A CONTENT BOOSTED COLLABORATIVE FILTERING APPROACH FOR MOVIE RECOMMENDATION BASED ON LOCAL & GLOBAL USER SIMILARITY AND MISSING DATA PREDICTION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GÖZDE ÖZBAL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2009

Approval of the thesis:

A CONTENT BOOSTED COLLABORATIVE FILTERING APPROACH FOR MOVIE RECOMMENDATION BASED ON LOCAL & GLOBAL USER SIMILARITY AND MISSING DATA PREDICTION

submitted by **GÖZDE ÖZBAL** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _______________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Müslim Bozyiğit _______________
Head of Department, **Computer Engineering**

Assoc. Prof. Ferda Nur Alpaslan

Supervisor, **Computer Engineering Dept., METU** _______________

Examining Committee Members:

Assoc. Prof. Dr. Nihan Kesim Çiçekli
Computer Engineering Dept., METU _____________________

Assoc. Prof. Ferda Nur Alpaslan
Computer Engineering Dept., METU _____________________

Dr. Ayşenur Birtürk
Computer Engineering Dept., METU _____________________

Asst. Prof. Dr. İlkay Ulusoy
Electrical and Electronics Engineering Dept., METU _____________________

Orkunt Sabuncu, Ph.D.
ORBIM _____________________

Date: 09.09.2009

ABSTRACT

A CONTENT BOOSTED COLLABORATIVE FILTERING APPROACH FOR MOVIE RECOMMENDATION BASED ON LOCAL & GLOBAL SIMILARITY AND MISSING DATA PREDICTION

ÖZBAL, Gözde
M.S., Department of Computer Engineering
Supervisor: Assoc. Prof. Ferda Nur ALPASLAN

August 2009, 90 pages

Recently, it has become more and more difficult for the existing web based systems to locate or retrieve any kind of relevant information, due to the rapid growth of the World Wide Web (WWW) in terms of the information space and the amount of the users in that space. However, in today's world, many systems and approaches make it possible for the users to be guided by the recommendations that they provide about new items such as articles, news, books, music, and movies. However, a lot of traditional recommender systems result in failure when the data to be used throughout the recommendation process is sparse. In another sense, when there exists an inadequate number of items or users in the system, unsuccessful recommendations are produced.

Within this thesis work, ReMovender, a web based movie recommendation system, which uses a content boosted collaborative filtering approach, will be presented. ReMovender combines the local/global similarity and missing data prediction techniques in order to handle the previously mentioned sparseness problem effectively. Besides, by putting the content information of the movies into consideration during the item similarity calculations, the goal of making more successful and realistic predictions is achieved.

Keywords: Recommender Systems, Personalization, User Modeling, Collaborative Filtering, Content Based Filtering, Information Extraction, Pearson Correlation Coefficient, Floyd Warshall Algorithm

ÖZ

FİLM ÖNERME SİSTEMLERİNDE EKSİK VERİ TAHMİNİ VE LOKAL & GLOBAL BENZERLİK METODLARINI BİRLEŞTİREN İÇERİK DESTEKLİ KOLABORATİF FİLTRELEME YAKLAŞIMI

ÖZBAL, GÖZDE
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Ferda Nur ALPASLAN

Ağustos 2009, 90 sayfa

Son zamanlarda, dünya çapındaki ağın (WWW) içerdiği bilgi alanı ve bu alandaki kullanıcı sayısı açısından hızla büyümeye başlaması, web tabanlı sistemlerin gerekli bilgiyi oluşturup kullanıcılarına sunmasını giderek zorlaştırmaktadır. Bunun yanında, günümüz dünyasındaki birçok system ve yaklaşım, sundukları öneriler sayesinde insanların makale, haber, kitap, müzik ve film gibi birçok dalda kendilerine uygun seçeneklerle ilgili karar vermelerine yardımcı olmaktadır. Ancak geleneksel yöntemlerden yararlanan birçok sistem, önerme işlemleri sırasında kullanılan veri seyrek olduğunda başarısız olmaktadır. Başka bir deyişle, sistemde yetersiz sayıda nesne veya kullanıcının bulunması, yetersiz önerilerin üretilmesine yol açmaktadır.

Bu tez kapsamında, içerik destekli kolaboratif filtreleme yaklaşımı kullanan, ReMovender isimli bir web tabanlı film önerme sistemi tanıtılacaktır. ReMovender, veri eksikliği problemini etkili bir şekilde çözme amacıyla local/global benzerlik ve eksik veri tahmini metodlarını birleştirmektedir. Bunun yanı sıra, film benzerliği hesaplamalarında filmlerin içeriklerini de göz önünde bulundurarak daha başarılı ve gerçekçi tahminler elde etmektedir.

Anahtar Kelimeler: Öneri Sistemleri, Kişiselleştirme, Kullanıcı Modelleme, Kolaboratif Filtreleme, İçerik Bazlı Filtreleme, Bilgi Çıkarımı, Pearson Korelasyon Sabiti, Floyd Warshall Algoritması

To my family

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to Assoc. Prof. Ferda Nur Alpaslan for her encouragement and support throughout this study.

I am deeply grateful to my family for their love and support. Without them, this work could never have been completed.

I am deeply indebted to my friends Candan Ceylan, Hilal Karaman and Erbil Karaman whose help, stimulating suggestions and encouragement helped me in all the time of research and development of this thesis. Furthermore, I thank Davide Alberani for sharing his deep expertise with me.

I would thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing the financial means throughout this study.

Finally, my special thanks go to my friends, Candan Ceylan, Gizem Öztürk, Fulya Tunçer, Olcay Aydemir, Gül Deniz Demirel, Bora Yıldız, and Özgen Bulut Erdoğan for their help, support and cheerful presence through not only the course of this study but also any course of my life. Thanks for giving me a shoulder to lean on, whenever I need.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

RS	Recommender Systems
IR	Information Retrieval
IF	Information Filtering
CB	Content Based
CBR	Content Based Recommender
CF	Collaborative Filtering
PCC	Pearson Correlation Coefficient
VS	Vector Similarity
EMDP	Effective Missing Data Prediction
CBCF	Content Boosted Collaborative Filtering Approach of ReMovender
LU&GU	Local & Global User Similarity
SF	Similarity Fusion
UPCC	User Based Using PCC
IPCC	Item Based Using PCC

CHAPTER 1

INTRODUCTION

1.1 Background

Recently, the Web, globally and within intranets, has been expanding both in the size of the information space and in the number of the users of that space. As a result, the task of locating relevant information and navigating that space in order to make choices of good items has been proving more and more difficult [1].

This information overload has created a great interest in automated filtering, refinement, and organized presentation of relevant information to the users. There have been efforts to automate filtering of relevant information and presenting organized information to the user. Such automated methods, commonly referred to as intelligent information retrieval are used to locate and retrieve information with respect to a user’s individual preferences [2][3][4][5][6]. Also, efforts to rank and sort information based on user preferences, which are often conflicting, have generated interest over the past few years [7]. One key area of research to achieve this goal is targeted around *recommender systems* (RS).

In today's world, many systems and approaches make it possible for the users to be guided by the recommendations they provide about new items such as news, web pages, articles, books, music, and movies. In addition, recommender systems are being used in an ever-increasing number of e-commerce sites such as Amazon [8] for books, IMDb [9], MovieLens [10], and MovieFinder [11] for movies, Pandora [12] and Last.fm [13] for music domains, in order to assist “buyers” in finding suitable precuts [14].

The amount of available information in the Web, as a result of increasing of the electronic business activities, is greater than a consumer can manage. In order to be competitive, E-commerce systems need to provide users with mechanisms for selective retrieval of web information. A way to obtain new customers and retain existing ones is the personalized product recommendation. E-commerce applications that incorporate recommender systems provide users with intelligent mechanisms to search products to purchase [15]. At the online world, RSs act as vendors who connect the items to customers by predicting their preferences. Such systems not only reduce the searching time of the user, but also enhance the selling rates and build consumer's loyalty [16].

RSs are often combinations of information filtering and intelligent agent systems [17] that use computational intelligence and that can be trained to find patterns between different users and subject matter [18]. In order to achieve this, RSs use the stored preferences (history of user's likes and dislikes) to locate and suggest items of interest to the users they serve. Other than making use of the stored preferences of the user, they also use the opinions of members of a community for helping individuals in that community to identify the information or products most likely to be interesting to them or relevant to their needs [19].

Many of the recommendation strategies used today rely on the modeling of intrinsic attributes about each item (e.g. the keywords for a document or the genre of a movie) so that the items can be categorized, and the level of interest that a user has can be expressed in terms of these attributes [20]. This knowledge is usually gathered over time, by monitoring and logging various user interactions with the system so that the knowledge base is updated dynamically. An RS usually combines the values (ratings) of the elements of every dimension according to some evaluation scheme before obtaining a recommendation value (rating) of an item [21].

Due to the explosive improvements especially in the WWW and internet services lately, recommendation systems based on AI technology have begun to arise. Besides, the evolution of the WWW has led various researches on some information retrieval techniques [22]. With the help of those researches and the techniques developed, users have begun not to have so much difficulty while making their choices.

1.2 Recommender Systems As a Research Area

Recommender systems have become an important research area since the appearance of the first papers on *collaborative filtering* (CF) in the mid-1990s. There has been much work done in the industry and academia on developing new approaches to RSs over the last decade [23].

The roots of recommender systems can be traced back to the extensive work in cognitive science, approximation theory, information retrieval, forecasting theories, and they also have links to management science and to consumer choice modeling in marketing. However, recommender systems emerged as an independent research area in the mid-1990s when researchers started focusing on recommendation problems that explicitly rely on the ratings structure. In this manner, the recommendation problem was reduced to the problem of estimating ratings for the items that have not been observed by a user yet [24].

The interest in this research area maintains its popularity today because it constitutes a problem-rich research area and there is an abundance of practical applications that help users to deal with information overload and provide personalized recommendations, content, and services to them [23]. The current RSs are built by combining approaches from different areas including Artificial Intelligence, Natural Language Processing, Human-Computer Interaction, Sociology, Information Retrieval and the technology of the WWW [24].

1.3 Problem Definition

This thesis focuses on the development and evaluation of a web based movie recommendation system called ReMovender. A variety of features are integrated by ReMovender in order to attack the so-called ‘data sparsity’ problem in recommendation systems. In addition, the cultural metadata of the movies are exploited by the CF approach in order to make more successful and realistic predictions.

In this thesis study, the first fact that is being focused on is that traditional collaborative filtering methods may not achieve success when the training data is sparse. Thus, a synthesis of two separate methods (Local & Global User Similarity [25] and Effective Missing Data Prediction [26], which have been proven to handle this problem in a reasonable way, is used to attack the data sparsity problem with a stronger approach. Secondly, the positive impact of using content information in a collaborative filtering approach is addressed by developing a content boosted collaborative filtering prediction technique.

1.4 Structure of This Thesis

This thesis will be structured in the following way:

In chapter 2, a detailed explanation about recommender systems and a more formal description of the recommendation process is presented. In addition, current approaches and theories used throughout the existing recommendation systems are stated.

Chapter 3 covers the related work about pure content based, pure collaborative filtering, and hybrid recommendation systems. A variety of recommendations systems from different domains are described. Besides, the pros and cons of these systems are discussed.

In chapter 4, the architecture and the system components of ReMovender are presented. Besides, the prediction mechanism of the system is examined in detail.

In chapter 5, the evaluation scheme used to test the performance of ReMovender is explained. In addition, the results of the experiments that were carried out during the evaluation process are discussed.

Chapter 6 draws the conclusions of this thesis work. Other than that, some possible future work in terms of both ReMovender and the related area is stated.

CHAPTER 2

RECOMMENDER SYSTEMS

2.1 Definition of a Recommender System (RS)

An RS is an agent-based system which is a combination of an information filtering and intelligent agent system. It uses the stored preferences to locate and suggest items of interest to the users being served [27]. RSs can also be considered as services which recommend users new items such as news, articles, books, music, and movies they would like [28]. These systems produce a ranked list of items on which a user might be interested, in the context of his/her current choice of an item [29]. That is, the existence of actor information provides RSs to implement automated selection assistance [30]. [31] gives a more specific definition of RSs like "Systems that produce individualized recommendations as output or have the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options." The keyword terms individualized and personalized indicate that each user in an RS will be presented with different information sources or items according to the information gained about him/her. The detailed explanation about the 'personalization' term will be covered in 2.2.

It can be realized from the above definitions that all RSs require any kind of information about the preferences of the users and a method to decide if an item is interesting for a specific user.

The recommendation problem can be formulated more formally as follows [23]: Let C be the set of all users and let S be the set of all possible items that can be recommended, such as books, movies, or restaurants. The space S of possible items can be very large, ranging in hundreds of thousands or even millions of items in some applications, such as recommending books or CDs. Similarly, the user space can also be very large—millions in some cases. Let u be a utility function that measures the usefulness of item s to user c, i.e., u : C X S -> R, where R is a totally ordered set (e.g. nonnegative integers or real numbers within a certain range). Then, for each user c ∈ C, we want to choose such item s` ∈ S that maximizes the user's utility. More formally:

$$\forall c \in C, s_c^{`} = \underset{s \in S}{argmax}\, u(c,s) \qquad \textbf{(1)}$$

The main problem which recommender systems try to deal with is that utility u is most of the time not defined for the whole C x S space. The utility of an item is often represented by a rating, and is initially defined only on the items that have already been experienced and then rated by the users. An example user-item rating matrix for a movie recommendation system can be observed in Table 1.

Table 1 - An example user-item matrix

	Titanic	Brave Heart	Harry Potter	Twilight
Mary	4	0	2	0
Julia	0	5	0	3
John	5	0	1	2
Robert	0	4	3	0

This table shows the ratings that were given by four users Mary, Julia, John, and Robert for the movies Titanic, Brave Heart, Harry Potter, and Twilight. For instance, the rating that Mary has given for the movie Titanic is 4 and the one John has given for the movie Harry Potter is 1. Actual ratings are specified on a scale of 1 to 5, and the value 0 is used for the ratings of the items that haven't yet been experienced by the user. To illustrate, we can understand from the table that Robert hasn't seen the movie Twilight yet. In such a system, the first mission of the recommendation system can be defined as predicting the ratings of the non-rated movie/user rating combinations with the 0 rating value based on the training set (i.e. existent ratings). In other words, the RS tries to replace all 0 values by some optimal guesses. And the next mission of the systems is presenting the appropriate recommendations. Once the prediction process is complete, recommendations to a user are made by selecting the highest rating among all the estimated ratings for that user [32] or N best items to a user or a set of users to an item [23].

In Figure 1, recommendation process at a very high level is depicted. As can be observed from this figure, two sources of information, including users' profiles and the information about items and products, are required for the process. The information stored in the profiles is related with the preferences of the users and can be explicitly given by the user or can be extracted from other external sources such as web pages, buying behavior, etc. The information about the items can range from special metadata of the product, information extracted from the item, or the item itself in the case of electronic documents. In audio or video recommendation systems, this information could produce databases of huge dimensions. It can also be distinguished from the figure that the final product of the system is a set of recommendations for the user. And as stated before, the representation of these recommendations depends on the system itself but may range from ordered lists of items, snapshots of the items, or the whole items [33].

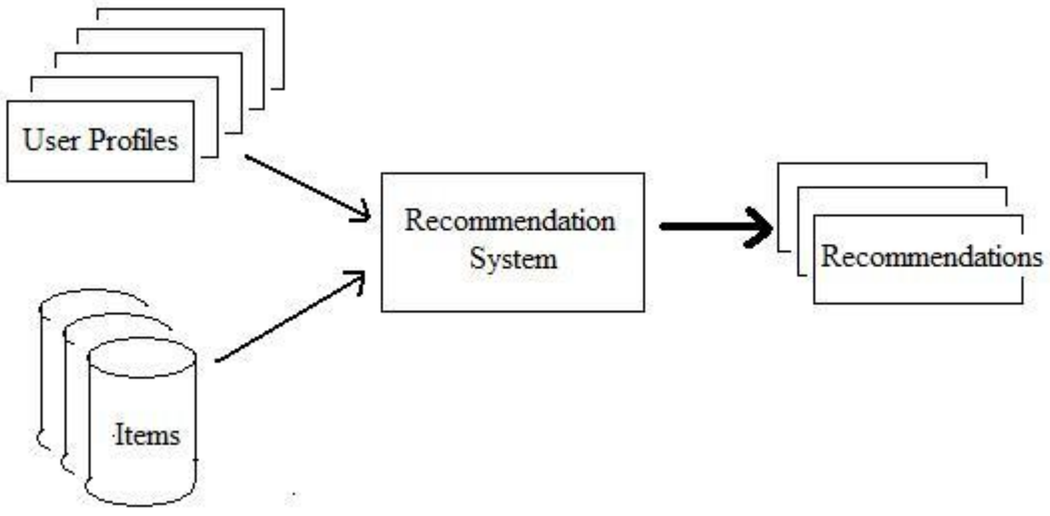


Figure 1 - Recommendation process at a very high level

2.1.1 Terms and Concepts Common to Recommender Systems

There exist some terms and concepts that are commonly used in existing Recommender Systems. Thus, it is essential to first understand these basics well before proceeding with the details of recommender systems. The following definitions are based on the work presented in [34], [35] and [36].

- **Resources:** The targets of the recommendation process

- **Recommenders:** Entities that radiate opinions about resources. In practice, recommenders are actors, and in principle, they could also be artificial agents. Other than that, they may be resources at the same time. A recommender can be considered as any entity that gives personalized recommendations as output to users' preferences. It may be possible that a recommender does not produce a specific output, such as a list, but they might guide somehow the users in an individual way to useful or interesting items.

- **Descriptions:** Those information resources that include opinions. The descriptions may explicitly be designed for expressing opinions, or they may include opinions about resources implicitly, in which case they must be inferred.

- **Preferences:** Recommendation seeker's position towards resources

- **Algorithms for Computing Recommendations:** The system's means for automatically evaluating resources by using descriptions and preference information

- **Recommendations**: The concrete results of the evaluation process for the recommendation seeker. The recommendations may be presented for example by filtering out resources, informing about new resources, displaying recommendations along the resources, or ordering resources, simply in an ordered list, or perhaps by presenting sophisticated 2D or 3D visualizations.

- **User's Interest:** This is an abstract representation of how much a user appreciates an item. User's interest is a subjective concept and it is hard to represent it in an objective way.

- **Prediction:** The expected interest of a user in one item. This concept is different from the concept of recommendation. While some systems might present predictions with the actual recommendations, others can produce recommendations only.

- **Rating:** An objective measure representing a user's interest. The possible values of this measure are given according to a scale established by the designer of the recommendation system.

- **Predicted Rating:** An objective measure representing the expected interest of a user in a certain item. This measure is estimated by the system and its possible values are elements of a specific scale.

- **Actual Rating:** Objective measure representing the real interest of the user in a specific item. This value is given by the user himself/herself according to the scale of rates in the system.

- **Prediction Accuracy:** A measure that indicates at which extent the predicted rating agrees with the user's actual rating. The more accurate the prediction, the better the performance of the recommendation system is.

- **Prediction Technique:** The specific algorithm that the recommendation system will use in order to calculate the predicted rating of an item.

2.2 Personalization

As stated previously in 2.1, RSs are directly related to the personalization on a web site and the development of electronic e-commerce. [37] considers personalization as the future of the Web since it has achieved great success in industrial applications. For instance, online stores, such as Amazon and Netflix, provide customized recommendations for additional products or services based on a user's history. Besides, recent offerings such as My MSN, My Yahoo, My Google and Google News infer a user's interests from his/her history.

Personalization includes a series of fundamental and independent processes [38]:

User data acquisition

- Model building
- Identification of adaptation tasks

And the main goals of personalization technology can be summarized as below [39]:

- Relevant and precise recommendations must be delivered according to the tastes and preferences of each user in the system
- These preferences must be delivered with minimal involvement from users
- Recommendations must be delivered in real time with an immediate interaction

2.3 User Modeling

Obtaining any kind of knowledge about the taste or preferences of a specific user helps the recommender systems achieve more success in the recommendation process. For this aim, RSs make use of two kinds of user modeling. The term user modeling commonly means only the modeling of the local user. The modeling of other users can be referred as actor modeling.

In an environment where different users require different kinds of resources, user modeling helps the system to present useful recommendations for different users. At the end of the modeling process, a user stereotype or a user profile is created. Stereotype tells user’s type in some predefined taxonomy, whereas profile provides a more detailed description with several attributes of varying value ranges. Since there is a great variety of quality preferences and available resources, a simple stereotype cannot be sufficient for user modeling in a recommender system. On the other hand, stereotype often provides an adequate solution for actor modeling, as most of the time there exist no information available to build a detailed actor profile. Besides, final recommendation is usually maintained by processing the opinions of many actors. That’s why the details of one actor do not have to be considered as so important [30].

Although the easiest way of modeling a user is asking his/her preferences explicitly, it does not seem to work well in practice, since the preferences of a user may change over time. Moreover, keeping a user profile up to date manually is a tedious work. Besides, it could be so hard for a user to explicitly state his/her preferences by just setting some attribute values.

The other way that can be used for user modeling is collecting implicit feedback, in which the user profile is built without disturbing the user by unobtrusively watching the user interactions with the system [30]. Buying, evaluating, or recommending a resource, repeated usage, and the save/print, delete, refer, reply, mark, examine/read, consider, glimpse, associate or query actions can be listed as the clues for maintaining implicit feedback [40]. In addition to these, there exist various other possibilities like capturing the number of mouse clicks, the time spent moving the mouse, the number of clicks on the horizontal and vertical scroll bars, the time spent, number and duration of pressing some special action keys, like page up or down arrow [41].

As for the actor modeling, implicit feedback methods do not seem to be bearable solutions due to the privacy issues. Because, publishing any information about actors, which is gathered by automatic surveillance techniques, without asking permission may not be a good idea all the time [30].

2.4 Information Overload

The increase and widespread of different technologies have changed the way people access, manage, and distribute content. One direct consequence of this is the corresponding information overload that affects any user looking for some specific or new items [33].

Information overload can be defined as the following [31]:

"…the state of having too much information to make a decision or remain informed about a topic. Large amounts of historical information to dig through, a high rate of new information being added, contradictions in available information, a low signal-to-noise ratio making it difficult to identify what information is relevant to the decision, or the lack of a method for comparing and processing different kinds of information can all contribute to this effect"

2.5 Information Retrieval and Filtering

In order to cope with the information overload problem mentioned in 2.4, a set of techniques and tools are provided by the research field of information retrieval (IR). Recommender systems can be considered as a kind of information retrieval system according to [42]. However, there's little distinction between them such that whereas user queries are considered to be one-time episodic tasks ordered to the system to complete in IR systems, there is a continuing task in recommenders like deriving good estimates of ratings for items that the user has not experienced yet [24]. RSs adapt the IR approaches in conjunction with knowledge about the users, to suggest personalized and interesting items to each user in the system [33].

Information retrieval is defined as the following according to [31]:

"…the art and science of searching information in documents, searching documents themselves, searching metadata which describes documents, or searching within databases, whether relational stand alone databases or hypertext networked databases such as the Internet or intranets, for text, sound, images or data"

As for the Information Filtering (IF) systems, they focus on filtering information based on a user profile. Filtering within an IF system is done when the user automatically receives the information required based on his/her profile. The advantage of IF is its ability to adapt to the user's long term interest and bring him/her the filtered information [24]. Information retrieval also deals with finding information by using filtering approach when large volumes of information should be sorted and only the interesting sources should be presented to the user [33].

2.6 Cultural Metadata

People often depict their thoughts and feelings about something through their word-of-mouth by talking almost every day. However, since these talks are temporal and vanishing, they easily fly away.

On the other hand, WWW has a characteristic to transform these temporal 'paroles' into spatial 'texts'. Texts are lasting and preserved in the Web. Indeed, the WWW is a mine of numerous thoughts and impressions. A part of paroles on the Web is about cultural contents like music, soap operas, books, and movies. Reviews and comments on cultural contents are published on user blogs. Qualitative discourses such as ratings, reviews, and comments have been accumulated on cultural contents DB sites. These paroles about cultural contents have been aggregated by numerous users spontaneously and culturally, and formed colorful qualitative discourses and reputations. They are considered as indeed 'cultural metadata' with a great potential according to [28].

[43] defines cultural metadata can as 'the information that is implicitly present in huge amounts of data and needs to be extracted with information retrieval techniques.'

2.7 Recommendation Techniques

Since one of the most important components of every recommender system is the one in charge of making predictions, it is logical to classify them according to the prediction technique they use. Based on the prediction technique used, [33] classifies the RSs into three main groups:

- Systems that use information-based prediction techniques
- Systems that use social-based prediction techniques
- Systems that combine the first two techniques in several different ways

2.7.1 Information-based Prediction Techniques

Information-based prediction techniques base their results in the analysis of all the items in addition to the preferences of the user. Only the actual user is taken into consideration and the information related to other users is not processed. Furthermore, these techniques are considered domain specific, since they have to analyze the information stored in the item or the metadata associated with them. Some examples of these techniques are: case-based reasoning or content-based recommendation, information filtering, attribute-based techniques [34].

The main assumption under case-based reasoning or content-based recommendation techniques is that a user has similar preferences over similar items. The more similar the items, the more equal the preferences of the user on those items. In information filtering, the system is in charge of ordering huge amounts of items and delivering to the user only the items that are relevant for him/her. Finally, attribute-based techniques estimate their results on the base of the attributes of an item. Each attribute of an item has an importance weight and the overall prediction is calculated taking into account the value of each attribute [33].

The main assumption under content-based approaches is that an item or document can be identified by a set of features extracted directly from their content [44]. As stated before, content-based recommendation techniques are special cases of information-based techniques; in the way that they attempt to make predictions based on the analysis of the items or the metadata associated with them. In [34], these techniques are described as techniques that *"look at all items a user has rated in the past and determines how similar they are to the current item. For those items that are similar enough, the old ratings are used to calculate a predicted rating for the new item".* This description introduces the notion of similarity among items and also takes into account the idea of previously rated items [33].

In content-based recommendation methods, the utility u(c,s) of item s for user c is estimated based on the utilities u(c,s_i) assigned by user c to items $s_i \in S$ that are "similar" to item s. For example, in a movie recommendation application, in order to recommend movies to user c, the content-based recommender system tries to understand the commonalities among the movies user c has rated highly in the past (specific actors, directors, genres, subject matter, etc.) Then, only the movies that have a high degree of similarity to user's preferences are recommended [23]. The similarity calculation between user preferences and movie pieces in content based recommender (CBR) system can be observed in Figure 2.

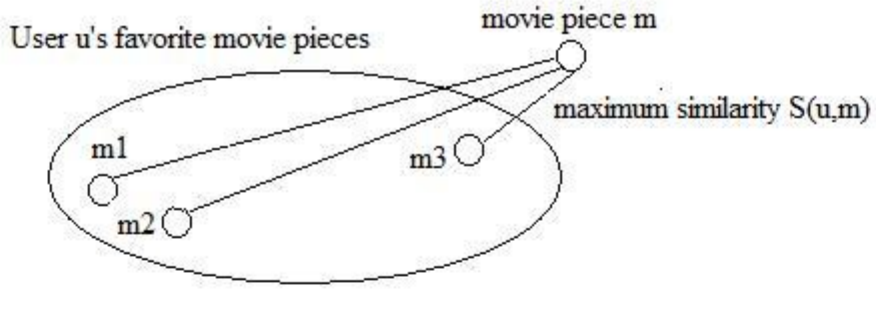


Figure 2 - Similarity calculation in CBR

More formally, let Content(s) be an item profile, i.e., a set of attributes characterizing item s. It is usually computed by extracting a set of features from item s (its content) and is used to determine the appropriateness of the item for recommendation purposes.

In content-based recommender systems, various candidate items are compared with items previously rated by the user and the best matching items are recommended. More formally, let ContentBasedProfile(c) be the profile of user c containing tastes and preferences of this user. These profiles are obtained by analyzing the content of the items previously seen and rated by the user and are usually constructed using keyword analysis techniques from information retrieval.

The utility function in u(c,s) in content-based systems is often defined as below [23]:

$$u(c,s) = score\ (ContentBasedProfile(c),Content(s)) \qquad \textbf{(2)}$$

CBR has its roots in the IR (Information Retrieval) techniques and, therefore, has the advantage in dealing with text-based items like news articles, books, URL, and so on [45][46]. Recently, as text-based metadata for audio/visual/video items such as music, pictures, drawings, or movies are created and provided widely, it’s possible for CBR to deal with a variety of items [28]. Besides the traditional heuristics that are based mostly on information retrieval methods, other techniques for content-based recommendation have also been used, such as Bayesian classifiers [47][48] and various machine learning techniques, including clustering, decision trees, and artificial neural networks [48]. These techniques differ from information retrieval-based approaches in that they calculate utility predictions based not on a heuristic formula, such as a cosine similarity measure, but rather are based on a model learned from the underlying data using statistical learning and machine learning techniques.

The strengths of CBR can be listed as below [28][33]:

- One of the most important advantages of content-based methods is that they can recommend completely new items.

- Since the idea of algorithm is simple to understand, it’s easy to explain a process of recommendations to users. The more users understand the system, the more they trust it. And when they appreciate the system, they provide their preferences more willingly.
- CBR is practical and serviceable. It requires not many resources for a computation process, although the total number of items managed in the system is huge.
- Non-personalized services are possible without user preference profiles.
- Its coverage is wide since there is no constraint except item analysis techniques.
- Since the content is usually constant, an item should be analyzed only once. That it, it’s not necessary to analyze it again if new items are included in the system or if new information about users is obtained.

As for the weaknesses of CBR, they can be summarized as below [23][28][33]:

- CBR do not take the opinions of other users to into consideration in order to recommend items. This can be seen as a weakness since users’ preferences are normally not unique and the information provided by other users could provide some important information that otherwise is impossible to find out by only analyzing content.
- Recommended items in CBR are apt to be ‘*over-specialized’*. That is, since the system can only recommend items scoring highly against a user’s profile, the user is restricted to see items similar to those already experienced. For example, a person with no experience with Greek cuisine would never receive a recommendation for even the greatest Greek restaurant in town. This problem, which has also been studied in other domains, is often addressed by introducing some randomness.
- This weakness can be shortly named as ‘limited content analysis’. Most of the time, it is really hard work to extract the truly relevant and significant features from the content. Besides, content-based techniques are limited by the features that are explicitly associated with the objects that these systems recommend. Therefore, in order to have a sufficient set of features, the

content must either be in a form that can be parsed automatically by a computer (e.g., text) or the features should be assigned to items manually. While information retrieval techniques work well in extracting features from text documents, some other domains have an inherent problem with automatic feature extraction. For example, automatic feature extraction methods are much harder to apply to multimedia data, e.g., graphical images, audio streams, and video streams. Moreover, it is often not practical to assign attributes by hand due to limitations of resources [49]. Another problem with limited content analysis is that, if two different items are represented by the same set of features, they are indistinguishable. Therefore, since text based documents are usually represented by their most important keywords, content-based systems cannot distinguish between a well-written article and a badly written one, if they happen to use the same terms [49].

- The user has to rate a sufficient number of items before a CBR system can really understand the users' preferences and present the user with reliable recommendations. This is often called the *"New User Problem*".

2.7.2 Social-based Prediction Techniques

In social-based prediction techniques, both the information about the actual user and the whole set of users are analyzed. Since item information is not used, these techniques are domain independent. Some examples of social-based prediction techniques are collaborative filtering (CF), item-item filtering, stereotypes and demographics, popularity, average [33].

CF is based on the assumption that people who rate items in a similar way probably have similar preferences and tastes. On the other hand, in item-item filtering the main assumption is that items that have similar rates are probably alike. The use of stereotypes and demographics are related with the fact that people who comes from the same background (i.e. age, gender, occupation, education, demographic data, etc.) usually exhibits exchangeable preferences. Finally, popularity and average are simple prediction techniques that recommend items based on their popularity among all users, or on the average of all the set of ratings of an item [33].

Since the main attention of this thesis will be on collaborative filtering, only the details of CF will be explained in the following section.

2.7.2.1 Collaborative Filtering Method

CF is the method which automatically predicts the interest of an active user by collecting rating information from other similar users or items. The underlying assumption of CF is that the active user (the user that the prediction refers to) will prefer those items which the similar users prefer [26]. This approach is based on the idea that tastes of people are not randomly distributed, and there exist general trends and patterns within the tastes of a person and between groups of people.

Actually, experiences and opinion of friends, or other users serve as a good reference when making a choice of new items [28]. To illustrate, if there is a target user who likes pieces A and B and if there are many users who like A, B, and C, C will be recommended to the target user. This technique is widely utilized in practical web shopping services like Amazon and iTunes music store, and has been demonstrated to be rather effective [50].

In order to produce recommendations using CF, the system first collects and maintains information about the user. This information includes specific interest of users in certain items and it is stored in separate profiles [33]. Once all the user profiles have been collected, the active user’s similarities with the remaining of the users are calculated. This calculation is system specific and it depends on the algorithm used. Then, the group (neighborhood) of the users that are most similar to the active user is selected and their ratings are combined to produce predictions. Predictions of ratings may typically lead to the presentation of a ranked or a top-n-list of the most relevant items [51].

CF is based on the subjective “overall taste” criterion expressed through the rating and therefore is applicable to any type of content without requiring its analysis to features.

2.7.2.2 Classification of Collaborative Filtering Method

Algorithms for collaborative recommendations can be grouped into two main classes.

2.7.2.2.1 Memory-Based Algorithms

These are heuristics that make rating predictions based on the entire collection of previously rated items by the users. That is, the value of the unknown rating for a user and an item is usually computed as an aggregate of the ratings of some other (usually the N most similar) users for the same item [23].

Memory-based approaches are the most popular prediction methods and are widely adopted in commercial collaborative filtering systems.

The major types of memory-based approaches are user-based approaches and item-based approaches.

- **User-based Approaches**

  User-based CF predicts an active user’s interest in a particular item based on rating information from similar user profiles, where each user profile corresponds to a row vector sorted in the user-item matrix. First, all similarities of any two row vectors are calculated. Then, for predicting the rating of a user for a particular item, a set of top-N similar users are identified. The ratings of those top-N users are averaged as the prediction by weighted [25].

  Reduced to a simple formula, user-based CF suggests that users who choose A will prefer B and C since other users in the database who chose A preferred B and C [39].

The detailed formula for the predicted rating $r'_{a,y}$ of test item y by test user a, where w_{a,u_k} denotes the similarity between a and his/her neighbors u_k, is given below:

$$r'_{a,y} = \frac{\sum_{k=1}^{K} w_{a,u_k}\ r_{u_k,y}}{\sum_{k=1}^{K} |w_{a,u_k}|} \quad \textbf{(3)}$$

Since a user-based CF system simply looks at a target user's pattern of choices and has no understanding of the underlying content being selected, it can often make poor-quality recommendations, presuming similarity in consumer behaviors where none may exist. Besides, these systems are extremely data-intensive, typically requiring a large number of user choices (ratings) before they can make reasonable recommendations. And depending on how actively a user selects (rates) content, the system may be slow to accumulate enough information about a user's preferences to make accurate recommendations, resulting in poor recommendations for a prolonged period [39].

- **Item-based Approaches**

These approaches use the similarity between items instead of users. After, the similarity of items (column vectors in the user-item matrix) are calculated, unknown ratings can be predicted by averaging the ratings of other similar items rated by the active user [25].

Reduced to a simple formula, item-based CF says that if the target user likes A, the system will recommend items B and C, if those items are determined to be the most similar to item A, based on their correlations or attributes. Since detailed content attributes are typically not commercially available, most implementations of item-based CF are based on user ratings [39].

The detailed formula for the predicted rating $r'_{a,y}$ of test item y by test user a, where w_{y,i_k} denotes the similarity between the test item and the most similar items i_k, is given below:

$$r'_{a,y} = \frac{\sum_{k=1}^{K} w_{y,i_k} \; r_{a,i_k}}{\sum_{k=1}^{K} |w_{y,i_k}|} \qquad \textbf{(4)}$$

The main advantage of item-based CF over user-based CF is its scalability. Item-based CF does not have to scour databases containing potentially millions of users in order to find users with similar tastes. Instead, it can pre-score content based on user ratings and/or their attributes, and then make recommendations without incurring high computation costs [39].

However, item based CF has some shortcomings, too. If similarity of items is based on correlations of user ratings and there is no information about the consumption/usage of an item rated by the target user, *cold start problem* occurs. And this prevents the system from accurately recommending new content or frequently changing content. On the other hand, if similarity is based on item attributes, calculation of similarity becomes highly inaccurate, since it can only be as accurate as the available attributes, which are quite limited in commercial applications. Besides, just like user-based CF, item-based CF systems typically require a large number of user choices/ratings before meaningful patterns can be identified in order to make accurate recommendations. Furthermore, item-based CF suffers from poor quality recommendations, since it simply recognizes patterns among ratings and does not learn about the individual preferences of a particular user. And lastly, item-based CF lacks the ability to learn more about a target user’s preferences by analyzing groups of similar users, which can seriously result in recommendations of poor quality [39].

2.7.2.2.2 Model-Based Algorithms

In contrast to memory-based algorithms, model-based algorithms use the collection of ratings as a training dataset to learn a *model*, which is then used to make rating predictions [23]. Examples of model-based approached include clustering models, aspect models, and latent factor models [26]. The model-based approaches are often time-consuming to build and update, and cannot cover a user range as diverse as the memory approaches [52].

2.7.2.3 Algorithms for Collaborative Filtering Method

As mentioned before, one important aspect to be considered in CF method is the way similarity between the profiles of users is computed. There are several possible algorithms for this computation and some important ones will be explained in detail below.

2.7.2.3.1 Mean Squared Differences Algorithm

This algorithm estimates the degree of dissimilarity between user profiles by calculating the mean squared difference between them. After taking into consideration all the profiles with the degree of dissimilarity below certain threshold L and computing a weighted average of the ratings provided by those profiles, final predictions can be made. And the weights used by this algorithm are inversely proportional to the dissimilarity [33].

The formula used for calculating the similarity between two users is given below:

$$D_{x,y} = \frac{\sum_{\mathrm{n}}^{\mathrm{Na}} C_{xn} . C_{yn} . (S_{xn} - S_{yn})^2}{\sum_{\mathrm{n}}^{\mathrm{Na}} C_{xn} . C_{yn}} \qquad \textbf{(5)}$$

where C_{xn} = [1,0]: depending on whether item n is rated by user x or not

C_{yn} = [1,0]: depending on whether item n is rated by user x or not

S_{xn} is the rate of item n given by user x

S_{yn} is the rate of item n given by user y

It can be easily observed from the formula that the similarity between two users depends only on the items that have been rated by both of the users, since the values of C_{xn} and C_{yn} will be 0 if one of the users does not give a rating to n [33].

2.7.2.3.2 Vector Similarity (VS)

This algorithm looks at the arrays of user ratings as vectors, and uses the cosine of the angle between the vectors as an index of similarity. The equation for similarity between users in the vector similarity algorithm becomes the normalized dot product of the two vectors, or the cosine of the angle between them [53].

$$w_{up,uq} = \frac{\sum_{\{i|r_{p,i} \,\&\, r_{q,i} != 0\}} r_{p,i} \; r_{q,i}}{\sqrt{\sum_{\{i|r_{p,i} != 0} r_{p,i}^2} \cdot \sqrt{\sum_{\{i|r_{q,i} != 0} r_{q,i}^2}} \qquad \textbf{(6)}$$

where $w_{up,uq}$: similarity between users u and q

$r_{p,i}$: the rate user p gave item i

$r_{q,i}$: the rate user q gave item i

2.7.2.3.3 Pearson Correlation Coefficient (PCC)

User-based CFF engaging PCC is used in a number of RSs, since it can be easily implemented and can achieve higher accuracy than other similarity computation methods. In user-based CFF, PCC is employed to define the similarity between user a and user u based on the items that they rated in common. The related formula, where Sim(a,u) denotes the similarity between users a and u, and i belongs to the subset of items which were rated by both of the users, $r_{a,i}$ is the rate user a gave item i, and avg(r_a) represents the average rating of user a, is given below [26]:

$$\mathrm{Sim}(a,u) = \frac{\sum_{i \in I(a) \cap I(u)} (r_{a,i} - avg(r_a)).\,(r_{u,i} - avg(r_u))}{\sqrt{\sum_{i \in I(a) \cap I(u)} (r_{a,i} - avg(r_a))^2} \cdot \sqrt{\sum_{i \in I(a) \cap I(u)} (r_{u,i} - avg(r_u))^2}} \qquad \textbf{(7)}$$

On the other hand, the basic idea in similarity computation between two items i and j by using PCC is to first isolate the users who have rated both of these items and then apply a similarity computation technique to determine the similarity Sim(i,j) [54]. The related formula, where Sim(i,j) denotes the similarity between items i and j, and u belongs to the subset of users who rated both of the items, $r_{u,i}$ is the rate user u gave to item i, and avg(r_i) represents the average rating of item i, is given below [26]:

$$\mathrm{Sim}(\mathrm{i},\mathrm{j}) = \frac{\sum_{\mathrm{u}\in \mathrm{U(i)}\cap \mathrm{U(j)}} (r_{u,i} - avg(r_i)).\ (r_{u,j} - avg(r_j))}{\sqrt{\sum_{\mathrm{u}\in \mathrm{U(i)}\cap \mathrm{U(j)}} (r_{u,i} - avg(r_i))^2}\ .\ \sqrt{\sum_{\mathrm{u}\in \mathrm{U(i)}\cap \mathrm{U(j)}} (r_{u,j} - avg(r_j))^2}} \qquad \textbf{(8)}$$

When PCC is used for the similarity calculations of both user and item, the possible similarity values range from -1 to +1 including 0. A larger similarity value of two items that is calculated via PCC means that the related items are more similar. Values near -1 indicate a negative correlation while values close to +1 indicates a positive correlation. And a value of 0 shows no correlation at all. One important characteristic of this algorithm is that it takes into account not only positive correlation but also negative correlation in order to make the predictions [33]. PCC-based CF generally can achieve higher performance than VS, since it considers the differences of user rating styles [56]. And the main difference between this method and mean squared difference is that it is not necessary in PCC to select users that are close to the testing user, since this method provides a way to take into account all the users of the set even when they have a negative correlation [33].

2.7.2.4 Strengths and Weaknesses of Collaborative Filtering Method

Strengths of the CF method can be listed as below: [33][55]

- No electronic representation of the items in order to be parsed by the computer is required

- The content of the recommendations might be very different from the original preferences of the user so that the user can also taste items other than his/her previous likes.
- CF techniques are domain independent and can work perfectly in domains where it's hard to extract content information from the items or where there is not content at all associated with the items.

As for the shortcomings of CF method, they can be summarized as the following:

- Most users rate just a few of the items in the collection, which causes the user-item rating matrix to become very sparse. And this leads to a reduced probability of finding a set of users with many ratings in common. This is often called the *sparsity* problem [51].

- When the number of users is small, it becomes difficult to find nearest neighbors, which causes the performance of recommendations drop considerably [28].

- When the system does not have enough information about a user or an item, it cannot provide any recommendation to this user or about this item. This situation is usually called the *bootstrapping* or *cold start* problem. [34] explains three kind of bootstrapping problems including new item, new user, and new system. New user problem occurs when a user is new in the system and he/she has not provided enough information about his/her preferences yet. New item problem occurs when a new item is introduced and none of the users in the system has an opinion about this item. In such a situation, it is not possible for the RS to recommend the new item. And as can be guessed, new system problem is a bigger combination of the previously mentioned problems [33].

- Collaborative RSs tend to fail when there exist little information about preferences or when a user has quite uncommon interests [56]. The sparsity of nearest neighbors for such users cause this failure. Besides, these techniques only work correctly if a reasonable amount of reliable data about user preferences is available online [57].

- Recommendations using collaborative techniques may not correspond to actual preferences but could be biased by the popularity of a certain item [58].

- CF systems produce interesting recommendations only for naïve profiles. They get stuck when bigger profiles are used and they cannot handle heterogeneous profiles correctly [59].

- If two similar items have never been rated by the same user, their relationship is lost. Thus, those two items cannot be classified into the same neighborhood and this will definitely affect the performance of the system [60].

- Although recommendation performance improves when a scale of users and items data increases, large resources are required for computing in this situation, which results in lowering practicality [28].

2.7.3 Hybrid Techniques

The main idea behind hybrid recommendation techniques is that "a combination of algorithms can provide more accurate recommendations than a single algorithm and disadvantages of one algorithm can be overcome by other algorithms" [31]. In order to exploit the advantages of the CB and CF recommendation methods that were mentioned in the previous sections, several hybrid approaches have been proposed, concerning combinations of CB and CF. A significant part of research in hybrid RSs concerns the techniques that can be used to combine the approaches since they may significantly affect the prediction outcome [51].

According to [56], incorporating content into collaborative filtering systems allows increasing the quality of a recommendation system. Besides, when data is too sparse additional content information is a need in order to fit global probabilistic models. The work presented in [50] explains that a method that integrates both ratings and content data enables more accurate recommendations with a richer variety than pure content-based or pure collaborative filtering techniques.

2.7.3.1 Methods for Combining Recommendation Techniques

According to [23], there is a variety of methods to combine collaborative and content-based techniques into a hybrid RS, and they can be classified as below:

2.7.3.1.1 Implementing collaborative and content-based methods separately and combining their predictions:

There exist two ways of combining the separate predictions. As the first way, the outputs (ratings) obtained from individual RSs are combined into one final recommendation using either a linear combination of ratings or a voting scheme. Alternatively, one of the individual recommenders is chosen at any given moment in order to select the one that has a higher performance based on some recommendation quality metric.

2.7.3.1.2 Incorporating some content-based characteristics into a collaborative approach

Several hybrid RSs are based on traditional collaborative techniques but also maintain the content-based profiles for each user. These profiles, and not the commonly rated items, are then used to calculate the similarity between two users. This allows to overcome the sparsity-related problems of a purely CF approach, since typically not many pairs of users will have a significant number of commonly rated items. As another advantage, this approach provides to recommend an item not only when it has been rated highly by users with similar profiles, but also when this item scores highly against the user’s profile.

2.7.3.1.3 Incorporating some collaborative characteristics into a content-based approach

Some dimensionality reduction technique on a group of content-based profiles is used as a very common approach in this category. For instance, a collaborative view of a collection of user profiles represented by term vectors can be created, which results in a higher performance compared to the pure content-based approach.

2.7.3.1.4 Constructing a general unifying model that incorporates both content-based and collaborative characteristics

This approach has been used frequently by many researchers in recent years. An example of this approach is using content-based and collaborative characteristics like age or gender of the users or the genre of movies in a single rule-based classifier [61].

Another classification of hybridization techniques have been proposed by [62]:

- **Weighted:** Each of the recommendation approaches that makes predictions are combined into a single prediction.
- **Switching:** One of the recommendation techniques is selected to make the prediction when certain criteria are met.
- **Mixed:** Predictions from each of the recommendation techniques are presented to the user.
- **Feature Combination:** A single prediction algorithm is provided with features from different recommendation techniques.
- **Cascade:** Output from one recommendation technique is refined by another.
- **Feature Augmentation:** Output from one recommendation technique is fed to another.

- **Meta-level:** Entire model produced by one recommendation technique is utilized by another.

CHAPTER 3

RELATED WORK

In this chapter, the related work in the area will be focused on. In order to achieve this, a variety of recommendations systems from different domains making use of pure content based, pure collaborative filtering and hybrid techniques will be described. In addition to examining the recommendation approaches that these systems are making use of, the pros and cons of these approaches will be discussed in detail so that some knowledge can be gained about how common recommendation problems have been solved so far.

3.1 Pure Collaborative Filtering Methods

The study in [26] focuses on the memory-based collaborative filtering problem. It is stated that sparsity of user-item matrix leads to inaccurate recommendations. In order to handle this problem, an enhanced Pearson Correlation Coefficient (PCC) algorithm, which adds a parameter to overcome the potential decrease of accuracy during user/item similarity computation, is used. In addition, an effective missing data prediction algorithm, in which both user and item information is taken into consideration, is proposed. In this algorithm, some similarity thresholds for both users and items are set so that the prediction algorithm decides whether to make a prediction or not. This research also addresses how to predict the missing data by employing a combination of user and item information.

The evaluation results of [26] show that the proposed method outperforms other state-of-the-art collaborative filtering algorithms and it is more robust against data sparsity.

Throughout the study of [25], the concept of local and global similarity, based on surprisal-based vector similarity and an application of the concept of maximin distance in graph theory are presented. Besides, a user-based collaborative filtering framework based on these concepts is explained.

Surprisal-based vector similarity expresses the relationship between any two users based on the quantities of information (called surprisal) contained in their ratings. The intuition behind this method is that less common ratings for a specific item provides more discriminative information than the most common ones. As for the global similarity, it defines two users as similar if they can be connected through their locally similar neighbors.

As stated before, due to the data sparsity problem, there exist neither a sufficient amount of similar neighbors, nor a sufficient amount of ratings for a particular item. With the help of the concepts introduced in this research, the data sparsity problem is handled in a successful way. Moreover, the proposed framework improves the accuracy of prediction. Under the sparse data set condition, global user similarity improves the performance of the algorithm introduced by [25], which uses only local user similarity. The overall approach explained in [25] is only an improvement of user-based algorithm. Therefore, it asserted that the approaches that make use of both user and item-based algorithms can employ the approach followed by [25] to replace the traditional user-based approach so that they can achieve a higher performance.

As for the study in [65], it presents a Multi-Agent approach to the problem of recommending training courses to engineering professionals. Through user modeling and data collection from a survey, collaborative filtering recommendation is implemented using intelligent agents. The agents work together for recommending meaningful training courses and updating the course information.

The Multi-Agent solution designed and implemented for the recommendation problem in this system consists of two main agents, which are recommendation agent and the information retrieval agent. The Recommendation Agent proposes a personalized list of training modules, and the Information Retrieval Agent searches a predefined list of service providers' websites for course information and updates. The system uses a users profile and keywords from courses to rank courses. A ranking accuracy for courses of 90% is achieved while flexibility is maintained by using an agent that retrieves information autonomously using data mining techniques from websites.

The system achieves high accuracy in ranking using user information and course information. It is stated that the final system is scalable and has possibilities for future modification and adaptability to other problem domains. Further improvements can be made using clustering and recording user feedback as a future work.

Within the project conducted by [29], a recommendation system is built based on multiple CF approaches including item-based K-nearest neighbor, item-based EM, sparse SVD and a mixture of item-based KNN and sparse SVD.

As a contribution, item-based correction and near-integer round-off methods are used in both sparse SVD and EM algorithms. Besides, the EM algorithm is changed from user-based to item-based in this study. The performance of these algorithm and their mixtures are tested on part of Netflix data. With the blend item-based KNN and sparse SVD algorithm, 6.14% improvement of the baseline is maintained.

When the system has a newcomer, that is, a user without any existent ratings, it is very difficult to predict his/her rating on any item. In this case, the average rating of the item is used as his/her prediction. When the predictions that are obtained by the recommendation algorithms are above 5 or below 1, they are truncated, which strictly improves the performance of the system. Besides, the predictions are rounded off if its distance to the nearest integer is less than or equal to 0.1, which again provides a better performance for the RS.

Given predictions as outcomes of some collaborative filtering algorithms, the performance is further enhanced by first applying the newcomer prediction algorithm, then carrying out the item-based correction, then prediction truncation, and at last near-integer round-off. With this sequence of implementation of post-processing tricks, the RMSE of the recommendation system can be reduced by approximately 0.02.

3.2 Pure Content Based Methods

The research in [63] proposes a hybrid recommender system where attributes used for content-based recommendation are assigned weights depending on their importance for the users. These weight values are estimated from a set of linear regression equations obtained from a social network graph that captures human judgment about item similarities. The underlying principle is to use existing recommendations by users to construct a social network graph with items as nodes.

The proposed RS is based on the presumption that feature weights are almost universal for different sets of users and movies. These features along with their weights that are obtained from the regression equations are used to obtain the recommendations.

The evaluation results of this research show that the proposed method agrees well with IMDB recommendation, and it outperforms other pure content-based methods, which can be considered as a demonstration for the effectiveness of feature weighting.

[28] presents a simple and low-cost movie recommendation system harnessing vast cultural metadata about movies on the Web and analyzes the strengths of this system. The system uses 5 types of cultural metadata including user comments, plot outline, synopsis, plot keywords, and genres provided by IMDb (Internet Movie Database) for movie recommendation.

Similarities between movies with the above 5 types of metadata are computed to make a recommendation list. In order to achieve this, a vector space for cultural metadata is created, and a cosine measure method from information retrieval domain in addition to a sense reasoning toolkit developed by MIT Commonsense Computing Initiative is used.

The strength of this approach is that its conceptual process is simple, intuitive, and has a very low-cost. Also it can recommend movies without user preference profiles. Moreover, it provides serendipities by using a few cultural metadata, which represent a variety of emotional, factual, qualitative attributes of movies.

In LIBRA, which is a recommender system presented by [47], the user rates selected titles on a 1 to 10 scale. A naïve Bayesian text-categorization algorithm is used to learn a profile from these rated examples. According to LIBRA, ratings between 6 and 10 have a positive meaning and ratings between 1 and 5 have a negative meaning. The learned profile is used to rank all other books as recommendations based on the computed probability that they are positive. The users can also provide explicit positive/negative keywords, which are used as priors to bias the role of these features in categorization.

In the light of the research conducted by [68], a multi-agent TV recommender system is presented. This system encapsulates three user information streams including implicit view history, explicit preferences and feedback information on specific shows into adaptive agents and generates program recommendation for a TV viewer.

In the system, there exist two implicit recommender agents, one based on Bayesian statistics and one on Decision Trees, both of which are considered alternatives for implicit recommendation. As a future work, it is proposed that some methods can be used to combine them. As for the explicit recommender agent, it relies on explicit profiles of TV viewers. These profiles are obtained from a question-answer session with the viewer and the viewer’s explicit likes and dislikes towards particular TV channels, show genres and preferred days and times of TV viewing are gathered in this session. The third agent in the system is the feedback agent, which works in collaboration with the other agents in order to fine-tune their recommendation quality by enabling the user to provide the implicit and explicit recommenders with show-specific information, which the recommenders lacked previously.

The evaluation results of this study show that the combination of implicit and explicit agents works best in the related environment.

Throughout the study of [24], a framework for content-based film recommendation system where user models from CF recommenders are enhanced is proposed. A vector of explicit ratings on a set of objects is provided by the user to the CB user model represented as a list of preferences of different movie features.

The user models built by the proposed approach facilitate the identification of commonalities in positively or negatively rated objects as derived by the collaborative user models. Movie profiles are stored as a vector of feature weights, which are assigned according to the degree of how well they can discriminate one movie from another considering the characteristics of the whole movie domain. The required features for each movie are collected from IMDB.

With the help of the feature weights and collaboratively built user models, a weighted list of features that are liked or disliked by the user are found out. Thus, user preferences, interests and needs are modelled by using only collaborative-based user model and item profiles. The system also provides the users to view their proposed models, which enables them to become aware of their implicit or unknown preferences.

The conducted experiments show that optimization mechanisms on the built user models result in better recommendations. Focusing on the effect of the whole dimension together with the effect of features only is planned by [24] as a future improvement.

3.3 Hybrid Methods

AVATAR introduced by [64] is a TV recommender suggesting a user contents semantically related to those he/she watched in the past. AVATAR involves user profiles, content-based reasoning and collaborative methods to make recommendations. It also uses an explicit technique which provides an initial profile from the information entered at registration and after that, the feedback information (watched programs, changes in preferences, etc) needed for the explicit interaction and collaboration of the user through specific Web pages. This system also uses a mechanism to represent formally the knowledge of their specific application domain. In the Semantic Web, one of the well-known methodologies intended for this purpose is the *ontologies*. So, the system implements an ontology using the OWL (Web Ontology Language) language, where resources and relations typical in the TV domain are identified by means of classes, properties and specific instances. In order to model the preferences, a dynamic subset of TV ontology is used, built incrementally by adding new classes, properties and instances. Specifically, when AVATAR knows a new content, the system adds to the profile this instance, the class referred to it, the hierarchy of super classes defined in knowledge base related to this class and some properties defining the main characteristics of this TV content.

AVATAR refers to a filtering methodology in order to avoid retrieving a massive amount of meaningless associations. This methodology ignores those instances of the analyzed sequences which are not relevant enough according to the personalization requirements, and retrieves only significant associations. Obviously, the quantification of such relevance depends on the user preferences: if an instance is closely related to the programs the user liked, this one is not filtered and consequently, it will be contained in the inferred semantic associations. This phase returns a set of programs semantically related to the user preferences, together with the type of discovered association.

If the user wants to watch programs similar to ones he liked in the past, AVATAR infers very direct associations between the suggested programs and his/her preferences. On the contrary, if the user does not mind watching varied programs, less similar to those he knows but always related to them, the inference methodology considers indirect associations. So, AVATAR discovers contents especially appealing for the user, which would not be suggested by many other previous approaches.

PTV presented by [66] is a recommender system designed to make TV program suggestions to users based on their learned viewing preferences. PTV profiles created in this system contain lists of positively and negatively rated TV programs. In this recommendation system, users register themselves in the PTV server (Web site) and then they can access to personalized programming guides presented as HTML or WML pages. The system incorporates user profiles, content-based reasoning and collaborative methods to make recommendations. When registering a new user, the system creates a profile which stores preferences about programs, channels, genres, timetables, etc. Although this system seeks similar user profile information, it does not include a 'dynamic' learning algorithm that tracks a person's changing TV preferences over time. However a dynamic approach to this learning problem is essential to capture the evolving personal TV preferences of a viewer.

In PTV like recommendation systems, two main lacks can be identified: the way in which user information is gathered to elaborate the profile, and the inability to maintain historic logs about which programs have been watched by which users. As a result, in this kind of systems, it is not possible to deploy implicit techniques for managing the above information. The only solution is to use an explicit technique which elaborates an initial profile from the information entered at registration, and after that, the feedback information (observed people, changes in preferences, etc.) needed for the explicit interaction and collaboration of the user through specific Web pages. Despite the fact that good recommendations can be achieved by this explicit mechanism, it entails the user to do a lot of work.

Having this information in mind, in this kind of systems it is possible to set up implicit learning techniques about user interests and behavior. Additionally, this information can be continuously improved by taking into account all decisions adopted by the user any time he/she interacts with the system. This allows a great reduction in the information that must be explicitly given by the user. So, in this case, the first time the user turns on the system, he/she must inform only about a few characteristics to build a preliminary profile.

Throughout the study of [67], a web based recommendation system called MatchBook that has been developed by using personalization techniques and combining machine learning and automated reasoning methods is represented.

In Matchbook, the users are able to navigate through the system freely in order to find the right matches for themselves in terms of both character and appearance. In this progress, there are two main concepts that have been covered, namely; machine learning and automated reasoning.

For the machine learning part, by observing just the implicit behavior of the user (i.e. visiting other user's profiles) the interests of the user are learnt. "Add to Favorite List" means a positive example and "Skip to the Other User" means a negative example. According to these positive and negative examples, some hypotheses about the user's taste are maintained by using Candidate Elimination Algorithm. In this manner, the users in the system are personalized so that the system is able to provide them with the appropriate matches. After learning the preferences, the suitable candidates for the user are matched and displayed by means of some automated reasoning techniques. As another machine learning algorithm, k-Nearest Neighbor Algorithm is used in order to find the people sharing the most common interests specified during the registration.

In order to make recommendation to a user, the system finds the relation defined for that user and tries to match the related properties with the users in the system. After the matches are found by the reasoner, Naïve Bayes Classifier is applied and the matches are ordered according to the results calculated from the classification method.

As a shortcoming of the system, when the user first registers to our system, the system does not have any idea about the user’s tastes. For this purpose, a collaborative filtering method is used at this step. However, when there is little number of users, it could be difficult to find someone similar to the specific user.

Another shortcoming of MatchBook is that the users do not always look at the profiles of others since they liked them. Sometimes these data can cause some noise and there can be inconsistency in the training examples and the version space could be converged to an empty set. Therefore, an inconsistency check feature can be added to find the data that causes inconsistency, and to eliminate those data before applying the learning algorithms.

The approach used throughout the study in [69] describes a framework that combines missing data scores with content-based recommendations in order to produce a hybrid recommendation system.

In the first stage, personalized user agents produce recommendations for items with a content-based method. Then, a second agent models the likelihood that the user already finds this item interesting. This model of the missing data is combined with the personalized content agents to form a model of stacked agents using CF.

With the help of the proposed approach, improved results over a baseline content model are maintained. Besides, since the system combines content based methods with collaborative filtering ones, it attacks the cold-start problem in an effective way and outperforms approaches that rely on pure collaborative filtering.

The research in [50] presents a hybrid music recommendation method that solves the problems of both collaborative filtering and content-based recommendation.

The proposed method integrates both rating and content data by using a Bayesian network called an aspect model. Unobservable user preferences are directly represented by introducing latent variables that are statistically estimated.

The evaluation results of this research show that this method outperforms the two conventional recommendation methods in terms of recommendation accuracy and artist variety. High recommendation accuracy is achieved by the reliable modelling of user preferences and the integration of rating and content data. Besides the proposed system attacks the “new item problem” by recommending reasonable pieces even if they have no ratings.

The movie recommendation system called MoRe, presented in [51] also uses a hybrid approach based on content-based and collaborative filtering techniques. The system uses switching and substitute techniques by monitoring certain parameters that trigger either a CB or CF prediction. Besides, an empirical comparison of the hybrid approach to the base methods of CF and CB is provided.

The system collects user ratings concerning movies on one-to-five scale through the graphical user interface. In order to handle the new user problem, the user is asked to provide several ratings after registration. In this manner, the prediction process can be initiated by the system. The two versions of the hybrid algorithm are differentiated by the parameter that controls the switch from CF to CB method.

A web crawler is used in order to seek the movie description features from the website of internet movie database. The system creates the set of most similar movies for all available movies at an offline phase in order to speed-up run time predictions. And the size of the neighbour set is determined by the system administrator.

In order to make movie recommendations, CF uses the ratings matrix whereas the content-based predictor uses mainly the movie data files. As for the hybrid methods, they make use of both CB and CF engines. Although the system is able to produce recommendations based on more than one method, only one method which is specified by the administrator is applied at any given time.

Users of this system receive the recommendations in a ranked list of movies where the prediction appears to the user in a “five-star” scale, while users provide their feedback directly on the recommended movies.

MOVIES2GO is a web-based movie recommender system introduced by [21], which catches and reasons with user preferences to pro-actively recommend movies. The approach used in the system combines voting based ranking procedure with guaranteed properties that use syntactic features like actor/actress of movies together with a learning based approach that processes semantic features of movies like its synopsis.

Throughout the system, a reasoning procedure that can meaningfully and systematically trade off between user preferences is developed. Multiple query modalities by which the user can pose unconstrained, constrained, or instance-based queries are also provided.

It is claimed in the related research paper that even in the presence of conflicting user preferences, the system provides satisfactory recommendations by using voting theory-based reasoning scheme.

As for the study in [70], it presents a framework for combining content and collaboration. Personalized suggestions through CF are maintained by using a content-based predictor to enhance existing user data.

Sparsity and first-rater problems, which are the drawbacks of the CF systems, are overcome by exploiting content information of the items already rated. Basically, content-based predictions are used to convert a sparse user ratings matrix into a full ratings matrix, and then CF is used to provide the recommendations.

The content information of each movie is collected from IMDb by using a simple crawler and the content information of each movie is represented as set of slots (features), each of which is simply a bag of words. The features that are used for this system are movie title, director, cast, genre, plot, summary, plot keywords, user comments, external reviews, newsgroup reviews, and awards.

The conducted experiments show that content-boosted collaborative filtering performs better than a pure content-based predictor, pure collaborative filter, and a naïve hybrid approach. Naïve Bayesian text-classifier currently used as a content-based predictor is considered to be not ideal, since it disregards the fact that classes represent ratings on a linear scale. This problem is planned to be overcome by using a learning algorithm that can directly produce numerical predictions, such as logistic regression and locally weighted regression. In addition, the CF component is expected to improve by using a Clustered Pearson Predictor.

The main objective of the work in [33] is to find out whether the main disadvantages of content-based and collaborative filtering recommendation methods can be solved using hybrid methods. In order to do this, a hybrid recommendation system, which possesses the best characteristics of both methods and produce better results than each method individually, is built.

At the end of this study, it is concluded that although CB methods present reduced item variety, they provide more accurate recommendations than other methods and they normally succeed on items that CF cannot predict. Other conclusions drawn for pure recommendation methods are related with the fact that most of the computations required to make predictions can be done offline so that predictions can be made easily online while the user is waiting for a fast response. It is also concluded that CF methods can outperform CB ones with the help of correct amount of data. As for the main conclusion, content-based and collaborative filtering methods are proven to be complementary ones. CB techniques are stated to solve problems when the users' preferences data is inadequate or inexistent, whereas CF techniques can obtain relations between songs that are simply impossible to obtain by using only the audio files.

The system described in [27], MovieReco, introduces the concept of testing the knowledge of the user to filter out "bad users". The overall research focuses on the mechanism used to provide robust and effective recommendations.

In MovieReco, recommendations are made available to the user as soon as he/she logs in and a film is strongly recommended, recommended or not recommended based on the predicted rating. It is explained that CF approach totally depends on the ratings of different users and if any user rates a movie blindly without watching the film, it may affect the result. Therefore, the user is asked some questions about the movie while rating a movie so that bad users can be filtered out by the system. And since users may user different styles of rating for the same level of interest, each user is asked to rate the best movie he/she has ever seen. As for technique used for filtering in this system, it is basically a combination of Item-Item and Pearson-r algorithm.

3.4 Discussion

By means of making a deep literature survey and a detailed review of the current recommender systems, the "state of the art" topics in RSs and the most important problems to be dealt with in this area have been realized.

It has been found out that there are so many systems and research studies trying to cope with the various problems of recommender systems by using many different approaches. Due to its success in many cases that were mentioned in the previous sections, collaborative filtering has been the most popular one among these approaches for a long time. However, in order for CF to be successful, it requires a considerable amount of rating information, which can be named as the data sparsity problem. Besides, CF suffers from the new item problem where it is not possible to make a rating prediction for the items that have not been rated by any user yet. On the other hand, the technique of using the content information of the items has been proven to overcome these problems by many related studies.

To achieve the goal of minimizing the disadvantages of CF and CB approaches when used separately and creating a stronger prediction mechanism, latest systems make use of hybrid recommender techniques by integrating multiple approaches.

This idea of improving predictions by integrating the content information to the collaborative filtering techniques has inspired the overall approach of this thesis work in a great amount. Especially the study in [63], which has a presumption that some feature weights are almost universal for different sets of users and movies, forms the basis of the approach in ReMovender in terms of the usage of content information. In addition, Global Similarity and Effective Missing Data Prediction concepts, which were previously explained in section 3.1 as techniques attacking the user item data sparsity problem, have a great importance for the constitution of ReMovender's prediction mechanism.

CHAPTER 4

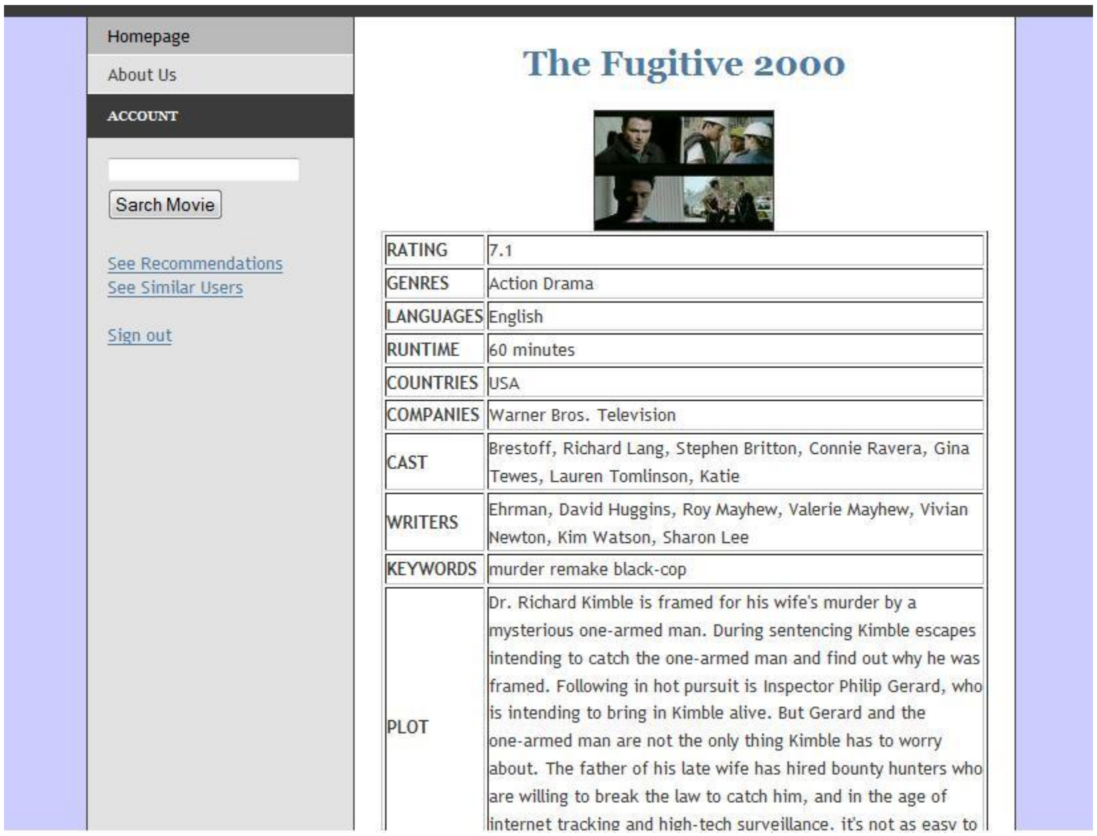
ReMovender: A CONTENT BOOSTED COLLABORATIVE FILTERING MOVIE RECOMMENDER

4.1 System Overview

ReMovender is a web based movie recommendation system where people can freely navigate through, make comments on, and give rating to the movies. Besides, the users of this system are able to search for specific movies, and make discussions about movies with the other users. In the meantime, the system tracks the actions of the users and tries to learn their movie taste in order to make some movie recommendations to them.

In addition to these features, all users in the system have the chance of meeting the similar users who share the same kind of movie preferences. ReMovender also provides the users to see similar movies to a specific one.

The users of ReMovender are capable of viewing the details of a specific movie. These details include the information about movie’s genres, languages, countries, companies, writers, keywords, runtime, cast, plot and rating on IMDb, which is the current biggest movie database in the world. And the related interface for displaying the movie details can be seen in Figure 3.



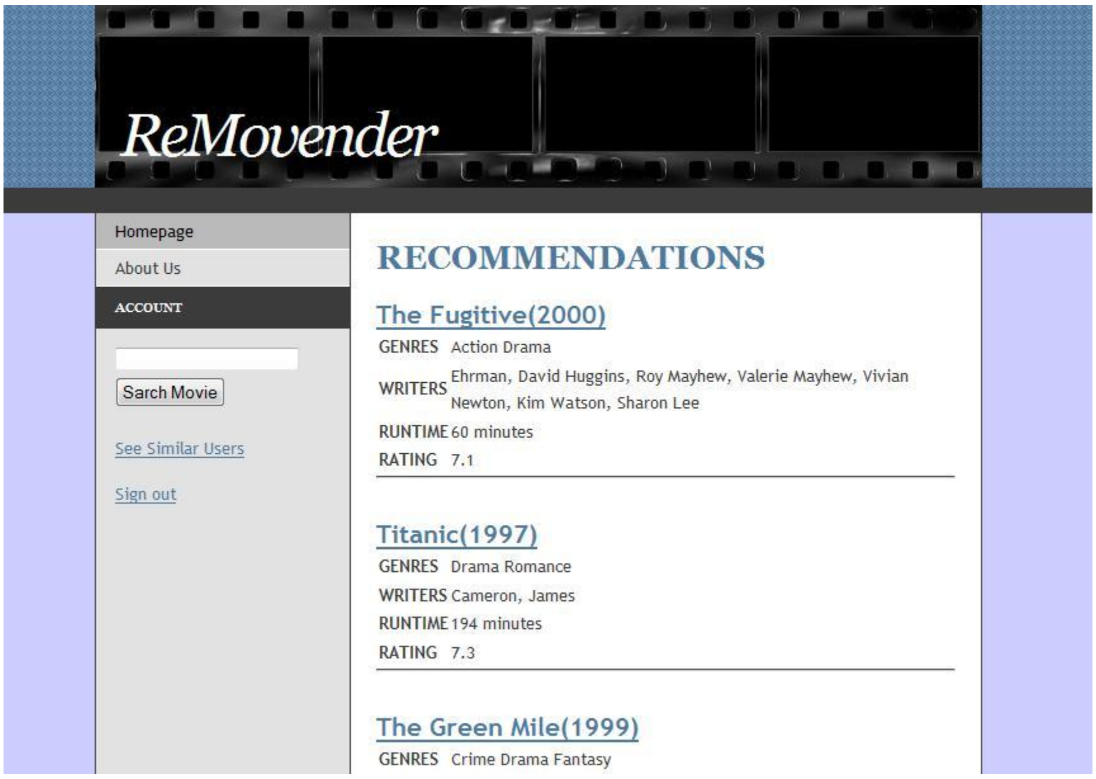
The Fugitive 2000

RATING	7.1
GENRES	Action Drama
LANGUAGES	English
RUNTIME	60 minutes
COUNTRIES	USA
COMPANIES	Warner Bros. Television
CAST	Brestoff, Richard Lang, Stephen Britton, Connie Ravera, Gina Tewes, Lauren Tomlinson, Katie
WRITERS	Ehrman, David Huggins, Roy Mayhew, Valerie Mayhew, Vivian Newton, Kim Watson, Sharon Lee
KEYWORDS	murder remake black-cop
PLOT	Dr. Richard Kimble is framed for his wife's murder by a mysterious one-armed man. During sentencing Kimble escapes intending to catch the one-armed man and find out why he was framed. Following in hot pursuit is Inspector Philip Gerard, who is intending to bring in Kimble alive. But Gerard and the one-armed man are not the only thing Kimble has to worry about. The father of his late wife has hired bounty hunters who are willing to break the law to catch him, and in the age of internet tracking and high-tech surveillance, it's not as easy to

Figure 3 - Movie page

The user is also able to rate any movie with the help of this interface. The profile of a user in the system is created automatically from the ratings this user has given for the movies that he/she has watched before. Each time a user gives a rating for a movie, the profile of the related user is updated accordingly so that the predictions in the future can be more successful and satisfactory.

Each time the user logs in to the system, some movie recommendations are provided to the user as a list, which has been created by various prediction techniques of ReMovender. The predicted ratings of the user for the movies in this list are in decreasing order so that the most appealing ones occur at the top. The related screen can be observed from Figure 4:



RECOMMENDATIONS

The Fugitive(2000)

GENRES	Action Drama
WRITERS	Ehrman, David Huggins, Roy Mayhew, Valerie Mayhew, Vivian Newton, Kim Watson, Sharon Lee
RUNTIME	60 minutes
RATING	7.1

Titanic(1997)

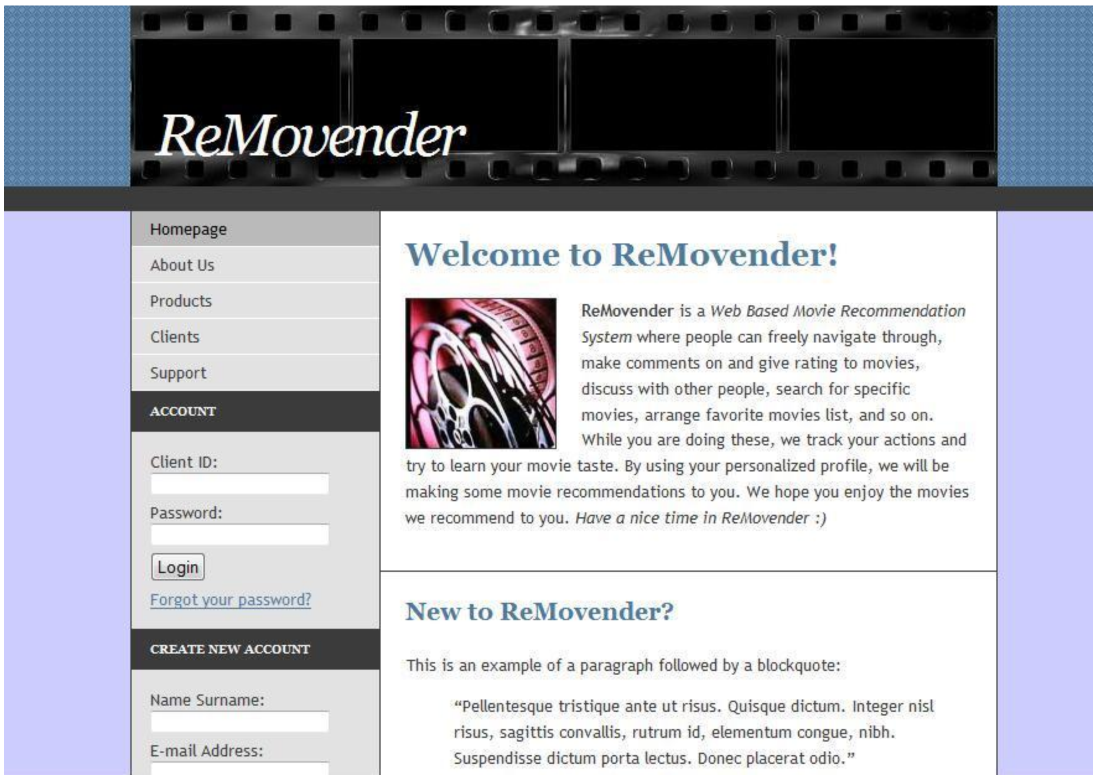
GENRES	Drama Romance
WRITERS	Cameron, James
RUNTIME	194 minutes
RATING	7.3

The Green Mile(1999)

GENRES	Crime Drama Fantasy

Figure 4 - Recommendation page

As for the screen designed as the main page of ReMovender, it can be found below:



Welcome to ReMovender!

ReMovender is a *Web Based Movie Recommendation System* where people can freely navigate through, make comments on and give rating to movies, discuss with other people, search for specific movies, arrange favorite movies list, and so on. While you are doing these, we track your actions and try to learn your movie taste. By using your personalized profile, we will be making some movie recommendations to you. We hope you enjoy the movies we recommend to you. *Have a nice time in ReMovender :)*

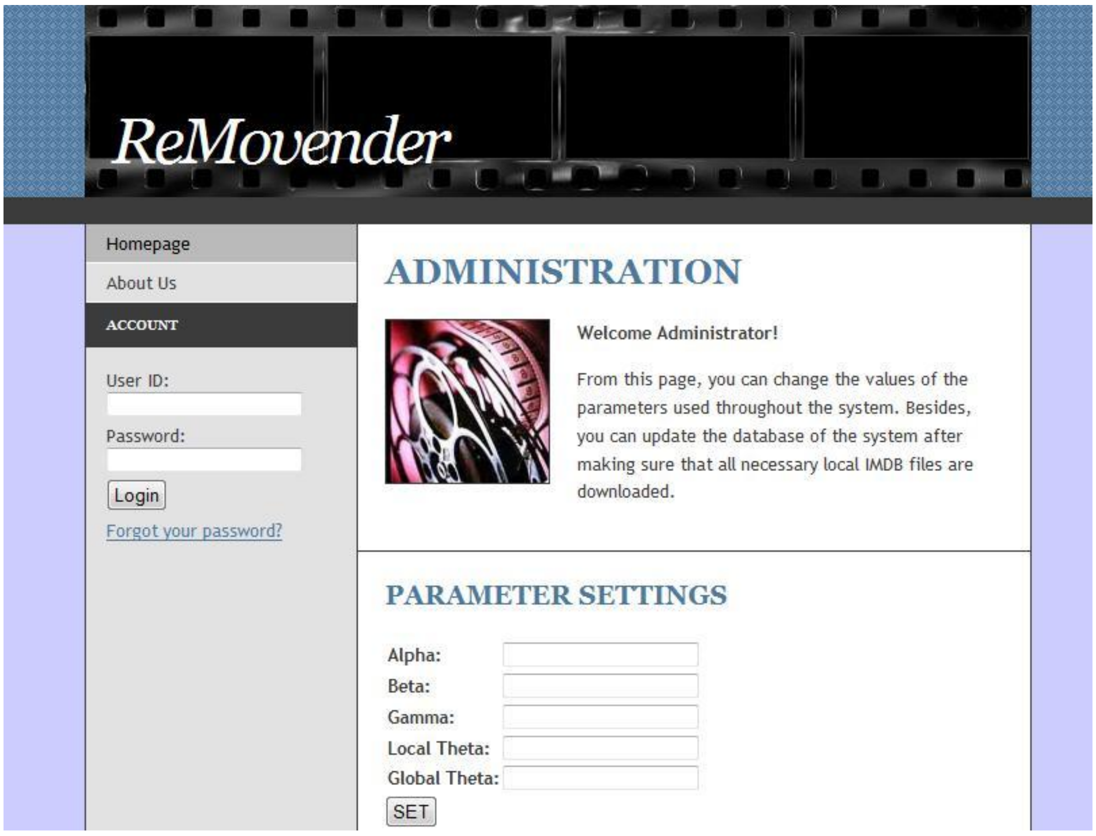
New to ReMovender?

This is an example of a paragraph followed by a blockquote:

"Pellentesque tristique ante ut risus. Quisque dictum. Integer nisl risus, sagittis convallis, rutrum id, elementum congue, nibh. Suspendisse dictum porta lectus. Donec placerat odio."

Figure 5 - Main page

ReMovender has also a page designed for the system administrator to enable him/her to set some parameters that are used throughout the prediction phase. With the help of this interface, the administrator can also update the movie database, the recommendations for the users in the system, and the information about similar users and movies. These processes are completed offline in order to decrease the response time of the system for providing the user with the necessary recommendations, and similarity information. The related interface is shown in Figures 6 and Figure 7 as two parts:


ReMovender

Homepage

About Us

ACCOUNT

User ID:

Password:

Login

Forgot your password?

ADMINISTRATION

Welcome Administrator!

From this page, you can change the values of the parameters used throughout the system. Besides, you can update the database of the system after making sure that all necessary local IMDB files are downloaded.

PARAMETER SETTINGS

Alpha:

Beta:

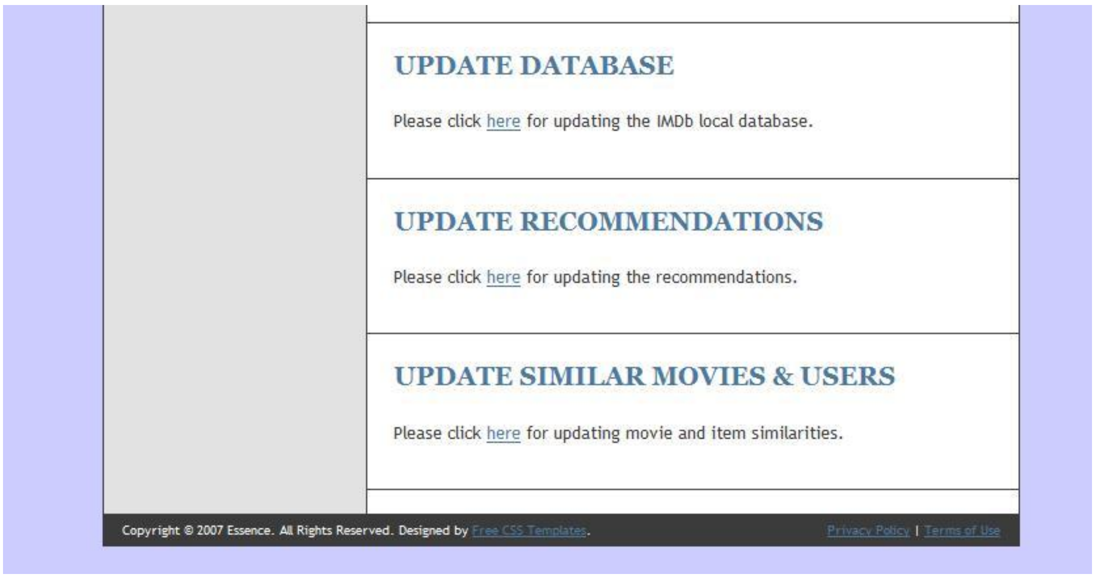
Gamma:

Local Theta:

Global Theta:

SET

Figure 6 - Administration page part 1


UPDATE DATABASE

Please click here for updating the IMDb local database.

UPDATE RECOMMENDATIONS

Please click here for updating the recommendations.

UPDATE SIMILAR MOVIES & USERS

Please click here for updating movie and item similarities.

Copyright © 2007 Essence. All Rights Reserved. Designed by Free CSS Templates. Privacy Policy | Terms of Use

Figure 7 - Administration page part 2

4.2 System Architecture

The overall design of ReMovender can be seen in Figure 8. The system functionality is maintained by three main components.

1. **Information Extractor:** This component is used to store the necessary metadata about all movies in IMDb to the local database of ReMovender by means of information extraction techniques.

2. **User Interface:** With the help of this component, the interaction of all the users including the system administrator with the system is maintained. The administrator can update the movie database; calculate similarities and predictions offline via user interface. In addition, by tracking the behavior of the user throughout the system like rating movies, making comments, etc., the necessary updates are made in the knowledge base of the system.

3. **Recommender:** This component makes the appropriate recommendations to the user by making use of movie content and user profile (rating) information with the help of some prediction techniques.

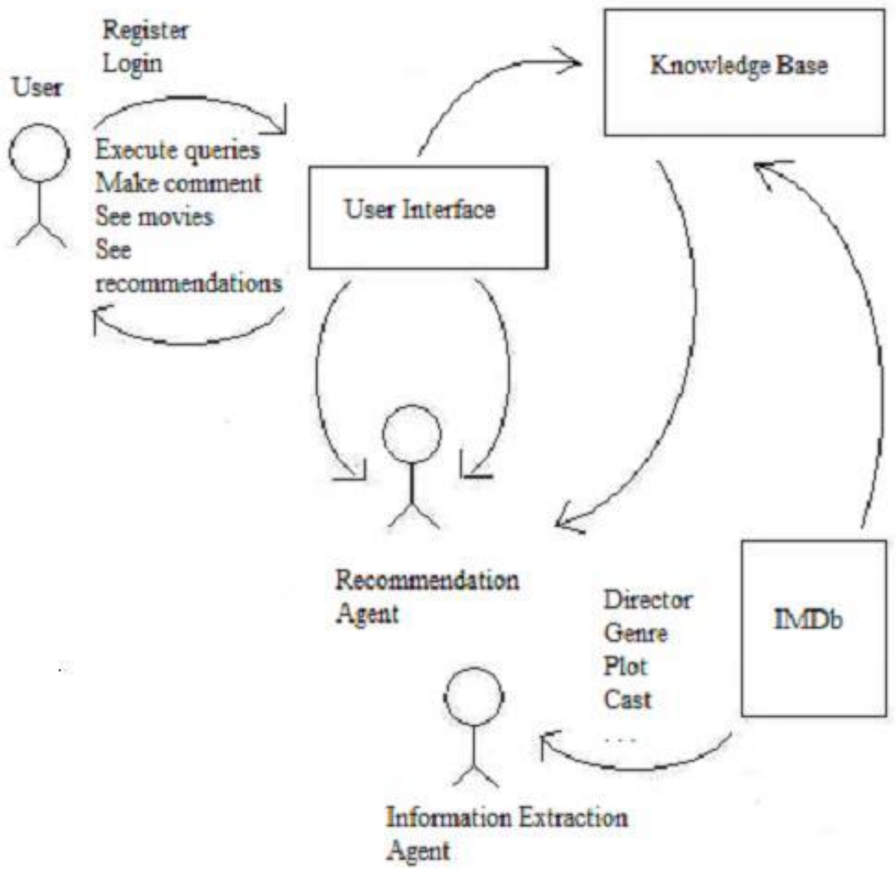


Figure 8 - Overall design of the system

4.3 Design Issues

4.3.1 General Description of the Proposed Approach

In general, ReMovender uses a content boosted collaborative filtering approach during the process of predicting the rating information of a user for a specific movie. More specifically, it attacks the data sparsity problem with the help of *Effective Missing Data Prediction* and *Global User Similarity* concepts.

The collaborative filtering approach is strengthened by using both an item and user based approach into which the global user similarity is integrated. While *Global User Similarity* provides to overcome the difficulty in making prediction under sparse user data, *Effective Missing Data Prediction* determines whether to predict the missing data by using information of user, items, or both. In addition, by taking the content information of the movies into consideration during the calculation of item similarity, the quality of the recommendations is increased.

4.3.2 User Modeling

As stated previously, the main prediction technique that is used in ReMovender is Collaborative Filtering. The input to a CF system is a matrix of users' ratings on a set of items, where each row represents ratings of a single user and each column ratings for a single item [24].

Given a CF recommendation system consisting of M users and N items, there exist an M x N user-item matrix R. Each entry $r_{m,n} = x$ of this matrix represents the rating that user m gives to item n, where $x \in \{1,2,..,r_{max}\}$ [25]. Since the possible ratings are in a range of [1,5] in ReMovender, r_{max} has the value 5. This user-item matrix can be decomposed into row vectors like below:

$R = [u_1, u_2, \ldots, u_M]^T$, $u_M = [r_{m,1}, r_{m,2}, \ldots, r_{m,N}]^T$ where $m = 1, 2, \ldots, M$. Here, the row vector u_M represents the ratings of user m for all of N items. This row vector can be considered as the user model in ReMovender.

4.3.3 Domain Description

The movie domain has been chosen for the application of a content-boosted collaborative filtering recommendation system. The first reason for this choice is that the content information of movies can be easily accessed through information extraction techniques. Secondly, various datasets that provide to evaluate the success of a movie recommender system are currently available.

The cultural metadata about each movie is stored in the database of ReMovender. This is achieved by the information extraction module of the system, which creates a local copy of the IMDb database. Details of these features and the extracted knowledge will be explained in the following section.

4.3.4 Features and Dimensions

In order to make the content information available for the Content Boosted Collaborative Filtering approach, each movie is represented by a set of features where each feature belongs to a dimension. The dimensions used throughout ReMovender are type, rating, production year, runtime, country, cast, genres, languages, companies, writers, keywords, and plots.

Although all of these dimensions are used for displaying the information about a movie on the user interface, only the dimensions type, writer, genre, keyword, cast, country, language, and company are required for the prediction process.

A movie can have a set of possible features for some of the dimensions, whereas it can have a single value for some of them. To illustrate, it’s possible for a movie to be written by more than one writer, whereas the runtime information of a movie consists of only one string.

In table 2, the set of dimensions used in ReMovender along with their type, domain and distance measures are shown. Distance measures are available only for the dimensions used in the prediction mechanism.

The system uses two possible methods for the distance measure calculation. The first method is valid for calculating the distance between features with string type. If the two values are equal, the distance is calculated as 1, else it is calculated as 0. As for the second method, it is appropriate for calculating the distance between features with string list type and the result is equal to the cardinality of the intersection of the two lists divided by the cardinality of the first list. Although the divisor for the related calculation in [63] is specified as the cardinality of the list with maximum cardinality, ReMovender uses this approach in order to preserve equality. To illustrate, let’s suppose that the cast of movie A consists of Kate Winslet, Jack Nicholson, and Demi Moore, the cast of movie B consists of Kate Winslet, Demi Moore, and Brad Pitt, and lastly the cast of movie C consists of Kate Winslet, Demi Moore, Angeline Jolie and Richard Gere. While calculating the similarity of movie A to the other movies in the system, the similarity between A and B should not be calculated as greater than the one between B and C just because the cast of the movie C is more crowded. The similarity calculation mechanism in ReMovender considers these two similarities as equal since the number of common actors or actresses is the same.

Table 2 - Dimensions used in ReMovender

Dimension	**Type**	**Domain**	**Distance Measure**
Rating	Integer	[1,5]	
Production year	Integer	1913, 1986, etc.	
Runtime	Integer	30, 60, 90, etc.	
Type	String	Movie, TV, etc.	$T_1 = T_2 ? 1 : 0$
Country	String	France, Italy, etc.	$\frac{\lvert C_1 \cap C_2\rvert}{\lvert C_1\rvert}$
Cast	String List	Natalie Portman, Mel Gibson, etc.	$\frac{\lvert C_1 \cap C_2\rvert}{\lvert C_1\rvert}$
Genre	String List	Comedy, etc.	$\frac{\lvert G_1 \cap G_2\rvert}{G_1}$
Language	String List	English, etc.	$\frac{\lvert L_1 \cap L_2\rvert}{\lvert L_1\rvert}$

Company	String List	Warner Bros, etc.	$\frac{\lvert C_1 \cap C_2\rvert}{\lvert C_1\rvert}$
Writer	String List	Vivian Newton, Kim Watson, etc.	$W_1 = W_2?1 : 0$
Keyword	String List	Murder, love, etc.	$\frac{\lvert K_1 \cap K_2\rvert}{\lvert K_1\rvert}$
Plot	String List		

4.3.5 Data Representation

As previously stated, all the cultural metadata of the movies required for the user interface and the prediction mechanism are extracted from IMDb. This information is stored in ReMovender's database. In addition there are two more tables including the *users* table, which keeps the necessary information about users, and the *ratings* table, which stores the user ratings that are given for the movies.

In Table 3, a list of the tables in the database of ReMovender is given with some brief explanation. Although the tables obtained by the information extraction module are exactly the copies of the ones in IMDb, only the tables and columns used throughout the system are included in Table 3.

Table 3 - Database of ReMovender

TABLE	COLUMNS	SUMMARY
title	id, title, kind_id, production_year	Stores title, kind, and production year information of a movie

cast_info	id, person_id, movie_id, person_role_id, role_id	Stores cast information of a movie
company_type	id, kind	Stores all available types of companies including distributor, production, special effects, and miscellaneous
info_type	id	Stores all available types of information about a movie, including runtime, genre, language, etc.
movie_info_idx	movie_id, info_type_id, info	Stores the information about distribution, number of rates and rating for each movie
keyword	keyword, id	Stores all possible keyword values
movie_keyword	movie_id, keyword_id	Stores the keywords for each movie
name	id, name, imdb_index	Stores the names of actors and actresses. The column imdb_index is used for distinguishing the ones with the same name.
role_type	id, role	Stores whether the role of the person in the movie is an actor, actress, writer,

		etc.
company_name	id, name	Stores the names of the companies
movie_companies	id, movie_id, company_id, company_type_id	Stores the company information for each movie
Users	user_id, password, name, nickname, email, gender, birth, occupation, location	Stores the necessary information about users in ReMovender
Ratings	user_id, movie_id, rating, type	Stores the user ratings for the movies. *type* column provides to distinguish whether the rating is predicted or real.
movielensToIMDb	movielens_Id, imdb_id	Stores the imdb id values for the movies in MovieLens dataset
movielens_movies	movie_id, movie_name_year	Stores the id, name and year information for the movies in the MovieLens dataset

Table 3 – Database of ReMovender

4.4 System Components

In this section, the details of the Information Extractor and Recommender components will be presented.

4.4.1 Information Extractor

As stated before, information extractor in ReMovender extracts all necessary metadata about the movies in IMDb. In order to achieve this, a Python package called IMDbPY is used. IMDbPY is a free and open source software which is platform-independent and written in pure Python. With the help of this package, data from both the IMDb’s web server and a local copy of the whole database can be retrieved. The reason for creating a copy instead of using web crawling techniques is to make ReMovender a real recommender system application.

The imdbpy2sql.py script in this package is used to populate a database using the data in the IMDb’s plain text data files. SQLObject package, which is a popular Object Relational Manager for providing an object interface to the local database, is also required for this aim. Since MySQL is stated to be the fastest database for this extraction process, MySQL is used in ReMovender. In addition, as InnoDB is a lot slower for this process, MyISAM tables have been used to improve the performance. At the end of the extraction process, database requires between 2.5 and 5 GB of disc space.

In order to achieve the goal of making ReMovender an up-to-date recommender system, the related script is converted into an executable file so that the system administrator can update the database from time to time through the user interface just after downloading the IMDb’s plain text data files that he/she desires. With the help of this feature, the information about the newly released movies and other new information about actors/actresses, companies, writers can be extracted.

In addition to the creation of a local copy of IMDb’s database, the thumbnails of the movies to be displayed in the user interface are extracted from the web via Google AJAX Search API, since the images are copyrighted and IMDb does not provide this information itself. Google AJAX Search API puts Google Search in web pages with JavaScript. With the help of this API, the information extractor dynamically searches the movie thumbnail in Google Images by using the name and year of the movie as the search keywords. The first search result is displayed as the thumbnail on the related movie page.

4.4.2 Recommender

4.4.2.1 Content Boosted Collaborative Filtering

ReMovender uses both user and item based, and content boosted collaborative filtering approach as a prediction technique. In general, the user's rating for an item is predicted by processing both the content of that item and the rating information of similar users and similar items.

As explained previously, user-based CF predicts the missing data by using the ratings of similar users, whereas item-based CF makes a prediction by using the ratings of similar items. And ReMovender uses a combination of these approaches in order to avoid the possibility of ignoring some valuable information that will make the prediction more accurate.

The details of the prediction and recommendation processes will be explained in the following sections.

4.4.2.2 Significance Weighting in PCC

During the user and item similarity calculations, PCC method is used since it takes the factor of the differences in user rating styles into account. However, as stated in [71], PCC overestimates the similarities of users who happen to have rated a few items identically, but may not have similar overall preferences. So, as proposed in [26], a correlation significance weighting factor is added in order to devalue the similarity weights that are based on a small number of co-rated items.

According to this, the new user similarity calculation, where $I_a \cap I_u$ is defined as the number of items rated in common by user a and user u, becomes as below:

$$CollabSim(a,u) = \frac{\mathrm{Min}(|I_a \cap I_u|, \gamma)}{\gamma} . Sim(a,u) \qquad \textbf{(9)}$$

And the updated formula for the item similarity calculation, where $U_i \cap U_j$ is defined as the number of users who rated both item i and item j, is as below:

$$CollabSim(i,j) = \frac{\mathrm{Min}\left(|U_i \cap U_j|, \delta\right)}{\delta} . Sim(i,j) \qquad \textbf{(10)}$$

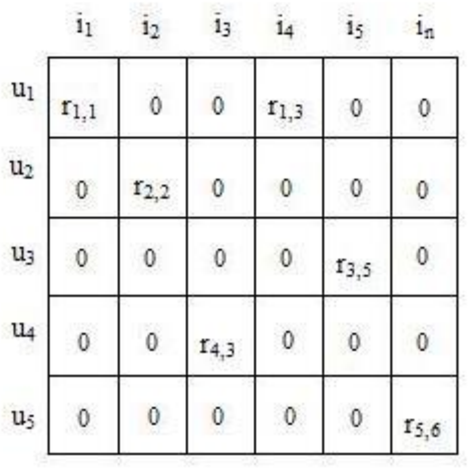
4.4.2.3 Content Boosted Item Similarity Calculation in ReMovender

Traditional collaborative filtering approaches do not take the content information of the two items into account while calculating the similarity of these two items. As explained previously, the PCC algorithm for the similarity calculation between two items is:

$$\mathrm{CollabSim}(\mathrm{i},\mathrm{j}) = \frac{\sum_{\mathrm{u} \in \mathrm{U(i)} \cap \mathrm{U(j)}} \left(r_{u,i} - avg(r_i)\right).\ \left(r_{u,j} - avg(r_j)\right)}{\sqrt{\sum_{\mathrm{u} \in \mathrm{U(i)} \cap \mathrm{U(j)}} \left(r_{u,i} - avg(r_i)\right)^2 . \sum_{\mathrm{u} \in \mathrm{U(i)} \cap \mathrm{U(j)}} \left(r_{u,j} - avg(r_j)\right)^2}} \qquad \textbf{(11)}$$

This algorithm can work without any problem for a very dense user-item matrix. However, there might be very crucial problems when the algorithm deals with sparse data. To illustrate this, let's assume that the user-item matrix is as in Figure 9 and we are trying to find the neighbors of i_2 by using PCC.

As can be observed from the figure, i_2 is most probably a newly released movie since it has been rated by only one user, u_2. And u_2 is most probably a new user in the system since he/she has rated only one movie so far. In order for PCC to be able to calculate a similarity between i_2 and another item, there must be a subset of users who both rated i_2 and the other item.



	i_1	i_2	i_3	i_4	i_5	i_n
u_1	$r_{1,1}$	0	0	$r_{1,3}$	0	0
u_2	0	$r_{2,2}$	0	0	0	0
u_3	0	0	0	0	$r_{3,5}$	0
u_4	0	0	$r_{4,3}$	0	0	0
u_5	0	0	0	0	0	$r_{5,6}$

Figure 9 - Another example of user item matrix

However, in that situation, PCC cannot find any user who has rated an item together with i_2, which causes the algorithm not to be able to return any neighbor for i_2.

To overcome such problematic cases, ReMovender also processes the content information of the items while calculating the similarity of them.

It is asserted in [63] that human judgment of similarity between two items often gives different weights to different attributes. For example, while choosing a movie to watch, the writer of a movie can be more important than the genre. Thus, users base their preferences on some latent criteria, which is a weighted linear combination of the differences in individual attribute. Accordingly, [63] defines similarity S between items I_i and I_j as:

$$ContentSim(I_i, I_j) = w_1.f(A_{1i}, A_{1j}) + w_2.f(A_{2i}, A_{2j}) + \cdots + w_n.f(A_{ni}, A_{nj}) \quad \textbf{(12)}$$

where w_n is defined as the weight given to the difference in value of attribute A_n between the items and the distance between the attributes is given by $f(A_{ni}, A_{nj})$. The distance measures defined in Table 2 are used for the attribute distance calculations in ReMovender. These measures provide the f's to return a value in range [0,1].

As for the feature weights, the mean values that [63] estimates from a social network graph of items are used in ReMovender. The list of these weights can be found in Table 4. This estimation is based on the presumption that feature weights are almost universal for different sets of users and movies. To test this presumption, different sets of regression equations have been considered and they have been solved for the weights by [63].

Table 4 - Feature weight values

Feature	**Mean**
Type	0.18
Writer	0.36
Genre	0.04
Keyword	0.03
Cast	0.01
Country	0.07
Language	0.09
Company	0.21

As a conclusion, the formula for the overall item similarity calculation used in ReMovender is:

$$OverallItemSim(i,j) = (1 - \beta).CollabSim(i,j) + \beta.ContentSim(i,j) \qquad \textbf{(13)}$$

CollabSim(i,j) is calculated by the formula in section 4.4.2.2 and the parameter β in this formula determines the extent to which the item similarity relies on collaborative filtering methods or content similarity. With β=0, it indicates that the similarity depends completely on collaborative similarity, whereas it depends completely on content similarity with β=1.

4.4.2.4 Local Neighbor Selection

The process of selecting the similar users and items has a great importance for the overall prediction mechanism. However, commonly used Top-N algorithm generates a lot of dissimilar users and if selected neighbors are not very similar with the current user, missing data prediction mechanism will calculate inaccurate values. In order to overcome this problem, ReMovender makes use of thresholds introduced in [26] by adding an update to the related algorithm. If the similarity between the neighbor and the current user is larger than η, then this neighbor is added to the potential neighbor list which is sorted in terms of similarity values. And the real neighbors of the user are determined as the minimum of N and the size of the potential neighbor list. The item similarity calculations are made just in the same way.

The pseudo-code for the local user neighbor selection algorithm is given below:

```
UserList findLocalNearestUserNeighbors (User user1, UserList trainingUserList,
                                        float threshold, int numberOfNeighbors)
    UserList potentialSimilarNeighbors = {}
    for each u ∈ trainingUserList
        if ( similarity (user1,u) > threshold )
            add u to potentialSimilarNeighbors
    sort potentialSimilarNeighbors in decreasing similarity order
    sublistSize = min (size (potentialSimilarNeighbors),numberOfNeighbors)
    return the first sublistSize elements of potentialSimilarNeighbors
```

And the pseudo-code for the local item neighbor selection algorithm is given below:

```
ItemList findLocalNearestItemNeighbors (Item item1, ItemList movieList,
                                        float threshold, int numberOfNeighbors)
    ItemList potentialSimilarNeighbors = {}
    for each m $\in$ movieList
        if ( similarity (item1,m) > threshold )
            add u to potentialSimilarNeighbors
    sort potentialSimilarNeighbors in decreasing similarity order
    sublistSize = min (size (potentialSimilarNeighbors),numberOfNeighbors)
    return the first sublistSize elements of potentialSimilarNeighbors
```

4.4.2.5 Local & Global User Similarity

These concepts are introduced by [25] in order to address the data sparsity problem so that two users can become more similar if they can be connected through their locally similar neighbors. Although [25] makes use of surprisal-based-vector space similarity for local user similarity calculations, PCC method has been preferred in ReMovender.

As stated before, the formula for user similarity calculation with PCC is:

$$\mathrm{Sim}(\mathrm{a},\mathrm{u}) = \frac{\sum_{\mathrm{i}\in\mathrm{I(a)}\cap\mathrm{I(u)}}(r_{a,i}-avg(r_a)).\ (r_{u,i}-avg(r_u))}{\sqrt{\sum_{\mathrm{i}\in\mathrm{I(a)}\cap\mathrm{I(u)}}\ (r_{a,i}-avg(r_a))^2}\ .\ \sqrt{\sum_{\mathrm{i}\in\mathrm{I(a)}\cap\mathrm{I(u)}}\ (r_{u,i}-avg(r_u))^2}} \qquad \textbf{(14)}$$

Just like the problem explained in 4.4.2.3, when there exist no sufficient amount of items which user a, and user u both rated, this formula may not return realistic similarity values.

With the help of global similarity, more neighbors of an active user can be found when he/she has few or no immediate neighbors using local user similarity (using

PCC). In another sense, global user similarity prevents to underestimate the similarity of users who have not rated common items. And the global similarity between two users is evaluated when the local similarity between them is below the determined threshold value.

In order to fulfill this goal, first a user graph is constructed using the users as the nodes and the local similarity values as the weight of edges. Since there is a possibility that the local similarity calculated by PCC has a negative value, the negative values are set to 0. Then, the maximin distance of two users in the graph is calculated as the global similarity value between them.

In ReMovender, the Floyd-Warshall algorithm is adopted to effectively compute all-pairs of maximin distances. The details will be explained in the following section.

4.4.2.6 Calculation of Global Similarity Using Floyd-Warshall

Given an RS consisting of M users, we can construct an MxM user-user matrix R, each entry of which represents the local similarity as the weight of edges, as previously explained.

For the Floyd-Warshall recurrence, we can define c^k_{ij} as the maximin distance between two nodes i and j with intermediate vertices belonging to the set {1,2,...,k}.

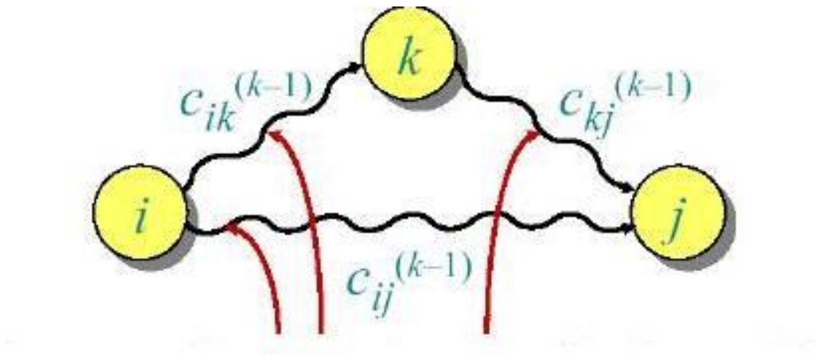


Figure 10 - Floyd Warshall

Thus, the recurrence becomes:

$$c^k{}_{i.j} = max^k\{ c^{k-1}{}_{i.j}, \min\left(c^{k-1}{}_{i.k}, c^{k-1}{}_{k.j}\right)\} \quad \textbf{(15)}$$

And the pseudo-code for Floyd's algorithm, which runs in $O(n^3)$ can be written as follows:

Procedure: Floyd (W[1:m,1:m], P[1:m,1:m], S[1:m, 1:m])

Input: W[1:m,1:m], weight matrix which has been constructed according to the weighted graph G mentioned in 4.4.1.

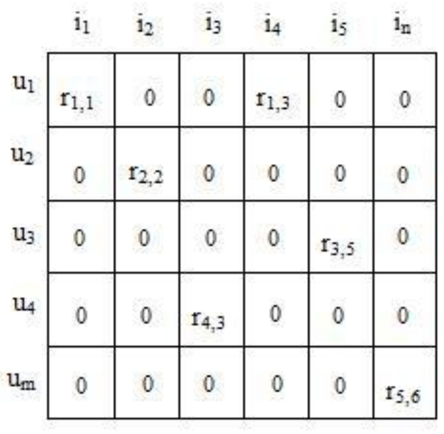
Output: P[1:m,1:m] matrix implementing maximin distance path

S[1:m, 1:m], distance matrix, where S[u,v] is the length(cost) of a maximin distance of u to v in G.

```
for i=1 to n do { //Initialize P and S
    for j=1 to n do {
        P[i,j]=0;
        S[i,j]=W[i,j]
    }
}
for k=1 to n do //Update S and P using the recurrence relation
    for i=1 to n do
        for j=1 to n do {
            if min(S[i,k],S[k,j]) > S[i,j] {
                P[i,j]=k
                S[i,j]=min(S[i,k],S[k,j])
            }
        }
EndFloyd
```

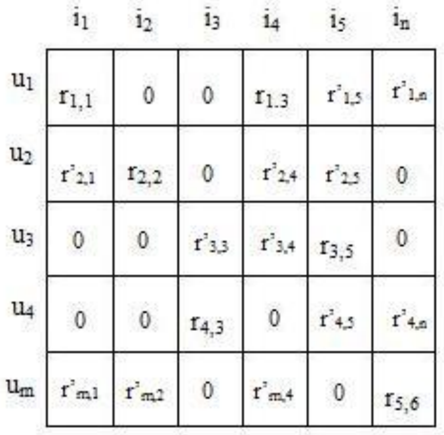
4.4.2.7 Effective Missing Data Prediction (EMDP)

As can be observed from Figure 11, user-item matrix can be very sparse initially when the users have rated only a few movies. And using this matrix for the prediction of ratings for active users often results in recommendations with very low quality. However, with the help of EMDP algorithm, each missing data is evaluated by using the available information. If the evaluation achieves confidence, the predicted rating is stored in the entry of the new matrix as can be seen in Figure 12. Otherwise, no prediction takes place and the value of the missing data remains as zero.



	i_1	i_2	i_3	i_4	i_5	i_n
u_1	$r_{1,1}$	0	0	$r_{1,3}$	0	0
u_2	0	$r_{2,2}$	0	0	0	0
u_3	0	0	0	0	$r_{3,5}$	0
u_4	0	0	$r_{4,3}$	0	0	0
u_m	0	0	0	0	0	$r_{5,6}$

Figure 11 - User item matrix without missing data prediction



	i_1	i_2	i_3	i_4	i_5	i_n
u_1	$r_{1,1}$	0	0	$r_{1.3}$	$r'_{1,5}$	$r'_{1,n}$
u_2	$r'_{2,1}$	$r_{2,2}$	0	$r'_{2,4}$	$r'_{2,5}$	0
u_3	0	0	$r'_{3,3}$	$r'_{3,4}$	$r_{3,5}$	0
u_4	0	0	$r_{4,3}$	0	$r'_{4,5}$	$r'_{4,n}$
u_m	$r'_{m,1}$	$r'_{m,2}$	0	$r'_{m,4}$	0	$r_{5,6}$

Figure 12 - User item matrix with missing data prediction

To illustrate this, let's assume that user a's rating for item i will be predicted. Then, the main steps for the rating estimation process used in ReMovender can be summarized as below:

1. During the first step, the user-item rating matrix is initialized according to the user ratings for the movies. The entries which correspond to the rating of a movie that has not been rated by the related user are set to 0.
2. The local and global nearest user neighbors having the similarity above the specified threshold are calculated. If the number of both local and global neighbors is equal to 0, no user based prediction is made according to EMDP. Else, the formula for the user based prediction, where $nn_L(u_{a,})$ denotes the local neighbors of $u_{a,}$, $nn_G(u_{a,})$ denotes the global neighbors of $u_{a,}$ and sim'_{L} is calculated from the formula in section 4.4.2.2 is used:

$$\mathrm{ub\_}r_{a,i} = (1-\alpha)\frac{\sum_{u_k \in nn_L(u_{a,})} sim'_{\mathrm{L}}(u_k,u_a)\, r_{k,i}}{\sum_{u_k \in nn_L(u_{a,})} sim'_{\mathrm{L}}(u_k,u_a)} + \alpha\frac{\sum_{u_k \in nn_G(u_{a,})} sim'_{\mathrm{G}}(u_k,u_a)\, r_{k,i}}{\sum_{u_k \in nn_G(u_{a,})} sim'_{\mathrm{G}}(u_k,u_a)} \quad \mathbf{(16)}$$

3. The nearest neighbors of the item are calculated. If the number of neighbors is equal to 0, no item based prediction is made according to EMDP. Else, the

prediction is made according to the below formula, where $\text{OverallItemSim}(i_k, \text{i})$ is calculated according to the formula in section 4.4.2.3, $nn(\text{i})$ denotes the nearest neighbors of item i, and avg (i) denotes the average rating of item i.

$$\text{ib\_}r_{a,i} = \text{avg (i)} + \frac{\sum_{i_k \in nn(\text{i})} \text{OverallItemSim}(i_k,\text{i})\ (r_{\text{a},i_k} - \text{avg}(i_k))}{\sum_{i_k \in nn(\text{i})} \text{OverallItemSim}(i_k,\text{i})} \quad \textbf{(17)}$$

4. If the number of both user and item neighbors is 0, and the rating to be predicted belongs to a training user, the return value is 0.

$$ubib\_r_{a,i} = 0 \quad \textbf{(18)}$$

Else if the number of both user and item neighbors is 0, and the rating to be predicted belongs to an active user, the formula for the predicted value where $avg\ (r_a)$ denotes the average rating of user a, and $avg\ (r_i)$ denotes the average rating of item i is:

$$ubib\_r_{a,i} = \lambda\,.\,avg\ (r_a)\ +\ (1\ -\ \lambda)\,.\,avg\ (r_i) \quad \textbf{(19)}$$

Else if the number of user neighbors is 0, the overall predicted value is equal to the result of the item based prediction.

$$ubib\_r_{a,i} = ib\_r_{a,i} \quad \textbf{(20)}$$

Else if the number of item neighbors is 0, the overall predicted value is equal to the result of the user based prediction.

$$ubib\_r_{a,i} = ub\_r_{a,i} \quad \textbf{(21)}$$

Else the overall prediction depends both on the result of the user and item based prediction, and the resulting formula is as below:

$$ubib\_r_{a,i} = \lambda\,.\,ub\_r_{a,i} + (1\ -\ \lambda)\,.\,ib\_r_{a,i} \quad \textbf{(22)}$$

During the EMDP prediction for training users, the obtained results are stored in another matrix in order to make this step fair for all entries. And although the predicted ratings can have a value smaller than 1 or greater than 5, no rounding procedure is used in that step.

4.4.2.8 Recommendation

In order to recommend a set of movies to a user, all the missing values in the related row of the user-item matrix are predicted according to the calculations explained in 4.4.2.7. Then, the movies of which ratings are predicted are sorted in decreasing rating order so that the recommendations can be presented to the user in that order on the user interface.

4.5. Implementation Details

For the information extraction module of ReMovender, a Python package called IMDbPY is used. IMDbPY is a free and open source software which is platform-independent and written in pure Python. The required Python scripts are run on Eclipse IDE with the help of the Pydev plugin that enables users to use Eclipse for Python development. In addition, the thumbnails of the movies to be displayed in the user interface are extracted from the web via Google AJAX Search API, which provides to put Google Search in web pages with JavaScript.

As for the user interface and prediction mechanism of the system, they have been implemented in Java with its servlet and JSP technologies by using Eclipse Ganymede.

The implementation environment also includes Apache-Tomcat Application Server v6.0.18 used together with the MySQL v5.1.34 database, which has become the world's most popular open source database because of its consistent fast performance, high reliability and ease of use.

CHAPTER 5

EVALUATION

Several aspects of the web based recommendation problem are somewhat different than many other problems. The most fundamental distinction is that the core of this problem is to model and maximize the user satisfaction after making recommendations, which is not only difficult to define but also a changing function over time.

This chapter presents the details of how the proposed approach of ReMovender is evaluated in order to test its performance. First, the dataset and the metrics used for the experimental evaluation are introduced. Then, how the prediction performance is affected by the parameter β, which determines the extent to which the item similarity relies on content similarity, is explained. Lastly, the results of the conducted experiments are stated and discussed.

5.1 Data Set

The experimental evaluation of ReMovender was conducted using the MovieLens dataset maintained by the GroupLens Research group at University of Minnesota. Among the three available datasets, the one containing 100,000 ratings on a scale of 1 to 5 for 1682 movies by 943 users, where each user has rated at least 20 movies, was preferred in order to make the evaluation results comparable to the studies [25] and [26] that used the same dataset.

The density of the user-item matrix created from the MovieLens dataset is:

$$\frac{100{,}000}{943 \text{ x } 1682} = 6.30\ \%$$

which can be considered to be appropriate enough in terms of sparsity for the evaluation of the system.

In order to make the contents of the movies in the data set available for the Content Boosted Collaborative Filtering prediction approach of ReMovender, the title and year of each movie in the MovieLens dataset were used for retrieving the related IMDb id from the local copy of the IMDb database. However, approximately 400 movies could not be correlated due to the language, year, 'and'/'&', capital letter inconsistencies. Besides, it was observed that the titles of some movies in the MovieLens data set use the aka (also known as) titles in IMDb. All of these inconsistencies were corrected manually so that all of the movie ids in the MovieLens dataset could be correlated against the IMDb ids, which provided the system to obtain all the content required during the content boosted prediction.

In order to evaluate the prediction mechanism of ReMovender, cross validation method was used and among the various cross validation methods, the holdout method was preferred. Following this method, the data set was separated into two sets, called the training set and the testing set. Thus, after a subset of 500 users was extracted randomly from the data set, 300, 200 and 100 of them were selected as the training users respectively. And the rest 200, 300, 400 were selected as the active (testing) users. The respective sets were named as MovieLens300, MovieLens200 and MovieLens100. As for the ratings from the active users, the number of the ratings provided by the user was varied from 5 to 10 and 20, which were named as Given5, Given10 and Given20 respectively. This resulted in a total of 9 configurations which represented different training data sparsity and test item of active user sparsity. The most important reason for adopting this protocol during the experimental setup of ReMovender is the aim of comparing ReMovender with the other experiments in the literature, which also made use of this protocol.

5.2 Metrics

For the purpose of measuring the prediction quality of the proposed approach and comparing with other CF methods, Mean Absolute Error (MAE) metrics was used. The MAE is computed by first summing the absolute errors of the N corresponding ratings-prediction pairs and then averaging the sum. And it can be more formally defined as:

$$\mathrm{MAE} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} | r_{\mathrm{i}} - r_{\mathrm{i}}' |}{\mathrm{N}} \tag{23}$$

where r_{i} denotes the actual rating that the related user gave for item i, and r_{i}' denotes the rating predicted by ReMovender's approach, and N denotes the number of tested ratings. As can be observed, a larger MAE indicates a lower accuracy.

5.3 Comparison

In order to test the performance of ReMovender's prediction approach, the MAE values obtained for the 9 configurations, which were explained in detail in section 5.1, were compared with the state-of-the-arts algorithms on MovieLens.

The parameters or thresholds that were used throughout the prediction process were empirically set to $\lambda = 0.6$, $\gamma = 30$, $\delta = 25$, $\eta = \theta = 0.6$, numberOfNeighbors = 35, and $\alpha = 0.5$ just like the experimental setup of [25] so that the result values could be compared to the ones reported in [25]. As for the parameter β, which was introduced by ReMovender's approach, it was set to 0.5. In Table 5, MAE comparison of the content boosted collaborative filtering approach of ReMovender (CBCF) with state-of-the-arts algorithms including user-based using PCC (UPCC) [72], item-based using PCC (IPCC) [54], Similarity Fusion (SF) [73], Effective Missing Data Prediction (EMDP) [26], and Local & Global User Similarity (LU&GU) [25] are summarized. It can be easily observed from this table that ReMovender's prediction approach significantly improves the recommendation quality of collaborative filtering, and outperforms all other competitive algorithms in various configurations. As explained in the previous sections, [26]'s EMDP algorithm is a combination of a user-based and item-based predictor, whereas [25]'s LU&GU approach is an improvement of user-based algorithms. And as [25] also asserted, when EMDP employed LU&GU in ReMovender to replace traditional user-based approaches, a better performance was achieved. Besides, the contribution of using content information during item similarity calculations should certainly be taken into consideration while interpreting the experimental results.

Table 5 - MAE comparison with state-of-the-arts algorithms on MovieLens (A smaller value means a better performance)

Training Users	Methods	Given5	Given10	Given20
	CBCF	0.7889	0.7653	0.7561
	LU&GU	0.791	0.7681	0.7565
100	EMDP	0.7896	0.7668	0.7806
	SF	0.8446	0.7807	0.7717
	UPCC	0.8377	0.8044	0.7943
	IPCC	0.9639	0.8922	0.8577
	CBCF	0.7816	0.7648	0.7533
	LU&GU	0.7937	0.7733	0.7719
200	EMDP	0.7997	0.7953	0.7908
	SF	0.8507	0.8012	0.7862
	UPCC	0.8185	0.8067	0.796
	IPCC	0.955	0.9135	0.871
	CBCF	0.7637	0.7562	0.7384
	LU&GU	0.7718	0.7704	0.7444
300	EMDP	0.7925	0.7951	0.7552
	SF	0.8062	0.7971	0.7527
	UPCC	0.8055	0.7910	0.7805
	IPCC	0.9862	0.9266	0.8573

5.4 Impact of Parameters

In this section, the impacts of the parameters and thresholds on the prediction performance of ReMovender are explained.

5.4.1 Impact of γ and δ

With the help of these parameters that are used for the significance weighting process, fake similarities are devalued. And the results of the experiments conducted by [26] state that significance weighting promotes the CF performance. In addition, the results show that there is a relationship between the density of the user-item matrix and γ / δ such that the density decreases with the increase of these parameters.

5.4.2 Impact of λ

This parameter is used to fuse information from both users and items to predict the missing data and to predict the rating for active users. According to the experiment results reported by [26], it is concluded that λ has a significant effect on prediction results. In addition, the fact that combining user-based and item-based method greatly improves the recommendation accuracy is demonstrated. Another observation made according to the results is that the information for users is more important than the information for items if more ratings for active users are given, while the information for items becomes more important if less ratings for active users are available.

5.4.3 Impact of η and θ

These parameters determine the number of missing data to be predicted. If they are set too high, many missing data cannot be predicted since a lot of users and items will not have similar neighbors. If the values of these parameters are set too low, users and items will have many similar neighbors, which will cause prediction inaccuracy and an increase in computation cost. The experimental results of [26] show that if the values of these parameters are around 0.70, EMDP achieves a very good MAE value.

5.4.4 Impact of α

α balances the prediction from local and global user similarity. In another sense, with $\alpha = 0$, the prediction depends completely on local user similarity, and with $\alpha = 1$, it depends completely on global user similarity. According to the experiments carried out by [25] to determine the sensitivity of this parameter, it is concluded that although global similarity provides to improve the prediction accuracy in a great amount whenever there exist few training users and few ratings of the active users, it cannot improve accuracy for the cases where the number of both training users and the ratings from the active users are sufficient.

5.4.5 Impact of β

As explained in the previous sections this parameter is introduced by ReMovender's prediction approach and it provides to determine the extent to which the item similarity relies on collaborative filtering methods or content similarity. With $\beta=0$, it indicates that the similarity depends completely on collaborative similarity, whereas it depends completely on content similarity when $\beta=1$.

For the purpose of determining the sensitivity of β, several experiments were carried out on all configurations in which the value of β was varied from 0 to 1. The results of these experiments are shown in Figures 11, 12, and 13. Before proceeding with interpreting these results, the item similarity related formulas (11, 13, and 17) should be analyzed in detail.

As can be observed from these formulas, during the item based prediction of a user rating for a specific item, the ratings of the other users in the system for that item are not taken into consideration directly. These ratings only have a contribution on the calculation of the average rating of the item.

As a design issue of ReMovender, while making user based prediction for an item, the users who have not rated that item are not considered as similar to the user, whereas while making item based prediction for an item, the items who have not been rated by the user are not considered as similar to the item. And due to the second statement, in order to be capable of using the content information of the items that are similar to the item for which rating will be predicted, the user should absolutely have rated these items. Thus, the number of ratings given by a user has great importance for the overall prediction mechanism of ReMovender.

For all the reasons explained above, a decrease in the MAE was observed for all of the configurations, when the number of the ratings of the user increased. The experimental results also show that more accurate and realistic predictions can be obtained when the value of β is around 0.5. Because in this way, the prediction can both exploit collaborative filtering and content based similarity in certain and sensible amounts, which shows that CF and CB approaches both have a very important and indispensable role for rating prediction.

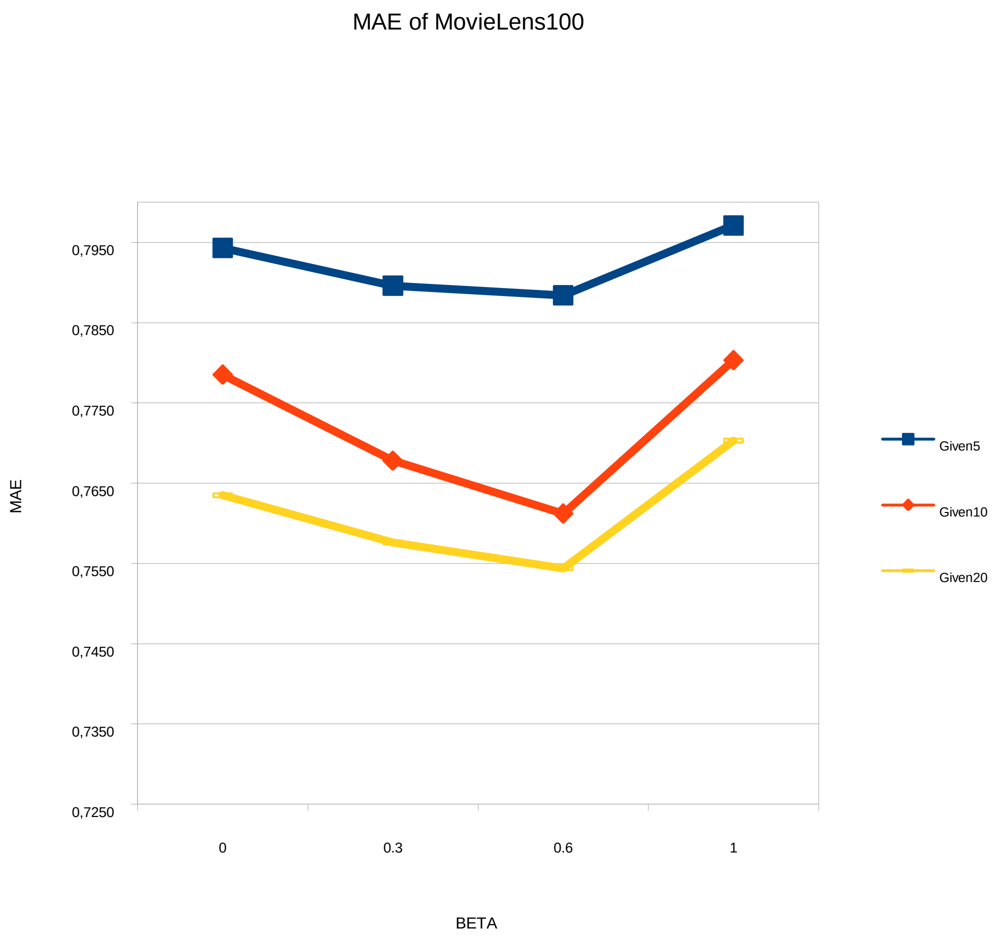


Figure 13 - Impact of Beta on MAE (on movieLens100)

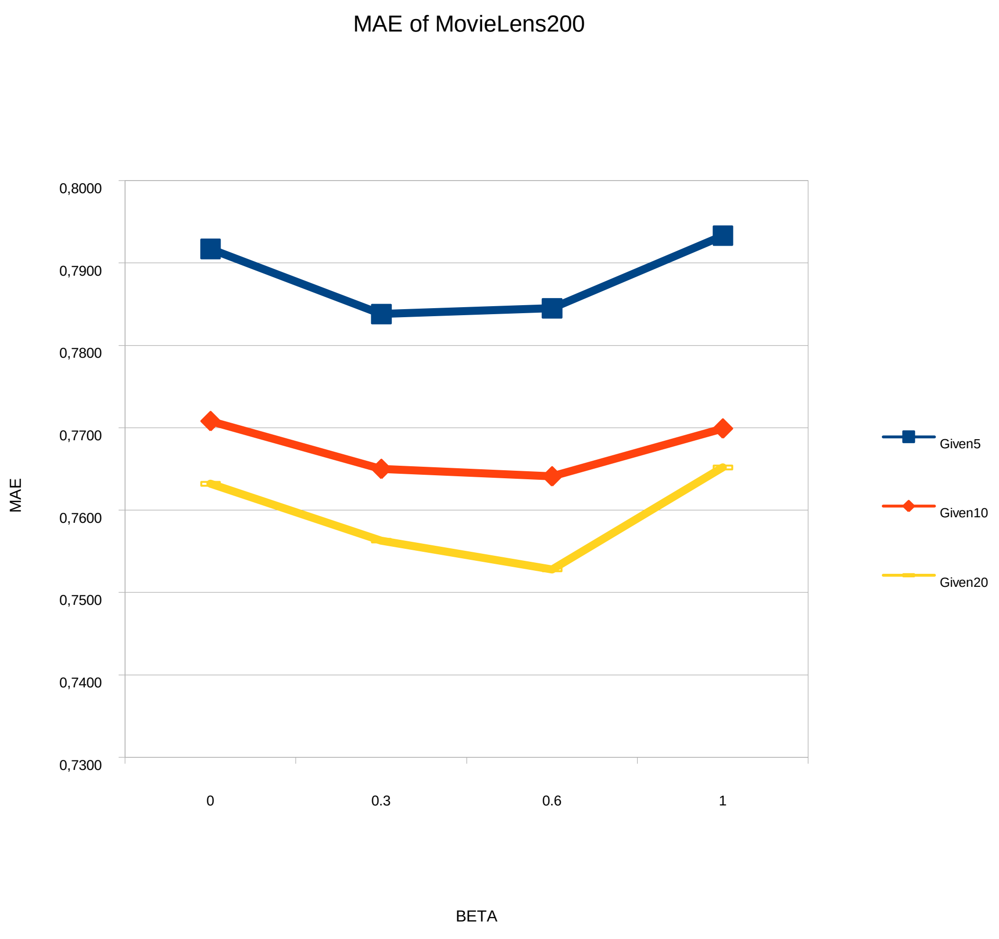


Figure 14 - Impact of Beta on MAE (on movieLens200)

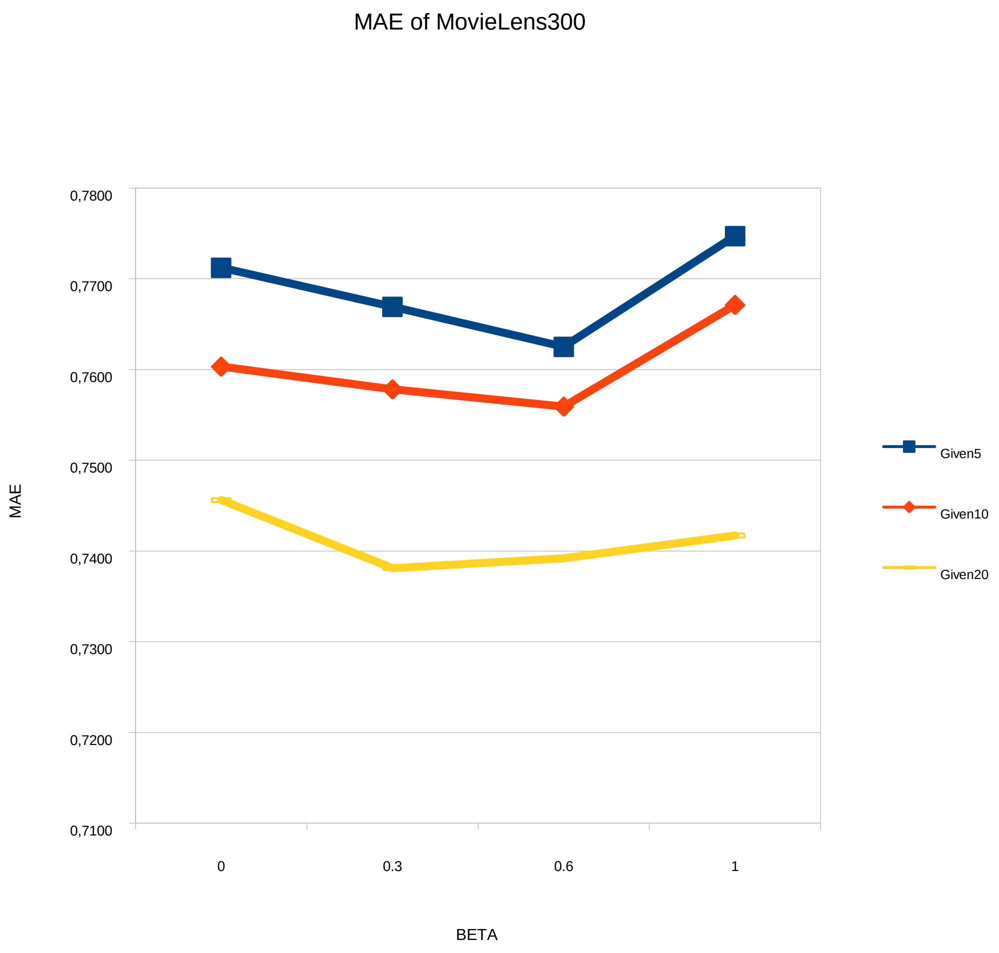


Figure 15 - Impact of Beta on MAE (on movieLens300)

CHAPTER 6

CONCLUSION AND FUTURE WORK

Within this thesis work, a web based movie recommender system called ReMovender has been presented. ReMovender uses a content boosted collaborative filtering approach combining the local and global user similarity introduced by [25] and effective missing data prediction technique introduced by [26] in order to handle the sparseness problem effectively. Content information of the movies, which are obtained from IMDb with the help of information extraction techniques, is exploited by the proposed system during the item similarity calculations.

First, a brief introduction to recommender systems has been given by stating the current approaches and theories used in these systems. Then, the related work in the related area has been covered by analyzing a variety of recommendation systems from different domains together with their advantages and disadvantages. After that, the architecture, and the prediction mechanism of ReMovender has been examined in detail. And lastly, the evaluation scheme used to test the prediction performance of ReMovender and determine the most appropriate value for the newly introduced parameter has been explained. In addition, the results of the conducted experiments have also been discussed.

Empirical analysis shows that the proposed prediction algorithm of ReMovender outperforms other state-of-the-arts collaborative filtering approaches in various configurations. When LU&GU is employed by EMDP to replace traditional user-based approaches, a better performance is achieved. Moreover, using content information during item similarity calculations significantly improves the recommendation quality of collaborative filtering approach.

As a future work, ReMovender will be made publicly available for daily use as an up-to-date movie recommendation system so that further evaluation about the performance of the prediction mechanism can be made. In addition, a technique making use of the text based metadata including comments, plot outlines, and synopsis can be developed in order to improve the content based item similarity calculations. Furthermore, more research on the relationship between user and item information can be conducted since the results of this thesis work show that the algorithm combining these two kinds of information generates better performance. Also, a more improved user modeling and personalization technique can be tried, which observes the implicit behavior of the user throughout the system and makes use of machine learning algorithms in order to learn the movie taste of the user. Lastly, further research on the methods of adding content information to collaborative filtering methods in a more natural way can be conducted.

REFERENCES

[1] Samhaa El-Beltagy, David De Roure and Wendy Hall, *The evolution of a practical Agent-Based Recommender System*, 2000.

[2] M. Balabanovic, “Learning to Surf: Multiagent systems for adaptive web page recommendation,” Ph.D Thesis, Stanford University, Stanford, CA, 1998.

[3] K Lang. Newsweeder, “Learning to filter news,” in *Proc. 12th International conference on machine learning*, pp. 331–339, Morgan Kaufmann, San Fransisco, 1995.

[4] M.Pazzani and D.Billsus, *Learning and revising user profiles: the identification of interesting web sites, Machine Learning*, pp. 313–331, 1997.

[5] P. Maglio and R. Barett, “How to build modeling agents to support web searches,” in *Proc. Sixth International Conference on User Modeling,* pp. 5–16, 1997.

[6] C. Thomas and G. Fischer, “Using agents to improve the usability and usefulness of the world wide web,” in *Proc. Fifth International Conference on User Modeling,* pp. 5–12, 1996.

[7] J. Ben Schafer, Nathaniel Good, and Joseph Konstan et. al., “Combining collaborative filtering with personal agents for better recommendations,” in *Proc. National Conference of the American Association of Artificial Intelligence,* pp. 439–436, 1999.

[8] Amazon.com, www.amazon.com, Last accessed on August 2009.

[9] The Internet Movie Database (IMDb), http://www.imdb.com/, Last accessed on August 2009.

[10] MovieLens, www.movielens.umn.edu, Last accessed on August 2009.

[11] E! Online, www.moviefinder.com, Last accessed on August 2009.

[12] PANDORA, www.pandora.com, Last accessed on August 2009.

[13] last.fm, www.last.fm, Last accessed on August 2009.

[14] J. Ben Schafer, Joseph Konstan, and John Riedl, "Recommender systems in e-commerce," *ACM Conference on Electronic Commerce (EC-99),* Denver, CO, November 1999.

[15] Saddys Segrera, Maria N. Moreno, "An Experimental Comparative Study of Web Mining Methods for Recommender Systems," in *Proc. 6th WSEAS International Conference on Distance Learning and Web Engineering,* Lisbon, Portugal, September 22-24, 2006

[16] Scharer, J.B., J.A. Konstan et al, "E-commerce recommendation applications," *Data Mining and Knowledge Discovery,* vol. 5, No.1 pp. 115-53, 2001.

[17] Dipankaj G Medhi, Juri Dakua, "MovieReco: A Recommendation System," *World Academy of Science, Engineering and Technology,* vol. 4, pp. 70-74, April 2005.

[18] V. N. Marivate, Student M\ember, IEEE, G. Ssali, T. Marwala, Member, "IEEE: An Intelligent Multi-Agent Recommender System for Human Capacity Building," *14th IEEE Mediterranean Electrotechnical Conference,* pp. 909-915, 2008.

[19] Olli Niinivaara, "Collaborative Social Recommender Agents," University of Helsinki, Dept. of Comp. Sci., Software Agent Technology Course Paper, 2004.

[20] Martin Szomszor, Ciro Cattuto, Harith Alani1, Kieron O'Hara, Andrea Baldassarri, Vittorio Loreto, Vito D.P. Servedio, "Folksonomies, the Semantic Web, and Movie Recommendation," *4th European Semantic Web Conference, Bridging the Gap between Semantic Web and Web 2.0,* 3-7th, June 2007.

[21] Rajatish Mukherjee, Partha Sarthi Dutta, Sandip Sen, “MOVIES2GO – A new approach to online movie recommendation,” in *Proc. IJCAI Workshop on Intelligent Techniques for Web Personalization*, Seattle, WA, USA, August 2001.

[22] Shinhyun Ahn, Chung-Kon Shi, “Exploring Movie Recommendation System Using Cultural Metadata,” *International Conference on Cyberworlds*, 2008

[23] Gediminas Adomavicius, and Alexander Tuzhilin, “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions,” *IEEE Transactions on Knowledge and Data Engineering,* VOL. 17, NO.6, June 2005.

[24] Oznur Kirmemis, Aysenur Birturk, “A Content-Based User Model Generation and Optimization Approach for Movie Recommendation,” *6th Workshop on Intelligent Techniques for Web Personalization & Recommender Systems*, 2008.

[25] Heng Luo, Changyong Niu, Ruimin Shen, Carsten Ullrich, “A collaborative filtering framework based on both local user similarity and global user similarity,” in *Proc*. 2008 *European Conference on Machine Learning and Knowledge Discovery in Databases,* Part I, 2008.

[26] Ma, H., King, I., and Lyu, M. R., “Effective missing data prediction for collaborative filtering,” in *Proc. 30th annual international ACM SIGIR conference on research and development in information retrieval,* pp. 39-46, NewYork, 2007.

[27] Dipankaj G Medhi, Juri Dakua, “MovieReco: A Recommendation System,” *World Academy of Science, Engineering and Technology 4,* 2005.

[28] Shinhyun Ahn, Chung-Kon Shi, “Exploring Movie Recommendation System Using Cultural Metadata,” *International Conference on Cyberworlds*, 2008.

[29] Souvik Debnath, “Machine Learning Based Recommendation System,” Master Thesis, Indian Institute of Technology, Dept. of Comp. Science and Engineering, May 2008.

[30] Olli Niinivaara, “Agent-Based Recommender Systems,” University of Helsinki, Dept. of Comp. Sci., Software Agent Technology Course Paper, 2004.

[31] Mark Van Setten, “Supporting people in finding information, Hybrid recommender systems and goal-based structuring,” *Telematica Instituut Fundamental Research Series*, vol. 016. Enschede, the Netherlands: Telematica Instituut, 2005.

[32] G. Adomavicius and A. Tuzhilin, “Expert-Driven Validation of Rule-Based User Models in Personalization Applications,” *Data Mining and Knowledge Discovery,* vol. 5, nos. 1 and 2, pp. 33-58, 2001.

[33] Hugo Siles Del Castillo, Hybrid Content-Based Collaborative-Filtering Music Recommendations, Master Thesis, University of Twente, November 2007.

[34] Mark Van Setten, “Supporting people in finding information, Hybrid recommender systems and goal-based structuring,” *Telematica Instituut Fundamental Research Series*, vol. 016. Enschede, the Netherlands: Telematica Instituut, 2005.

[35] Resnick, P., Varian, H., “Recommender Systems,” *Communications of the ACM,* vol. 40, no. 3, pp. 56-58, 1997.

[36] Terveen, L., Hill, W., *Beyond Recommender Systems: Helping People Help Each Other*, In Carroll, J., (Ed.), HCI in the New Millennium. Addison Wesley, pp. 475-486, 2001.

[37] Yi Zhang, Jonathan Koren, “Efficient Bayesian hierarchical user modeling for recommendation systems,” in *Proc. 30th annual international ACM SIGIR conference on Research and development in information retrieval*, Amsterdam, 2007.

[38] Gaudioso, E., Contribuciones, al Modelado del Usuario en Entornos Adaptativos de Aprendizaje y Colaboración a través de Internet mediante técnicas de Aprendizaje Automático. Tesis Doctoral. Dpto. de Inteligencia Artificial, Facultad de Ciencias, Universidad Nacional de Educación a Distancia, Madrid, 2002.

[39] ChoiceStream Technology Brief, Review of Personalization Technologies: Collaborative Filtering vs. ChoiceStream's Attributized Bayesian Choice Modeling.

[40] Nichols, D., "Implicit Rating and Filtering," *in Proc. 5th DELOS Workshop on Filtering and Collaborative Filtering,* pp. 31-36, Hungary, 1997.

[41] Claypool, M., et al., "Implicit Interest Indicators," in Proc. *Intelligent User Interfaces 2001 (IUI'01),* New Mexico, USA. ACM, pp. 33-40, 2001.

[42] Belkin N. J., Croft W. B., "Information filtering and information retrieval: Two sides of the same coin?" *Communications of the ACM 35,* 29–39, December 1992.

[43] S, Baumann, and O, Hummel, "Enhancing Music Recommendation Algorithms Using Cultural Metadata", *Journal of New Music Research,* 34(2), pp. 161– 172, 2005.

[44] Nicola Orio, "Music Retrieval: A tutorial and Review," *Foundations and Trends in Information Retrieval*, Vol. 1, Issue 1, pp 1-96, 2006.

[45] B. Krulwich, and C. Burkey, "Learning user information interests through extraction of semantically significant phrases", in *Proc. AAAI Spring Symposium on Machine Learning in Information Access*, 1996.

[46] K, Lang, "Newsweeder: Learning to filter netnews," in *Proc. 12th International Conference on Machine Learning,* 1995.

[47] R.J. Mooney, P.N. Bennett, and L. Roy, "Book Recommending Using Text Categorization with Extracted Information," in *Proc. Recommender Systems Papers from 1998 Workshop,* Technical Report WS-98-08, 1998.

[48] M. Pazzani and D. Billsus, "Learning and Revising User Profiles: The Identification of Interesting Web Sites," *Machine Learning,* vol. 27, pp. 313-331, 1997.

[49] U. Shardanand and P. Maes, “Social Information Filtering: Algorithms for Automating ‘Word of Mouth’,” in *Proc. Conf. Human Factors in Computing Systems,* 1995.

[50] Kazuyoshi Yoshii, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, Hiroshi G. Okuno, “Hybrid collaborative and content-based movie recommendation using probabilistic model with latent user preferences,“ *ISMIR*, pp. 296-301, 2006.

[51] George Lekakos, Petros Caravelas, “A hybrid approach for movie recommendation,” *Springer Science + Business Media,* LLC, December 2006.

[52] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen, “Scalable collaborative filtering using cluster-based smoothing,” in *Proc. of SIGIR*, 2005.

[53] Clifford Lyon, “Movie Recommender,” CSCI E-280 Final Project, 2004.

[54] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proc. WWW Conference*, 2001.

[55] Prem Melville, R. Mooney, and R. Nagarajan, “Content-boosted collaborative filtering,” in *Proc. SIGIR-2001 Workshop on Recommender Systems*, New Orleans, LA, September 2001.

[56] Alexandrin Popescul, Lyle Ungar, David Pennock, “Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments,” in *Proc. 17th Conference in Uncertainty in Artificial Intelligence*, pp 437-444, University of Washington, Seattle, Washington, USA, August 2-5, 2001.

[57] A. Berenzweig, B. Logan, D. P. W. Ellis, and B. Whitman, “A Large-Scale Evaluation of Acoustic and Subjective Music Similarity Measures,” in *Proc. 4th International Symposium on Music Information Retrieval (ISMIR'03),* Washington, D.C., USA, 2003.

[58] Pedro Cano, Markus Koppenberger, Nicolas Wack, “Content-Based Music Audio Recommendation,” in *Proc. ACM international conference on Multimedia*, November 6-11, 2005.

[59] Jean-Julien Aucouturier, François Pachet, “Representing Musical Genre: A State of the Art,” *Journal of New Music Research*, Vol. 32, No 1, pp 83-93. 2003.

[60] Qing Li, Byeong Man, Dong Hai, “A music recommender based on audio features,” in *Proc. 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp 532-533, Sheffield, UK, July 25-29, 2004.

[61] C. Basu, H. Hirsh, and W. Cohen, “Recommendation as Classification: Using Social and Content-Based Information in Recommendation,” Recommender Systems. Papers from 1998 Workshop, Technical Report WS-98-08, AAAI Press, 1998.

[62] Burke R, “Hybrid recommender systems: survey and experiments,” *User Model User Adapt Interact* 12:331–370, 2002.

[63] Souvik Debnath, Niloy Ganguly, Pabitra Mitra, “Feature weighting in content based recommendation system using social network analysis,” *WWW*, 2008.

[64] Yolanda Blanco Fernández, Jose J. Pazos Arias, Alberto Gil Solla, Manuel Ramos Cabrer, “A Flexible Approach to Improve the Personalized TV by Semantic Inference,” in *Proc. Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces (WPRSIUI-05)*, Reading, pp. 69-79, U.K., 2005.

[65] V. N. Marivate, Student M\ember, IEEE, G. Ssali, T. Marwala, Member, “IEEE: An Intelligent Multi-Agent Recommender System for Human Capacity Building,” *14th IEEE Mediterranean Electrotechnical Conference*, pp. 909-915, 2008.

[66] Smyth B. and Cotter P, “A personalized television listings service,” *Communications of the ACM,* 43(8):107.111, 2000.

[67] Gozde Ozbal, Hilal Karaman, “Matchbook, A Web Based Recommendation System For Matchmaking,” *International Symposium on Computer and Information Sciences (ISCIS'08),* Istanbul, Turkey, October 2008.

[68] Kurapati K., Gutta S., Schaffer D., Martino J. and Zimmerman J, “A multi-agent tv recommender,” in *Proc. Workshop on Personalization in Future TV (TV-01)*, 2001.

[69] Jason M. Adams, Paul N. Bennett, Anthony Tomasic, “Combining Personalized Agents to Improve Content Based Recommendations,” Technical Report CMU-LTI-07-015, Carnegie Mellon University, December 2007.

[70] Prem Melville, Raymond J. Mooney, Ramadass Nagarajan, “Content-boosted collaborative filtering for improved recommendation,” in *Proc. AAAI-02*, 2002.

[71] M. R. McLaughlin and J. L. Herlocker, “A collaborative filtering algorithm and evaluation metric that accurately model the user experience,” in *Proc. SIGIR*, 2004.

[72] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “Grouplens: An open architecture for collaborative filtering of netnews,” in *Proc. ACM Conference on Computer Supported Cooperative Work*, 1994.

[73] J. Wang, A. P. de Vries, and M. J. Reinders, “Unifying user-based and item-based collaborative filtering approaches by similarity fusion,” in *Proc.SIGIR*, 2006.