. Y., and McCalla, G.I. (2006). Active, context-dependent, data centered techniques for e-learning: a case study of a research paper recommender system. In

C. Romero and S. Ventura (Eds.) *Data Mining in E-Learning*, WIT Press, Southampton, UK, 97-116.

125. Tang, T. Y., and McCalla, G.I. (2007a). A multi-dimensional paper recommender. In *Proc. of the 13th International Conference on Artificial Intelligence in Education (AIED 2007)*. Marina Del Rey, CA, USA.
126. Tang, T.Y., and McCalla, G. I. (2007b). The social affordance of a paper. In *Workshop of Assessment of Group and Individual Learning Through. Intelligent Visualization (AGILEeViz)*, *13th International Conference on Artificial Intelligence in Education (AIED 2007)*. Marina Del Rey, CA, USA, 34-42. http://aied.inf.ed.ac.uk/AIED2007/AgileVizWorkshopAIED2007.pdf
127. Terveen, L., Hill, W., Amento, B., McDonald, D. and Creter, J. (1997). PHOAKS: a system for sharing recommendations. *Communications of the ACM*, 40(3): 59-62.
128. Terveen, L. and McDonald, D. (2005). Social matching: a framework and research agenda. *ACM Transactions on Computer-Human Interaction*, 12(3): 401-434.
129. Torres, R., McNee, S. M., Abel, M., Konstan, J.A. and Riedl, J. (2004). Enhancing digital libraries with TechLens. In *Proc. of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'2004)*, Tuscon, AZ, USA, 228-236.
130. Vassileva, J., McCalla, G. I. and Greer, J. (2003). Multi-agent multi-user modeling in I-Help. *Journal of User Modeling and User –Adapted Interaction*, 13, 179-210. Springer.
131. Weibel, S. (1999). The Dublin Core: a simple content description format for electronic resources. *NFAIS Newsletter*, 40(7): 117-119.
132. Winoto, P and Tang, T. Y. (2008). If you like the Devil Wears Prada the book, will you also enjoy the Devil Wears Prada the movie? A study of cross-domain recommendations. *New Generation Computing*, *Special Issue: Web-based Recommendation Systems Technologies and Applications*, 36-3 (May 2008).
133. Woodruff, A., Gossweiler, R., Pitkow, J., Chi, E. and Card, S. (2000). Enhancing a digital book with a reading recommender. In *Proc. of ACM Conference on Human Factors in Computing Systems (ACM CHI'00)*, The Hague, The Netherlands, 153-160.
134. Wold, S. (1995). PLS for multivariate linear modelling. In van de Waterbeemd H. (ed.), QSAR: *Chemometric Methods in Molecular Design*, vol 2. Wiley-VCH, Weinheim, Germany, 195-218.
135. Zan, H., Chung, W., Ong, T-H, Chen, H. (2002). A graph-based recommender system for digital library. In *Proceedings of Proc. of IEEE/ACM Joint Conference on Digital Libraries (ACM/IEEE JCDL'2002)*, Portland, Oregon, USA, 65-73.
136. Ziegler, C-N., McNee, S., Konstan, J. and Lausen, G. (2005). Improving recommendation lists through topic diversification. In *Proc. of the 14th International World Wide Web Conference (WWW '05)*, Chiba, Japan, 22-32.

APPENDIX

A. PAPER LIST

P 1: **How Software Engineers Use Documentation: The State of the Practice**
Author: Timothy C. Lethbridge, Janice Singer and Andrew Forward.
Publication place: IEEE Software
Issue no.: November/December 2003
Page no.: 35-39 (5 pages), 2003.

P 2: **Software Engineering: Project Management**
Author: Alan Howard
Publication place: Communications of the ACM
Issue no.: Vol. 44, No. 5, May 2001
Page no.: 23-24 (2 pages)

P 3: **Requirements Engineering: The State of the Practice**
Author: Colin J. Neill and Phillip A. Laplante
Publication place: IEEE Software
Issue no.: November/December 2003
Page no.: 40-45 (6 pages)

P 4: **ERP Requirements Engineering Practice: Lessons Learned**
Author: Maya Daneva
Publication place: IEEE Software
Issue no.: March / April 2004
Page no.: 26-33 (8 pages)

P 5: **Software Engineering as a Business**
Author: Ann Miller, Christof Ebert
Publication place: IEEE Software
Issue no.: November/December 2002
Page no.: 18-20 (3 pages)

P 6: **Never Go to a Client Meeting without a Prototype**

Author: Michael Schrage
Publication place: IEEE Software
Issue no.: March / April 2004
Page no.: 42-45 (4 pages)

P 7: **Project Failures in Small Companies**
Author: Mohammad A. Rob
Publication place: IEEE Software
Issue no.: November/December 2003
Page no.: 94-95 (2 pages)

P 8: **Oracle: The Design of User Interface**
Author: Jeremy Ashley, Kristin Desmond
Publication place: ACM Interactions
Issue no.: 2002
Page no.: 81-86 (2 pages)

P 9: **Amazon.com Recommendations**
Author: Greg Linden, Brent Smith, and Jeremy York
Publication place: IEEE Internet Computing
Issue no.: January/Feburary 2003
Page no.: 76-80 (5 pages)

P10: **Web User Interface Design: Forgotten Lessons**
Author: Uttara Nerurkar,
Publication place: IEEE Software
Issue no.: November/December 2001
Page no.: 69-71 (3 pages)

P11: **DateLens: A Fisheye Calendar Interface for PDAs**
Author: Benjamin B. Bederson, Aaron Clamage, Mary P. Czerwinski, George G. Robertson
Publication place: ACM Interactions
Issue no.: July + August 2004
Page no.: 9-10 (2 pages)

P12: **What Sounds Do People Love and Hate?**
Author: Jonathan Effrat, Lisa Chan, B.J. Fogg, Ling Kong
Publication place: ACM Interactions
Issue no.: September + October 2004
Page no.: 64-66 (3 pages)

P13: **Seven Great Myths of Usability**
Author: MARC CHRUSCH
Publication place: ACM Interactions
Issue no.: September + October 2000

Page no.: 13-16 (4 pages)

P14: **Using Heuristics to Evaluate the Playability of Games**
Author: Heather Desurvire, Martin Caplan, Jozsef A. Toth
Publication place: ACM CHI
Issue no.: April 24–29, 2004,
Page no.: 1509-1512 (4 pages)

P15: **Learning the Ropes: Human-Centered Design Skills and Patterns for Software Engineers' Education**
Author: Ahmed Seffah
Publication place: ACM Interactions
Issue no.: September + October 2003
Page no.: 37- 46 (10 pages)

P16: **The Windows® UI: A Case Study in Usability Engineering**
Author: Kent Sullivan
Publication place: ACM CHI
Issue no.: APRIL 13-18, 1996
Page no.: 473-480 (8 pages)

P17: **The Usability Engineering Life Cycle**
Author: Jakob Nielsen, Bellcore
Publication place: IEEE Computer
Issue no.: March 1992
Page no.: 12-22(11 pages)

P18: **Automated Bug Tracking: The Promise and the Pitfalls**
Author: Laurianne McLaughlin
Publication place: IEEE Software
Issue no.: January / February 2004
Page no.: 100-102(3 pages)

P19: **Seven Hot Outsourcing Practices**
Author: Donald J. Reifer
Publication place: IEEE Software
Issue no.: January / Febuary 2004
Page no.: 14-16(3 pages)

P20: **Don't Make Me Read: Use and Abuse of Text in Web Page Design**
Author: William Hudson
Publication place: ACM Interactions
Issue no.: July + August 2003
Page no.: 55-56(2 pages)

P21: **Insights on Outsourcing**

Author: Aaron Marcus, President
Publication place: ACM Interactions
Issue no.: July + August 2004
Page no.: 12-17(6 pages)

APPENDIX B. PRE-QUESTIONNNAIRE

Student ID # : _____ Student Name # :_____

You will read many papers based on a match between the topics of the papers and your interests, experience and expectations. Please tell me your interests/expectations by filling in this short survey.

My Interests:

Please rate all items below using the following criteria:

5: very interested 4: interested 3: moderately 2: a little bit 1: not interested

	very interested				not interested
Software requirement engineering	5	4	3	2	1
Web design and Web engineering	5	4	3	2	1
User interface design	5	4	3	2	1
Software reuse	5	4	3	2	1
Agile programming	5	4	3	2	1
Software testing & quality management	5	4	3	2	1
Project management and managing people	5	4	3	2	1
E-commerce/ E-banking/ Online shop	5	4	3	2	1
Search engine & recommender system	5	4	3	2	1
Trust & reputation system on the Internet	5	4	3	2	1
Social network (chat, collaborative work, etc.)	5	4	3	2	1
Case study (engineering issues)	5	4	3	2	1
Case study (management issues)	5	4	3	2	1

My Background Knowledge:

Please tell me how familiar you are with the topics below:

	very familiar				not familiar
Programming (Basic, Java, C, etc.)	5	4	3	2	1
Statistics (variance, t-test, Pearson corr., etc)	5	4	3	2	1
Network & Internet (TCP/IP, public key, etc.)	5	4	3	2	1
Logic & Mathematics (sets, directed graph, etc)	5	4	3	2	1
Finance and management (ROI, NPV, etc)	5	4	3	2	1

Please tell me the nature of your current job (please tick the most relevant one):

___ Programming (coding) ___ Software Development (testing, design, etc)

___ Academic/Teaching ___ Management/Marketing/Finance
___ Engineering (electrical, mechanical, etc.) ___ Others

My Expectations in Taking this Class:

I want to gain knowledge on the area of (you may tick more than one):

___ Software engineering in general ___ Software project management
___ Agile programming ___ Software testing
___ Web engineering ___ User interface design
___ Other (please specify):

APPENDIX C. PAPER RATING FORM

Feedback Form

Paper Title:

Student Name:
Student No.

The following questions concerning your perceptions on the paper. You answer is very valuable for improving the recommendations.

Part I. How would describe the topics of this paper? **(Circle all that apply)**

a.	Software Requirements Engineering
b.	Web Design and Web Engineering
c.	User Interface Design
d.	Software Reuse
e.	Agile Programming
f.	Software Testing & Quality Management
g.	Project Management and Managing People
h.	E-commerce/ E-banking/ Online Shop
i.	Search Engine & Recommender System
j.	Trust & Reputation System on the Internet
k.	Social Network (chat, collaborative work, etc.)
l.	Case Study (Engineering Issues)
m.	Case Study (Management Issues)

Part II. Please check/circle your answers.

1. Is the paper difficult to understand?
   4. very difficult 3. difficult 2. easy 1. very easy

2. Is the content of paper related to your job?
   4. very much 3. relatively 2. not really 1. not at all

3. Is the paper interesting?
   4. very much 3. relatively 2. not really 1. not at all

4. Is the paper useful to aid your understanding of the SE concepts and techniques learned in the class?
   4. very much 3. relatively 2. not really 1. not at all

5. Do you learn something "new" after reading this paper?
   4. absolutely 3. relatively 2. not really 1. not at all

6. What is your overall rating towards this paper?
   4. very good 3. good 2. relatively 1. bad

7. Will you recommend this paper to your fellow classmates?
   3. absolutely yes 2. maybe 1. no

Part III.

1. Briefly describe the content of the paper in three or four sentences.
2. Give some critical comments on the paper.

THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

MEMO

To : TANG Ya, Department of Computing
From : CAO Jiannong, Chairman, Departmental Research Committee, Department of Computing

Ethical Review of Research Project Involving Human Subjects

I write to inform you that approval has been given to your application for human subjects ethics review of the following research project for a period from 12/09/2005 to 31/08/2006:

Project Title : The Study of the Pedagogical Paper Recommendation

Department : Department of Computing

Principal Investigator : TANG Ya

Please note that you will be held responsible for the ethical approval granted for the project and the ethical conduct of the research personnel involved in the project. In the case the Co-PI has also obtained ethical approval for the project, the Co-PI will also assume the responsibility in respect of the ethical approval (in relation to the areas of expertise of respective Co-PI in accordance with the stipulations given by the approving authority).

You are responsible for informing the Departmental Research Committee Department of Computing in advance of any changes in the research proposal or procedures which may affect the validity of this ethical approval.

You will receive separate notification should you be required to obtain fresh approval.

CAO Jiannong
Chairman
Departmental Research Committee
Department of Computing

CONSENT TO PARTICIPATE IN RESEARCH

The Study of the Pedagogical Paper Recommendation

I _____ hereby consent to participate in the captioned research conducted by Tiffany Ya Tang.

I understand that information obtained from this research may be used in future research and published. However, my right to privacy will be retained, i.e. my personal details will not be revealed.

The procedure as set out in the attached information sheet has been fully explained. I understand the benefit and risk involved. My participation in the project is voluntary.

I acknowledge that I have the right to question any part of the procedure and can withdraw at any time without penalty of any kind.

Name of participant

_____

Signature of participant

_____

Name of researcher

_____

Signature of researcher

_____

Date

_____