A CONTENT BOOSTED COLLABORATIVE FILTERING APPROACH FOR RECOMMENDER SYSTEMS BASED ON MULTI LEVEL AND BIDIRECTIONAL TRUST DATA

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

BY

FERHAT ŞAHİNKAYA

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

JUNE 2010

Approval of the thesis

A CONTENT BOOSTED COLLABORATIVE FILTERING APPROACH FOR RECOMMENDER SYSTEMS BASED ON MULTI LEVEL AND BIDIRECTIONAL TRUST DATA

submitted by **Ferhat ŞAHİNKAYA** 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, **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı ______________________________
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Ferda Nur Alpaslan ______________________________
Supervisor, **Computer Engineering**

Examining Committee Members:

Assoc. Prof. Dr. Nihan Kesim Çiçekli ______________________________
Computer Engineering, METU

Assoc. Prof. Ferda Nur Alpaslan ______________________________
Computer Engineering, METU

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

Dr. Ayşenur Birtürk ______________________________
Computer Engineering, METU

M.S. Özgür Alan ______________________________
ORBIM Information Technologies

Date: ______________________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: FERHAT ŞAHİNKAYA

Signature :

ABSTRACT

A CONTENT BOOSTED COLLABORATIVE FILTERING APPROACH FOR RECOMMENDER SYSTEMS BASED ON MULTI LEVEL AND BIDIRECTIONAL TRUST DATA

ŞAHİNKAYA, Ferhat
M.S., Department of Computer Engineering
Supervisor: Assoc. Prof. Ferda Nur ALPASLAN

June 2010, 121 pages

As the Internet became widespread all over the world, people started to share great amount of data on the web and almost every people joined different data networks in order to have a quick access to data shared among people and survive against the information overload on the web.

Recommender systems are created to provide users more personalized information services and to make data available for people without an extra effort. Most of these systems aim to get or learn user preferences, explicitly or implicitly depending the system, and guess “preferable data” that has not already been consumed by the user. Traditional approaches use user/item similarity or item content information to filter items for the active user; however most of the recent approaches also consider the trustworthiness of users. By using trustworthiness, only reliable users according to the target user opinion will be considered during information retrieval.

Within this thesis work, a content boosted method of using trust data in recommender systems is proposed. It is aimed to be shown that people who trust the active user and the people, whom the active user trusts, also have correlated opinions with the active user. This results the fact that the rated items by these people can also be used while offering new items to users.

For this research, www.epinions.com site is crawled, in order to access user trust relationships, product content information and review ratings which are ratings given by users to product reviews that are written by other users.

Keywords: Recommender Systems, User Modeling, Collaborative Filtering, Content Based Filtering, Trust Based Social Networks

ÖZ

ÖNERİ SİSTEMLERİNDE ÇOK SEVİYELİ VE İKİ YÖNLÜ GÜVEN VERİSİNE DAYALI İÇERİK DESTEKLİ KOLABORATİF FİLTRELEME YAKLAŞIMI

ŞAHİNKAYA, Ferhat
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Ferda Nur ALPASLAN

Haziran 2010, 121 sayfa

İnternet dünya üzerinde yaygınlaştıkça, insanlar web üzerinden daha fazla bilgi paylaşabilir hale geldi. Bunun yanısıra, neredeyse her internet kullanıcısı paylaşılan verilere daha hızlı bir şekilde erişebilmek ve aşırı veri yoğunluğuna karşı direnebilmek amacıyla kullanıcılar arasında veri paylaşımının olduğu farklı veri ağlarını kullanmaya başladı.

Geliştirilen öneri sistemleri, kullanıcılara daha fazla kişiselleştirilmiş veri sağlanmasını ve kullanıcıların fazla çaba sarfetmeden verilere ulaşmasını sağladı. Sistemden sisteme değişmesine rağmen, çoğu öneri sisteminde amaç, kullanıcının tercih bilgilerini açıkça ya da gizli bir yolla öğrenip kullanıcının henüz fikir belirtmediği "tercih edilebilir verileri" tahmin etmektir. Geleneksel yaklaşımlarda kullanıcı/madde benzerlikleri ya da madde içerik bilgileri aktif kullanıcıya sunulan verilerin seçilmesinde kullanılmaktadır, ancak yeni yaklaşımlarda kullanıcı güven bilgileri de kullanılarak, sadece hedef kullanıcının güvenilir bulduğu kullanıcıların görüşleri bilgi çıkarma sürecinde göz önünde bulundurulmaktadır.

Bu tez çalışması kapsamında, güven verisini de göz önünde bulunduran içerik destekli bir öneri yöntemi sunulmaktadır. Aktif kullanıcı tarafından güvenilir bulunan ya da aktif kullanıcıyı güvenilir bulan kullanıcıların, aktif kullanıcı ile benzer görüşlere sahip olduğunu göstermek hedeflenmektedir. Bu düşünce ile, bu kullanıcıların oy verdiği maddeler, aktif kullanıcıya sunulacak olan maddelerin bulunmasında kullanılabilir.

Bu araştırma için, kullanıcı güven ilişki bilgileri, ürün içerik bilgileri ve ürünler hakkında yorum yazan kullanıcılara başka kullanıcılar tarafından verilen oylar, www.epinions.com web sitesi taranarak elde edilmiştir.

Anahtar Kelimeler:, Öneri Sistemleri, Kullanıcı Modelleme, Kolaboratif Filtreleme, İçerik Bazlı Filtreleme, Güven Verisine Dayalı Sosyal Ağlar

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 would also like to express my deepest gratitude to my parents and my brother, who always encouraged me with their love, understanding, patience and emotional support. They always gave me the courage and strength I needed to achieve my goals.

I thank to Betül Bilge whose help, stimulating suggestions and encouragement helped me in all time of the research and development of this thesis. She shared her deep expertise with me both in work and throughout this thesis study.

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 Eda Ercan, for her help, support and cheerful presence through not only the course of this study, but also any course of my life. Thanks for your never-ending support and giving me a shoulder to lean on every time I need.

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Research Motivation

Recently, Internet has come into our lives and its popularity is increasing day by day. This popularity has caused the Internet to contain huge amount of information in a wide range of topics. This information overload is foreseen by Denning in early 1980's. He supported that due to the increasing use of electronic platform, at some time, users would not be able to reach the information that they really need [1].

After Web 2.0 came on the scene at the end of 1990's, people have gained a great ability of information handling on the Web. The term "Information handling" includes user centered design, information sharing and collaboration which provide people to own and control their data on the Web. These abilities also introduced various web applications, social networking sites, video sharing sites, blogs etc. where people can share any kind of data ranging from their recorded videos in the previous day party to music they are currently listening to. While Web technologies are evolving so fast that has never occurred in the past, web users anticipate to have all control on the Internet and want to get what they need from it, easily.

As previous experiences show us, in every field of technology and also in daily life, expectations determine the way to go and most people always desire more than they have in their hand. Up to now, people resist the growing web individually or with a little help of already generated systems. However, when Semantic Web and Web 3.0 is started to be used with their innovations, more sophisticated techniques will be required in order to cope with the already grown giant information on the web. Even to compete with the current information and user load on the Web, several automated information filtering techniques are introduced.

According to [2], Information Filtering user has a passive role during information gathering process and expects the system to push or send him information of interest according to some previously defined profiles. In other words, Information Filtering user can be described as the user who wants to access information with the possible minimum effort.

As previous studies proposed, use of social networks and relationships between users during information filtering process improves the overall performance. In the light of this fact, in this thesis, we aim to make the Information Filtering user comfortable by presenting a content boosted collaborative filtering approach for recommender systems based on multi level and bidirectional trust data.

1.2 Background Information

In the previous section, motivation for this study is discussed. However, it is also important to note the state-of-the art solutions and current status in recommender system research area. Details of state of the art solutions and recent approaches will be presented in Chapter 3.

Most of the traditional recommenders are based on content based approach; item and user based collaborative filtering approaches. These approaches provided an initial attempt for the area of recommender systems. Despite the existence of their drawbacks, these approaches occupy a significant space and subsequently proposed methods are mostly based on these initial attempts.

On the other hand, recent approaches use trustworthiness of users in recommendation process. Trust based collaborative filtering methods weights each user depending on his trustworthiness in the system. By this way, fake users are tried to be eliminated from recommendations. Trust based systems increased the recommendation accuracy according to evaluation results of proposed methods.

In fact, recommender systems aim to simulate recommendations in our daily life. A person looks for his friends when he needs guidance on a topic and collaborative filtering methods have an idea based on this statement. For this reason, even if various methods having simple and complex algorithms are proposed, the goal of each method is generating a personalized guidance for each user with a better quality.

1.3 Organization of the Thesis

This thesis comprises six chapters, which are organized as follows:

In this chapter, we have discussed the research motivation and why recommender systems become crucial in today's world.

In Chapter 2, already proposed methods are summarized and a formal definition for recommender systems is given. Additionally, categorization of recommendation algorithms, already proposed basic algorithms, their advantages and disadvantages are discussed.

In Chapter 3, related works used as base methods in the scope of this study are presented in a detailed manner to capture their underlying properties. Indeed, this chapter gathers a literature survey and state of the art solutions are described as parts of this survey.

Chapter 4 covers construction process of proposed methods in this study and base methods that are used as components of proposed methods.

Chapter 5 presents evaluation results of the proposed methods. In this chapter, comments on each method and its evaluation results are given. Advantages and disadvantages of these methods are discussed. Additionally, effects of base methods extracted from related works in Chapter 3 on each proposed method are investigated in Chapter 5.

Chapter 6 concludes this thesis work and draws possible paths for future works.

CHAPTER 2

RECOMMENDER SYSTEMS

2.1 Recommender Systems as a Research Area

Although previous studies in cognitive science, approximation theory, information retrieval, forecasting theories and even in management science and consumer choice modeling are roots of today’s recommender system approaches, recommender systems emerged as a new independent research area in mid-1990s with the introduction of rating structure [13]. Today, almost all recommender system applications use a rating data in order to formalize evaluation of a product by a user.

With the introduction of profile extraction techniques from textual information, which is a significant improvement in information retrieval techniques in mid-1990s, content based methods are concentrated on text based information since textual information plays an important role at that time, such as documents, Web sites (URLs) and USENET news messages [13]. By the help of new approaches, item and user profiles are started to be extracted from textual information, such as Fab System [14] which uses word weights in a web page to recommend a web site to a user. Similarly, the Syskill & Webert [15] system creates profiles according to “most descriptive words” of documents.

In addition to traditional methods in information retrieval, new techniques are also applied to recommendation process. For instance, in order to model the structure of problem domain, Bayesian classifiers, decision trees, artificial neural networks, clustering methods, statistical data analysis and various machine learning algorithms are used in different systems [13].

Collaborative filtering based recommender systems also occupy a significant place in this area. The first system using stereotypes is the Grundy System [16], which creates a user profile from a limited amount of information to recommend books to users. Tapestry [17] is another collaborative filtering system, on which users manually identify like-minded users and recommendation is performed accordingly. GroupLens [18][19] recommending Usenet news to people, Video Recommender [20] selecting videos from a large set by considering other like-minded, similarly situated users and communal history of use and Ringo [21] making personalized recommendation for music albums and artists are other systems relying on collaborative filtering.

Unfortunately, Netflix Prize competition [9] was a great chance for increasing development pace in collaborative filtering algorithms, since Netflix dataset was presented to interested users. This is the first time that a large scale dataset became publicly available. Even today, almost all of the commercial sites do not share their datasets and only some sites let people crawl their web pages to extract dataset information manually. Netflix Prize competition was started on October, 2006 and ended on September 21, 2009. Netflix Prize sought to substantially improve the prediction accuracies about how much someone is going to enjoy a movie based on their movie preferences. 51051 contestants on 41305 teams from 186 different countries are competed for $1M Grand Prize. “BellKor’s Pragmatic Chaos” team is the winner of the competition with 10.06% improvement over Netflix’s original algorithm [9]. Netflix 2 Prize competition plans are also announced by Netflix community and this competition is expected to be shorter than the first one, again grand prizes wait for best results in the competition.

Nowadays, with the help of Netflix Prize and interests of people both in academia and industry, various developed recommender systems, recommending various kinds of items, are available. Some of these applications recommend books, CDs, and other products like Amazon.com [4], some of them recommend movies like IMDb [5] and MovieLens [6], some others deal with music recommendation like Pandora [7] and Last.fm [8] and there are some systems like VERSIFI Technologies (formerly known as AdaptiveInfo.com) [9]recommending news. Most of these systems are created by vendors of products in order to increase sales, however quite different recommenders also exist on the web like MoleSkiing [11], where people share their experiences related to ski mountaineering and the system recommend available ski tours to its users. Furthermore, Jester [22] is an online joke recommendation system which has a method comprising eigen-taste collaborative filtering algorithm and principle component analysis. Additionally, PHOAKS [23] recommends information on WWW based on the principles of role specialization and reuse.

In today's computer science development, the topic "Recommender Systems" is one of the hottest and problem rich areas. New improvements on recommendation processes are expected as information load on the web and people needs increase like today. All approaches, mentioned in this section and several different disciplines interact with each other and that provides further improvements in all participants of the interaction. Some of the related disciplines that recommender systems are dealing with can be listed as Information Retrieval, Information Filtering, Artificial Intelligence, Natural Language Processing, WWW, Sociology and Human Computer Interaction.

2.2 Generic Recommender Systems

In today's world, every person interacts with World Wide Web (WWW) and searches for products that they can consume such as news, articles, web pages, movies, CDs, books etc. over the Internet. Due to the rapid growth of user and product load on the Internet, today's people want to access data with a relatively easy way in every field of life. In order to satisfy this need, many different recommender systems (RS) are created and many of them are currently being used on the web. Each system uses different aspects of its users to guess preferences of the target user, however aim of all recommender systems are almost same even in different contexts: trying to suggest the most related items with users' needs.

2.2.1 Formal Definition of Recommender Systems

Recommender Systems can be described as computer-based intelligent techniques to deal with the information load which benefit both the customer and the merchant. A recommender system benefits the customer by suggesting items that he is probably going to like. At the same time, it benefits the merchant by the increase of the sales which is expected to happen when more items in his area of interest is presented to the customer [3].

In order to make technical details of Recommender Systems clear, a formal description can be given as the following [13]:

Let C be the set of all users and let S be the set of all possible items that can be recommended such as books, movies, or restaurants.

Let u be a utility function that measures the usefulness of item s to user c, i.e. C x S → R, where R is a totally ordered set (e.g. non-negative integers or real numbers within a certain range. This set corresponds to a set of estimated or already given rating values for a user-item pair).

Then, for each user c ϵ C, recommender system aims to choose such item s' ϵ S that maximizes the utility.

$$\forall c \in C, s_c^{'} = \arg\max_{s \in S} u(c,s)$$

(EQUATION 2.1)

The main problem during recommendation process is that the utility function *u* is not defined on the whole of its domain, *C x S*. For this reason, utility function needs to be extrapolated to the whole space of *C x S*, however, again utility function may not cover its whole domain and for some user-item pairs in its domain, utility function may not estimate any rating value.

Table 1 - A sample User-Item matrix

	The Godfather	Avatar	Frozen	Amelie	Sherlock Holmes
John	5	2	-	-	3
Bob	-	3	-	4	-
Kate	3	-	-	-	-
Michael	-	5	-	2	3
Julie	2	5	-	3	2
Ann	4	-	-	-	1

A sample User x Item matrix is shown on Table 1. This table contains rating values for corresponding user-item pairs. If user *c* rated the item *s* with rating *r*, then cell *(c,s)* contains the given rating, *r*, however if no rating is explicitly given by the user *c* for the item *s*, then the cell value is represented with a *dash* (“-“).

As mentioned in 0, Recommender Systems aim to generate suggestions about new items or to predict the benefit of a specific item for a particular user [3]. This goal can be degraded to the goal of predicting *dash* values in User-Item matrix by the help of the utility function *u*. In order to predict these *dash* values, the utility function utilizes several kinds of already available data. Some of these data used by various utility functions for estimating rating value of user *c* for item *s* can be listed as below [3]:

- **Ratings (votes):** Ratings already provided by users in an already defined range, which are specified in User-Item matrix. Binary rating scheme is also commonly used technique and allows only rating values 0 (dislike) and 1 (like).
- **Demographic data:** This data refers to age, gender and education of users. Demographic data is not available in most of the datasets and is not retrieved easily.
- **Content data:** Item content information. Artists, date of a movie, length of a song, height of the mountain for a ski tour are samples of this data type.
- **User relationships:** User relationship weights are also started to be used in recommendation processes, recently. Trust, distrust, friendship, like, dislike ratings among users are examples of this data type. Reputation can also be classified in this type of data. User relationship weights can also be extracted from ratings in some applications like [24].

The user for whom the Recommender System is trying to produce its output at a moment is named as the active user and the output of a Recommender System can be either Prediction or Recommendation. Prediction is a rating value representing the estimated rating of the active user *c* for item *s*; where recommendation is a list of N items that is anticipated to be liked by the active user. Recommendation can be Top-N Recommendation or Ranked Scoring [3]. It is also remarkable that a utility function returns the cell value of User-Item matrix if existing value is not a *dash*; however, if no rating is specified by user *c* for item *s* in User-Item matrix, the utility function tries to find an output in Recommender System's output format as a prediction or a recommendation.

Most of the existing systems recommend a list of items as Top-N Recommendation or Ranked Scoring as in [4][7][8][11]. Items shown in recommendation list are ordered as item with highest predicted rating at the top and item with Nth highest predicted rating is at the bottom. All other items are filtered out and not shown to the user.

2.2.2 Generic Similarity Calculation Methods in Recommender Systems

Basic difference varying recommender system approaches is profile similarity calculation between users and items. There are two commonly used techniques to calculate the similarity between profiles, namely Cosine/Vector Similarity and Pearson Correlation Similarity. Cosine similarity is calculated as follows:

$$\begin{aligned} sim(x, y) = \cos(\overrightarrow{w_x}, \overrightarrow{w_y}) &= \frac{\overrightarrow{w_x} \cdot \overrightarrow{w_y}}{\left\|\overrightarrow{w_x}\right\|_2 \cdot \left\|\overrightarrow{w_y}\right\|_2} \\ &= \frac{\sum_{i=1}^{K} w_{i,x} w_{i,y}}{\sqrt{\sum_{i=1}^{K} w_{i,x}^2} \sqrt{\sum_{i=1}^{K} w_{i,y}^2}} \end{aligned} \qquad \textbf{(EQUATION 2.2)}$$

where w_x and w_y can be either item or user profile weight vectors depending on applied algorithm.

Pearson Correlation Similarity is mostly used for user-user similarity and calculated as follows:

$$sim(x, y) = \text{pcc}(\overrightarrow{w_x}, \overrightarrow{w_y}) = \frac{\sum_{s \in S} (r_{x,s} - \overline{r_x})(r_{y,s} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \overline{r_x})^2 \sum_{s \in S_{xy}} (r_{y,s} - \overline{r_y})^2}} \qquad \textbf{(EQUATION 2.3)}$$

where w_x and w_y are profile weight vectors for x and y, respectively. Additionally, $r_{x,s}$ and $r_{y,s}$ is rating provided by users x and y to item s, respectively. $\overline{r_x}$ and $\overline{r_y}$ are average ratings provided by users x and y. The set S_{xy} stands for commonly rated items by user x and y.

2.2.3 Previous Recommendation System Categorization

During previous studies, various recommendation techniques are classified into two broad categories, namely, memory based and model based methods. However, today some people in academia believe that this classification is no more appropriate, since decisive characteristics of model based techniques are used in memory based methods and vice versa [25]. Additionally, [64] proposes a combined memory and model based approach and it demonstrates use of a combined approach provide better recommendations compared to pure memory or model based approaches. Even some people do not agree with this classification in recommendation process any more, it will be useful to see the history of previous approaches and why this classification came on the scene.

2.2.3.1 Memory-Based Recommender Systems

Memory-based recommender systems make their predictions depending on the original user-item matrix. Always, up to date user-item matrix is used during recommendation process and this matrix shall be kept in memory throughout the procedure. This set of procedures is not so fast and requires greater amount of memory. Speed of recommendation is degraded since excessive processing is needed to return a single recommendation for a user-item matrix with high dimensions. On the other hand, memory-based predictions are always in agreement with the most current user ratings [3] and any off-line procedure is not required to synchronize recommendations with user-item matrix.

Some of the methods that can be included in memory-based recommender systems are basic collaborative filtering that are discussed in 2.2.5, item based collaborative filtering [63], algorithms using SVD/LSI for Prediction Generation [41].

2.2.3.2 Model-Based Recommender Systems

Model-based recommender systems make their predictions based on a model that is initially created according to previously given ratings (i.e. user-item matrix). Infrastructure model is usually constructed with a probabilistic approach that is envisioning expected rating value for a user-item pair. Expected rating value for a user-item pair is nothing but recommended rating for user-item pair. Initial model is created preliminarily based on user-item matrix; however, as model is trained with new ratings, model is updated and user-item matrix is no longer needed to update the existing model. Since there is no need to store user-item matrix in memory, almost all model based methods has less space complexity compared to memory based methods. Additionally, model based methods lead to rapid recommendation generation, since extracted model may implicitly store user profile similarities and same actions will not be performed for each recommendation. Most significant drawback of model based methods is synchronization of user-item matrix and existing model. A time consuming synchronization operation shall be performed to maintain the model. Thus, off-line model construction shall be executed periodically (i.e. as sufficient changes exist on user-item matrix) in order to keep the model alive.

Already proposed model based methods can be listed as, machine learning classification problem, personality diagnosis and Bayesian network modeling [3].

2.2.4 Content Based Methods

2.2.4.1 General Algorithm

A recommendation system using a content-based recommendation approach recommends items that are most similar to the ones that the active user preferred in the past [13]. An item is said to be preferred by a user if it is highly rated, bought, listened or experienced by the user depending on what the item is.

In addition to this brief explanation of content based approach, a generic content based recommendation system utilizes profiles as its building blocks. All recommendation process relies on profile structures of users and items. User profiles are constructed by analyzing the content information of already rated items by the active user. Furthermore, keyword analysis techniques play important roles during user profile construction [13]. Keyword analysis enumerates content information with several keywords according to its context. For instance, genre information for movies can be enumerated with *action, comedies, drama, adventure, western, etc.* as mentioned in [26].

Formally, content based methods aim to estimate the utility $u(c,s)$ of item s for user c based on the utilities $u(c,s_i)$ assigned by user c to item $s_i \epsilon S$ that are similar to item s [13]. For instance, a content based music recommendation system, uses singer, year, genre, length of music, etc. data to find out commonalities among songs that user c, has already rated in the past.

In general content based approaches, user profiles can be represented by a vector of weights $c(w_{c1},w_{c2},\ldots, w_{ck},)$ where each weight w_{ci} can be extracted from already rated items by the active user c and corresponds to the importance of keyword k_i, for the active user, c.

Item profiles are also constructed with item content information in a simpler and similar manner. As in the case of user profiles, item profiles are also weighted vector having same number of dimensions with user profiles and each component of the vector corresponds to the weight of each keyword k_i for the item. Item profiles will not be modified as users rate items in the execution, since they are built-in and come up with item content information.

According to these definitions of user and item profiles, user-item similarities and user-user similarities can be calculated by the help of Cosine/Vector or Pearson Correlation similarities which are already defined in (EQUATION 2.2) and (EQUATION 2.3), respectively. For each similarity calculation technique, profile weight vectors shall be replaced with user or item profile vectors.

Content based systems have an advantage of low computational complexity and high coverage. Low computational complexity is satisfied by simple management of profiles and low complexity of similarity calculations. Coverage is also high because a wide range of items will be available in a recommender system and there will always be a set of items that can be presented to users. Moreover, one of the most important features that shall be held by a recommender system is generation of understandable results because the more users understand the system, the most they trust it [26]. Understandability is satisfied by content based recommenders, since they can explain the reasoning behind recommendations. For instance, a user who has already liked “Mr. & Mrs. Smith” may be recommended “Troy” and it will be possible to explain the result as: You have liked “Mr. & Mrs. Smith” where the leading actor is Brad Pitt and recommender thought that you may like another Brad Pitt film “Troy”.

Even if content based approach is useful to extract user preferences easily and is a basis method for recommendation systems, there are certain problems underlying behind the approach. These problems are discussed in following paragraphs.

2.2.4.2 Drawbacks of Algorithm

2.2.4.2.1 Limited Content Analysis

The success of content based approaches is limited to the success of data extraction from items. That means the more features can be extracted from items; the more powerful recommendation can be performed. When there is not much content associated with items in a domain or there is no knowledge readily available to maintain, a content based recommender system cannot construct user and item profiles including significant information about items and that causes poor recommendation of items for users [29].

As mentioned in 0, the very first recommender systems contain only the analysis of text based data such as documents, Web URLs and Usenet news. Due to the simplicity of text based data compared to other currently available data types, automatic data extraction tools are currently available only for text data types [13]. However for multimedia data such as graphical images, audio or video streams, it is not easy to extract content information. For instance, a user may like a painting of Leonardo da Vinci because of its fascinating figures, but it will be really hard to extract this content information without complex image processing techniques.

Additionally, limited content analysis causes indistinguishable items if different products are specified with same set of keywords [13]. For instance, a poem recommender system cannot distinguish the differences between two poems if both have same set of words and the system only considers the number of words mostly appeared in the poem.

2.2.4.2.2 Overspecialization

As stated in former paragraphs of this section, content based recommendation process suggest people new items similar to the items that they liked in the past [13]. This guidance is not good enough for recommending different kind of products to people, because if generic content based approach is used, only most similar items to the active user profile will be presented to the user. For instance, an anticipated recommender system shall not present all movies of a director if a user has already highly rated a single movie of him. Genetic algorithms have already been proposed in order to solve overspecialization problem [27]. Another approach against this problem is Daily Learner [28] which is a news recommender system and tries to find a solution to overspecialization problem by filtering out the items not only very dissimilar to already liked items but also very similar items. In order to solve this problem, recommender systems shall somehow benefit from a kind of diversity.

2.2.4.2.3 Cold Start User Problem

A user newly signed into a system cannot be directed well enough by a content based recommender system. The reason behind this problem is that optimized user profile construction cannot be performed until a sufficient number of items are rated by the user. In order to solve this problem, some systems ask users to rate sample items for creating user profiles initially.

2.2.5 Collaborative Methods

2.2.5.1 General Algorithm

A recommendation system using a collaborative recommendation approach recommends items that are already liked by similar users. In generic collaborative filtering approach, content data related to items are not processed, however similarity of users and items are explored with an indirect way compared to content based approach. Different similarity functions that are used to calculate user similarities reveal different collaborative filtering approaches.

Formally, utility function $u(c,s)$ of item s for user c is estimated based on the utilities $u(c_j,s)$ assigned to item s by those users $c_j \epsilon C$ who are “similar” to user c [13]. A collaborative music recommender system uses only ratings of users that are similar to the active user when estimating rating value of the active user for a specific song. This corresponds to recommendation of songs to a user via his similar users. So that, a user having similar tastes with other users on music preferences, can be recommended similar songs without any content information analysis. In other words, if a user has already indicated his admiration about Elvis Presley and John Lennon songs, his similar users will also be recommended the songs of these singers which are already highly rated by the user.

According to descriptions in the previous paragraphs of this section, still a similarity calculation is needed to produce a prediction or a recommendation for a user. In generic collaborative filtering approaches, Cosine/Vector as presented in (EQUATION 2.2) or Pearson Correlation as shown in (EQUATION 2.3) similarities are generally used user similarity calculation methods. User similarities are calculated based on user profiles, where they are constructed with previous ratings of users.

Collaborative filtering approach resolves most of the shortcomings specified in 2.2.4. Recommendation through similar users inherently resolves the problem of limited content analysis. No content analysis is performed in basic collaborative filtering approaches so that various kinds of items ranging from text based documents to video streams can be recommended to users without any need for modification or manual handling on the system.

The overspecialization problem that is already described in 2.2.4 is also resolved by this approach, due to the possibility of similar users that have diverse tastes. For instance, while a person who likes action films will always be recommended other action films with the help of a content based recommender using only movie genres that will not be the case for the collaborative recommender. A collaborative system will search for similar users of the active user and it will possibly find out similar users who have also liked comedies or dramas. As in this case, diversity can be inherently achieved by collaborative approach.

Even if collaborative filtering resolves most of the shortcomings specified in 2.2.4, it has its own limitations. Some of the problems that can be observed in this approach are discussed in following paragraphs.

2.2.5.2 Drawbacks of Algorithm

2.2.5.2.1 Cold Start User Problem

The same problem already defined in the previous section, 2.2.4, still continues for basic collaborative methods. Some techniques are proposed in order to solve this problem such as recommending favorite items among users and presenting decisive items for determining a passive user profile in an easy manner [30]. However, whatever the selected technique is, a good solution to this problem shall identify a user profile with anticipating a small set of items to be rated by the user.

2.2.5.2.2 First Rater Problem

In a collaborative recommender system, an item cannot be recommended to users unless a substantial number of users rate it [3]. The reason behind this problem is non-existence of item analysis in generic collaborative methods. A newly added item to a collaborative recommender system cannot be interpreted until an association between users and the item is established, because there is no way to recommend a new item to a user if no similar user has already rated the item.

2.2.5.2.3 Sparsity

As mentioned in first paragraphs of this section, generic collaborative filtering approach aims to find similar users in order to recommend items to the active user. The recommendation step, weighting scheme or similarity calculation depend on the selected algorithm, however, finding similar users by the help of already provided data to common items is the key point in the execution of a generic collaborative algorithm. This is the reason why sparsity of existing data such as sparse user-item matrix degrades the overall performance of the algorithm. Even if this reason causes a lower performance, it also provides collaborative filtering approach to suggest people various kinds of items without any content analysis.

Basic collaborative filtering approaches use ratings given to common items to evaluate similarities between users. When a system has a sparse user-item matrix, number of ratings given to same set of items by any two users will be low and that will cause bad or totally no recommendation. Furthermore, a user whose tastes are unusual compared to the rest of the population, there can be no other user who is particularly similar and this will lead poor recommendations [14].

Table 1 is an example of user item matrix, where that is not the case in real world. For an application actively used on the web, every moment a new user can register to the system and a new item can be added to item list. All of these facts increase the dimensions of user item matrix and causes more sparse matrices. However, in order to keep a system alive, recommendations shall keep users, even with less number of ratings, within the system through good recommendations. Sparsity is the most significant problem of collaborative filtering and various methods are introduced in order to reduce sparsity in the recommendation process. In the remaining part of this section, sparsity reduction techniques will be discussed.

2.2.5.2.3.1 Use of Demographic Data

One way to overcome the problem of rating sparsity is to use other types of input data existing in the system. Inputs of a recommender system are already discussed in 2.2. For this purpose, demographic data of users can be used during recommendation process. The use of demographic data such as age, gender or education of the user for recommendation is called as demographic filtering and is an already used technique in [31]. [31] discusses learning user interests for web page and news articles recommendations and proposes a new restaurant recommendation application depending on the age, gender, area code, education and employment information of users.

2.2.5.2.3.2 Preprocessing User Item Matrix

As mentioned in preceding paragraphs, the main reason for sparsity problem is the sparsity of user-item matrix in which all data for a generic collaborative filtering system is stored to make recommendations. Therefore, another proposed way to overcome the problem of rating sparsity is trying to make user-item matrix denser.

Default voting is the simplest technique to reduce the sparsity of user-item matrix [42]. This technique proposes to use a default rating *d* to put some appropriate places of user-item matrix where no rating already exists.

Preprocessing of user-item matrix with user and item averages is another technique to reduce user-item matrix sparsity [41]. Preprocessing the user-item matrix before the main collaborative filtering algorithm application provides a better performance because of the reduction of sparsity.

In the user average scheme, average rating for each user is calculated beforehand and missing values for items that user has not rated are predicted by the help of the user's past ratings. The user-item matrix is filled according to formula given below:

$$r_{i,j} = \begin{cases} \overline{r_i}, \text{if user i has not rated item j} \\ r, \text{if user i rated item j with r} \end{cases} \qquad \textbf{(EQUATION 2.4)}$$

Preprocessing with item averages is still another method to create a denser user-item matrix [3]. According to this scheme, item averages are taken into account to replace missing values and user-item matrix can be filled as the following:

$$r_{i,j} = \begin{cases} \overline{r_j}, \text{if user i has not rated item j} \\ r, \text{if user i rated item j with r} \end{cases} \qquad \textbf{(EQUATION 2.5)}$$

Preprocessing of user-item matrix with either user or item averages completely removes the sparsity problem and newly generated matrix will be complete and full dense. However, both techniques use only one dimension of the user-item matrix to extrapolate existing ratings to whole space. User average scheme only considers a single row of the matrix and item average scheme only deals with a column. In order to generalize these approaches and unify it to a single preprocessing method, composite scheme is proposed.

One of composite preprocessing scheme is to use active user rating average and rating deviations of other users for the item.

$$r_{i,j} = \begin{cases} \overline{r_i} + \dfrac{\sum_{p=1}^{l} (r_{p,j} - \overline{r_p})}{l}, \text{if user i has not rated item j} \\ r, \text{if user i rated item j with r} \end{cases}$$ **(EQUATION 2.6)**

where *l* is the number of users who have already rated item i.

An alternative way of utilizing composite scheme is to use item average and rating deviation of the active user for items that he has already rated.

$$r_{i,j} = \overline{r_j} + \frac{\sum_{q=1}^{k} (r_{q,j} - \overline{r_q})}{k}$$ **(EQUATION 2.7)**

where *k* is the number of items rated by user i.

2.2.5.2.3.3 Use of Filterbots [3]

Filterbots are automated rater agents in a recommendation process. Filterbots rate items as they are inserted into the system. They are very similar to a user in the system, however, they can generate numerous predictions at a time, they are very generous and they never request predictions for themselves although they exist in user-item matrix. Filterbots can be intelligent as much as the amount of intelligence in their rating algorithm. As items are added to a recommender system, existing filterbots rate these items according to their built-in algorithms and update user-item matrix with these ratings. Additionally, user profiles can be mapped either to a single filterbot or to a combination of filterbots. A user is tried to be modeled with a combination of filterbots that is previous user ratings is examined and a best fitting combination of filterbots (i.e. filterbot algorithms) is tried to be mapped to this user. This provides the system to make recommendations even for new items, since already mapped filterbot combination to this user may predict ratings for these items. There are some proposed methods for combining filterbot ratings as described below:

- **Rating Average:** This is the simplest combination method and this method proposes to average ratings of different filterbots.
- **Regression:** Regression can be applied in order to create best fitting combination of filterbots. By this way, a linear combination of filterbot ratings is constructed and depending on user ratings, weights in linear combination is updated. In this scheme, different filterbots are independent participants of the combination.
- **Filterbots as users:** Filterbot ratings are inserted into recommender system and are presented as final prediction to users.

Moreover, as time passes, some of the filterbots can be worn out (i.e. useless in recommendation process, since it is no more a participant of any existing filterbot combination) and they can be removed from the system safely.

2.2.5.2.3.4 Use of Dimensionality Reduction Techniques [3]

Singular Value Decomposition is a dimensionality reduction technique that may be used to convert a sparse matrix to a denser one. This technique can also be used to reduce dimensions of user-item matrix. After a dense user-item matrix is generated, rows and columns of this matrix are used as pseudo-user and pseudo item vectors, respectively, to calculate proximities between users and items [41][65]. However, during the application of this technique, dimension reduction rate shall be large enough to capture relationships between users and shall be small enough to avoid over-fitting errors.

2.2.5.2.4 Loss of Neighbor Transitivity [3]

In generic collaborative filtering methods, transitivity of user similarity is not taken into consideration. Assume that, ratings of two users A and B are highly correlated and user B has also a high correlation with another user C. Then there is a high possibility that users A and C will have similar tastes because of common correlation with user B. Nevertheless, users A and C will not be qualified as similar users unless they have rated many common items with a similar rating scheme.

This deficiency may be resolved by similarity propagation, but recent works in the area have not highly concentrated on such an improvement for collaborative filtering methods. A proposed method on Local and Global user similarities is discussed in [38][39]. The intention of this method is to improve the similarity calculation between users by the help of the algorithms already employed in graph theory.

2.2.5.2.5 Computationally Expensiveness [36]

A generic collaborative filtering algorithm has a lazy evaluation technique. It compares the active user ratings with each of the remaining users in order to find out most similar users and this can cause much time at query time. For a system that contains millions of users and billions of items, recommendation even for a single user wastes a great amount of time which is not feasible for a real world application. For this reason, such a huge system can only be active with a periodically offline processing of recommendations. This will cause recommendations not to be up to date.

2.2.5.2.6 Easy Attacks by Malicious Users [36]

A system using a generic collaborative filtering approach is attractive for malicious users since it is easy to intrude into the system and direct the system according to their own desires. The simplest attack is the copy-profile attack, where the attacker copies ratings of a target user and highly rates some other items that are desired to be preferred to the target user. By this way, attacker will be most similar user to the target user profile and items that are not rated by the target user but the attacker will be recommended to the target user. With the introduction of FOAF and RVW formats which propose a decentralized Semantic Web concept, these type of attacks will be more apparent.

2.2.6 Hybrid Methods

2.2.6.1 General Algorithm

A recommendation system using a hybrid recommendation approach recommends items to users with a combined method of content based and collaborative filtering. The reason behind using this combined approach is to eliminate the deficiencies already discussed in 2.2.4 and 2.2.5. Four different ways to combine content based and collaborative filtering approaches can be specified as the following [13].

2.2.6.1.1 Separate Content-Based and Collaborative Filtering Components

There are two proposed ways to combine separate content based and collaborative filtering components. As specified in the method name, there are different components to get estimations and their results will be combined somehow. One way is to combine estimated rating results of all components with a weighting or voting scheme. The weighting scheme can be a linear combination of estimated ratings as proposed in [40]. An alternative way is to select the component that is better than other components based on a recommendation quality metric as in the case of [28]. In this approach, not only two different components but also many other components implementing various content based or collaborative filtering approaches can be combined.

2.2.6.1.2 Content-Boosted Collaborative Filtering

One of the very first approaches, namely Fab System [14] and another method using Demographic Filtering as proposed in [31] are examples of this hybrid method. In this technique, content based profiles are created as in generic content based method and collaborative user similarities are calculated between these profiles. The use of content based profiles of users also reduces the sparsity problem of collaborative filtering. In addition to that, overspecialization problem of content based filtering is eliminated by this technique, since recommendations are performed not only based on user content profiles but also commonly rated items.

2.2.6.1.3 Collaboration in Content Based Filtering

Dimensionality reduction techniques on content based profiles can be classified in this category. [66] describes a use of latent semantic indexing to create a collaborative view on user profiles which are represented as term vectors and based on content information.

2.2.6.1.4 Unified Content Based and Collaborative Filtering

Many different approaches in this category are already proposed. [67] is a rule based method, which extracts rules by the help of both content based and collaborative filtering approaches. Another technique based on Bayesian networks that employ Markov Chain Monte Carlo methods for parameter estimation is proposed in [68] and [69]. This technique extracts item and user profiles and already defined parameters to make a recommendation are estimated by the help of already given ratings and these constructed profiles. Thus, both content based profiles and user similarities are used in prediction. Moreover, another technique unifying content based and collaborative filtering methods is case-based reasoning which can be classified as a knowledge-based technique [70]. This approach addresses new item and new user problems of collaborative filtering. In [70], a knowledge-based restaurant recommender system is proposed to suggest foods for people. In this case, background knowledge on foods shall be readily available for recommendation. “Seafood is not vegetarian” can be an example for this kind of knowledge. Recommendations are performed depending on existing knowledge in specified domain and user-item matrix; however, knowledge acquisition for each item can still be a problem for knowledge based systems. Domains, for which knowledge information can be gathered easily in a machine readable format, shall be a target for knowledge based systems. The most time consuming and crucial job is knowledge acquisition and as this knowledge is obtained, the drawback of cold start user or new item problems will be substantially degraded.

CHAPTER 3

RELATED WORK

3.1 Use of Trust Data

Problems of generic collaborative filtering approach are discussed in 2.2.5.2. Sometimes, these problems became showstopper in a recommendation process. In order to solve deficiencies of collaborative filtering approach, possible solutions are proposed and some of them are actively being used in several systems. Most of the solutions are based on inclusion of other input data types existing in the system or managing already used data in collaborative filtering approach with a different way.

In addition to already discussed methods in 2.2.5.2, recent progress of the area concentrates on social relationships among users. In most of the existing systems using social relationships [11][24][32][33][34][35][36][37], social networks are constructed based on trust relationships.

According to previous studies, it is shown that trust relationships among users provide the system to increase the coverage of recommender systems while preserving the quality of recommendations [36]. Additionally, using trustworthy users in recommendation process filters out malicious users who are trying to influence recommendation accuracy. This makes the system attack-resistant. Trust propagation is a method to extract not specified trust relationships among users with the help of already existing trust statements. By using trust propagation during recommendation process, cold start users, who have rated only few items, can also benefit recommender system even if they have trusted a few users. Furthermore, computational performance of trust metric seems more reasonable compared to generic collaborative filtering since a basic trust metric method uses only trusted users in recommendation process and trust between users is either explicitly stated or can be easily extracted through trust propagation, where in collaborative filtering approach ratings of the active user shall be compared with ratings of each remaining user to calculate user similarities. Most recent approaches also refer to distrust between users, however, there is no commonly accepted method to exploit this new feature.

In the scope of this thesis, two generic approaches, namely global and local trust data use in recommender systems will be discussed in this section.

3.1.1 Global Trust Data

Global trust metric approach stores an overall trustworthiness metric for each user which can be named as reputation. Each user has a single reputation metric and this metric is determined by votes of other users. If a user feels another user has a similar taste with him, he can give positive vote for that user and the reputation of the user who gets a positive vote will be incremented by one. In a similar way, a negative vote causes the user reputation to be decremented. Global trust metric summarizes what the community as a whole thinks about a certain user without any inspection of individual differences. Highly reputable people in the system will have more effect on recommendations compared to people whose reputation is not high enough. At first glance, using reputations during recommendation process seems as a smart idea; however, this approach also has its own deficiencies. In a dynamic system using reputations in its recommendation engine, any user can increase reputation of any other user so that fake users who have high reputations can exist in the system with the help of other fake users that are registered to the system. This possible system attack has annihilated the reputations in the area of recommender systems, where systems regarding reputations still exist in various applications [32].

3.1.2 Local Trust Data

Local trust metric approach for social relationships among people employs trust relationships between users. Instead of reputation metric, there is no global view of reliable or trustworthy users, so that a user can be considered reliable by one user and unreliable by another one [36]. Trust relationship exists between a single pair of users and that is why a similar attack as in the case of global trust cannot occur in the system as long as the user does not trust a fake user. People who are trusted by the active user and people who are similar to the active user according to user-item matrix can contribute to the recommendation. In fact, some systems only use trust relationships in recommendation [11][24][32][33][34][35][36][37] and some other systems propose new unified techniques to merge both user rating similarity and trust relationships [36]. Local trust metric allows people personalizing user trust metrics independent of other users based on their web of trust. Even if local trust can be seen as an upgraded version of the global one, its space and computational complexity is much worse [11]. Trust based methods that can be referred as state of art solutions are discussed in the next section.

3.2 Related Studies on Recommender Systems

3.2.1 MovieReco: A Recommendation System [29]

3.2.1.1Proposed Technique

A recommender system is a combination of information filtering and intelligent agent system and acts as a personal decision guide to lead people about unlimited number of items such as movies, songs, restaurants, travels etc. according to their tastes. Initial approaches in the area of recommender systems ignore trustworthiness of users and focus on statistical accuracy of proposed algorithms. However, locating bad users, who are misleading the system, is so crucial to keep the system alive. As discussed in 2.2.5 and 2.2.6, systems including flavor of collaboration try to find out similar users of the active user and recommend items to him by the help of his similar users. If bad users have become similar to the active user, the recommender system cannot estimate items that the active user will like, precisely. Although bad users in the system can be neglected if their count is small enough, but if more such users exist in the system, it will not be possible for the recommender system to arrive at accurate solutions.

One of the very first ideas to get out of bad users in the system, testing user knowledge with content information of items is proposed. Users, who can pass the test, will be eligible to rate products. Content information test for each product waits for the name of three leading actors, two leading actresses, name of the director and type of movie along with voting each of these attributes. The overall rating for the movie is automatically calculated by the system with the following formula:

$$r_{u_i f_j} = f_s(u_i) \sum_{k=0}^{7} r_k \qquad \textbf{(EQUATION 3.1)}$$

where $r_{u_i f_j}$ is the standardized rate of user u_i on film f_j, r_k is the rating assigned by the user u_i to each component of content information test and $f_s(u_i)$ is the standardization factor for u_i and it is calculated to unify different rating styles of users. The system calculates the standardization factor as shown below, when the user is newly registered to the system by the help of content information test for best three movies that the user has ever seen:

$$f_s(u_i) = \frac{\text{highest rating in best three movie ratings}}{\frac{1}{3} \sum_{j=1}^{3} br_j} \qquad \textbf{(EQUATION 3.2)}$$

where br_i is the best rating assigned to movie j by the user i.

The algorithm of MovieReco uses an item-item algorithm and Pearson-r algorithm that is already utilized in GroupLens [18] project. MovieReco algorithm can be described as follows step by step:

- Let F_S be the set of all films that the active user has not already rated. Find correlations between each pair of films in F_S.
- Based on the calculated correlations, select S films in F_S that are mostly correlated with film F.
- Calculate correlations between every pair of users by considering only the ratings given to S similar films that are already selected.
- Select X users that are mostly correlated with the active user.
- Plot the graph of ratings for film F vs. correlation coefficient values of each user in X with the active user.
- Find best fitting straight line through the points in the graph and rating at correlation coefficient = 1 is the estimated rating of the active user for film F.

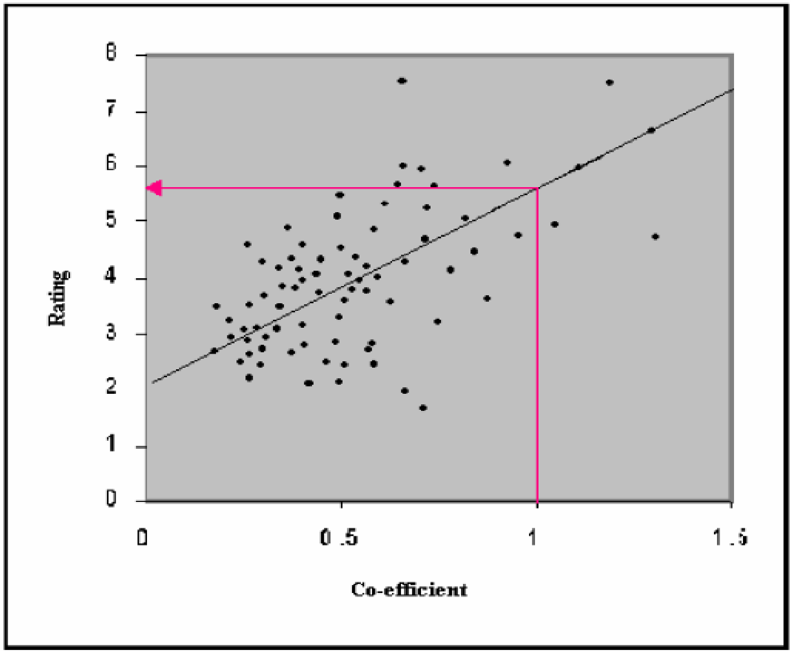


Figure 1 – Ratings for film F vs. Correlation Coefficient values of each user in X with the active user

3.2.1.2 Evaluation

Knowledge test for each user before taking their ratings into consideration was a new approach for recommender systems. During the initial use of MovieReco on an intranet filtered out 13.23% of users in the first hundred users. Additionally, since only content based similarity calculation is used, first rater problem discussed in 2.2.5 for collaborative methods is eliminated. This approach again suffers sparsity problem. The reason behind that is the best fitting line acts as a hypothetical user who is the most similar user to the active user and due to data sparsity correlation coefficient values may not reflect real correlations between users and this will cause results to be inaccurate.

3.2.2 Exploring Movie Recommendation System Using Cultural Metadata [26]

3.2.2.1 Proposed Technique

Evolution of WWW has provided people an opportunity to state their thoughts, feelings about various kinds of items on WWW. Blogs, video sharing sites, forums are some of platforms that people can depict their opinions about items through their word of mouth. The possibility of indicating thought with word of mouth is a great simulation of real life on a virtual world, since in real life almost every day; items are evaluated with talks by various people. Relocation of opinions from real life to WWW also increased the researches on information retrieval techniques such as natural language processing.

Actually, WWW has a characteristic to transform temporal ‘parols’, i.e. vanishing talks in real world, into spatial texts on the web. A part of parols in people’s mind is related to songs, movies, travels etc. Reviews, comments and ratings that are published on user blogs and forums, are accumulated on cultural contents in databases of different systems. Aggregation of such discourses constitutes the ‘cultural metadata’ that has a great potential for recommendation systems area.

Cultural metadata is defined [49] as the following:

The information that is implicitly present in huge amounts of data needs to be extracted with techniques for information retrieval.

Five types of cultural metadata in IMDb [5] can be listed as below:

- **User Comments:** User comments are reviews about movies they have seen to share their impressions and opinions with other people. User comments will benefit people who have not seen a movie yet.
- **Plot Outlines:** Plot outlines are brief plot summaries of movies which do not include any judgment about the quality of them.
- **Synopsis:** Longer plot outlines**.**
- **Plot Keywords:** Controlled keywords to describe a film such as “Sabotage”, “Genetic War”, “Honor” etc. Keyword controls are managed by IMDb.
- **Genres:** Genre information of movies.

Above cultural metadata types can be classified into two types, namely text-type and keyword-type. User comments, plot outlines and synopsis fit into natural language and labeled as text-type cultural metadata. Plot keywords and genres are controlled keywords and labeled as keyword-type cultural metadata. Most of the movies have at least a genre description or user comments and utilizing cultural metadata seem reasonable.

By the help of ConceptNet2.1, text-type data can be processed to extract words of impression and feeling in comments and reviews. This method provides an easy handling of text-type data as keyword-type data. For each preprocessed text-type data word and keyword-type data word, TFIDF metric is calculated as below:

$$TFIDF_{i,j} = TF_{i,j} x DF_{i,j}$$

(EQUATION 3.3)

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

(EQUATION 3.4)

where $n_{i,j}$ is the number of occurrences of term t_i in document d_j

$$IDF_i = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

where $|D|$ is total number of documents **(EQUATION 3.5)**

$|\{d : t_i \in d\}|$ is number of documents where term t_i appears

where $TF_{i,j}$ is the frequency of term i in a document j and IDF_i is the inverse document frequency as a measure of the general importance of the term.

TFIDF metric assigns a weight for each word. Words having high weights are more significant for similarity calculation. User profiles comprise keywords from text-type and keyword-type data. Each user profile is a weight vector where each word weight corresponds to a dimension of the vector. User-user similarities are calculated with cosine similarity metric in (EQUATION 2.2).

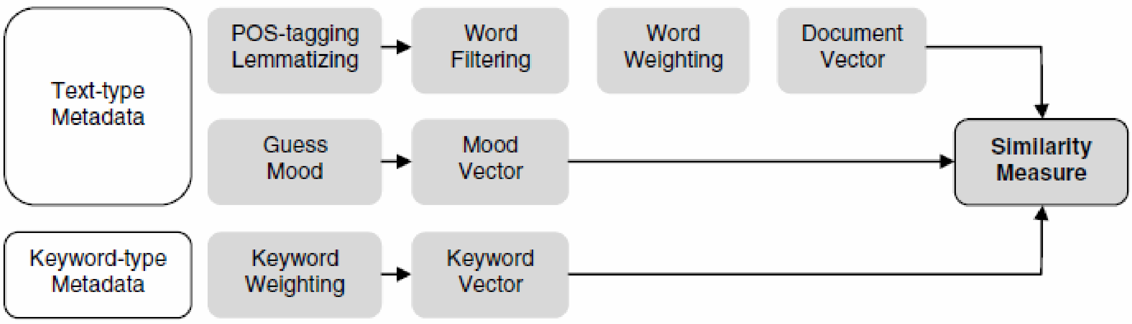


Figure 2 – Overall design of [26]

Additionally, ConceptNet2.1 can extract mood of a user from reviews and comments, this data can also be used in text-type cultural metadata assessment. Overall system design can be shown as in Figure 2.

3.2.2.2Evaluation

According to the given results in [26], user comments have a wide spectrum of implicit information about movies and have a great significance for extracting user impression where plot outlines and synopsis did not show a good performance because of data sparsity. User comments perform well even for actors/actresses and modifiers in sentences play an important role in representing impression. Furthermore, keyword-type data, i.e. genres and plot keywords are still important to be effective on recommendation, since they are controlled and precise. Actually, keyword-type data usage in recommendation lowers computational costs of the system, since text-type data includes a preprocessing phase.

3.2.3 Improving Recommender Systems with Social Networking [44]

Traditional approaches in collaborative filtering aim to filter all possible options according to user tastes expressed through prior item evaluations. During previous researches on collaborative recommender systems, almost all improvement ideas are on implementation of algorithms and the potential use of any new data such as social networks and relationships between people is totally ignored. In daily life, people who will make a choice on a situation, seek advice from peers or other trusted resources. However, applications that were emerged as products of previous studies did not provide any interface related to social relationships.

In fact, in order to get better results from a system, learning the psychology of the user is one of the most important points [43]. Additionally, Human Computer Interaction focuses on how to match user’s mental model with the system and how to manipulate system interface to make system functionality transparent to users [45][46][47]. Trust metric use in a recommender system rather than any effort to extract user similarities from previous user ratings can also provide user expectations in real life to be met, namely advice seeking. In short, the reason to use trust data in recommendation process aims to consider a broader range of factors with simulation of advice taking and decision making in real world.

According to [48], the quality and usefulness of friends' recommendations are found as more preferable compared to collaborative filtering method recommendation, since a subjective choice (i.e. friend recommendation) seems to be more accurate than an objective choice (i.e. a collaborative filtering method recommendation) in a recommendation process. The main idea behind this reasoning can also be stated as:

Friends are seen as more qualified to make good and useful recommendations compared to recommender systems mainly because they are assumed to know more about the recommendee.

Work explained in [44] contains a phase of interviews to determine how people seek advice and how people decide when choosing an option through many possible ones. Specifically, 12 one-on-one interviews and five focus groups (32 participants) were conducted for this purpose and the results collected can be listed as below:

- The relationship between advice seeker and giver is a key indicator for taste overlap and mutual knowledge.
- Decision makers differentiate between objective (factual & specification driven) and subjective domains (taste) and apply different advice seeking strategies for each.
- Past experience with the recommender impacts on trust.
- Aggregation of user opinions used as a popularity indicator.
- Ulterior motives of a recommender are perceived to have a negative effect the quality of the advice given.
- Decision making transfer & sharing of responsibility as a motivator for seeking advice.

Unfortunately, one of the most significant results revealed is that the relationship between advice seeker and giver is key point during recommendation process and this relationship can be mapped to a trust metric. Furthermore, as mentioned in previous paragraphs, subjective choices have better performance compared to objective ones in recommendations according to previous works. This result can be extracted from the second fact listed above because understanding user tastes is the main consideration in recommendation of a movie, song or a restaurant. The above survey can be analyzed in a more detailed way, however analyzing only the psychology of a user and the interactions between users during a taste based recommendation is enough for the time being.

3.2.4 Trust in Recommender Systems [24]

3.2.4.1 Proposed Technique

According to [50], trust metric can be divided into two classes:

- **Context-Specific Interpersonal Trust:** A user may trust another user with respect to one specific situation, but does not have to trust in another. For instance, a person can trust another for movies having genre Drama, but not for other genres.
- **System/Impersonal Trust:** A user trust in the whole system, similar to reputation described in 2.2.5.

Similar classification is also used in TrustMail system [51]. The importance of mails is determined according to reputation or trustworthiness of the sender. A mail participant can be trusted for a specific topic or for all topics.

A benchmark algorithm Resnick’s [19] formula can be represented as below:

$$c(i) = \bar{c} + \frac{\sum_{p \in P(i)} (p(i) - \bar{p}) sim(c, p)}{\sum_{p \in P(i)} \left| sim(c, p) \right|} \qquad \textbf{(EQUATION 3.6)}$$

where p's refer to producers who are acting as participants of a recommendation process for consumer c. $\overline{p}$ and $\overline{c}$ refer to average ratings of profiles p and c, respectively. P(i) is a set of producers who have already rated item i and p(i) stands for rating that is given for item i by producer p. Moreover, similarity sim(c,p) can be calculated with standard Pearson Correlation Coefficient as described in (EQUATION 2.3). This formula is a generic approach for a recommendation process; similarity calculation alternatives create different approaches.

According to [24], a user A trusts user B if correlation between ratings of (A, i) and (B, i) user-item pairs are high enough. This proposition states that [24] tries to extract trust relationships among users from user-item matrix where no explicit trust data exists. In order to formulate this statement, one can formulate what a correlation means for a single item and then generalize it to whole set of items. A recommendation for consumer c can be labeled as correct if (EQUATION 3.7) holds and this corresponds to a good correlation between producer p and consumer c for item i.

$$Correct(i,p,c) \Leftrightarrow \left| p(i) - c(i) \right| < \varepsilon \qquad \textbf{(EQUATION 3.7)}$$

Full set of recommendations that producer p is involved by Resnick's formula (EQUATION 3.6) is named as RecSet(p) and CorrectSet(p) is a subset of RecSet(p) which includes only recommendations that p and c are highly correlated according to (EQUATION 3.7):

$$\mathrm{Re}\, cSet(p) = \{(c_1, i_1), (c_2, i_2), \ldots, (c_1, i_n)\} \qquad \textbf{(EQUATION 3.8)}$$

$$CorrectSet(p) = \{(c_k, i_k) \in \mathrm{Re}\, cSet(p) : Correct(i_k, p, c_k)\} \qquad \textbf{(EQUATION 3.9)}$$

After the given definitions of base formulas profile level and item level trusts can be described as the following, respectively:

$$Trust^{P}(p) = \frac{|CorrectSet(p)|}{|\mathrm{Re}\,cSet(p)|}$$ **(EQUATION 3.10)**

$$Trust^{I}(p,i) = \frac{|\{(c_k, i_k \in CorrectSet(p) : i_k = i\}|}{|\{(c_k, i_k \in \mathrm{Re}\,cSet(p) : i_k = i\}|}$$ **(EQUATION 3.11)**

Profile level trust is very coarse grained since a user can be very reliable for a small set of items and not much for other items. (EQUATION 3.10) does not give any interpretation about variety in a profile for different items. However, item level trust is fine grained because trustworthiness of a producer for each item will be calculated independently.

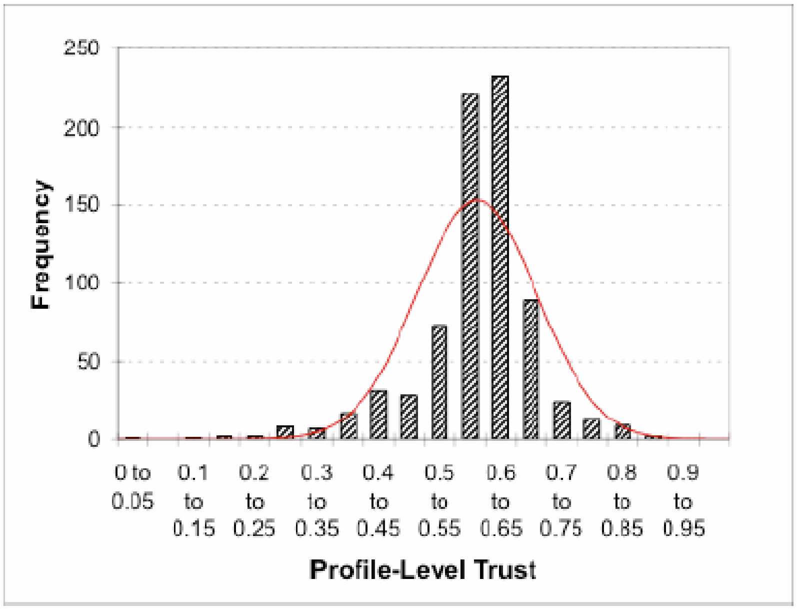


Figure 3 – Profile Level Trust value vs. Frequency

One of the common characteristics of frequencies in Figure 3 and Figure 4 are that they are both normally distributed; however, their variations are not similar. As estimated, item level trust has greater variety, while averaging process during profile level trust value calculation degrades the variation in Figure 3.

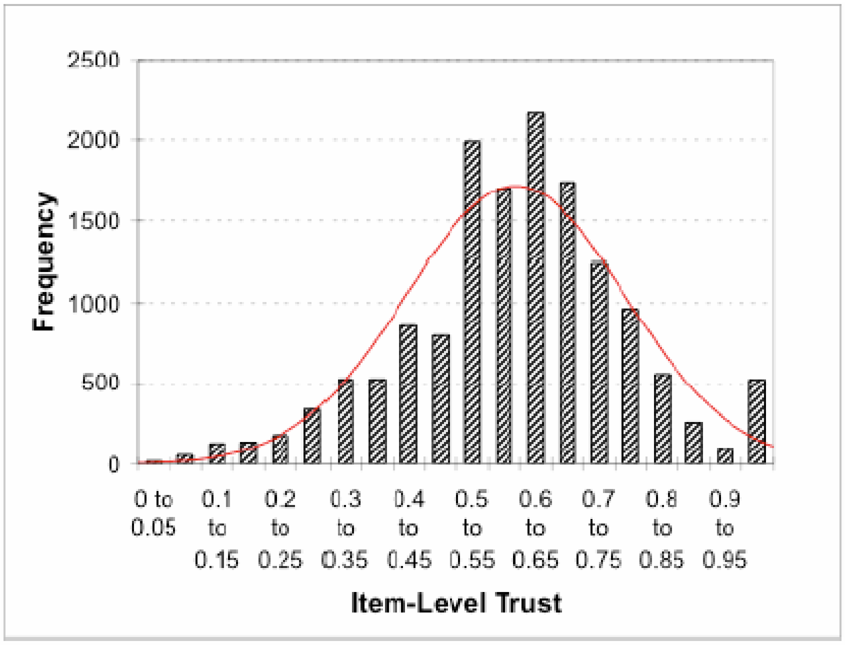


Figure 4 – Item Level Trust value vs. Frequency

There are three methods in [24]:

- **Trust-Based Weighting:** As mentioned in previous paragraphs of this section, weight in Resnick's formula can be modified in different approaches and one of the proposed approaches is as the following:

$$c(i) = \bar{c} + \frac{\sum_{p \in P(i)} (p(i) - \bar{p}) w(c,p,i)}{\sum_{p \in P(i)} |w(c,p,i)|}$$ **(EQUATION 3.12)**

$$w(c,p,i) = \frac{2(sim(c,p))(trust^{I}(p,i))}{sim(c,p) + trust^{I}(p,i)}$$ **(EQUATION 3.13)**

This method weights rating deviations of producers according to trust and similarity between producer and consumer. Here, the weighting scheme is harmonic mean and this can be replaced with another weighting scheme to propose new trust-based weighting approaches.

- **Trust-Based Filtering:** Producers who are not trusted enough can be filtered with a preprocessing step so that a producer that will participate in the recommendation process will have a trust score higher than a threshold value.

$$c(i) = \bar{c} + \frac{\sum_{p \in P^{T}(i)} (p(i) - \bar{p}) sim(c,p,i)}{\sum_{p \in P^{T}(i)} |sim(c,p,i)|}$$ **(EQUATION 3.14)**

where $P^{T}(i) = \{p \in P(i) : Trust^{I}(p,i) > T\}$.

- **Combined Trust-Based Weighting & Filtering:** In this approach, filtering process is initially applied as in Trust-Based Filtering method and then Trust-Based Weighting is performed.

$$c(i) = \bar{c} + \frac{\sum_{p \in P^{T}(i)} (p(i) - \bar{p}) w(c,p,i)}{\sum_{p \in P^{T}(i)} |w(c,p,i)|}$$ **(EQUATION 3.15)**

3.2.4.2 Evaluation

Error calculations with standard leave-one-out technique for each proposed method is presented in Figure 5. Error rates are calculated with temporarily treating a producer as if it were a consumer and rating predictions are generated by the help of remaining producer profiles and in each step a single producer is used as a lone recommendation partner. Comparing results of known and estimated rating values outputs the error rates shown in figures.

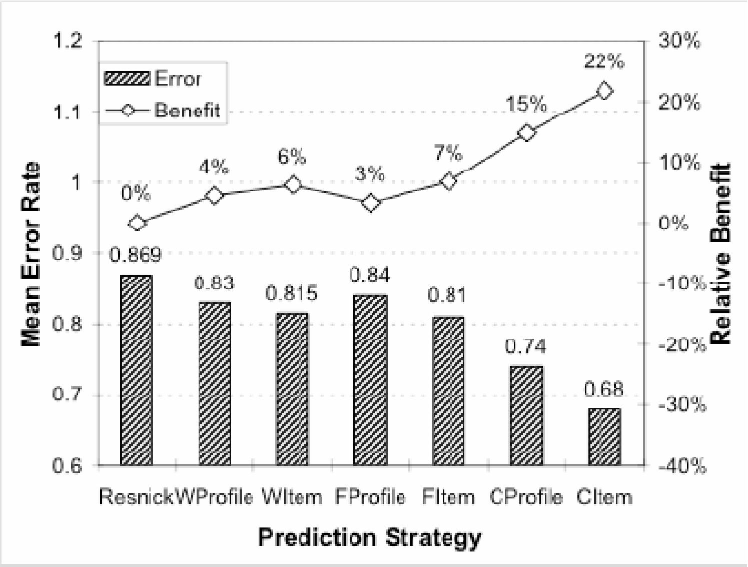


Figure 5 – Prediction Strategy vs. Error Rate

According to Figure 5, strategies depending on item level trust outperform profile level trust strategies. This is expected because item level trust values are more fine-grained and accurate. An individual can be very trustworthy on profile level where he may not very trustworthy for some items. This resolution is lost during averaging process. Furthermore, mean absolute error is far more decreased when combined methods are used instead of stand-alone methods. For this reason, it seems more reasonable to eliminate untrustworthy users i.e. users whose trustworthiness is below a threshold value, in trust-based weighting phase. Even if untrustworthy users are participated in weighting scheme with a little weight, their existence in trust-based weighting (EQUATION 3.12) reduces the performance.

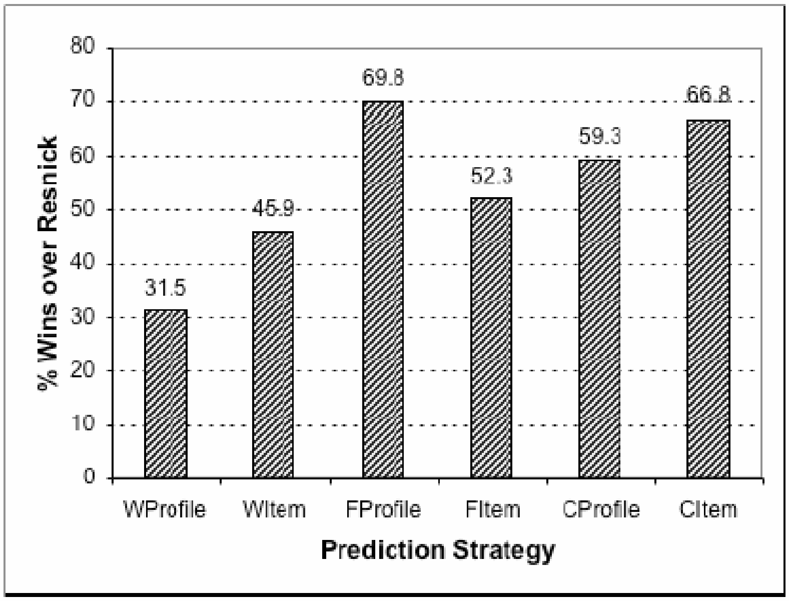


Figure 6 – Prediction Strategy vs. %Win over Resnick's standard formula

Figure 6 shows winning ratio of proposed strategies over Resnick's standard formula for each consumer/competition used in leave-one-out technique. For instance, profile level trust-based weighting loses most of the competitions while it has already reduced the mean error rate. The reason behind that is analogous to difference between profile level and item level trust strategies. Average error rate can be decreased because of very good performance in some competitions, where no such a performance in others. Due to the use of mean error rate, competition level result details disappeared. Winning rates deal with this fact, where it loses error rate differences in each competition, since it only cares the winner or the loser of the competition without considering error rates in competitions.

Most of the results in Figure 6 are correlated with results in Figure 5, however, profile level trust-based filtering method wins 69.8% of competitions with only 3% mean error rate improvement over Resnick's standard formula. This corresponds to the fact that this method wins most of the competitions with very small improvements on mean error rate.

3.2.5 Moleskiing: A Trust-aware Decentralized Recommender System [11]

Moleskiing system is created to provide information about lately available ski tours. The most important expectation about a ski tour is the security of the tour i.e. to prevent mountaineers from avalanches hazard. For this reason, through Moleskiing system, people can annotate their last experiences and can share both the quality of the route and snow conditions. However, the conditions of a particular ski route can vary day by day, thus ski mountaineering domain differ from a standard collaborative filtering domain such as movies, songs etc. where people's taste does not change such fast. This results the fact that a collaborative filtering based system for ski mountaineering is required to decide how to recommend a route by taking time into consideration.

The notion of local trust is used in Moleskiing which is discussed in 3.1.2. Users express their annotations about various ski tours that they have recently experienced in a decentralized and open publishing environment. Additionally, a user can issue trust statement about any other ski mountaineer and this trust metric is assumed to be expressed as a profile level trust i.e. trust statement is extrapolated to all annotations of the trusted user.

There are two kinds of ratings for a ski trip in Moleskiing system. One is trip rate and the other is route rate. Trip rate indicates security of the route and route rate stands for quality of the route. Each user states these ratings for a trip that they have already experienced and recommendation for a user is generated by using his trusted users with their ratings. Firstly, ski routes that have low trip ratings are filtered out since recommending insecure routes will be meaningless. After that, routes are ordered based on the general appreciation of trustable users, where trustable users are identified as explicitly or implicitly trusted by the active user with a minimum degree of 6 out of 10. Implicit trust metric is generated by trust propagation algorithms and the general appreciation of users is indicated by route ratings. Overall formula to reach rating estimations for each route is given as the following:

$$Rating(Route, A) = \frac{\sum_{U \in TrustableUsers(A)} \left[Trust(A,U) Rating(Route,U) \right]}{\sum_{U \in TrustableUsers(A)} \left[Trust(A,U) \right]} \qquad \textbf{(EQUATION 3.16)}$$

As mentioned in preceding paragraphs, for a ski tour recommendation system, security of the route is so significant and local trust metric provide a more attack resistant use of trust input compared to global trust. If the global trust metric were used, fake users having high reputations would cause recommendation of dangerous trips.

As a disadvantage of Moleskiing prediction formula (EQUATION 3.16) cannot estimate a rating for a route if the active user has no user in his web of trust. In order to produce a recommendation list for this kind of users, Moleskiing orders ski routes by using user reputations (i.e. global trust) and provides an option to trust highly reputed users for adding the active user into the social network.

3.2.6 FilmTrust: Movie Recommendations using Trust in Web-based Social Networks [33]

3.2.6.1 Proposed Technique

FilmTrust is a one directional trust based collaborative filtering movie recommender system. A user in the system can specify a trust rating in a range 0 to 10 and this rating is only shown to the rater. The reason behind that is to keep the rated user objective for his further trust ratings. In addition to explicitly stated trust ratings among people, FilmTrust also uses an algorithm, TidalTrust, to estimate trust ratings that are not explicitly stated. TidalTrust is a trust network inference algorithm that recursively looks for trust estimation between any given two people based on the paths that connect them in the network. By the help of TidalTrust, system uses inferred trust whenever no explicit trust exists. FilmTrust system assumes that trust must be correlated with user preference similarities. Ziegler and Lausen have already verified the existence of a correlation between trust and user similarities with their empirical study on a real online community [52]. Moreover, [36] and [53] are other studies on this issue. They support even an incorporation of a binary trust increases the coverage and sparsity problem will be mostly resolved if enough trust metric is available.

Recommendation algorithm steps of FilmTrust system can be listed as below:

- Trust ratings between rater (i.e. the active user) and each remaining user is calculated by the help of TidalTrust.
- Mostly trusted users are added to a set named as S.
- Retrieved trust scores stand for similarity between rater and used as weights of trusted people as shown in (EQUATION 3.17)

$$Rating(Route, A) = \frac{\sum_{U \in TrustableUsers(A)} \left[Trust(A,U) Rating(Route,U) \right]}{\sum_{U \in TrustableUsers(A)} \left[Trust(A,U) \right]} \qquad \textbf{(EQUATION 3.17)}$$

where r_{sm} is recommended rating for user s to movie m and t_{si} is estimated or explicitly stated trust score from user s to user i.

In FilmTrust, a recommendation list is presented with two ratings attached to each movie, which are average rating for the movie and inferred rating, respectively. Inferred rating is calculated with trusts among people and user-item matrix. Thus, inferred rating shall reflect the difference between active user's preference and the average rating of the movie. Additionally, reviews for movies are sorted from the most trusted user's review to the least, because of the correlation between trust and user preference similarities.

Analysis in FilmTrust system is rather different than already proposed analysis methods. As mentioned in previous paragraph, inferred rating shall reflect the difference between the active user's preference and the average rating of the movie. This means that recommendations shall reflect preferences of users, who have dissimilar tastes compared to population. In order to examine the performance of FilmTrust, three base difference metrics are defined: the difference between the actual rating and the recommended rating (call this dr), the difference between the actual rating and the average rating (call this da) and the difference between the actual rating and the rating generated by a generic collaborative filtering (ACF) (call this dcf).

For users who gave similar ratings with average rating cannot sufficiently benefit a personalized recommendation, because the correct recommended rating will not be so different than the average rating. However, the point of the inferred rating can become more apparent when a user has a more distinct taste compared to remaining population. For these users, system shall recommend rating far away from the average and close to user preferences.

3.2.6.2 Evaluation

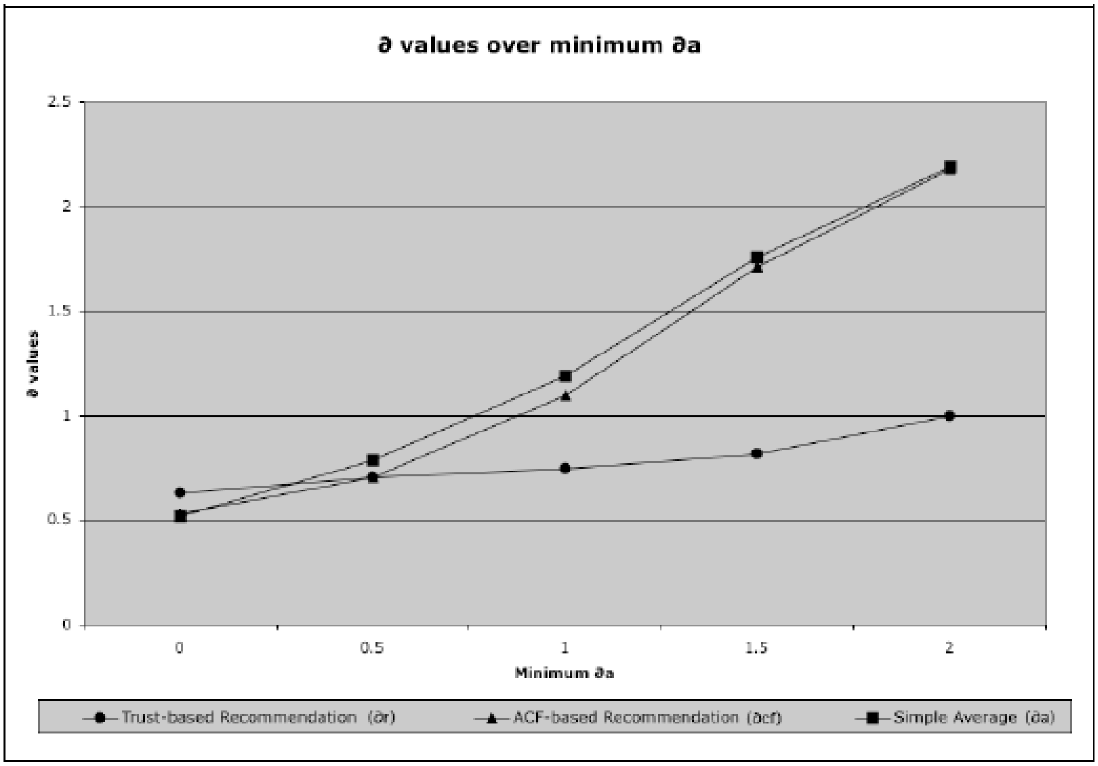


Figure 7 – da, dcf, dr changes with minimum da threshold

Figure 7 presents the results of da, dcf and dr differences with respect to minimum da threshold. With a minimum da threshold 0, where users having similar actual ratings with average ratings of movies also included, there is not so much difference in the results of methods and personalized ratings do not offer a benefit over average rating. As threshold value is increased with 0.5 increments, the ratio of users whose ratings diverge from the average increases. With each increment of threshold, difference between actual rating and inferred rating gets smaller, where ACF rating and the average rating move far away from user preferences. The reason behind the poor performance of ACF can be explained as it only captures the overall correlation between users. Additionally, number of users whose actual ratings are similar to average ratings will be much more compared to the number of other unusual users (i.e. users whose ratings are divergent from the average) similar to the active user. Thus, according to ACF weighting scheme, estimated ACF rating converges to average item rating.

A potential drawback of FilmTrust system is same with Moleskiing discussed in previous section 3.2.5. A trust cannot be estimated by TidalTrust if no path from the active user to another user exists. For this reason, users, who cannot be accessed from the active user with a path in social network, cannot be used during recommendation process. However, this case is rare and estimation is produced for 95% of the user item-pairs.

3.2.7 Propagation of Trust and Distrust [34]

3.2.7.1 Proposed Technique

People believe statements from a trusted acquaintance rather than from a stranger. This belief is called as trust on web applications and trust metric usage provides possible replacement or improvement on existing collaborative filtering based approaches. In a real world application, a user can not state his trust to all remaining users. User can only select a set of users that he trust/distrust among users that he has already encountered (i.e. according to their reviews, comments, ratings etc.). A trusted acquaintance will also trust beliefs of his friends; thus, trust propagation is a possible extrapolation of existing trust values to whole social network.

[34] is the first attempt incorporating distrust in a computational trust propagation. Beforehand, many approaches dealing with trust and trust propagation are proposed in several disciplines, such as decision making in political science [54][55][56], cryptography and authentication in computational security [57][58], marketing in business management [59] and in e-commerce applications [60].

[34] proposes a method of trust and distrust propagation by using predefined set of matrices. Table 2 contains descriptions of matrices that are used in different phases of proposed trust propagation algorithm.

Table 2 – Predefined Matrices

Matrix Name	Meaning
T	Trust matrix - T_{ij} is i's trust of j
D	Distrust matrix - D_{ij} is i's distrust of j
B	Beliefs - total trust belief inferred from T and D (generally either T or T-D)
$\mathbf{C_{B,\alpha}}$	Combined atomic propagation matrix
$\mathbf{P^{(k)}}$	k-step propagation matrix
F	Final beliefs - F_{ij} is i's trust of j

Even both trust and distrust relationships among users are not gathered in most of the systems, initially, the input of the algorithm is assumed to be trust and distrust matrices. According to the proposed algorithm, propagation operations that can be put into words are converted into matrix operations and their meanings are described below. Each propagation operation corresponds to a matrix operator and applying (i.e. Cartesian product operation) the related operator to B matrix results a single step atomic propagation.

According to Table 3, there are four basic atomic propagation methods represented by matrix operators. By weighting each propagation method, a combined atomic propagation matrix, $C_{B,\alpha}$, can be obtained, $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$. Entry (i,j) of matrix $C_{B,\alpha}$

represents trust/distrust flow from user i to user j through a single atomic propagation step.

$$C_{B,\alpha} = \alpha_1 B + \alpha_2 B^T B + \alpha_3 B^T + \alpha_4 BB^T$$

(EQUATION 3.18)

Table 3 – Predefined Atomic Propagation Operators

Atomic Propagation	Operator	Description
Direct propagation	**B**	If a trusts b and b trusts c, then a should trust c
Co-citation	$\mathbf{B^T B}$	If a trusts c and d, and b also trusts c, b should trust d
Transpose trust	$\mathbf{B^T}$	If a trusts b then trusting b should imply trusting a
Trust coupling	$\mathbf{BB^T}$	a, b trust c, so trusting a should imply trusting b

All operator applications are on B and this matrix should be initially constructed from matrices T and D. There are three proposed methods for belief matrix construction and iterative propagation of trust and distrust:

- **Trust only:** Distrust data is completely ignored.

$$B = T \qquad P^{(k)} = C_{B,\alpha}$$

(EQUATION 3.19)

- **One-step Distrust:** Distrust propagation is included for a single step, while trust propagation is included repeatedly.

$$B = T \qquad P^{(k)} = C_{B,\alpha}(T - D)$$

(EQUATION 3.20)

- **Propagated Distrust:** Both trust and distrust propagation is included repeatedly.

$$B = T - D \qquad P^{(k)} = C_{B,\alpha}$$ **(EQUATION 3.21)**

Given $P^{(k)}$ matrices represent k step atomic propagation matrix and as previously mentioned, $C_{B,\alpha}$ is constructed with a weighting scheme adaptation on atomic propagation operators. Moreover, for smaller values of k, $P^{(k)}$ may be more reliable, since smaller number of propagation operators will be applied and this equals to only use of closer users in social network. For this reason, each propagation step may not be equally weighted in a real world application. With this knowledge, weight of each propagation step can differ and this fact results two different types of final belief matrix construction:

- **Eigen-value Propagation:** Let K is a suitably chosen integer. This model uses equally weighted K step propagation.

$$F = P^{(K)}$$ **(EQUATION 3.22)**

- **Weighted Linear Combinations:** Let γ is a discount factor constant (that is smaller than the largest eigen-value of $C_{B,\ \alpha}$) and K is a suitably chosen integer. This model uses a weighted K step propagation and penalizes lengthy propagation steps.

$$F = \sum_{k=1}^{K} \gamma^k P^{(k)}$$ **(EQUATION 3.23)**

Up to now, final belief matrix is constructed by the application of propagations. Entries in F can also include real numbers. When discrete values are used to represent trust and distrust metrics, one can round real valued entries of F to discrete values in valid range of trust and distrust values. For this reason, different rounding techniques are generated, however, algorithm given in this section, has the ability to still work with real numbers.

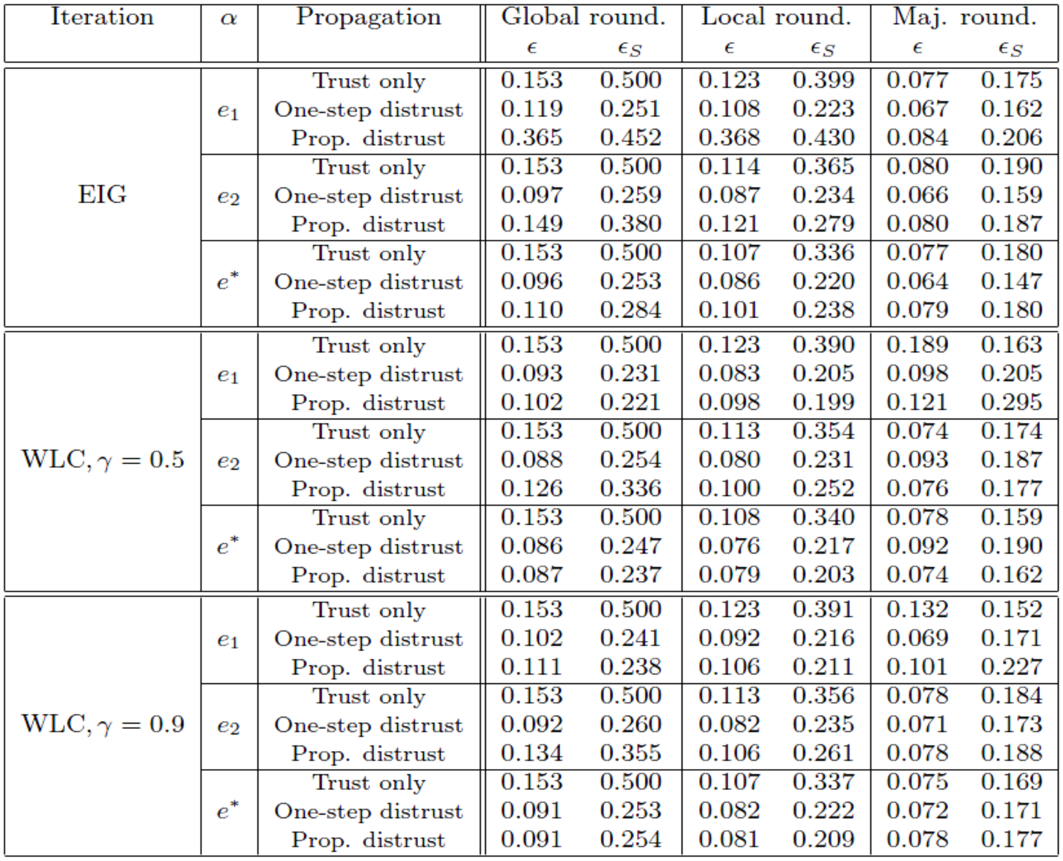
Rounding is performed in order to get a yes/no answer to question: “Should user i trust j?” Here are three techniques:

- **Global Rounding:** This technique tries to align the ratio of trust to distrust in F as in overall initial input. We assume that i trusts j if and only if F_{ij} is within the top μ fraction of entries of the vector Fi, under the standard < ordering. μ is chosen based on the relative fractions of trust and distrust in initial input.
- **Local Rounding:** This technique tries to align the ratio of trust to distrust in initial input for each user. We assume that i trusts j if and only if F_{ij} is within the top μ fraction of entries of the vector F_i under the standard < ordering. μ is chosen based on the relative fractions of trust and distrust judgments made by i.
- **Majority Rounding:** This technique predicts label of j to be that of the majority of the labels in the smallest local neighborhood surrounding it where the majority is well defined. We assume that i trusts j if and only if the label of the majority in the smallest local neighbors of i surrounding j is trust.

3.2.7.2 Evaluation

Cross validation method for evaluation of proposed method similar to leave-one-out technique is used in this approach. In this technique, a single trial edge (i,j) is masked from the graph and trust/distrust value for this edge is tried to be estimated. Moreover, overall evaluation results are presented in Table 4. In this table, evaluation results for each different method constructed by incorporation of various approaches for atomic propagation, iteration technique, rounding which are discussed in the previous section, are presented. By this way, all combinations of partial techniques can be observed.

Table 4 – Prediction errors of various algorithms, where e* = (0.4,0.4,0.1,0.1), K=20



Iteration	α	Propagation	Global round.		Local round.		Maj. round.	
			ϵ	ϵ_S	ϵ	ϵ_S	ϵ	ϵ_S
EIG	e_1	Trust only	0.153	0.500	0.123	0.399	0.077	0.175
		One-step distrust	0.119	0.251	0.108	0.223	0.067	0.162
		Prop. distrust	0.365	0.452	0.368	0.430	0.084	0.206
	e_2	Trust only	0.153	0.500	0.114	0.365	0.080	0.190
		One-step distrust	0.097	0.259	0.087	0.234	0.066	0.159
		Prop. distrust	0.149	0.380	0.121	0.279	0.080	0.187
	e^*	Trust only	0.153	0.500	0.107	0.336	0.077	0.180
		One-step distrust	0.096	0.253	0.086	0.220	0.064	0.147
		Prop. distrust	0.110	0.284	0.101	0.238	0.079	0.180
WLC, $\gamma = 0.5$	e_1	Trust only	0.153	0.500	0.123	0.390	0.189	0.163
		One-step distrust	0.093	0.231	0.083	0.205	0.098	0.205
		Prop. distrust	0.102	0.221	0.098	0.199	0.121	0.295
	e_2	Trust only	0.153	0.500	0.113	0.354	0.074	0.174
		One-step distrust	0.088	0.254	0.080	0.231	0.093	0.187
		Prop. distrust	0.126	0.336	0.100	0.252	0.076	0.177
	e^*	Trust only	0.153	0.500	0.108	0.340	0.078	0.159
		One-step distrust	0.086	0.247	0.076	0.217	0.092	0.190
		Prop. distrust	0.087	0.237	0.079	0.203	0.074	0.162
WLC, $\gamma = 0.9$	e_1	Trust only	0.153	0.500	0.123	0.391	0.132	0.152
		One-step distrust	0.102	0.241	0.092	0.216	0.069	0.171
		Prop. distrust	0.111	0.238	0.106	0.211	0.101	0.227
	e_2	Trust only	0.153	0.500	0.113	0.356	0.078	0.184
		One-step distrust	0.092	0.260	0.082	0.235	0.071	0.173
		Prop. distrust	0.134	0.355	0.106	0.261	0.078	0.188
	e^*	Trust only	0.153	0.500	0.107	0.337	0.075	0.169
		One-step distrust	0.091	0.253	0.082	0.222	0.072	0.171
		Prop. distrust	0.091	0.254	0.081	0.209	0.078	0.177

- **Basis elements:** At first glance, direct propagation seems the most significant propagation method and the more weight it has, the better the performance would arise. However, when only a single propagation method is used among the methods described in Table 3, the best performer is co-citation. This is quite surprising because it looks obvious that when i trusts j and j trusts k, then this implies i trusts k. Direct propagation could have lower prediction errors than co-citation, however that is not the case in Figure 8. Even if single propagation method application gives quite well results, a weighted combination of atomic propagations performs better.

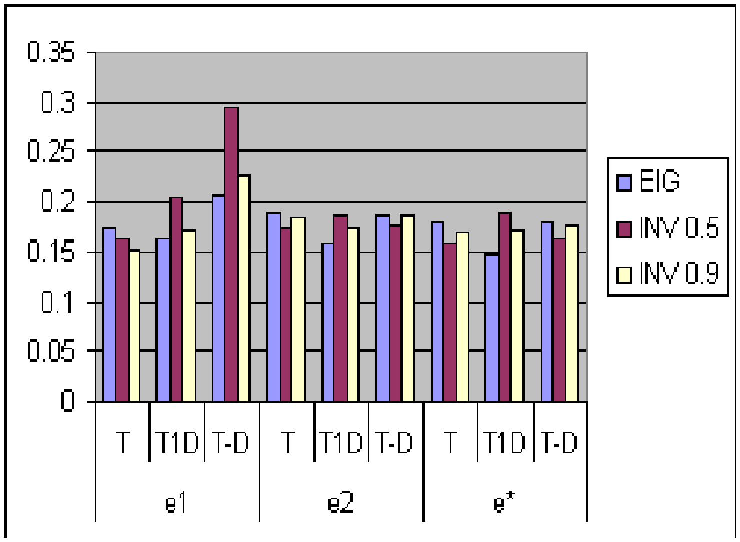


Figure 8 – Results for different values of α, majority rounding, against error rate

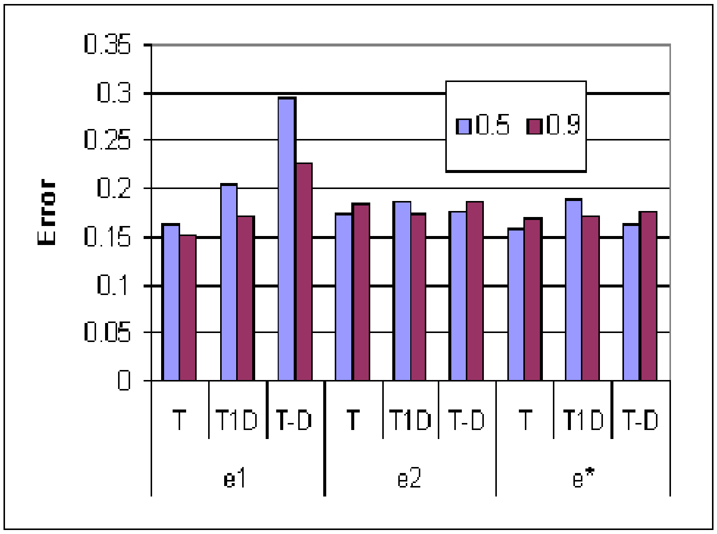


Figure 9 – Results for WLC iteration, γ ϵ {0.5,0.9}, showing iteration methods and basis vectors against error rate

- **Incorporation of Distrust:** According to Table 4, one-step distrust propagation with EIG iteration is the overall best performer. With WLC type of iteration, one-step distrust propagation and propagated distrust performs quite well compared to only trust use as belief matrix. γ=0.9 case which favors long paths, performs worse in one-step distrust propagation compared to γ=0.5. For this reason, it seems more reasonable to weight shorter paths higher than longer ones. Moreover, as seen in Figure 9, distrust propagation hurts the performance when only direct atomic propagation is used. The reason can be explained with multiplicative distrust propagation. Multiplicative distrust propagation method causes distrust values to be meaningless when propagation of distrust is used in further steps rather than a single step.

- **Rounding:** Figure 10 illustrates best scores for EIG and WLC iteration techniques with different rounding strategies. In all cases, order of performances is same for rounding methods and majority rounding beats local rounding which in turn beats global rounding. In Figure 10, set S stands for a data set sampling 50% trust edges and 50% distrust edges. Additionally, as it can be understood from its name, All Trials stands for the complete data set.

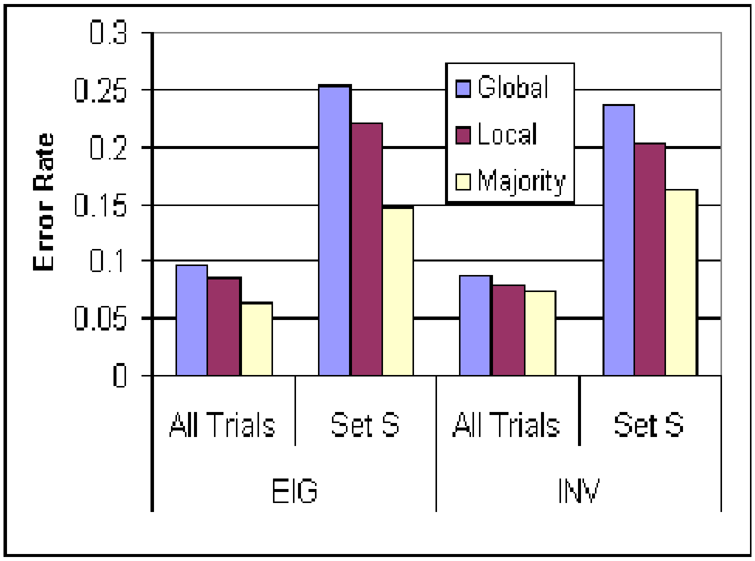


Figure 10 – Results for rounding using the best overall settings for the EIG and the WLC iteration against error rate

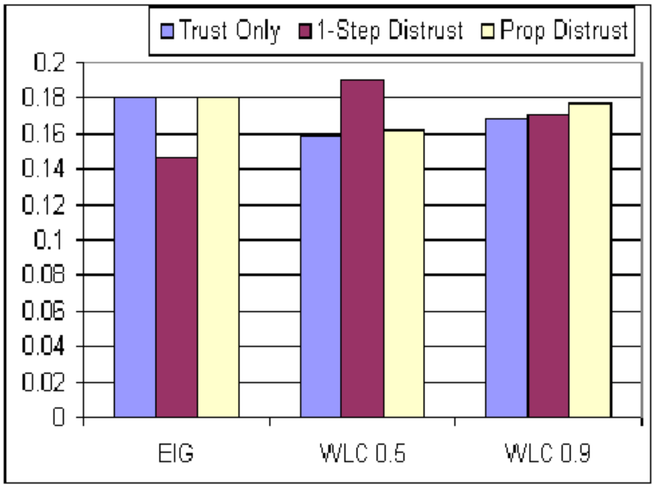
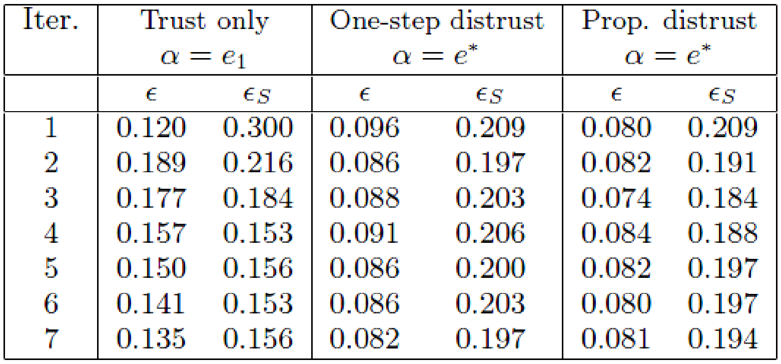


Figure 11 – Results for all iteration methods with α=e*, majority rounding, against error rate

- **Iteration Models:** Figure 11 restricts results to overall best performer rounding method (i.e. majority rounding) and best basis element vector (i.e. α=e*) and compares results of EIG, WLC with γ=0.5 and WLC with γ=0.9.

Table 5 – Effect of number of iterations



Iter.	Trust only $\alpha = e_1$		One-step distrust $\alpha = e^*$		Prop. distrust $\alpha = e^*$	
	ϵ	ϵ_S	ϵ	ϵ_S	ϵ	ϵ_S
1	0.120	0.300	0.096	0.209	0.080	0.209
2	0.189	0.216	0.086	0.197	0.082	0.191
3	0.177	0.184	0.088	0.203	0.074	0.184
4	0.157	0.153	0.091	0.206	0.084	0.188
5	0.150	0.156	0.086	0.200	0.082	0.197
6	0.141	0.153	0.086	0.203	0.080	0.197
7	0.135	0.156	0.082	0.197	0.081	0.194

- **Iterations K:** Table 5 contains error rates. According to this table, increasing number of iterations in $\alpha=e_1$ case has more dramatic effect compared to other propagation methods. This effect is expected because of direct propagation moves along the directed path in trust network. However, it is interesting not to see this improvement with similar weight on $\alpha=e^*$ case. The reason can be assumed to be the smaller weight of direct propagation in $\alpha=e^*$ case, where for $\alpha=e_1$ case it solely contains direct atomic propagation.

Already discussed algorithms in this section have a significant deficiency that is worth considering. Propagation of distrust may cause poor performance because of possible transitions that will result meaningless statements. There are two options for direct atomic propagation of distrust. One is if i distrusts j and j distrusts k, then i should trust k (i.e. the enemy of your enemy is your friend). The other is if i distrusts j and j distrusts k, then i should strongly distrust k (i.e. don't respect someone not respected by someone you don't respect). The former notion is multiplicative and the latter is additive distrust propagation. The virtue of matrix multiplication implements the multiplicative notion. However, there is a problem about this approach because it can cause some unexpected statements. For instance, if a directed cycle including a set of users exists with trust/distrust edges between them, then a negative product will lead a user to distrust himself, where this case shall never occur.

Even if some problematic fragments exist in proposed algorithms, propagation of trust and distrust increases the performance of a recommender system in certain circumstances. Results of this study can orient people in the area to use trust/distrust metric and also can lead merchants to add trust/distrust interfaces for their users. This will provide a better learning of user preferences and items that are mostly suitable for user preferences will be presented to him.

3.2.8 Trust-aware Collaborative Filtering for Recommender Systems [36]

3.2.8.1 Proposed Technique

Because of the deficiencies of collaborative filtering approach described in 2.2.5, a trust network where users rate each other can be constructed. Users highly rate another user if reviews and ratings of the user are found to be valuable [61]. Google's PageRank algorithm [62] is one of the powering algorithms of google.com and it tries to infer the authority of a single page on the whole network of web pages through a global trust metric. In fact, PageRank algorithm uses a network of web pages and relates web pages with each other. If a web page has an out link to another, it represents a positive rating from from-page to to-page of the link in web page networks and this concept is similar to trust statements among users in a trust network. In addition to PageRank algorithm, [61] and [4] has been actively using trust statements and has already created their social networks. Moreover, peer to peer networks and open marketplaces are also moving in this direction.

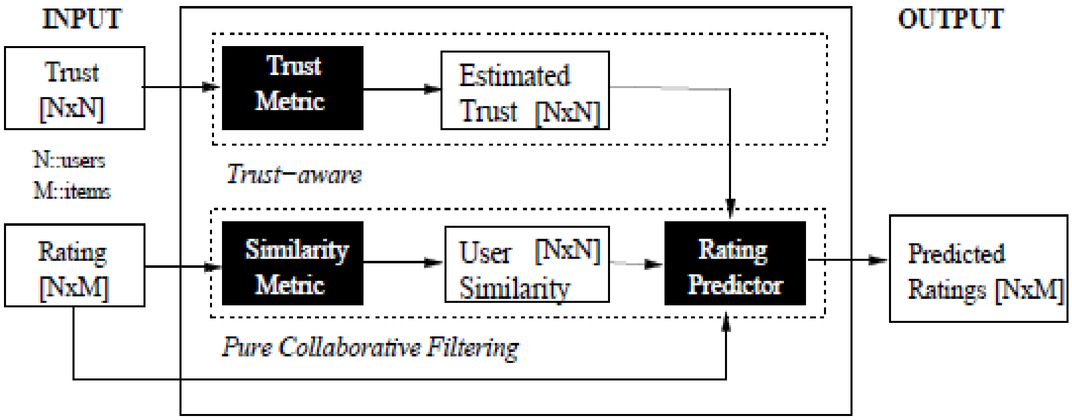


Figure 12 – Trust-aware Recommender System Architecture

Figure 12 illustrates the overall architecture used in [36]. Each represented module is designed to be replaceable with another module providing that input and output specification is retained. Each module shown in Figure 12 can be described in a more detailed way.

- **Trust Metric Module:** Trust metric module gets trust network as its input and outputs an estimated trust matrix by exploiting a trust propagation algorithm. The output matrix contains a value t in entry (i,j) representing how much user i should trust user j. If global trust is used, all rows of the estimated trust matrix contain a single value, since a user's reputation is defined in whole trust network instead of between any two users.

  Epinions.com contains only full positive trust instead of weighted trust statements and trusts propagation strategy of [36] is proposed accordingly. Hereunder, trust propagation is calculated for any (source user, target user) pair. Propagated trust at level n is $(d-n+1)/d$, where n is the distance between source user and target user and d is the maximum propagation distance that is the maximum path distance of target user from the source user.

- **Similarity Metric Module:** This module gets user rating matrix as its input and outputs a similarity matrix containing the entry similarity values or correlation between two users as its entries. For instance if user i and user j is correlated with a value 0.4, then the entry (i,j) of similarity matrix will contain a value 0.4. The correlation between two users is calculated through Pearson Correlation Coefficient and considers only overlapping items that are rated by both users. Similarity calculation between two users having only a single commonly rated item will not be meaningful, for this reason, user pairs that have commonly rated at least two items are taken into account. Moreover, negative similarities are discarded because of their mass and their misleading values. In short, this module performs the similarity calculation task of a generic collaborative filtering system with its little variations.

- **Rating Predictor:** This module gets user similarity and estimated trust matrices and outputs a predicted ratings matrix that contains values of estimated ratings for each user item pair. Rating predictor can predict output matrix entries by either using a single matrix or a combination of them. Since both of input matrices are very sparse, it seems reasonable to combine them. However, [36] does not explicitly state a merge algorithm, it leaves this study to the user, because its aim is to present how incorporation of trust can improve the performance of a recommender system.

3.2.8.2 Evaluation

[36] uses leave-one-out technique to estimate overall system performance as in most of the systems already described in this thesis work. By hiding a single rating value at a time, the system tries to predict it and differences between real rating values and the estimated ones becomes the error rate of proposed technique. In this work, Massa and Avesani has used Mean Absolute Error (MAE) which can be described as an error calculation strategy by averaging absolute differences between real ratings and estimated ratings.

Another important metric for performance evaluation is coverage. A system may perform well according to error rate, however, if only small number of ratings can be predicted, it will not be so reasonable to use this system. Rating coverage stands for the portion of ratings that a recommender system is able to predict. Still, rating coverage is not informative about the quality of a recommender system. For instance, a system can contain a set of users: one user is a heavy rater and has 100 ratings; where remaining 100 users have only a single rating. This kind of rating distribution may cause the system to predict only ratings of the heavy rater and none of the ratings for remaining users. Rating coverage of such a recommender system is 50%, since 100 out of 200 ratings can be predicted. However, most of the users can not benefit such a system. Thus, a new metric, user coverage, is proposed [36]. User coverage can be obtained by calculating the portion of users for which the recommender system can make at least one prediction.

As a complement of mean absolute error, mean absolute user error (MAUE) is proposed. This error function is calculated by averaging MAEs for each user. By this way, each user is taken into account once and a cold start user is influential as much as a heavy rater. The reason behind this strategy is that each user is equal to benefit a recommender system.

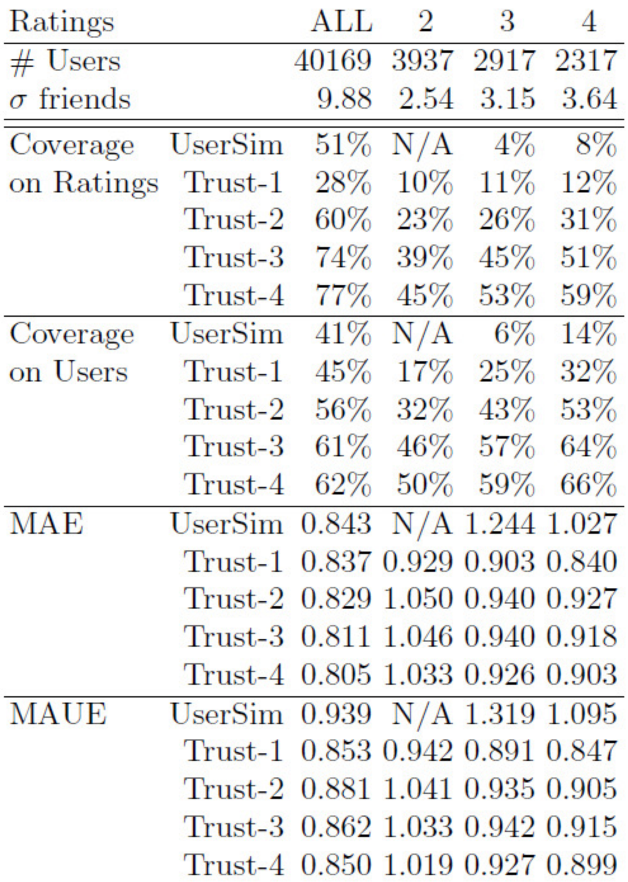
Table 6 illustrates overall results obtained from the evaluation. Rows represent different evaluation measures that have been already discussed in previous paragraphs of this section. At top of the table, number of users in each data set and average friend count of users that are considered in the corresponding data set, are listed. UserSim stands for a technique that only uses similarity metric module for evaluation. On the other hand, Trust-x stands for a technique that only uses trust metric module with a maximum propagation distance x.

It should be remembered that leave-one-out technique hides one rating but this does not affect the number of trust statements. That is why; users who expressed 4 ratings shall be compared with users who expressed almost 3 trust statements (friends).

According to Table 6, UserSim tends to perform well with users who have rated many items and poorly with users who have rated few items. This fact corresponds to sparsity problem of collaborative filtering. If a user has rated many items, it is more probable to find similar users since commonly rated item count will increase between user under consideration and others. In addition to that, for cold start users MAE and MAUE are also higher.

For every Trust-x evaluation, MAE and MAUE are close meaning that Trust-x methods have similar performance for every type of users. Even if Trust based recommendation has a better performance compared to similarity based recommendation based on MAE and MAUE, the extent of improvement is not a big deal. Though even a little improvement is significant, because trust based recommendation has also lower computational complexity and this will also be a benefit of trust in recommendation process. On the other hand, coverage is significantly improved by trust-aware systems. For instance, coverage for users who have only rated 3 items is almost 50% while the error rate is kept low.

Table 6 – Experiment Results



Ratings		ALL	2	3	4
# Users		40169	3937	2917	2317
σ friends		9.88	2.54	3.15	3.64
Coverage on Ratings	UserSim	51%	N/A	4%	8%
	Trust-1	28%	10%	11%	12%
	Trust-2	60%	23%	26%	31%
	Trust-3	74%	39%	45%	51%
	Trust-4	77%	45%	53%	59%
Coverage on Users	UserSim	41%	N/A	6%	14%
	Trust-1	45%	17%	25%	32%
	Trust-2	56%	32%	43%	53%
	Trust-3	61%	46%	57%	64%
	Trust-4	62%	50%	59%	66%
MAE	UserSim	0.843	N/A	1.244	1.027
	Trust-1	0.837	0.929	0.903	0.840
	Trust-2	0.829	1.050	0.940	0.927
	Trust-3	0.811	1.046	0.940	0.918
	Trust-4	0.805	1.033	0.926	0.903
MAUE	UserSim	0.939	N/A	1.319	1.095
	Trust-1	0.853	0.942	0.891	0.847
	Trust-2	0.881	1.041	0.935	0.905
	Trust-3	0.862	1.033	0.942	0.915
	Trust-4	0.850	1.019	0.927	0.899

For each proposed method, both rating coverage and user coverage increase as number of user ratings increases. In addition to that, as maximum trust propagation distance increases, both coverage results are better; however, error rates are fluctuating. This fact is because of the reduction of accuracy, as users far away from the active user is started to be included in recommendation process.

In conclusion, trust metric usage increases coverage of ratings and users significantly while keeping the error at least as low as similarity metric usage. The greatest the maximum trust propagation is, the highest the coverage is. For new users with few ratings and few trust statements, incorporation of trust metric module into recommendation process both improves error rate and coverage. However, when a user has not specified any trust statement, no item can be recommended by only usage of trust metric module. An algorithm combining both approaches can perform better and can take advantage of both methods.

3.3 **Discussion**

In this chapter, an analysis of trust data and its use in state of the art solutions are presented. In the former section, trust data use in recommender systems and a generally accepted classification of trust data are discussed. Advantages and disadvantages of each trust metric are also evaluated in order to keep them in mind during evaluation of our proposed method.

In the later section, related studies benefitted in the scope of this thesis work are investigated. Both proposed methods and their evaluation techniques are introduced in a detailed manner. Among methods discussed in this section, MovieReco [29] has tried to determine rating styles of users and it uses similar users and movies during recommendation process. Trust does not exist in MovieReco system; however, it is beneficial to see how a system can extract user rating styles.

Cultural metadata use in recommendation is a kind of content based recommendation [26], where content data quality in the overall system is considered. Common content data gets a lower quality while rare ones have high quality during similarity calculation between profiles. Additionally, data mining on text-based comments on the web can be mapped to word of mouth data in our daily life. Text-based data extraction provides a transformation of temporal parols into an input for recommendation.

In 3.2.3, a discussion on trust data in recommendation system is represented. This discussion reveals a concept of social network in recommenders. As a start point of trust-based systems, 3.2.4 presents a trust-based solution for a dataset without trust information. It is interesting to relate users with each other through trust metric, while no explicit data exists. [24] tries to find out a social network through already given ratings. Even trust data calculation through existing ratings provides better performance compared to generic collaborative filtering. Moreover, both profile and item level trusts are discussed in 3.2.4.

Moleskiing and FilmTrust systems are similar in their application while they are totally used on quite different domains. In Moleskiing system, a combination of trust-based filtering and trust-based weighting is used which are already discussed in 3.2.4. Both methods make use of different trust propagation techniques to discover invisible trust data on overall social network and use newly found trusts in application.

Section 3.2.7 points the use of distrust. Trust and distrust integration in application is discussed and matrix operations are used to propagate trust/distrust scores. According to this approach, a single matrix operation stands for a single propagation through an atomic operation while a sequential application of matrix operations corresponds to a total propagation technique. Various atomic operators, propagation schemes and rounding techniques are incorporated components of this system. In the evaluation part, all proposed combinations of each step are considered and an overall evaluation result table is produced.

In 3.2.8, a trust-aware system incorporating two separate components based on similarity and trust scores is presented. This system manages both components separately and compares evaluation results of the trust-based component and similarity-based component. 3.2.8 also proposes an aspect of user coverage and user error rates as extensions to rating coverage and overall system error rates, respectively. User based coverage and error rate calculation exposes the variation of coverage and error rates for each user, while overall coverage and error rate calculation lose details of analysis in the averaging process.

In conclusion, this chapter uncovers state of the art solution details and their evaluation techniques. Proposed methods in this section form the base of the proposed method in the next chapter. Most of the approaches discussed in this section explicitly used as components of the method proposed in the scope of this thesis work.

CHAPTER 4

THE PROPOSED SYSTEM

4.1 Dataset Overview: Epinions.com [61]

Epinions.com is a consumer opinion web site where people can express their opinions and ratings on wide range of items such as cars, computer software, travel, home and garden instruments etc. Each user is a reviewer and can write his reviews on any topic. Mostly, reviews include experiences or opinions of users on a product which may be beneficial to any other user wondering comments related to the product. Each review is attached a rating representing overall impression of the product to the reviewer.

Epinions users can also have an ability to express their trust and distrust opinions in other users. Each user constructs a web of trust with his trust expressions in the system. In Epinions terminology, web of trust for each user is a network of reviewers whose reviews and ratings have been consistently found valuable. Trust utilization is so important in Epinions, since reviews written about a product are listed in an order determined by already stated trust expressions. Reviews of most trustworthy users will be presented to the active user with a better ranking. Surely, each user will have distinct users in his web of trust (i.e. local trust) and this will provide a personalized recommendation. Instead of FilmTrust discussed in 3.2.6, Epinions shows users both their web of trust and who have already trusted them.

Moreover, each trust expression also affects users who are not actively in trust relationship. Users inherit some of the effects of their web of trust, because of trust propagation. User trust expressions also affect overall trust scores (i.e. global trust) stored for each user and this contributes to advisor selection processing in Epinions. Users are also categorized according to their global trust scores. Some set of best users are identified periodically, such as the following:

- **Category Leads:** These users are chosen for each category and they are the main contact point between Epinions and the community.
- **Top Reviewers:** Users who have the highest ratings among the community. Reviews are rated by advisors and rest of the community. It can be assumed that top reviewers are users with high quality reviews on specific category.
- **Advisors:** This set of users provides constructive feedback to rest of the community such as the used method in ordering reviews and try to improve content quality in Epinions.
- **Most Popular Reviewers:** This user list is compiled on a monthly basis for each category according to total number of visits to member reviews.
- **Most Popular Authors:** This user list is compiled on a monthly basis for each category according to user review hits.
- **Top Earning Reviewers:** This set of users is reviewers whose reviews are gained much value compared to rest of the reviews in Epinions.

Additionally, Epinions let people indicate their distrust. As a complement of trust, users can add a list of users, whose reviews or ratings are found offensive, inaccurate or low quality, to their block-list. By expressing distrust, users can create their block-list and make distrusted people contributions less likely during recommendation process. Distrust information is only presented to the rater in order to keep rated users objective against people who distrust them.

Both trust and distrust can be expressed only with a single degree. A user trust/distrust another user or not. For this reason, already discussed trust propagation algorithms, such as TidalTrust, can produce less accurate results. At any time, a user can change his mind and can express his trust to an already distrusted user or vice versa.

In the scope of this thesis work, Epinions.com site is crawled. As mentioned in 0, most of the existing recommendation systems let people extract a restricted set of data and do not disclose any data that is considered as confidential. One type of data considered as confidential by Epinions site is block lists and this data cannot be crawled because retaliation and hard feelings between users may come on the scene if this data becomes transparent for everyone.

Data extracted by the crawler contains 26522 users and 23260 products as of December 2009. Only movie products and their related ratings are collected by the crawler, because this work will not deal with different product categories. Originally, this dataset is a snapshot for this thesis work and whole dataset contains 92205 users and 27903 products with 169252 ratings at the end of data extraction. However, amount of users and products seems high enough to observe general dataset characteristics and to investigate advantages and disadvantages of newly proposed approaches. Dataset also incorporates 76902 product ratings and 255160 trust statements. Hence, the rating sparseness of the collected dataset is more than 99.99%. Furthermore, dataset trust statements are sparser than 99.97%. This is so because no peer reasonably experience and rate all existing items or no user can make a trust/distrust choice for each other user in Epinions. The crucial question that shall be asked is “how a good recommendation can be performed when less than 0.01% ratings and less than 0.03 trust statements are available?”

One remarkable property reveals difficulty of a successful recommendation is trust and rating data count provided by each user. In the extracted dataset, 22525 users (about 85% of total user count) have no other users in their web of trust. Moreover, 87% of total users have at most four users in their web of trust. One interesting point that shall be noted is the ratio of people who are not in any other user’s web of trust. Only 2% of all users are not in any other user’s web of trust. This is indeed surprising because most probably a randomly selected user will have no user in his web of trust, but he will be in web of trust of another user. This fact emerges because of the existence of active users in the system. An active user is a user who rate many items (i.e. at least more than two) and specify many (i.e. at least more than two) trust statements. When an active user A trusts another user B, this does not change the number of users who has no users in his web of trust, because user A has already users in his web of trust and his trust specification affects only his web of trust. On the other hand, trust rating specified by user A for user B increases the number of users who are in any other user's web of trust if user B is not found trustworthy (i.e. user B is not included any user's web of trust) before.

On the other hand, 90% of users are not provided even a single rating for any item and almost 95% users only have at most four rated items. These statistical data uncovers the difficulty of recommendation. Nevertheless, amount of trust data seems denser compared to ratings. Additionally, techniques like trust propagation can extract more trust relationships among users, where that is not the case in ratings (discussed in 2.2.5.2.4). That is the reason, why it is reasonable to test whether trust data gains better accuracy or not.

User review count in the dataset is 77006. These reviews are rated 2761716 times and that is the review rating count. A review rating is given by a user to another user's review. Actually, reviews and review ratings are not used in the scope of this study; however, they can be used in future works for a further improvement. Occurrence rates of each data type are significant when determining whether its use is reasonable or not. Moreover, at first glance, review ratings may be considered as implicit indicators of trust statements. In addition to explicit trust statements, review rating use may normalize Boolean trust scores by introducing a degree of trust as in [11][24][33] where a degree of trust exists for each trust statement.

4.2 Formal Domain Definition

A detailed domain definition is useful to depict existing data in an organized way. Dataset formalization and domain definition shall not be considered separately from the definition in 2.2.1; however this definition is a complement of previous definition.

The crawled dataset can be mapped to an environment similar to one in [36]:

- A set *P* of n uniquely identifiable peers.

  $P = \{p_1, p_2, p_3, \ldots, p_n\}$

  Each peer corresponds to a single user in Epinions. Additionally, the term "peer" stands for independent entities performing some actions in a system, such as nodes in peer-to-peer systems, software agents or intelligent web servers in other domains.

- A set *I* of m uniquely identifiable items.

  $I = \{i_1, i_2, i_3, \ldots, i_m\}$

  Each item is identified separately and in our case, each item stands for a movie. In fact, each item may be a movie, book, car etc. in Epinions, since each of these items exists in the Epinions system.

- A set R of k uniquely identifiable reviews.

  $R = \{r_1, r_2, r_3, \ldots, r_k\}$

  Each user can express his reviews related to an item. Each review contains opinions of the reviewer about the item through text-based information. Reviews are approved by the Epinions administrators before they are published in the system so that reviews with impermissible content are not displayed to peers. Multiple reviews can be entered to the system for same user-item pair.

- *n* sets of trust statements. Each peer has a set of trust statements representing his trust for any other user. Here is the set of trust statements for peer p_i.

  $t_{p_i} : P \rightarrow \{0,1\} \cup \perp$

  Each peer can trust (i.e. trust rating 1) or distrust (i.e. distrust rating 0) another user. Moreover, no trust rating may exist between a pair of peers. For instance, $t_{p_1}(p_2) = 1$ indicates p_1 trusts p_2.

- *n* sets of item ratings. Each peer can express a set of ratings for items in the dataset. In Epinions, a peer can give a rating within a range [1,5], where 1 is the lowest rating score indicating totally disliked item and 5 is the highest one which indicates maximum appreciation.

  $pr_{p_i} : I \rightarrow \{1,2,3,4,5\} \cup \perp$

- *n* sets of review ratings. These ratings are provided by peers to reviews of other peers. In fact, review ratings can be used as trust statement amplifiers between peers. It seems reasonable to disclose the opinion, "The higher review ratings peer A gives for peer B, the more A trusts B".

  $rr_{p_i} : R \rightarrow \{1,2,3,4,5\} \cup \perp$

4.3 **Base Methods**

Methods that are proposed in this thesis work aims to combine different approaches such as content based and collaborative recommender systems previously discussed as standalone solutions in Section 3.2.

The main idea behind proposed methods is searching an answer to the question: "Does trust information totally hold relationship between two users?" That stands for whether trust relationships comprise profile similarities between users and items in content based approach and user similarities in collaborative filtering approach or not. If trust data covers all relationships among users and items, then there will be no need to use content based and collaborative approaches as separate methods, because only trust degrees between users will provide a good estimation technique. On the other hand, if trust data does not cover all relationships examined in content based and collaborative approaches, we will try to find out in which cases trust data can be used to improve recommendation accuracy. In order to examine characteristics of trust relationship, a set of base methods are selected:

4.3.1 Trusted People Average (TPA)

This method basically estimates an unknown rating for a user-item pair depending on already given ratings to the item by user's friends (i.e. user's web of trust). It simply gets the average of ratings given to the item by users in the active user's friend list. In fact, this method is inherently a weighted average method. However, as mentioned in the previous section, degree of trust statements does not exist in Epinions so that this method converges to a basic average scheme on user friend ratings.

$$c(i) = \frac{\sum_{p \in W(c)} p(i) w(c,p)}{\sum_{p \in W(c)} |w(c,p)|}, \text{ where W(c) is c's web of trust} \qquad \textbf{(EQUATION 4.1)}$$

$$w(c,p) = \begin{cases} 1, \text{if p is in c's web of trust} \\ 0, \text{if p is not in c's web of trust} \end{cases}$$

Additionally, it is important to note that p(i) is the rating provided by user p to item i and c(i) is the estimated rating for user-item pair (c,i).

4.3.2 Trusted People Deviation (TPD)

This method implements a simpler version of Trust Based weighting scheme discussed in 3.2.4.1. Simplicity of this method is in weight calculation. TPD method only assigns a weight 1 for friends of the user, where weight 0 to other users. It will be helpful to see the employed formula to make things clearer:

$$c(i) = \bar{c} + \frac{\sum_{p \in W(c)} (p(i) - \bar{p}) w(c,p)}{\sum_{p \in W(c)} \left| w(c,p) \right|}, \text{ where W(c) is c's web of trust} \qquad \textbf{(EQUATION 4.2)}$$

$$w(c,p) = \begin{cases} 1, \text{if p is in c's web of trust} \\ 0, \text{if p is not in c's web of trust} \end{cases}$$

The distinction between TPA and TPD is incorporation of user rating styles. Different users may have different rating styles. For instance, a user can give a maximum rating 3 out of 5, while another user may give a rating 5 for items even if they are not the best movies he has ever seen. TPD weights each user with a different weight which is determined by user rating style, while TPA ignores this difference between users. MovieReco discussed in 3.2.1, is another system that is trying to expose rating styles of users, however, it utilizes an interactive solution which asks user to rate the best three movies he has ever seen.

4.3.3 Backward Trusted People Average (BTPA)

This method averages ratings given for item in consideration by users whose friend list contains the active user. This method is similar to TPA but only user set used in evaluation is different. The idea behind this approach is inverted version of idea behind traditional trust use. If user A trusts user B, then preferences of user B can be similar to preferences of user A. Rating prediction for user *c* and item *i* is performed by the following formula:

$$c(i) = \frac{\sum_{c \in W(p)} p(i) w(c,p)}{\sum_{c \in W(p)} |w(c,p)|}, \text{ where W(p) is p's web of trust} \qquad \textbf{(EQUATION 4.3)}$$

$$w(c,p) = \begin{cases} \text{1, if c is in p's web of trust} \\ \text{0, if c is not in p's web of trust} \end{cases}$$

4.3.4 Second Level Trusted People Average (2ndLevelTPA)

This base method is again similar to TPA and BTPA methods, however only second level friends of the current user are used in prediction process as presented in the formula below:

$$c(i) = \frac{\sum_{p \in W^2(c)} p(i) w(c,p)}{\sum_{p \in W^2(c)} |w(c,p)|},$$

(EQUATION 4.4)

$$\text{where } W^2\text{(c) is c's friends of friends not in c's web of trust}$$

$$w(c,p) = \begin{cases} \text{1, if p is in web of trust of a user who is in c's web of trust} \\ \text{0, if p is not in web of trust of a user who is in c's web of trust} \end{cases}$$

This method does not include users in web of trust of the active user but it only deals with users who are friends of the active user’s friends. This idea can be stated as “Friend of my friend is also my friend”. If a user is both in web of trust of the current user and in web of trust of current user’s friend, then this method will not include this user in weighting process, because 2ndLevelTPA solely uses second level trusted people ignoring the users in web of trust.

4.3.5 Cultural Metadata with Genre (CMG)

This base method implements the approach discussed in 3.2.2, where only genre information is used as a keyword-type data. This method can be classified as content based approach. As in generic content based approaches mentioned in 2.2.4.1, user and item profile similarities are calculated to recommend an item to a user. In order to recall the applied procedure, TFIDF method is used to calculate significance of keyword-type data (i.e. genre in this method). The idea behind CMG is "The more frequent a genre type is, the less significant it will be". For instance, in this technique, if movies with action genre are relatively more compared to other genres in the environment, action genre will get a low weight in similarity calculation. Because action movies are common and a high rating provided by a user for an action movie will bring less value compared to a high rating for a documentary movie, which is assumably less frequent.

In implementation part of this method, each movie profile is represented by a vector of weights as $m_p = (TFIDF_{action}, TFIDF_{comedies}, TFIDF_{dramas}, \ldots, TFIDF_{documentary})$ where each dimension of the vector stands for related genre TFIDF scores if movie incorporates the related genre elements. A movie can have multiple genre types in Epinions. In order to illustrate a movie vector for a movie which carries action and comedies genre elements in it, movie vector m_p for such a movie will be constructed as $m_p = (TFIDF_{action}, TFIDF_{comedies}, 0, \ldots, 0)$. Dimensions specified as zero are genres that are not included in the movie, where non-zero entries are included genres.

Additionally, each user profile is stored as a bag (i.e. multiset) of tuples $u_p = ((m_{i1}, r_{i1}), (m_{i2}, r_{i2}), \ldots, (m_{in}, r_{in}))$ where each tuple consists of a movie profile and its attached rating provided by the user. In fact, movie profiles can be same for several tuples as long as it contains same set of genre elements. However, movie profiles consisting same dimensions can be rated with different values.

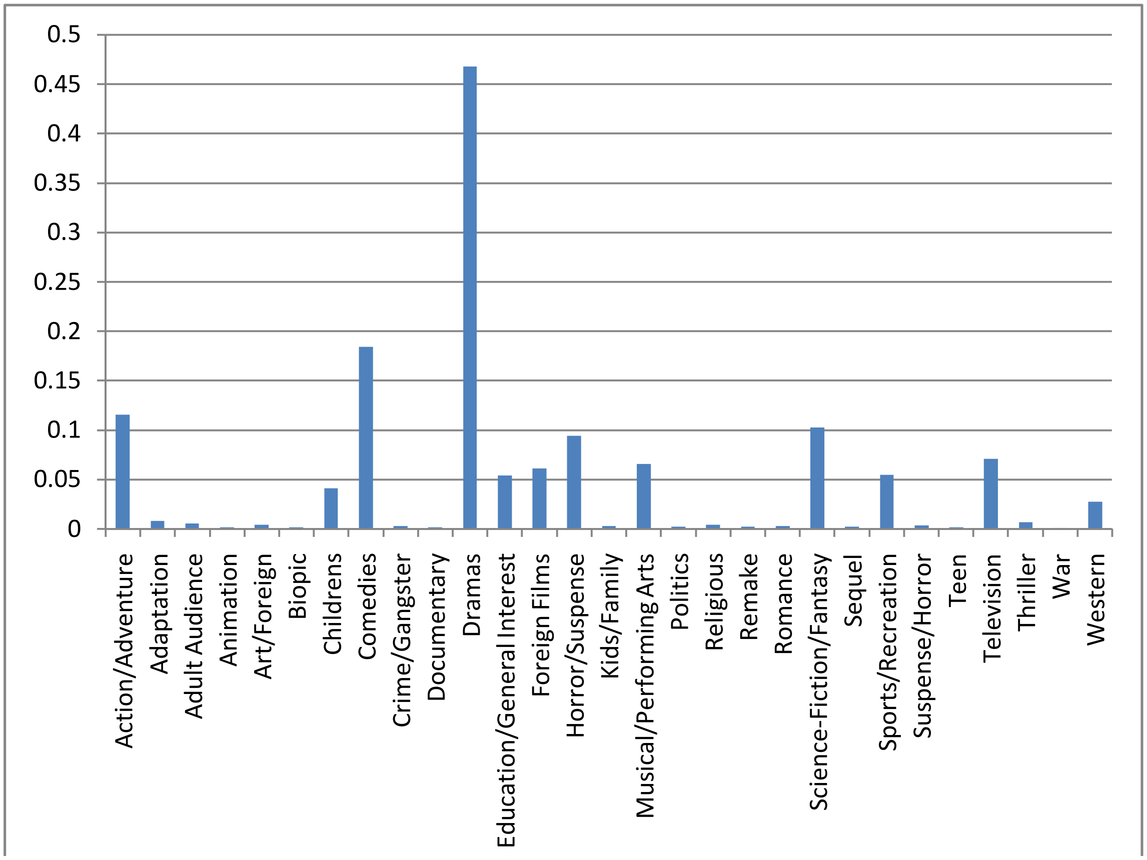


Figure 13 – Genre Frequencies in Epinions

As in generic content based approach, in this method, similarity stands for a user-item profile similarity and cosine similarity is used for this purpose. A common genre will have a lower TFIDF score compared to a less common one. Cosine similarity use on TFIDF scores reveals the property to acquire more similar profiles if profiles agree on a less common genre and less similar profiles if they agree on a common one. This fact is the reason behind the idea of weighting based on TFIDF cosine similarities. Prediction for a user-item pair (u, m) is calculated with a simple weighting scheme as presented below:

$$prediction(u,m) = \frac{\sum_{c \in u} c_r \cos(c_m, m)}{\sum_{c \in u} \cos(c_m, m)} \qquad \textbf{(EQUATION 4.5)}$$

where u and m stand for user and movie profiles, respectively. c_m is movie profile and c_r is rating in selected tuple in user profile multi-set. In this prediction technique, each already rated item by the user is compared to the item in consideration and similarities that are found out at the end of this process act as weights for given ratings to the rated items. Figure 13 illustrates genre distribution in Epinions. A single movie with multiple genres is incremented the frequency of multiple genre types. About 45% of movies are Dramas and similar preference between user and item profiles on Dramas will add less to similarity score. Comedies, Action/Adventure, Science-Fiction/Fantasy movies are still common genres and even they will have a disadvantage on similarity calculation, where War, Biopic, Teen genre type similarities will get a high weight because of their low density in the environment.

4.3.6 Cultural Metadata with Date (CMD)

This method is a counterpart of CMG with a single difference in handling of keyword-type data. This technique uses product release date information of movies as keyword-type data. In this approach, dates of movies are mapped to ten-year period classes, where older films than the lowest end point is mapped to a single class and newer films than the highest end point is mapped to another single class. In order to exemplify this approach, classes used to enumerate product release dates in this study are {*NO_STATEMENT, 1930S_AND_BEFORE, 1940S, 1950S, 1960S, 1970S, 1980S, 1990S, 2000S_AND_AFTER*}. According to this classification, the first movie of "The Godfather" series which is released in 1972, will be mapped to the class *1970S*.

Movie and user profile construction in this approach is just the same approach in CMG. Each movie is modeled with a profile which can be represented as a vector:

$$m_p = (TFIDF_{1930S\_AND\_BEFORE}, TFIDF_{1940S}, TFIDF_{1950S}, \ldots, TFIDF_{2000S\_AND\_AFTER})$$

Each dimension stands for TFIDF score of corresponding date interval. If the movie incorporates a specific date interval in its release date, then this dimension of the film will be non-zero and will contain corresponding TFIDF value. However, if the movie is not produced at a date interval, corresponding dimension score will be just zero. A movie having multiple release date seems meaningless; however Memento is released at 1999-2000 according to Epinions dataset and this movie will have two non-zero dimensions, 1990S and 2000S. Mostly series and DVD packs which are categorized under Movies tag in Epinions have a period of release date instead of a single year.

A user profile u_p is a bag of tuples and represented as $u_p = ((m_{i1}, r_{i1}), (m_{i2}, r_{i2}), \ldots, (m_{in}, r_{in}))$ where each tuple contains a movie profile and its rating given by the user, respectively.

Same similarity calculation method presented in (EQUATION 4.5) is used for CMD, but this time, similarity will be evaluated depending on release dates, because both movie and user profiles are constructed with release date TFIDF scores.

Figure 14 shows the date distribution of Epinions movies. Movies having a release date from 2000 to present constitute a significant part of the system, which is about 40%. Second common period of date release dates is 1990s with 15% intensity in the system and 1960s' movies come third. CMD algorithm rewards movies released in 1940s most and 2000s' movies least in similarity calculation.

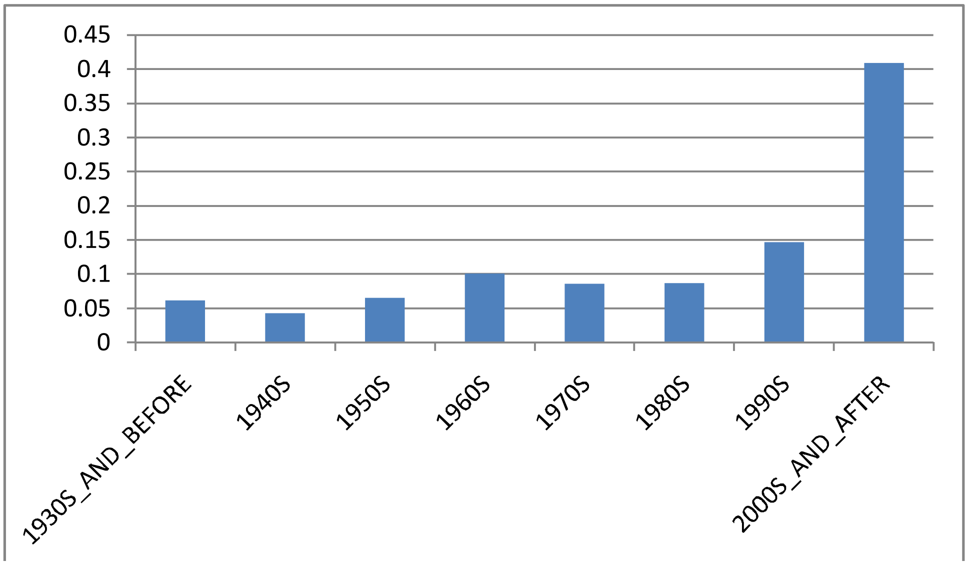


Figure 14 – Date Frequencies in Epinions

4.4 The Proposed Method

Base methods discussed in the previous section form the proposed methods of this study. Methods that will be proposed in this study are combination of base methods. Combination is performed via a simple weighting technique. Weighting approach simply weights estimations of two different approaches as shown in the following:

$$m(i,j) = \frac{w_1 m_1(i,j) + w_2 m_2(i,j)}{w_1 + w_2} \qquad \textbf{(EQUATION 4.6)}$$

where w_1 and w_2 are weights and $w_1 + w_2 = 1$

m(i,j) is the estimated rating for user i and item j pair. Estimated rating is evaluated with a simple weighting over estimated ratings of two different approaches m_1 and m_2. It is clear that when weight w_1 is greater than w_2 estimated rating will be closer to the estimation of m_1 and when w_2 is greater than w_1 estimated rating will be closer to the estimation of m_2. On the other hand, when both methods are equally weighted (i.e.

w_1=0.5 and w_2=0.5), the merged method estimation will be simply the average of both estimation techniques.

The aim is to find out best approach constructed by base methods with our weighting technique. This search discloses significance of each base method in recommendation process and implicitly searches answers to the following and many other questions:

- What is the importance of trust in recommendation process?
- Does standalone trust relationship use in recommendation process cover content based and collaborative based similarity calculation?
- Is trust propagation process needed in recommendation or are explicitly stated trust scores enough to perform a good recommendation?
- How can cold start users benefit recommender systems efficiently?
- Which content information is better to use genre, date or both?
- What are advantages and disadvantages of proposed methods in this work and which base methods shall be utilized for which type of users?

CHAPTER 5

EVALUATION

In the scope of this thesis work, leave-one-out evaluation technique is used to measure the accuracy of each method. According to this technique an existing rating for a single user-item pair is removed and the removed rating score is tried to be estimated by proposed methods. Moreover, the dataset already extracted is divided into eleven randomly generated smaller datasets with a thousand users in each. Each proposed method described and evaluated here has already run over these smaller datasets and presented evaluation results are averages of runs on smaller datasets. A single huge dataset use in evaluation may cause methods to be overstated. However, a random dataset selector provides an analysis of method performances on datasets with different characteristics. For instance, a method may be successful on a dense dataset (i.e. a dataset with less cold start users), however, it may perform worse on a sparse one. Random dataset selection inherently provides to see the performance of each method on many kinds of dataset characteristics and interpreting average scores as overall performance can provide a better evaluation.

For each proposed method, average RMSE score, average ranking, rating coverage and passive user counts are presented with the help of figures. RMSE is a commonly used error calculation technique and is evaluated as the following:

$$RMSE = \sqrt{\frac{\sum (m(i,j) - r_{existing})}{n}} \qquad \textbf{(EQUATION 5.1)}$$

where m(i,j) is predicted rating for user-item pair (i,j) and $r_{existing}$ is existing rating score for user-item pair (i,j). Additionally n stands for total number of ratings considered in numerator.

Average ranking is another criteria used in performance evaluation. Each run on datasets orders proposed method results according to their RMSE scores and average of all rankings gathered in this process is used as performance criteria of approaches. Average RMSE score of method A can be better than another method B; however, B can win most of the competitions (i.e. runs over smaller datasets). For instance, method A can be the winner for each competition except a single one for which it is the last and method B can be the 2nd in all of the competitions. This may provide a better ranking for B rather than method A, even if A is the chiefly winner. Better average ranking points out more stable error rates.

A passive user is a user for whom no prediction can be performed through the applied procedure. Even if a user provide sufficient number of ratings and trust statements in the system, no rating may be predicted for any item depending on the algorithm used during recommendation. Therefore, passive user count is strongly related to the used method. Coverage metrics will be presented in two parts; one part is user coverage [35] which is the percentage of users for whom at least one prediction can be performed and second part is the total rating coverage. Former part, user coverage, will be presented with the number of passive users, where later one will be rating coverage for all users in the system. We have already stated no rating can be estimated for a passive user so that a good recommender is supposed to have fewer passive users and high rating coverage. Since each coverage measurement method evaluates different aspects of a recommendation algorithm, coverage performance shall be analyzed by considering both user and rating coverage at the same time.

RMSE scores are calculated through user-item pairs for which a rating is explicitly stated. However, even a method outperforms many other techniques according to its RMSE scores or rankings, it is still important to cover most of the users and ratings existing in the system. For illustration, a single correctly estimated rating may cause an error rate 0, while coverage is almost 0, too and this will not be a good recommendation.

5.1 Base Method Evaluation

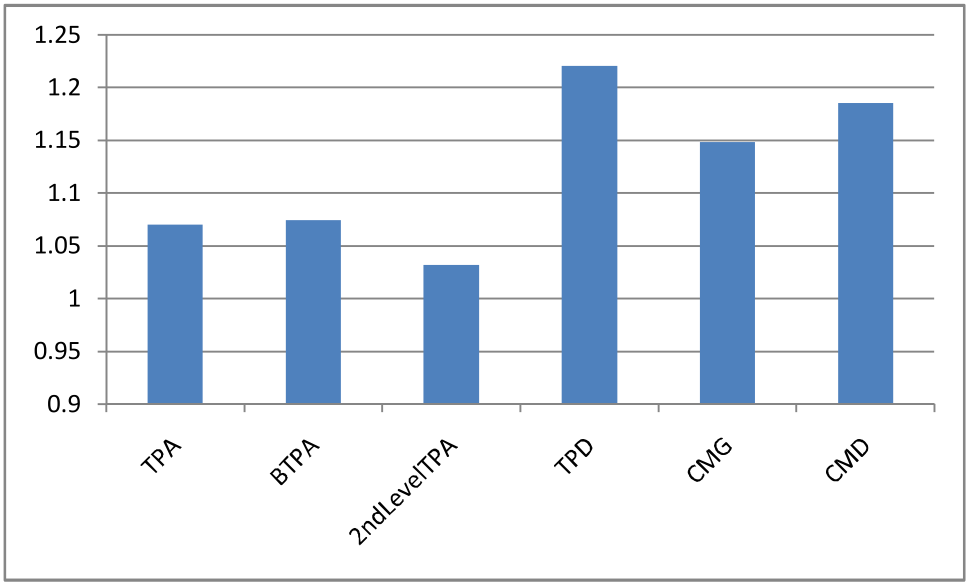


Figure 15 – Average RMSE scores for Base Methods

Figure 15 shows average RMSE scores for each base method described in 4.3. 2nd level TPA seems to perform better compared to other techniques. In fact, reason behind this result may occur because of its trust propagation advantage on other techniques. As mentioned in dataset overview, many people have no user in their web of trust and the fewer users in web of trust; the worse the performance of TPA will emerge. In 2nd level TPA, only people in second trust level will be used in estimation. Even if a user has fewer users in his web of trust, some of his friends may have a large web of trust and this will increase the success of 2nd level TPA. As the number of users included in estimation increase, predictions are expected to be normalized and converge to the optimal rating. TPA and BTPA have almost same average error rates and CMG has a better performance compared to CMD. It is reasonable to see only genre type use in recommendation is more powerful than only use of dates. People mostly prefers movies according to their genres and release dates, however, results reveal that genre has a greater precedence compared release dates.

Additionally, average ranking scores preserve same ordering as presented in Figure 16 except CMG and CMD rankings. In average, CMD has closer ranking to CMG compared to RMSE scores. This fact reveals the importance of release dates and they should be also kept in consideration during recommendation.

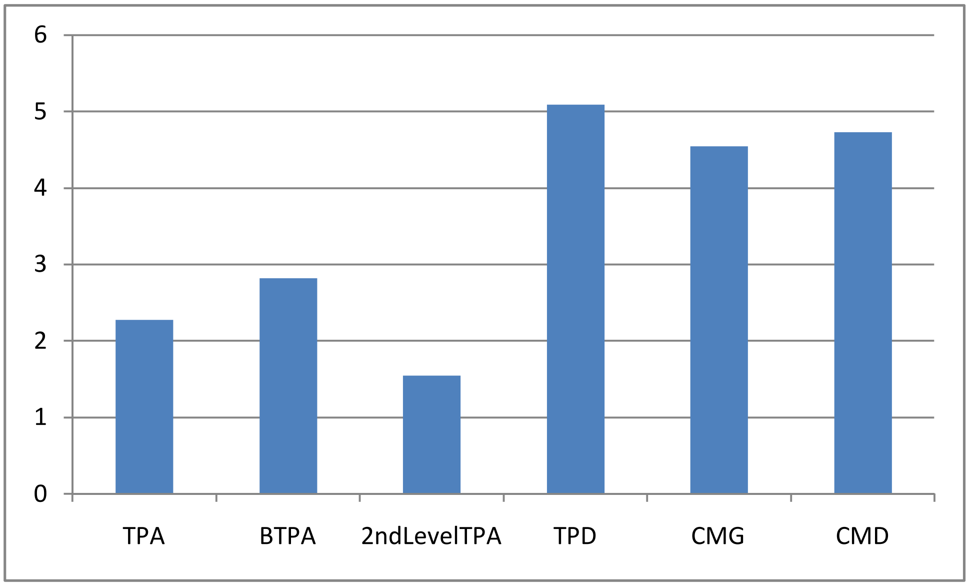


Figure 16 – Average Ranking for Base Methods

A surprising observation is TPD performs worse than TPA both in RMSE and ranking. In fact, at first glance, it is expected to get a better performance in TPD, because it includes rating styles of each user in its predictions. But dataset is so sparse to determine rating styles of users. Most of the users have rated few items in the system and this strongly affects calculation specified in (EQUATION 4.2). Both producer and consumer average ratings represented as $\overline{p}$ and $\overline{c}$, respectively, do not indicate overall average ratings in producer's and consumer's mind. For this reason, $p(i) - \overline{p}$ rating deviations for item i in consideration do not reflect the correct deviation and this formula becomes obsolete for such a sparse dataset. As dataset becomes denser, TPD is supposed to have a better ranking.

Passive user count is an important criterion to expose user coverage metric. Average passive user count for each base method is shown in Figure 17. BTPA has obviously great coverage over users. Only about 100 out of 1000 users will not get any recommendation from the system through BTPA, however, each of the other methods does not predict any rating for about 800 users. As discussed in 0, even if a randomly selected user A most probably will not have any other user in his web of trust, user A will be included in a very active user B's web of trust. This makes data used in BTPA dense and does not influence the recommendations of other trust based methods TPD, TPA and 2ndLevelTPA.

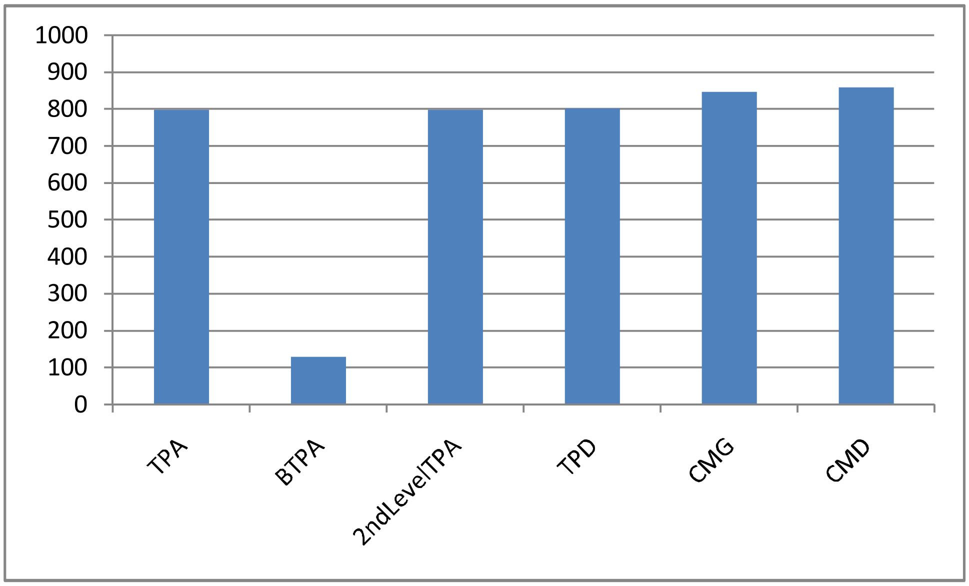


Figure 17 – Average Passive User Count for Base Methods

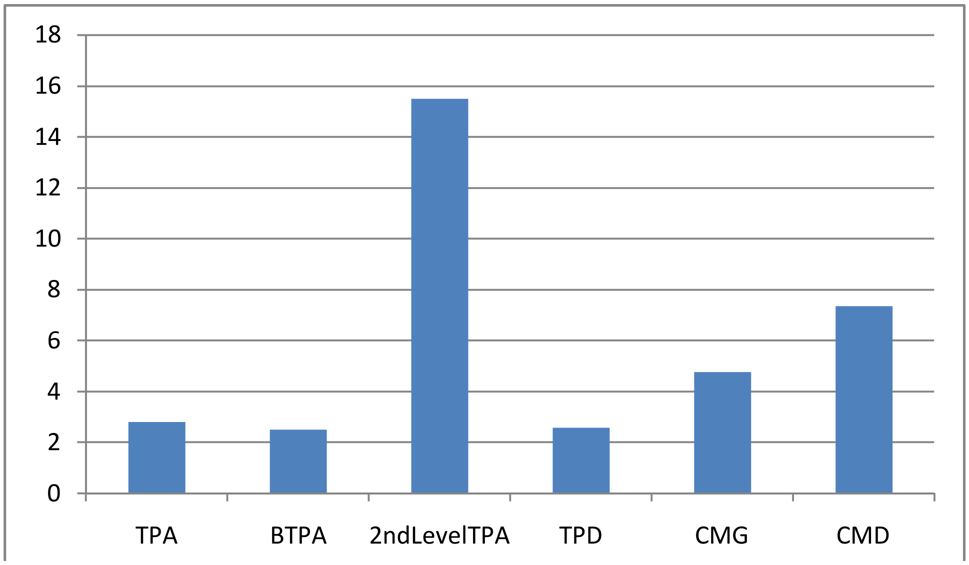


Figure 18 –Rating Coverages for Base Methods

Rating coverages for base methods are presented in Figure 18. Even if BTPA has high user coverage, it has very low rating coverage. The reason behind this result is that BTPA is able to predict at least one rating for most of the users, while for very active users who rate many items and trust many other users in the system; BTPA does not perform well. BTPA handles trust and rating data with a complementary approach. Even if a user A states trust for another user B, this will have no effect on predictions for user A, however, it affects the coverage and error rates for user B. Extreme results for BTPA may seem inaccurate first, but this results from the background idea of BTPA. BTPA uses an inverted idea of TPA, TPD, and 2ndLevelTPA. 2ndLevelTPA has a relatively better rating coverage among base methods with the help of excessive number of 2ndLevel friends compared to 1stLevel ones.

CMG has already better ranking and better RMSE rate; however, CMD is preferable according to rating coverage scores. Even if genre of a movie is more effective on user preferences compared to its release date, only use of release date has a better prediction rate. In fact, recommending movies to people only depending on their release dates does not seem reasonable, however its rating coverage can be benefitted as a component of an algorithm.

5.2 **2-Level TPA**

Overall best performer among base methods is definitely 2ndLevelTPA. Error rates and average rankings for 2ndLevelTPA are much better compared to other methods. Even if passive user count is not low enough and above 800 users cannot get any recommendation from the system, 2ndLevelTPA outperforms other methods with its rating coverage, too. Advantage of 2ndLevelTPA comes on the scene, when dataset is sparse enough, because users with a few 1stLevel friends (i.e. web of trust) may have huge number of 2ndLevel friends (i.e. 1st level friend's web of trust).

In order to benefit 2ndLevelTPA approach, we can propose to unify TPA and 2ndLevelTPA for a better performance with a weighting scheme described in 4.4. Even such a simple weighting scheme can expose benefits of both methods. TPA uses only explicitly constructed web of trust of users and 2ndLevelTPA uses 2ndLevel friends of users. With our weighting approach, unified method will find out optimal contributions for both methods to increase overall performance.

Average RMSE scores of combined TPA and 2ndLevelTPA methods are shown in Figure 19. First two bars show pure TPA and pure 2ndLevelTPA results and rest of the figure presents the several weighted versions of merged 2LevelTPAs. Each bar is named by its component methods with their weights attached to them. For instance, TPA01-2ndLevelTPA09 combined method uses weight 0.1 for TPA and weight 0.9 for 2ndLevelTPA in (EQUATION 4.6).

The best error rate is retrieved from TPA03-2ndLevelTPA07 and most of the combined approaches produce lower errors compared to pure components. The idea behind the approach is the tradeoff between data density and correctness. Ratings included in TPA processing can be seen as more reliable compared to ratings used in 2ndLevelTPA processing, while 2ndLevelTPA contains more ratings that are to be used during processing. In fact, the advantage of 2ndLevelTPA is the density of the data. The more ratings are used in the recommendation process, the more accurate recommendation can be performed. As dataset gets denser, evaluations are started to be normalized. TPA uses less data because of the shortage of user friends, while 2ndLevelTPA uses more data as we have already discussed in the previous section. 2ndLevelTPA can extract 2ndLevel friends and their count is expected to be more than the number of people in the user's web of trust. On the other hand, TPA uses explicitly mentioned trust statements by the user and each user intentionally adds another user to his web of trust. Additionally, 2ndLevelTPA purely uses 2ndLevel friends that are not in 1stLevel friends. Thus, 2ndLevelTPA infers a probable trust statement of the active user for his 2ndLevel friends, where no explicit trust statement exists in the system. Statement "Friend of my friend is also my friend" may not hold for each user and when that is the case for a user, recommendation accuracy degrades. 2LevelTPA tries to benefit advantages of both methods and improve their deficiencies. The tradeoff between correctness and data density ensures a partial utilization of each component and this provides lower average error.

Figure 20 illustrates a similar ordering in average ranking. This time TPA is not the worst performer but TPA09-2ndLevelTPA01 is. Average TPA error rates on each dataset are greater than all combined 2LevelTPA methods, however TPA gets a better average ranking in competitions compared to TPA08-2ndLevelTPA02 and TPA09-2ndLevelTPA01. Even if there is a small diversity between Figure 19 and Figure 20, TPA03-2ndLevelTPA07 still gets the best average ranking among all combinations. Both graphs exhibit that TPA03-2ndLevelTPA07 is the best method with respect to error rates and average rankings in competitions.

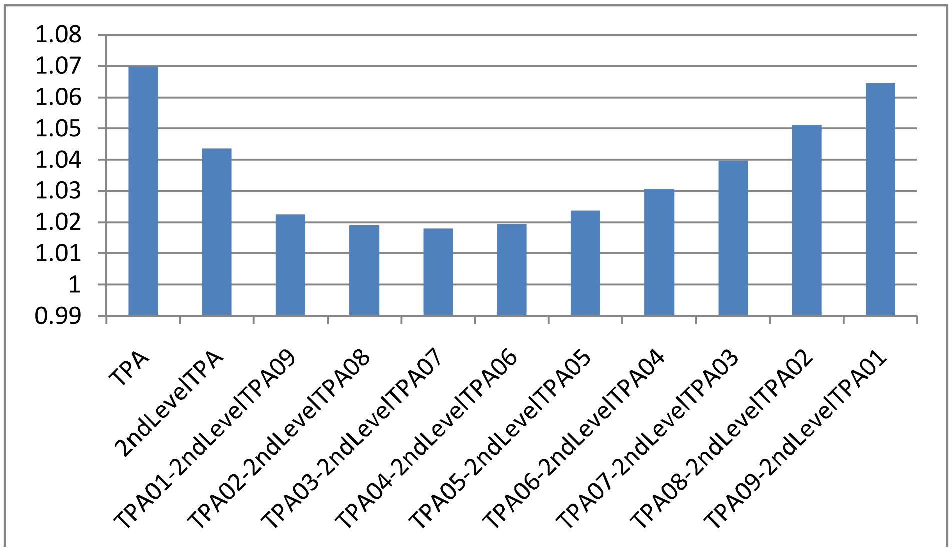


Figure 19 – Average RMSE scores for pure methods and Merged2LevelTPA

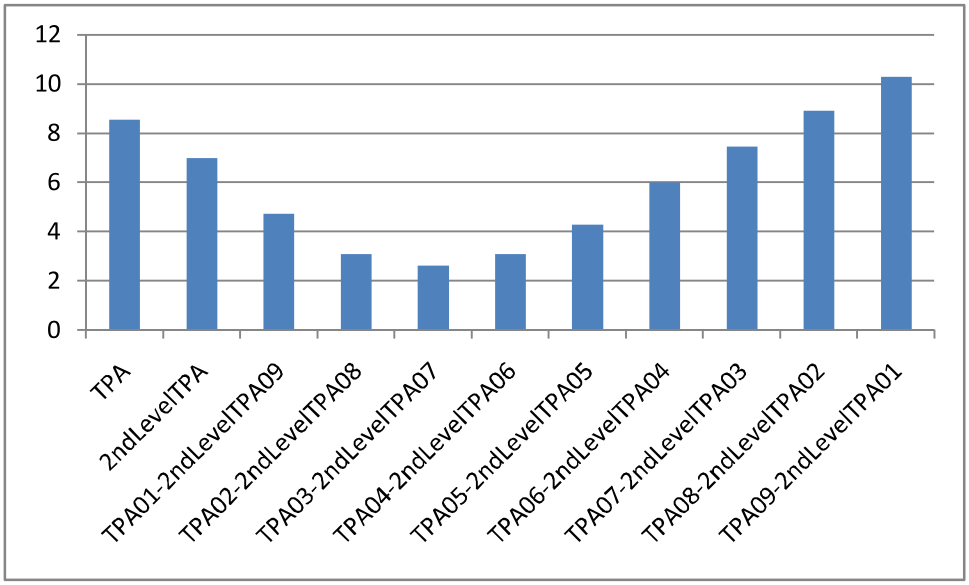


Figure 20 – Average Ranking for pure methods and Merged2LevelTPA

Average passive user count is very close for each method as figured out in Figure 21. Any combined method through (EQUATION 4.6) is always expected to have a fewer or at most same number of passive users. Because if one of the component methods does not predict a rating for a user-item pair, the other may return a prediction for the user and this may decrease the passive user count. However, a merged version of methods cannot make a prediction if and only if both components fail to predict a rating.

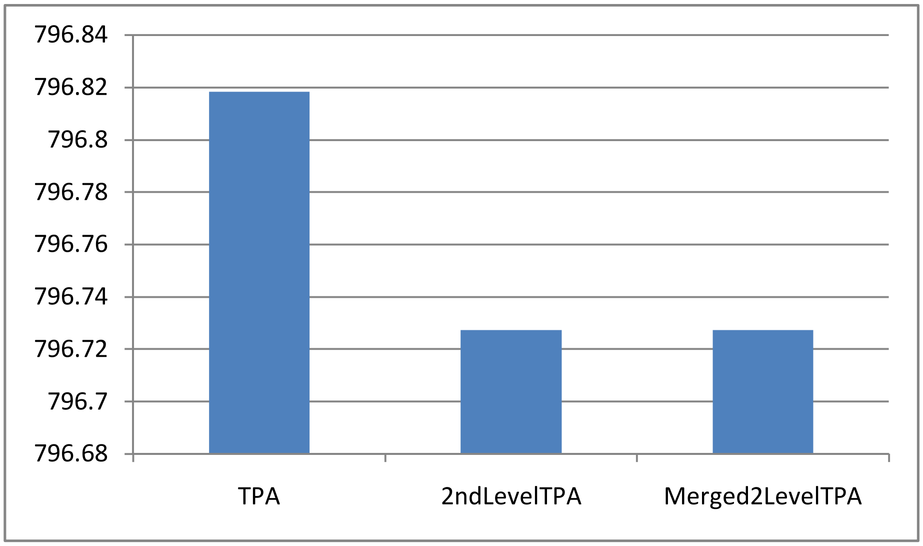


Figure 21 – Average Passive User Count for pure methods and Merged2LevelTPA

If a user has an empty web of trust, it is not possible for that user to have a 2ndLevel friend so that both methods will not predict a rating for the user-item pair. Inversely, if a user has a 2ndLevel friend, he should have at least one user in his web of trust and this will provide both methods to have a chance to perform a prediction. Of course, prediction can only be made when users in web of trust or 2ndLevel friends have already rated the item in the consideration. However, correlation in prediction possibilities is one of reasons keeping passive user counts closer.

Furthermore, it can be expected that users in the active user's web of trust will be more probable to rate the item in consideration compared to 2ndLevel friends. Because it is reasonable to assume that a user will have users, who have watched a similar set of movies, in his web of trust. This increases the probability of users included in TPA to rate the item in consideration. On the other hand, a user is expected to have much more 2ndLevel friends compared to 1stLevel ones and this increases the number of users included in 2ndLevelTPA that are potentially raters of the item in consideration. Both of these considerations make passive user counts approximately same.

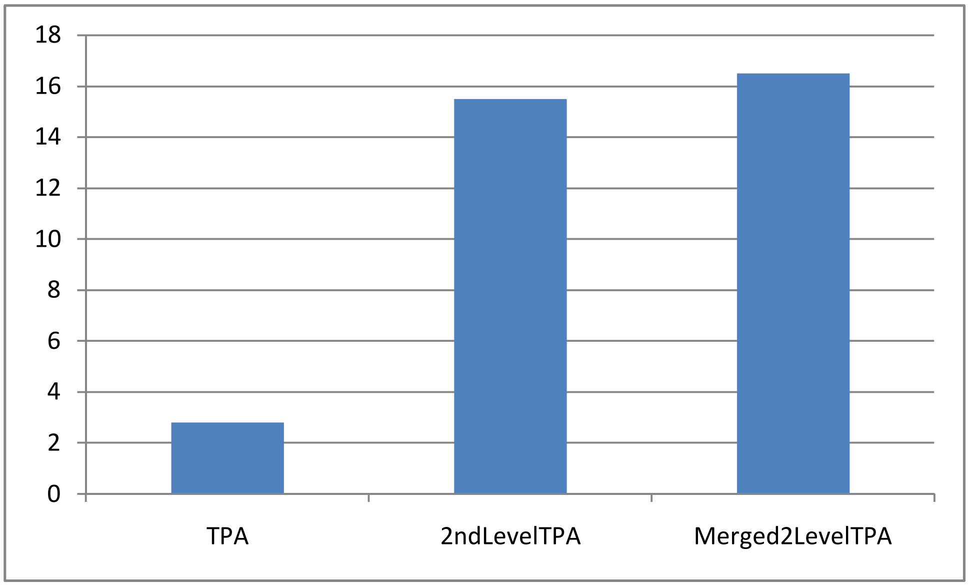


Figure 22 –Rating Coverage for pure methods and Merged2LevelTPA

As discussed in the previous section where base methods are evaluated, 2ndLevelTPA covers more ratings than TPA. Merged2LevelTPA shall have a higher coverage than its components. The reason behind is similar to fewer passive user count for a combined approach. If a user-item pair rating can be estimated by a component, combined method can automatically make a prediction. However, inverse of the statement is not true. That is if a combined method predicts a rating for a user-item pair, both of its components may not predict it. Therefore, any combined 2LevelTPA has a better rating coverage than TPA and 2ndLevelTPA.

5.3 2-Level & Bidirectional TPA

Overall best technique proposed in the previous section is TPA03-2ndLevelTPA07 where weight 0.3 assigned to TPA and weight 0.7 is used for 2ndLevelTPA. This combined method is the best performer for each criterion considered in this work. Error rate for TPA03-2ndLevelTPA07 is the least one among pure TPA, pure 2ndLevelTPA and all of their combined methods. According to average ranking, TPA03-2ndLevelTPA07 is still the best method. Lastly, combined methods are inherently better for passive user count and rating coverage performances as already mentioned in the previous section. For this reason, TPA03-2ndLevelTPA07 will be used in the remaining part of this study with its new name Best2LevelTPA.

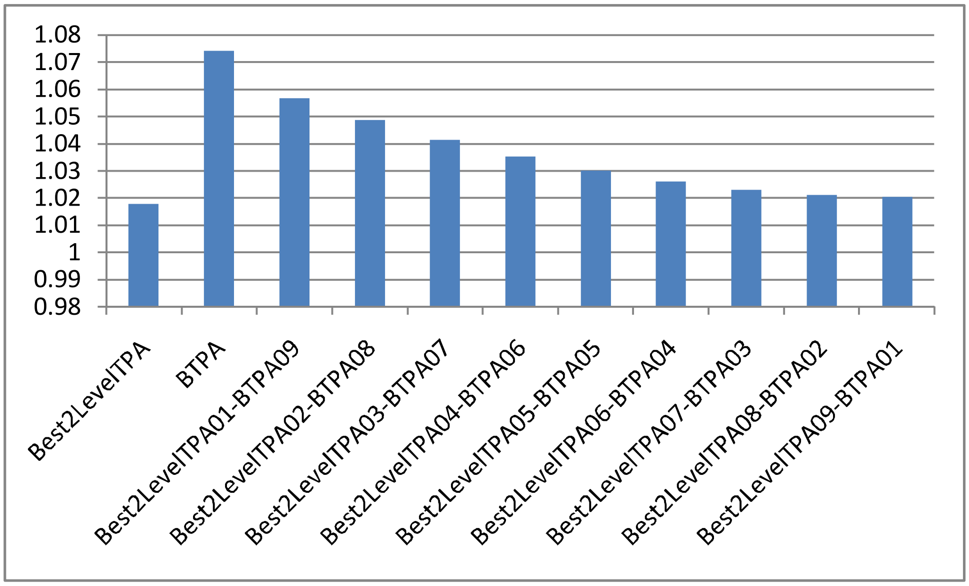


Figure 23 – Average RMSE scores for pure methods and MergedBest2LevelAndBTPA

In this section, Best2LevelTPA and BTPA methods are tried to be combined with our native weighting scheme for a better performance. Best2LevelTPA and BTPA combination is named as Best2LevelTPAAndBTPA as an abbreviation of combined Best2LevelTPA and BTPA. Best2LevelTPA considers both 1st level and 2nd level friends of the current user, where BTPA only evaluates users whose web of trusts include the current user. In fact, BTPA implements a modified version of transpose trust atomic propagation discussed in 3.2.7. It is reasonable to merge Best2LevelTPA and BTPA, because Best2LevelTPA is the overall best combination of TPA and 2ndLevelTPA, where BTPA is a method that can be used as a complementary method for TPA and 2ndLevelTPA and it is the best base method for user coverage with a far more better performance as confirmed in 5.1. Additionally, BTPA has totally different set of users compared to TPA and 2ndLevelTPA and this may add great value to Best2LevelTPA. For instance, even if a user does not specify any trust or rating statements, he can get a recommendation from the system when any other user adds him into his friend list.

Figure 23 illustrates weighted versions of Best2LevelTPA and BTPA. Best2LevelTPA has the smallest average RMSE score where Best2LevelTPA09-BTPA01 comes the second and Best2LevelTPA08-BTPA02 is the third with little differences less than 4%. It is obvious that all combined methods produce better predictions compared to pure BTPA.

It seems reasonable to have a worse performance with pure BTPA as it was the case in 3.2.7, where transpose trust atomic propagation gets a lower weight compared to direct propagation and co-citation. However, incorporation of BTPA in Best2LevelTPA adds variation to it and this is a plus for a recommendation algorithm to include different types of users or items in recommendation process. Three methods, pure Best2LevelTPA, Best2LevelTPA09-BTPA01 and Best2LevelTPA08-BTPA02, have close performances and they are the leading three methods. However, combined methods inherently have an advantage of variation.

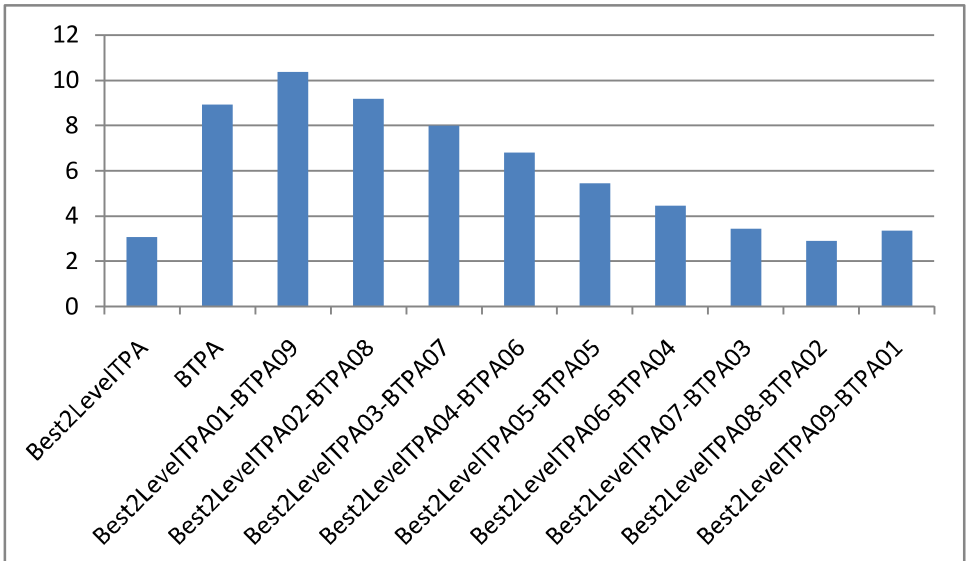


Figure 24 – Average Ranking for pure methods and MergedBest2LevelAndBTPA

Figure 24 shows average rankings for each method. BTPA gets a better ranking in this figure compared to average RMSE scores. Best2LevelTPA08-BTPA02 obtains the best ranking among all approaches. This indicates that Best2LevelTPA08-BTPA02 combined method has high error rates for some datasets where it has a relatively worse ranking against Best2LevelTPA (i.e. average RMSE score winner) and Best2LevelTPA09-BTPA01 (i.e. average RMSE score runner-up), but it wins against them with little differences for other datasets.

Passive user counts for all methods are illustrated in Figure 25. Obviously the expectation while combining Best2LevelTPA and BTPA is met in this figure. BTPA brings its advantage of having less number of passive users to the combined approach. As obtained in the previous section, even a merged method inherently decreases the passive user count and increases the rating coverage, Best2LevelTPA does not have a very good performance for passive user count criterion. Best2LevelTPA is not able to predict a single rating for about 800 users and this performance is not satisfactory for a recommender system. However, as figured out in Figure 17, BTPA has the least number of passive users among base methods. Combined method has 112.5 passive users out of 1000 users in average with the help of our combination scheme, while BTPA has 127.9 passive users. Here, combination benefit from dense input provided to BTPA explained in 0.

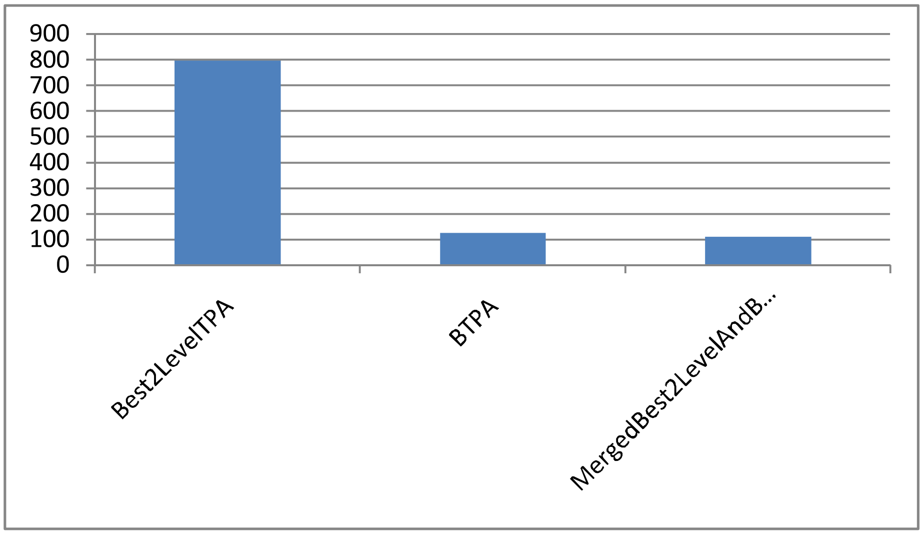


Figure 25 – Average Passive User Count for pure methods and MergedBest2LevelAndBTPA

As an overview of Figure 25, average passive user count for BTPA is low enough because of the density of the users that are included in its recommendation process. As already mentioned in 0, where Epinions dataset is presented, a randomly selected user will most probably have no users in his web of trust, while he will be included in web of trust of another user. This fact increases the number users included in (EQUATION 4.3) which in turn increases the number of predictions that can be performed by the recommender.

As in case of all unified methods, a combined method has a better rating coverage than its component approaches and rating coverage for Best2LevelTPA and BTPA combination is presented in Figure 26. A little increase in rating coverage is gained for the combined method, MergedBest2LevelAndBTPA. As it can be seen from Figure 25 and Figure 26, Best2LevelTPA has a better rating coverage while it has worse user coverage. According to these results, Best2LevelTPA predicts more ratings among all user-item pairs while it does not estimate even a single rating for most of the users. More predictions are performed by Best2LevelTPA for a small set of users; however a good recommender shall be able to present recommendations for each user in order to let them use the system, efficiently. Combined MergedBest2LevelAndBTPA method utilizes both methods advantages and acquires the best rating coverage.

Best2LevelTPA method covers more user-item pairs than BTPA in the system. It is important to note that users in the active user's web of trust or his 2nd level friends are potential users to rate the item in consideration, because people choose users with similar preferences as their friends. At first glance, it seems less possible for a user, who has the active user in his web of trust, to rate the item in evaluation. Furthermore, first level (i.e. users in his web of trust) and second level friends of the active user are obtained through explicitly stated trust statements where users having the active user in their web of trust are detected with inverse implicit relations of trust statements. These facts reveal the better performance of Best2LevelTPA compared to BTPA. In fact, it is already illustrated in Figure 18 that both TPA and 2ndLevelTPA covers more ratings compared to BTPA. Additionally, Figure 22 shows the great improvement in rating coverage of TPA when it is combined with 2ndLevelTPA. It is obvious that a better rating coverage is already gathered in previous sections for Best2LevelTPA; however, it is significant to mention reasons behind the fact.

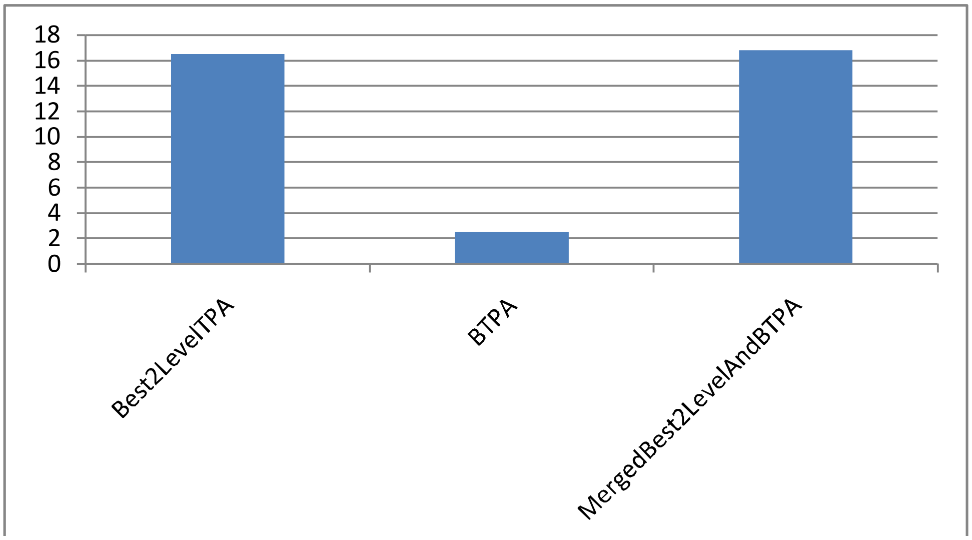


Figure 26 –Rating Coverage for pure methods and MergedBest2LevelAndBTPA

5.4 Cultural Metadata with Genre and Date

According to descriptions of base methods given in 4.3, in order to predict a rating for a user-item pair, CMG and CMD methods use already given ratings by the active user. Ratings given to any item by any other user are not taken into account. These methods are based on Cultural Metadata described in 3.2.2 and the idea behind these approaches is "A user, who has already specified his preferences and tastes, will act consistently in future and will like products with similar content information". Indeed, this is the idea of a generic content based recommender.

Furthermore, base method evaluation exposed poor performances of CMG and CMD on average error rate and ranking. They are the worst two methods next to TPD. Inability of these methods to predict even a single rating for each user causes another poor performance on passive user counts. However, with the intention of increasing average rating coverage, one may use these cultural metadata based algorithms. Even if

2ndLevelTPA has the best rating coverage, CMD and CMG come second and third, respectively.

Up to now, trust based methods such as TPA, BTPA and 2ndLevelTPA are combined to produce a better recommendation algorithm. However, as discussed in 2.2.4, content based recommender systems still occupy an important place in research area. It seems reasonable to combine CMG and CMD base methods independent from trust based methods to have a pure content based algorithm with a more preferable performance than pure CMG and CMD. In fact, this combination may not win competition against pure techniques; still this will provide us to interpret reasons of the failure.

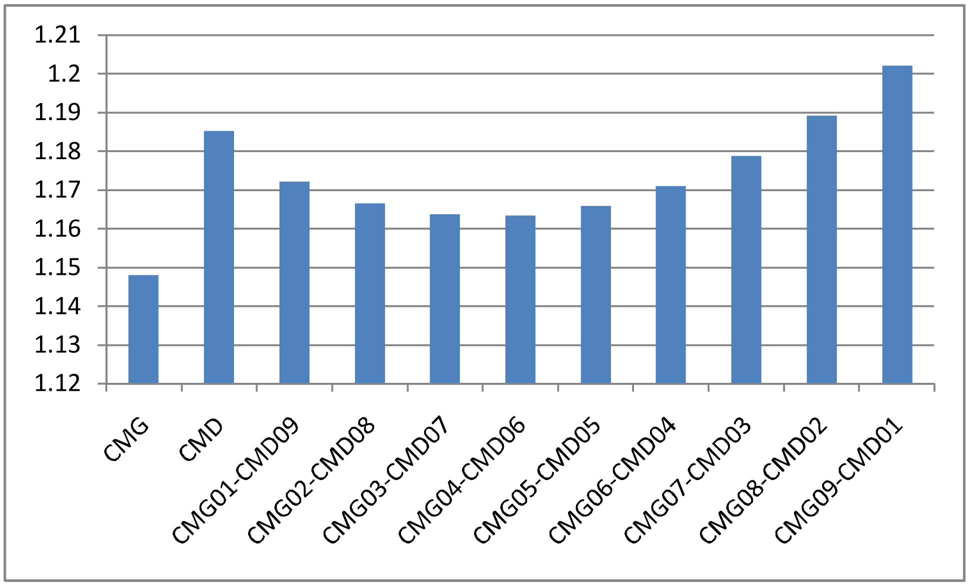


Figure 27 – Average RMSE scores for pure methods and MergedCMGD

In Figure 27, average error rates are presented. Error rates of pure CMG and CMD have already discussed in base method evaluation, however it is important to note advantages and disadvantages of merging these two techniques. None of the combined methods proposes a more qualified method with respect to RMSE rates. Best combination is achieved by CMG04-CMD-06 and it has a 13% worse performance compared to pure CMG. Observations on combination techniques represent that incorporation of release date to CMG performs less when weights for both methods are not close, where more identical weights of CMG and CMD has a better performance. According to given average RMSE scores, genre has a significant influence on user preferences that is "A simple automated system can determine movie preference of a user more precisely when it uses only genre information compared to only release date use." However, this does not state "Movie release dates are useless in recommendation". Some combinations of CMG and CMD perform almost same as the best performer.

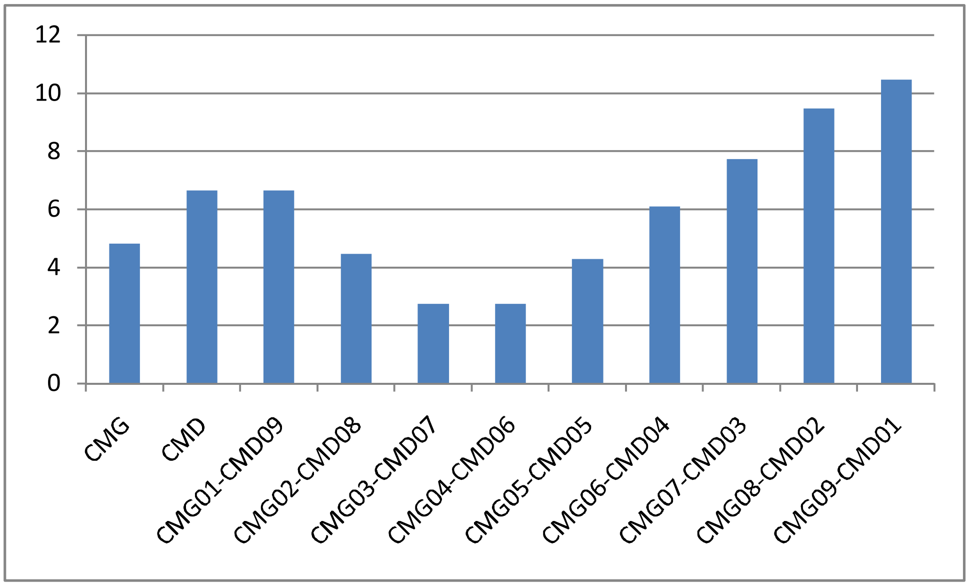


Figure 28 – Average Ranking for pure methods and MergedCMGD

Similar ordering comes on the scene when average rankings are considered. However as illustrated in Figure 28, rankings of CMG02-CMD08, CMG03-CMD07, CMG04-CMD06 and CMG05-CMD05 are superior to pure CMG. Obviously, combinations with better average ranking have high error rates in competitions that CMG wins and low error rates on competitions they win. This fact results the discrepancy between Figure 27 and Figure 28 and presents the importance of release dates in recommendation. Even in our daily life, some people adore films released in 1960s, where some others like only movies released in near past days. Movies released in a specific time period such as 1960s or 1990s etc. may attract some people and such people may prefer only similar movies in the same period to experience.

Figure 29 shows passive user count of CMG, CMD and combined method CMGD. Combined methods utilize advantages of CMD and eliminate disadvantages of CMG. Actually; there is not a huge difference between passive user counts of CMD and CMGD. However, this does not change the fact that combined methods shall have better performance in coverage analysis. CMGD has a better user coverage compared to CMD because ratings for a set of user-item pairs are determined by only CMG component of CMGD. For this reason, even if CMG has a worse performance in user coverage analysis, it is possible to utilize its features as in CMGD.

Different periods of release dates are enumerated with a less number of identifiers compared to genre types. There are about thirty different genre types where there are only eight enumerators for release date intervals. For this reason, when the system is trying to make a prediction for a specific user-item pair, it is more probable to encounter already rated movies by the active user that are released in the same period with the item in consideration compared to movies incorporating same genre type. That is one of the fascinating reasons stimulating average coverage and passive user count improvement of CMD.

Rating coverage of each method is figured out in Figure 30. A set of user-item pairs are covered by CMG, another set of user-item pairs are covered by CMD and since rating coverage of CMGD is less than total rating coverage of CMG and CMD, some set of user-item pairs are covered with the help of both methods.

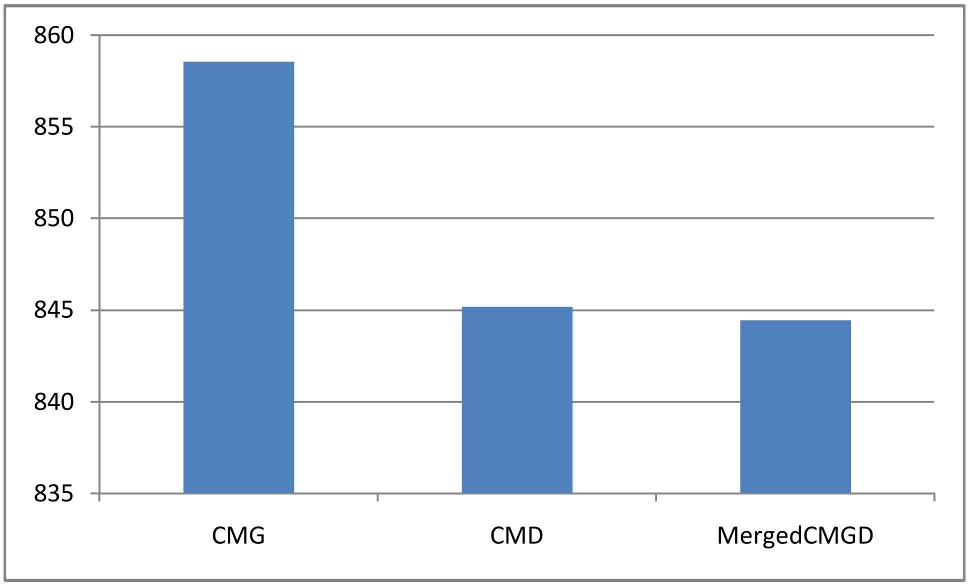


Figure 29 – Average Passive User Count for pure methods and MergedBest2LevelAndBTPA

Combined methods, mentioned in this section, use ratings already given by the active user as its components. During prediction process for a single user-item pair, CMGD lists already rated items by the active user with ratings attached to them and predicts two ratings for the item in consideration, where one of these ratings is predicted by CMG component (i.e. only genre of already rated items are taken into consideration) and the other rating is predicted by CMD component (i.e. only release data of already rated items are considered). Normalization phase of prediction is performed by our native weighting scheme (EQUATION 4.6) and normalized rating is nothing but the predicted rating for the item. In fact, CMG and CMD are not able to predict any rating when similarity between an already rated item and the item in consideration is not above a threshold. By this way, some inaccurate predictions of each component are filtered out.

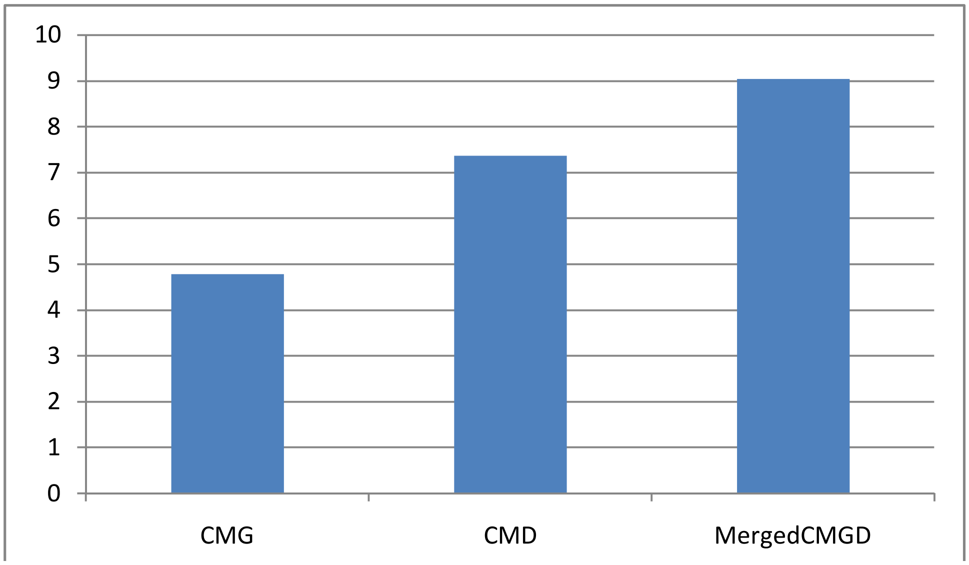


Figure 30 –Rating Coverage for pure methods and MergedBest2LevelAndBTPA

5.5 Best2LevelBTPA and BestCMGD

In 5.3, Best2LevelTPA and BTPA methods are unified to have a single method which bears web of trust, 2nd level friends and inverse friends (i.e. users whose web of trust include the active user) in mind. In fact, Best2LevelTPA has already emerged as a method inheriting characteristics of TPA and 2ndLevelTPA. Embedding BTPA into Best2LevelTPA has already created a method which gathers up all trust based methods discussed in the scope of this study. Combination Best2LevelTPA08-BTPA02 has got the highest average ranking even if it has a slight worse performance with respect to average RMSE scores. Moreover, because of the idea behind the combination process, Best2LevelTPA08-BTPA02 has also covered more users and ratings compared to pure methods. Moreover, unlike other combination evaluations, high user coverage of BTPA has provided a great improvement on Best2LevelTPA. In the remaining part of this study, overall best performer of 5.3, Best2LevelTPA08-BTPA02, will be renamed as Best2LevelBTPA from now on.

On the other hand, in 5.4, a more powerful content based recommendation algorithm is found out. Instead of previous methods, new combination only includes content based base methods CMG and CMD. In fact, again a pure method, CMG, is the winner with respect to average RMSE scores; however, it is not the case in average ranking. CMG is not even in very first three ranking methods. For this reason, with on-the-fly advantages of our combination scheme, CMG04-CMD06 can be selected as the overall best content based recommender and it will be called as BestCMGD in the remaining sections.

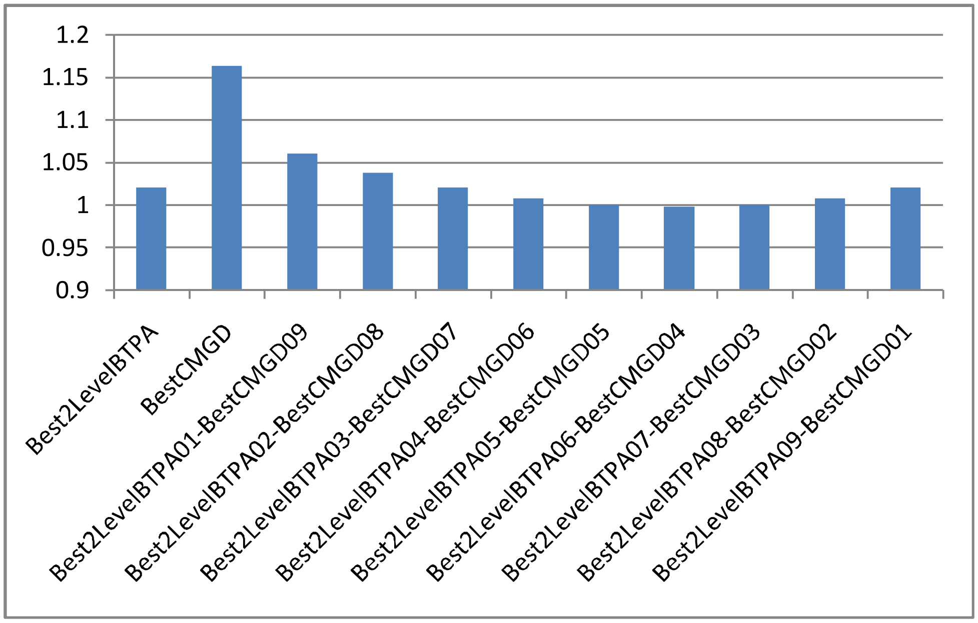


Figure 31 – Average RMSE scores for pure methods and MergedBest2LevelBTPAAndBestCMGD

Method merge operations have been performed in previous sections, but in this section this will be slightly different from previous ones. Until this section, all combined methods have same underlying idea with its components, such as a trust based method is combined with another trust based method and a content based method is combined with another one. All components that are used in weighting process make use of same data type with each other; however, each has a different kind of data handling. In this section, same weighting approach will merge two methods one is based on trust while the other is based on content information.

Both in Figure 31 and Figure 32, Best2LevelBTPA06-BestCMGD04 has the best ranking. According to these results, combination of trust and content based methods provide a better performance compared to its pure components. Additionally, among all methods proposed, it is the first time that we observe an average RMSE score less than 1. Most of Best2LevelBTPA and BestCMGD combinations have average RMSE scores about 1 and they also have better ranking compared to pure methods. This indicates combinations propose really successful methods. The correlation between average ranking and RMSE scores exposes the fact that we have found out "consistent" methods valid for each dataset.

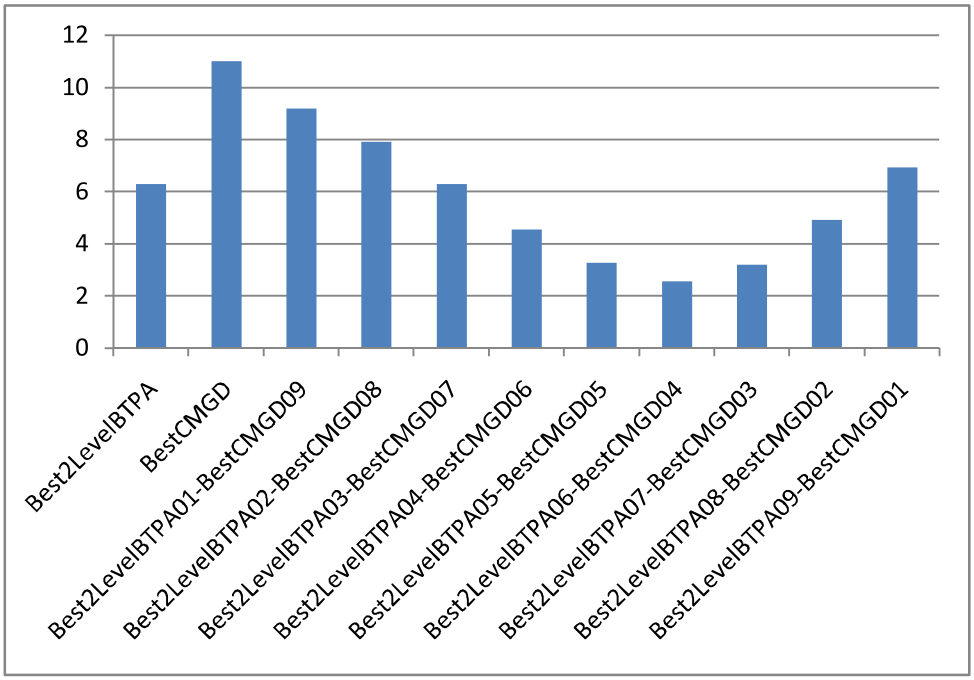


Figure 32 – Average Ranking for pure methods and MergedBest2LevelBTPAAndBestCMGD

Combined methods stand for a new approach incorporating cultural metadata with genre and date, 1st level friends, 2nd level friends and inverse friends (i.e. users used in BTPA prediction). According to this approach, a rating depending on relationships between the active user and rest of the population is retrieved by trust based component. This component is able to predict a rating for a user-item pair, if the item in consideration has been already rated by another user who is 1st level, 2nd level or inverse friend of the active user. Furthermore, already specified active user preferences play an important role in determining preferences of the active user in future. This consideration is taken into account with the help of content based component. Combination of both components seems a smart idea as displayed on Figure 31 and Figure 32. These figures also illustrates combined methods has converged to the optimal rating. Furthermore, both already experienced movies in the past and closer users in the social network affect user preferences.

Another evaluation aspect of combined methods is rating and user coverage. Even if a user does not have any other user in his web of trust, he can still make use of the recommender system, since his rating history will be used to find out his preferences. Lack of recommendations when no user exists in the active user's web of trust is the most important deficiency of Moleskiing and FilmTrust methods already discussed in sections 3.2.5, 3.2.6, respectively. Content based component eliminates this deficiency in Best2LevelBTPA.

Additionally, base method BTPA has already decreased excessive amount of passive users existing in Best2LevelTPA. However, BestCMGD, as already presented in Figure 29, has about 850 passive users and its user coverage is very low. It is obvious that combination scheme will provide better user coverage than both of its components so that this will increase the user coverage of BestCMGD. In the light of these comments, there is a mutual relationship between BestCMGD and Best2LevelTPA.

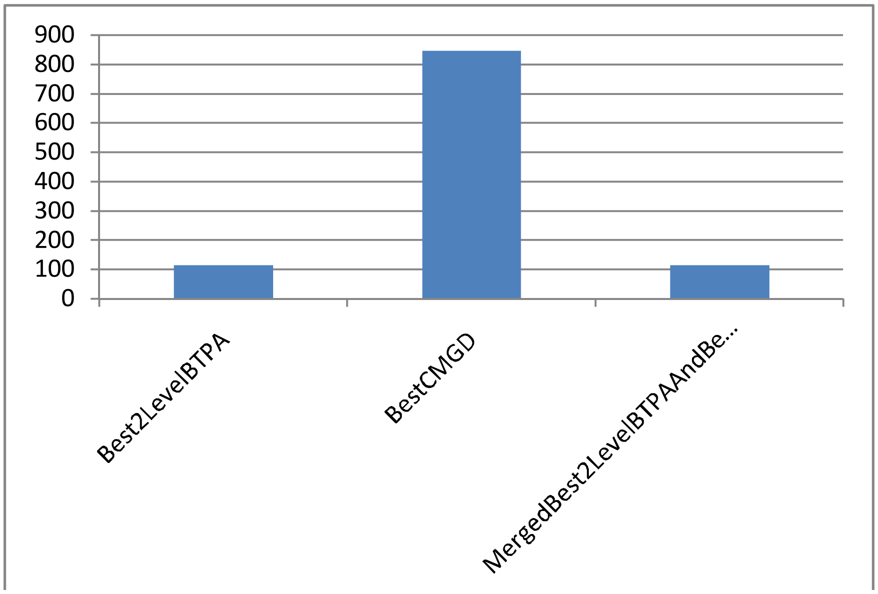


Figure 33 – Average Passive User Count for pure methods and MergedBest2LevelBTPAAndBestCMGD

As presented in Figure 33, combination does not provide a significant reduction in passive user count for Best2LevelBTPA, since it has an already embedded BTPA component and this component has already reduced the number of passive user count as shown in Figure 25. On the other hand, merged method has a better user coverage compared to BestCMGD with the help of weighting process.

Moreover, rating coverage of both methods is increased as shown in Figure 34. This result shows user preference and relationships of the user cover different parts of the user-item matrix. Previous comments related to passive user counts are also applicable for rating coverage analysis. Both methods utilize each other with different aspects and this provides a better rating coverage for the combined method.

In conclusion, the overall best performance is presented by Best2LevelBTPA06-BestCMGD04, since it is the best method proposed according to four criteria analyzed in the scope of this work. With the use of similar naming convention, Best2LevelBTPA06-BestCMGD04 can be renamed as Best2LevelBTPAWithCMGD.

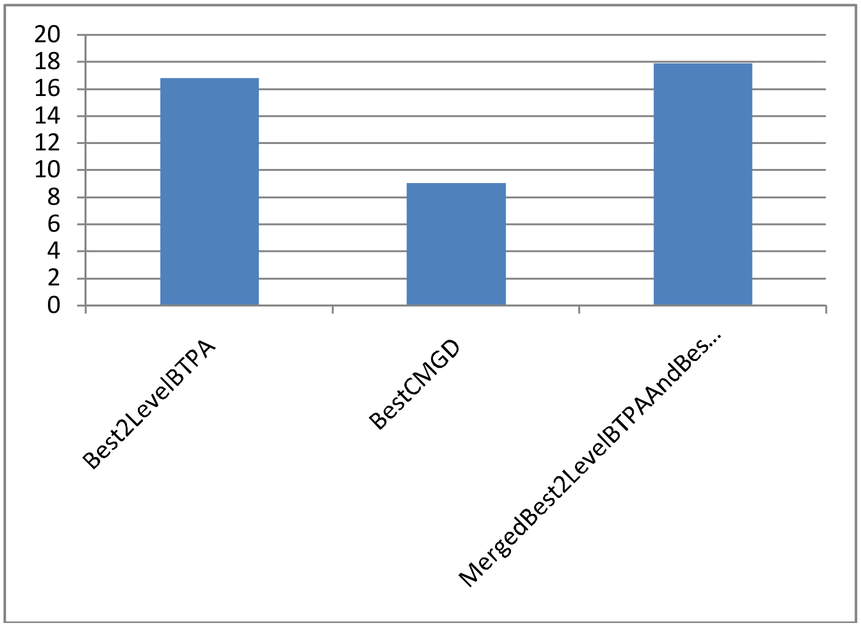


Figure 34 –Rating Coverage for pure methods and MergedBest2LevelBTPAAndBestCMGD

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

Within this thesis work, various kinds of methods have been presented with the help of already proposed approaches in Chapter 3. New techniques are proposed to increase accuracy and coverage of recommendation algorithms and they are constructed by already described methods in 3.2. A single weighting scheme (EQUATION 4.6) is used throughout the study to combine different base methods. Idea of weighting scheme has been already discussed in 2.2.6.1.1 for combining separate content based and collaborative filtering based recommenders.

As a different approach, this study merges different techniques based on trust statements of the active user and content information of items that the active user has already rated. In the scope of this thesis work, each proposed method benefits local trust metric. At first glance, trust statements among users seem to cover content information of items. Because, it is assumed that a user A trusts another user B if and only if common items rated by both users intersect according to their content. However, it is discovered that users do not adhere this kind of idea. Additionally, with a step-by-step procedure, a superior technique for recommendation is proposed at the end. At each step, a different data is started to be used in recommendation process. Combined methods always utilize advantages of its components and eliminate their deficiencies.

Strategy of weighting different components to get a unified method is inherently structured to express the reasons behind the prediction. A good recommender system is supposed to state why an item is recommended to the user. Proposed methods in this work can benefit its native idea to achieve this aim. This can be held by investigating the overall predicted rating, predicted ratings for each component and their weights in the complete approach. A high correlation with overall predicted rating disclose the related component has highly influence in recommendation, where a low correlation with overall prediction reveals less impact on recommendation result. By this way, a comparison of overall predicted rating and predicted ratings by each component, reason behind the recommendation can be explained to the user.

In addition, all base methods are generated with a state of the art solution to recommender systems based on discussed solutions in 3.2. But, lastly proposed approach, Best2LevelBTPAWithCMGD is the overall best performer in this study and it is the only method that produces an average RMSE score less than 1. As discussed in 0 where Epinions dataset is overviewed, dataset is so sparse to make rating estimations for user-item pairs. With such rate of sparseness, it is impossible to predict ratings for user-item pairs with a generic collaborative filtering as mentioned in 2.2.5.2.3. User similarity calculation of generic collaborative filtering approach needs to analyze a significant number of items that are commonly rated by users for whom similarity will be calculated. On the other hand, item based collaborative filtering needs a set of users who have commonly rated a set of items and the item in consideration. Such commonalities cannot be met in Epinions dataset, since user-item matrix is sparser than 99.99%.

Most of the described deficiencies in 2.2.4.2 and 2.2.5.2 are totally removed in Best2LevelBTPAWithCMGD. Collaborative filtering algorithm embedded in trust based components removes the limited content analysis problem encountered in content based recommenders. No content analysis is performed by TPA, 2ndLevelTPA and BTPA components and they totally make their predictions depending on social network among users. Again, overspecialization problem is resolved by trust base components. These components find out friends of the active user and these users may have different tastes which provide several kinds of recommendations for the active user different than his past preferences. Cold start user problem is also resolved with our approach. Even a user, who has not provided a single rating or trust statement yet, can get recommendation from the system. It is enough to be in another user's web of trust who has already rated a list of items.

On the other hand, the proposed method resolves deficiencies specific to a generic collaborative filtering approach. Possibility of malicious user attacks is degraded by the use of local trust metric. If a list of fake users trusts the active user, they can cause misleading recommendations because of the BTPA component in the proposed method. However, total weight of BTPA component is about 5% in Best2LevelBTPAWithCMGD and if the active user rates a set of items or trusts in other users, no problem will be encountered because of the high domination of other components compared to BTPA. Neighbor transitivity is not lost in our method, adversely; Best2LevelBTPAWithCMGD rewards neighbor transitivity with incorporation of 2ndLevelTPA. Computational complexity of collaborative filtering is higher than our proposed method, since it has to compare the active user with each remaining users in the dataset in order to select most similar users. In our proposed method, only users closer to the active user in social network are included in evaluation and this increases the performance of the algorithm. The last deficiency of the generic collaborative filtering approach resolved by the proposed method is the first rater problem. An item that is not rated by any user can be recommended to the active user according to the proposed method. It is enough to have similar items that are rated by the active user, because CMG and CMD components recommend items that are similar to past preferences of the active user.

In conclusion, the proposed method resolves deficiencies of the state of the art solutions by utilizing other state of the art solutions. Integration of new algorithms to our proposed method or incorporation of other data types into the algorithm can provide it extra advantages over previously proposed methods.

6.2 **Future Work**

The area of recommender systems is a hot topic and many ideas can improve the performance of existing methods.

One idea can be use of distrust data. Epinions site does not let people crawl block lists of users. By this way, site administrators aim to keep accurate trust statements in the system. Because if distrust data were explicitly displayed, people who are in block lists of other users can distrust these users with a feeling of revenge. However, if block lists become readily available, it will be beneficial to use this data in recommendation process.

Another idea to improve recommendation accuracy can be use of different algorithms for different product categories. In the scope of this thesis work, only movies are extracted from Epinions, however, Epinions also contain various kinds of product categories, such as cars, home & garden, music etc. For each of these categories, different trust statements can be stored in the system. Because, user A can trust user B in movies domain, however user A may not trust user B in any other domain, even he may distrust user B. In fact, this approach will correspond to different social networks for different domains. Furthermore, different recommendation algorithms can be used for different product categories. This seems reasonable at first glance. In our daily life, a person usually chooses to consult distinct people in distinct domains. Still, results for this kind of approach shall be presented in order to see its advantages and disadvantages.

Moreover, more content information is readily available in Epinions and each of this information can be used in recommendation. Actors, actresses, directors, producers of movies are already available and already crawled by our crawler. However, our aim is not to produce the best possible method with existing data, but finding out whether recommendation with trust data and item content information produce better results or not.

Combination technique proposed in this study can be improved and more complex combinations of methods can perform well. Even through a simple weighting algorithm to combine different approaches, we can guarantee an improvement on user and rating coverage. In this work, we aim to show trust based components and content based components can be unified with a better performance. Additionally, we have already represented trust data provided by users does not fully cover content based similarities of users. If this were the case, trust based methods will produce better results and combinations will not improve error rates.

All proposed future works in this section can be used for further improvements of this study. In fact, many extensions of proposed methods can be constructed to get better results than presented here, however, as mentioned, we do not aim to find out the best combination of trust based and content based methods. Use of new data types may improve performances of proposed algorithms and may expose new facts about recommender systems.

REFERENCES

[1] P. J. Denning. ACM President's letter: Electronic Junk, *Community ACM, pp 163-165, 1982*

[2] Improving Tag-Clouds as Visual Information Retrieval Interfaces, *International Conference on Multidisciplinary Information Sciences and Technologies, InSciT2006, Merida, Spain, October 25-28, 2006*

[3] Emmanouil Vozalis, Konstantinos G Margaritis, "Analysis of Recommender Systems' Algorithms", in *Proceedings of the Sixth Hellenic-European Conference on Computer Mathematics and its Applications – HERCMA 2003, 2003, pp. 732-745*.

[4] Amazon.com, www.amazon.com, Last accessed on May 2010.

[5] The Internet Movie Database (IMDb), http://www.imdb.com/, Last accessed on May 2010.

[6] MovieLens, www.movielens.umn.edu, Last accessed on May 2010.

[7] PANDORA, www.pandora.com, Last accessed on May 2010.

[8] Last.fm, www.lastfm.com, Last accessed on May 2010.

[9] Netflix Prize Competition, www.netflixprize.com, Last accessed on May 2010.

[10] D.Billsus, C.A. Brunk, C.Evans, B.Gladish and M.Pazzani, "Adaptive Interfaces for Ubiquitous Web Access", *Comm. ACM, vol. 45, no. 5 pp. 34-38, 2002*

[11] Paolo Avesani, Paolo Massa, Roberto Tiella, "A Trust-enhanced Recommender

[12] System application: Moleskiing", *Proceedings of the 2005 ACM Symposium on Applied Computing (SAC), 1589-1593.*

[13] Gediminas Adomavicius, 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*

[14] Marko Balabanovic, Yoav Shoham, "Fab: Content-based, collaborative recommendation (Special Section: Recommender Systems)", *Communications of the ACM, March 1997 v40 n3 p66(7)*

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

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

[17] D. Goldberg, D. Nichols, B.M. Oki and D.Terry, "Using Collaborative Filtering to Weave an Information Tapestry", *Communications of the ACM, vol. 35, no. 12, pp. 61-70, 1992*

[18] J.A. Konstan, B.N. Miller, D.Maltz, J.L. Herlocker, L.R. Gordon, and J. Riedl, "GroupLens: Applying Collaborative Filtering to Usenet News", *Communications of the ACM, vol. 40, no. 3, pp. 77-87, 1997*

[19] P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J.Riedl, “GroupLens: An Open Architecture for Collaborative Filtering of Netnews”, *Proc. 1994 Computer Supported Cooperative Work Conference, 1994*

[20] W. Hill, L.Stead, M. Rosenstein, and G.Furnas, “Recommending and Evaluating Choices in a Virtual Community of Use”, *Proc. Conference Human Factors in Computing Systems, 1995*

[21] U. Shardanand, P. Maes, “Social Information Filtering: Algorithms for Automating ‘Word of Mouth”, *Proc. Conference Human Factors in Computer Systems, 1995*

[22] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, “Eigentaste: A Constant Time Collaborative Filtering Algorithm”, *Information Retrieval Journal vol. 4, no. 2, pp. 133-151, July 2001*

[23] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter, “PHOAKS: A System for sharing Recommendations”, *Communications of the ACM, vol. 40, no. 3, pp. 59-62, 1997*

[24] John O’Donovan, Barry Smith, “Trust in Recommender Systems”, *IUI’05, January 9-12, 2005, San Diego, California, USA*

[25] Yehuda Koren, “Tutorial on Recent Progress in Collaborative Filtering”, *RecSys ’08, October 23-25 2008, Lausanne, Switzerland*

[26] Shinhyn Ahn, Chung-Kon Shi, “Exploring Movie Recommendation System Using Cultural Metadata”, *International Conference on CyberWorlds 2008*

[27] B.Sheth and P.Maes, "Evolving Agents for Personalized Information Filtering", *Proc.18th Conference Uncertainty in Artificial Intelligence, August 2002*

[28] D.Billsus and M.Pazzani, "User Modeling for Adaptive News Access", *User Modeling and User-Adapted Interation, vol. 10, nos. 2-3, pp.147-180, 2000*

[29] Dipankaj G. Medhi, Juri Dakua, "MovieReco: A Recommendation System". *Proceedings of World Academy of Science, Engineering and Technology vol. 4, February 2005*

[30] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, John Riedl, "Getting to Know You: Learning New User Preferences in Recommender Systems", *Intelligent User Interfaces '02, January 13-16, 2002*

[31] M. Pazzani, "A Framework for Collaborative, Content-Based and Demographic Filtering", *Artificial Intelligence Rev. pp 393-408*, *December 1999*

[32] Yao Wang, Julita Vassileva, "Trust and Reputation Model in Peer-to-Peer Networks", *Proc. of IEEE Conference on P2P Computing, Sweden, September 2003*

[33] Jennifer Golbeck, James Hendler, "FilmTrust: Movie Recommendations using Trust in Web-based Social Networks". *Proc. of IEEE CCNC 2006*

[34] R. Guha, Ravi Kumar, Prabhakar Raghavan, Andrew Tomkins, "Propagation of Trust and Distrust", *WWW 2004, Proc of ACM, May 17-22, New York*

[35] Philip Bonhard, "Improving Recommender Systems with Social Networking", *Workshop: Beyond Personalization 2005, IUI'05, January 9, 2005, San Diego, California, USA*

[36] Paolo Massa and Paolo Avesani, "Trust-aware Collaborative Filtering for Recommender Systems", *Proc. of 2007 ACM Conference on Recommender Systems*

[37] Hamza Kaya, "Using Social Graphs in One-Class Collaborative Filtering Problem", *MSc Thesis, August 2009*

[38] Heng Luo, Changyong Niu, Ruimin Shen, Carsten Ulrich, "A collaborative filtering framework based on both local users similarity and global user similarity", *Proc. of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases*

[39] Gözde Özbal, "A Content Boosted Collaborative Filtering Approach for Movie Recommendation based on Local & Global User Similarity and Missing Data Prediction", *MSc Thesis, September 2009*

[40] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murkinov, Dmitry Netes, and Matthew Sartin, "Combining content-based and collaborative filters in an online newspaper", *ACM SIGIR Workshop on Recommender Systems – Implementation and Evaluation, Berkeley, CA, 1999*

[41] Bardul M.Sarwar, "Sparsity, Scalability, and Distribution in Recommender Systems" *Ph.D. Thesis, University of Minnesota, 2001*

[42] John S.Breese, David Heckerman and Carl Kadie, "Emprical analysis of predictive algorithms for collaborative filtering", *14th Conference on Uncertainty in Artificial Intelligence, Madison, 1998*

[43] Helen Sharp, Yvonne Rogers, Jenny Preece, "Interaction Design: Beyond Human-Computer Interaction", *2002*

[44] Philip Bonhard, “Improving Recommender Systems with Social Networking”, *Proc. of Addendum of CSCW 2004, Nov. 6-10, Chicago, IL*

[45] J.L. Herlocker, J.A. Konstan, A. Borchers, J. Riedl, “Explaining Collaborative Filtering Recommendations”, *Proc. of the ACM 2000 Conference on Computer Supported Cooperative Work, pp. 241-250*

[46] S. M. MacNee, K. S. Lam, C. Guetzlaff, J.A. Konstan, J. Riedl, “Confidence Displays and Training in Recommender Systems”, *Proc. of INTERACT ’03 IFIP TC13 International Conference on Human Computer Interaction, pp. 176-183*

[47] K. Swearingen, R. Sinha, “Interaction Design for Recommender Systems”, *Interactive Systems (DIS2002), London, June 25-28, 2002*

[48] R. Sinha, K. Swearingen, “Comparing Recommendations made by Online Systems and Friends”, *Proc. of DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries*

[49] S. Baumann, O. Hummel, “Enhancing Music Recommendation Algorithms Using Cultural Metadata”, *Journal of New Music Research 2005, pp. 161-172*

[50] Alfarez Abdul-Rahman and Stephen Hailes, “A distributed trust model”, *New Security Paradigms 1997, pp. 48-60*

[51] Jennifer Golbeck, James Hendler, “Accuracy of metrics for inferring trust and reputation in Semantic Web based social networks”, *Proc. of EKAW ’04*

[52] Cai-Nicolas Ziegler, Georg Lausen, “Analyzing Correlation Between Trust and User Similarity in Online Communities”, *Proc. of Second International Conference on Trust Management, 2004*

[53] P. Massa, B.Bhattacharjee, "Using Trust in Recommender Systems: an Experimental Analysis", *Proc. of iTrust2004 International Conference*

[54] J. Coleman, "Foundations of Social Theory", *Harvard University Press, 1990*

[55] B.Misztal, "Trust in Modern Societies: The Search for the Bases of Social Order", *Polity Press, 1996*

[56] P. Sztompka, "Trust: A Sociological Theory", *Cambridge University Press, 1999*

[57] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication", *ACM Transactions on Computer Systems, pp. 18-36, 1990*

[58] U. Frendrup, H. Huttel, and J.N. Jensen, "Modal Logics for Cryptographic Processes", *Electronic Notes in Theoretical Computer Science, 2002*

[59] G. Ackerlof, "The market for lemons: Quality uncertainty and the market mechanism" *Quarterly Journal of Economics, pp. 488-500, 1970*

[60] P. Kollock, "The production of trust in online markets", *Advances in Group Processes, vol. 16, pp. 99-123, JAI Press, 1999*

[61] Epinions, http://www.epinions.com/, Last accessed on May 2010.

[62] L. Page, S. Brin, R. Motwani and T. Winograd, "The Pagerank citation ranking: Bringing order to the web", *Technical Report, Stanford, USA, 1998*

[63] Badrul Sarwar, George Karypis, Joseph Konstan, John Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms", *ACM, May 1-5, 2001, Hong Kong*

[64] D.M. Pennock and D.Billsus, “Collaborative Filtering by Personality Diagnosis: A Hybrid Memory and Model Based Approach”, *Proc. Int’l Joint Conf. Artificial Intelligence Workshop: Machine Learning for Information Filtering, Aug, 1999*

[65] Michael W. Berry, Susan T. Dumais and Gavin W. O’Brien, “Using linear algebra for intelligent information retrieval”, *SIAM Review, vol. 37, no.5, pp 513-523, 1988*

[66] I. Soboroff, C. Nicholas, “Combining Content and Collaboration in Text Filtering”, *Proc. Int’l Joint Conf. Artificial Intelligence, Workshop: Machine Learning for Information Filtering, August, 1999*

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

[68] A. Ansari, S. Essengaier, R. Kohli, “Internet Recommendations Systems”, *J. Marketing Research pp. 363-375, August 2000*

[69] M. Condliff, D. Lewis, D. Madigan, C. Posse, “Bayesian Mixed Effects Models for Recommender Systems”, *Proc. of ACM SIGIR ’99 Workshop Recommender Systems: Algorithms and Evaluation, August 1999*

[70] R. Burke, “Knowledge-Based Recommender Systems”, *Encyclopedia of Library and Information Systems, vol. 69, 2000*