AN ENHANCED HYBRID RECOMMENDATION SYSTEM BASED ON DIMENSIONALITY REDUCTION TECHNIQUES

TO THI THUAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN COMPUTING IN ENGINEERING SYSTEM

INTERNATIONAL COLLEGE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2015

KMITL-2014-IC-M-011-006

AN ENHANCED HYBRID RECOMMENDATION SYSTEM BASED ON DIMENSIONALITY REDUCTION TECHNIQUES

TO THI THUAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN COMPUTING IN ENGINEERING SYSTEM

INTERNATIONAL COLLEGE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2015

KMITL-2014-IC-M-011-006

Thesis Title	An Enhanced Hybrid Recommendation System Based on Dimensionality Reduction Techniques
Student	To Thi Thuan
Student ID	56610019
Degree	Master of Engineering
Program	Computing in Engineering System
Thesis Advisor	Asst. Prof. Dr. Sutheera Puntheeranurak

ABSTRACT

This thesis proposes a novel method based on dimensionality reduction techniques to enhance accurate prediction and reduce the computation cost of a hybrid recommendation algorithm. The aim of using dimensional reduction techniques is to extract off-line three novel factors: users' latent relationships; review helpfulness feature and review helpfulness trust, which are used in the on-line process of a hybrid recommendation system. Users' latent relationships weight, which is extracted and measured from users' preference by using column-sampling approximating singular value decomposition algorithm, are used in the predicted rating process. Review helpfulness feature and review helpfulness trust, which are extracted from review rating data, are used to build the hybrid model. The proposed method can deal with sparse and scalability problems to improve the prediction accuracy with low time complexity of the recommendation algorithm. The experiments use two datasets: Movielens dataset and Epinions.com dataset. The experiment is divided into three parts, corresponding three factors that influence the utilization of recommendation system. The experimental results show the significantly preferable efficiency of our proposed method, in terms of accuracy and low computation cost

ACKNOWLEDGEMENTS

This thesis is the outcome of my 2 years work and completed with supporting from many people and resources.

First of all, my sincere thanks to Assistant Professor Dr. Sutheera Puntheeranurak for her dedication to her student and patience in assisting me with this thesis. I appreciate her valuable advices and effort during my work. In addition, I would also like to thank my co-advisor Professor Dr. Kazuhiko Hamamoto from Tokai University for his useful lectures and suggestions for my research.

Special thanks for my scholarship from Japan International Cooperation Agency (JICA) under AUN/SEEDNet project for sponsoring me to study at King Mongkut's Institute of Technology Ladkrabang (KMITL), and the research Fund of King Mongkut's Institute of Technology Ladkrabang funding under the contract KREF- 125611.

Special thanks lecturers and staff in International College in KMITL as well with their lectures and co-operation during my study period. Besides, I would also like to thank my lab mate and other friends in KMITL, I appreciate their friendship and sympathetic help which has made my life here much easier.

Lastly, I would like to thank my parents and my family in Vietnam for their enormous encouragement and assistance. Without them, my work here would be impossible.

Bangkok, Thailand

2015

To Thi Thuan

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Background

The World Wide Web has an enormous amount of information. In January 2005 the number of pages in the publicly indexable web was exceeding 11.5 billion [1]. Recently, statistics also show that the number of Internet users is high and rapidly growing. Statistics from June 30th, 2014 shows that 42,3% of the world's population uses the Internet and that the number of users has grown with over 741% from 2000 to 2014 [2]. The tremendous growth of both information and usage has led to the information overload problem in which users are finding it increasingly difficult to find the right information at the right time [3]. As a response to this problem, many researches have done with the goal of providing users personalized information services and help users to find information quickly and accurately.

recommendation systems have proven to achieve this goal by using the opinions of users to help people in the community more effectively identify the content of interest from a potentially overwhelming set of choices [4]. These systems play an important role within interactive Media. Once a registered user performs an action, it is recorded, and a specific profile is being built up. Based upon this profile, the recommendation system builds, personalized recommendations for that user by predicting how that user would like. Currently, there are three main recommendation algorithms: the content-based algorithms [5], the collaborative filtering algorithms [6] and the hybrid algorithms [7].

The content-based algorithms [5] create a recommendation result by analyzing the textual information and finding regularities in the content. These representations of the items in the user profile are subsequently compared to possibly recommended items. If they matched with each other, the tested item would be recommended to the user. The collaborative filtering algorithms [6] analyze the preferences of users who have in the system to make predictions for the other users. The system recommends one certain item by matching the user's profile with profiles of other users. Then the system recommends items similar users liked before. Depending on whether an item can be used several times, an item will also be recommending several times. The hybrid algorithms [7] combine two or more other recommendation techniques together. In typical, the hybrid

combined between the content-based algorithms and the collaborative filtering algorithms which make much more effective than separate them.

recommendation systems have also been deployed within commercial domains, such as in e-commerce applications. A well-known example is Amazon, where a recommendation system is used to help people find items they would like to purchase. Many movie online communities use recommendation systems to gather user opinions on movies, and then produce recommendations based on these opinions. Examples are Netflix and MovieLens. New popular music services like Pandora and Last.fm also make use of recommendations to configure personalized music players.

These systems are used in a variety of contexts. These contexts range from online shops to personalized media streams. recommendation systems can be found in classical non-mobile appliances and lately also in mobile devices. In general, a recommendation system can be used in every context where a user is to choose among several different options. The system assists the user to select one good-fitting item.

1.2 Motivation

Collaborative filtering is the most concerned method in recommendation system. It gets a high accuracy of prediction if the user's product rating information is complete. However, in practical applications, because sparse data or new products do not have enough rating data, it will lead to poor results and cause sparse and cold start problems. Dimensionality reduction techniques are the most powerful techniques which used in a recommendation system to overcome the sparsity problem.

Singular value decomposition (SVD) is one of the most popular methods of dimensionality reduction techniques. For example, Sarwar et al. (2000) proposed the dimension reduction technique and singular value decomposition technique to reduce a sparse rating matrix dimension and eliminate rating matrix sparsity [8]. Reference [9] proposed a personalized recommendation algorithm based on approximating the singular value decomposition. SVD enables us to get the true dimensionality of data. However, these methods do not get a high accurate prediction.

Motivated by this, we decided to develop a novel method by combining dimensionality reduction technique and collaborative filtering algorithm. The proposed method aims to deal with sparse and scalability problem. Besides, it also improves the recommendation system performance by obtaining lower time complexity of the proposed algorithm. The dimensionality reduction technique is used to extract meaningful information about the user's preferences, called users' latent relationships. And then, the predicted result is calculated by implementing collaborative filtering algorithm on the extracted information.

Moreover, we aim to look for new factors to improve the prediction accuracy as well as the hybrid recommendation system performance. Review rating is researched and applied in this thesis. Based on the consumers' behavior that they always consider about reviews of the other consumers for a product besides product rating information. Besides, there are many attackers who aim to fake information of product rating. It causes the inefficiency system. Review rating is added to the product information to deal with this problem. Review rating determines which reviews are helpful. It plays an important role in the consumer's decision. The proposed method uses review rating to measure review helpfulness feature and review helpfulness trust. These factors are used to construct a hybrid model in a recommendation system.

1.3 Problem statement

This thesis aims to enhance accurate prediction and reduce time complexity of a recommendation algorithm. Different approaches for computing recommendations are designed, implemented and tested. The proposed method is implemented and evaluated to prove that a recommendation system can be enhanced by employing three novel factors: users' latent relationships weighting, review helpfulness features and review helpfulness trust. These novel factors are extracted by dimensional reduction techniques in offline phase and then they are used to make predictions in online phase.

1.4 Objective

The main objective of this thesis is to research a novel method to deal with sparse and scalability problem of a recommendation system. A novel method is proposed to deal

with these problems and enhance accuracy as well as performance of a recommendation system. In order to reach this goal, the dimensional reduction techniques are used to extract offline three factors: users' latent relationships weight; review helpfulness feature and review helpfulness trust. These factors are used in the online predicted rating process of a hybrid recommendation system, corresponding three parts of the proposed method. Fig 1.1 shows an overview workflow of the proposed method. The users' latent relationships weighting utilization is extracted based on the combination of the singular value decomposition algorithm and the collaborative filtering algorithm. The review helpfulness feature and the review helpfulness trust are extracted based on the employing review rating. Furthermore, the comparisons between methods on the same dataset are also presented.

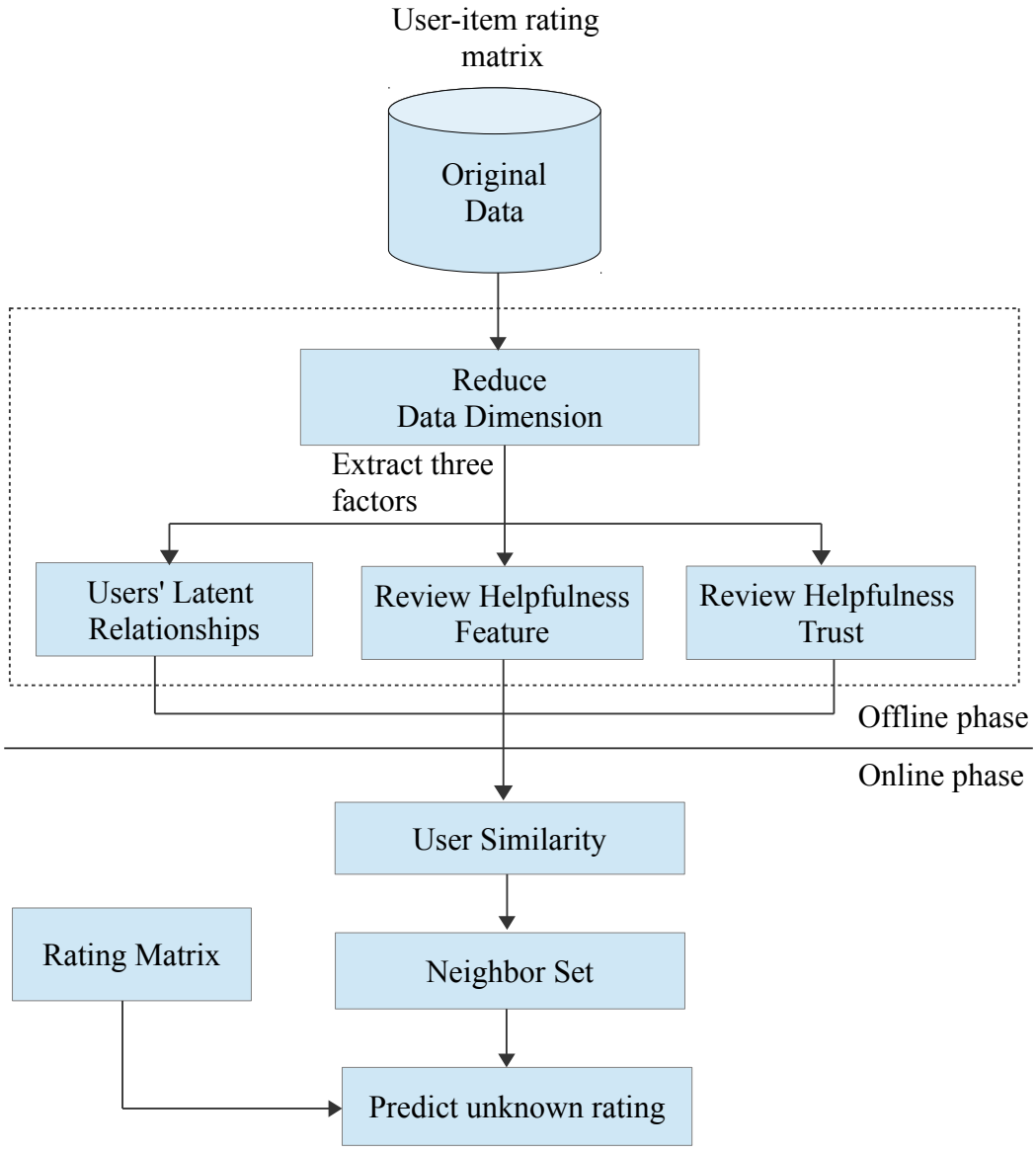

Figure 1.1: Overview workflow of the proposed method

1.5 Thesis structure

The thesis consists of the following chapters:

Chapter 2 – Literature review: introduces an overview of a recommendation system with three main methods: collaborative filtering; content-based and hybrid approaches. The strength and weaknesses of these approaches are also described. Besides, This chapter describes the use of the dimensional reduction techniques to resolve the weaknesses of a recommendation system and its motivation in this thesis.

Chapter 3 – The first part of the proposed method: proposes and implements the first part of the proposed method to enhance accuracy and performance of a recommendation system by using users' latent relationships weighting. This chapter introduces how users' latent relationships are extracted and applied in a recommendation system by using dimensionality reduction technique. This chapter also shows the evaluations of the proposed method comparing with the other existing methods and the discussions.

Chapter 4 – The second and the third part of the proposed method: proposed and implements the second and the third part of the proposed method to enhance accuracy and performance of a recommendation system by using review helpfulness features and review helpfulness trust. This chapter presents how review helpfulness features and review helpfulness trust are measured. It also shows the design of the proposed method by using a hybrid model to utilize the review helpfulness features and review helpfulness trust in it. The evaluation of the proposed method is compared and discussed with other existing methods.

Chapter 5 – Conclusion and future work: compares three parts of the proposed method and concludes the thesis with over all discussions as well as future research.

In addition to the chapters, the appendix contains additional information that is relevant to the thesis.

Appendix A: list of papers published

CHAPTER 2 LITERATURE REVIEW

2.1 Recommendation system

Recommendation systems are widely used in e-commerce applications. They attempt to predict unrated items for a particular user [10]. So, upon the predicted ratings, the system will be able to recommend items to the users. Many corporations such as Amazon or Netflix apply recommendation systems to recommend items or products to their customers.

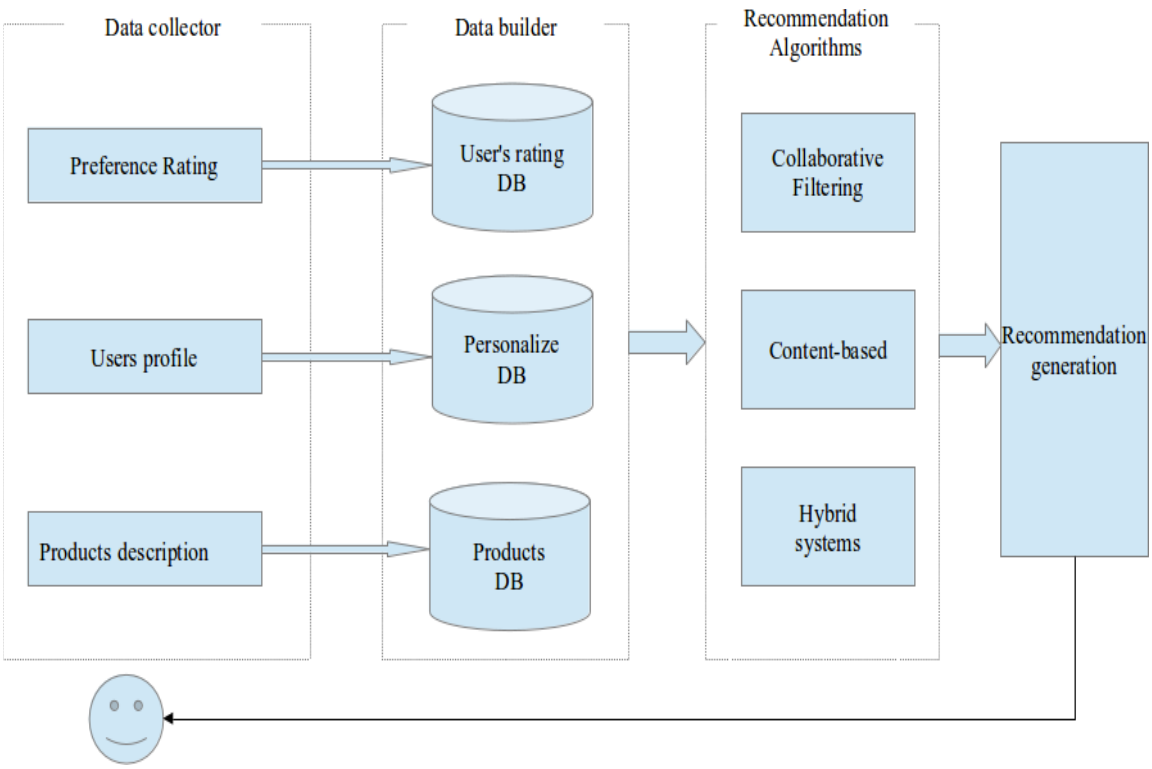

Figure 2.1: Basic architecture of a recommendation system

Figure 2.1 shows a basic architecture of a recommendation system. Firstly, it collects data from some different sources like users profile; product information or user's preferences; … so on. After that, these data are analyzed in the corresponding database. Finally, these databases are processed by different algorithms to generate the recommendation for the target users. Recommendation algorithms are the core of a recommendation system. There are three main kinds of recommendation algorithms: collaborative filtering algorithms; content-based algorithms and hybrid algorithms. These

algorithms will be introduced in detail the next sections.

Aim to provide more formal definition of a recommendation algorithm, let U be a set of all possible users, and let I be a set of all possible items. In many e-commerce applications, the space U and I can be very large. Let f be a utility function that measures the usefulness of item i to a user u such as $U\,x\,I \rightarrow R$ where R is an ordered set of non-negative integers or real numbers. Then, for each user $u \in U$, we want to choose an item $i_u \in I$ to maximize the user's utility as shown below [8]:

$$\forall u \in U, i_u = arg\,max\, f(u, i) \tag{2.1}$$

In the context of recommendation systems, the utility of an item is usually represented by a rating. For example, James gave the movie Spider Man a rating of 5. Basically, the utility of an item indicates how a particular user liked a particular item [10]. Table 2.1.1 shows an example of (User × Item) rating matrix:

Table 2.1: User x Item rating matrix

User/ Item	Spider Man	Star Wars	Toy Story	Titanic	Avatar
Jame	5	4	2	5	-
Hana	3	-	5	-	4
Michale	4	-	4	4	4
Anna	2	4	4	-	-

The table above shows the ratings for each movie that the users have watched. "-" indicates the movie has not been yet rated by the user. Therefore, the goal of recommendation systems is to predict unrated items. Based on that predicted rating, the recommendation systems will be able to select some items with highest predicted ratings and recommend them to the user.

All most commercial websites use recommendation system to improve the quality of customer support services. For example, some most popular commercial websites like Amazon.com[1], Last.fm[2] or Epinions.com[3]. The media store Amazon[1] is one of the biggest e-commerce sites in the entire world. Amazon sells goods like books, CDs or DVDs. Amazon has one of the best-known recommendation systems that are currently in use. It usually serves as an example for non-experts on what a recommendation system is

like. Amazon's recommendation system uses collaborative filtering based on the items a customer purchases before. Furthermore, the user can rate any item on a graphic 5-star rating scale.

Last.fm[2] is a social music platform offering personalized radio streams to its users. Last.fm recommendation system is also based on collaborative filtering, but also incorporates tagging functions into its service to generate meta-data for the tracks. Tags are usually used to describe a genre, mood or artist characteristics, but any other classifications are used, too.

Epinions[3] is a service of the eBay Commerce Network, a leading provider of comparison shopping services. The Epinions is a premier consumer reviews platform on the Web and a reliable source for valuable consumer insight, unbiased advice, in-depth product evaluations and personalized recommendations. The Epinions recommendation system is a trust-aware recommendation system which is created by using trust networks among users.

Besides, a famous non-commercial personalized recommendation system is Movielens[4]. It is a website that helps people find movies to watch. MovieLens is run by GroupLens[5], a research lab at the University of Minnesota. Based on movie ratings, MovieLens generates personalized predictions for movies which have not seen yet. The purpose of this website maintaines a database for students and researchers researching various aspects of personalization and filtering technologies.

2.2 Content-based algorithms

The content-based algorithms [5] select items based on the correlation between the content and the user's preference. Content-based recommendation systems [11, 12, 13] may be used in a variety of domains ranging from recommending web pages, news

[1] http://www.amazon.com/

[2] http://www.last.fm/

[3] http://www.epinions.com

[4] https://movielens.org/

[5] http://grouplens.org/

articles, restaurants, television programs, and items for sale based on information retrieval techniques [14, 15]; association rules [16, 17]; machine learning techniques [18, 19, 20, 21] ; etc .... The main disadvantage of the content-based recommendation system is that, user might miss the opportunity to find out new and interesting things because of "Lack of diversity". These approaches will be described further, and their strengths and limitations will be addressed.

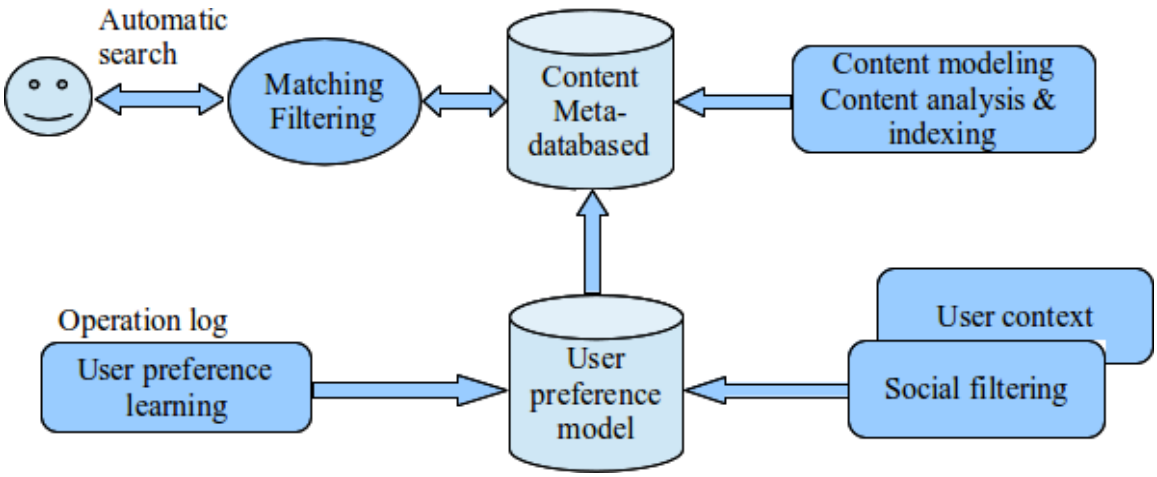

Figure 2.2: The content based recommendation process

Figure 2.2 shows a general content-based recommendation process. Firstly, the user preference model is created by combining user preference and user context. User preference is learned from user's operation log. User context is user profile which was collected or analyzed from a social trend. After this, contend meta-database is built which matches the user preference and content to measure the "similarity" between user's preference and content. When recommendations are produced, the preferences stored in meta-database is compared against the features of the items stored in the system, and the items of which the features are most similar to the user's content-based preferences are retrieved and presented as recommended content to the user.

In content-based recommendation systems, items are recommended according to a comparison between their content or also under the name of item profiles and user profiles, which contain information about the users' tastes, interests, and needs. Data

structures for both these components are created using extracted features from the content of the items. The profiling information can be obtained from users explicitly, for example through manual ratings, or implicitly learned from their transactional behavior in the system over time.

The user profile is a profile of a user which contains the preferences of this user. User profiles can be defined as a vector of weights $(w_{i1}, w_{i2}, ..., w_{ik})$, where each weight w_{ui} represents the importance of the keyword k_i to the user u. Item profiles are attributes or features of an item. They are usually automatically extracted from the item's content. The content of an item is usually described with keywords. Table 2.1 shows an example of item profiles, which is a profile containing a set of attributes that describes an item.

Table 2.2: An example of item profile

Title	Genre	Author	Type	Price	Keywords
The night of the gun	Memoir	David Carr	Paperback	29.9	Press and journalism, drug addiction, personal memoirs, New York
The lace reader	Fiction, Mystery	Brunonia Barry	Dardcover	49.9	American contemporary fiction, detective, historical
Into the fire	Romance, suspense	Suzanne Brockmann	Hardcover	45.9	American fiction, murder, neo-nazism

The utility of item i_n for the user u_m can be defined as a score function $f(u,i)$ below [8]:

$$f(u,i) = score(userprofile(u), itemprofile(i)) \tag{2.2}$$

The score function combines the different item description and user profile components. There are many techniques to obtain utility prediction function. They are distinguished into the different content-based recommendation techniques. These techniques can be classified into two main approaches: heuristic-based and model-based approaches. The first ones calculate utility predictions based on heuristic formulas that

are inspired mostly on information retrieval methods, such as the cosine similarity measure. The second ones obtain utility predictions based on a model learned from the underlying data using statistical learning and machine learning models, such as Bayesian classifiers, clustering algorithms, decision trees, and artificial neural networks. Below, the detail of the two main approaches is presented.

2.2.1 Heuristic-based

Heuristics is based mostly on Information Retrieval (IR) methods. The heuristic-based approach makes predictions based on the local neighbor of the active user, or can base their predictions on the similarities between items. So, the main task of heuristic-based approach is to compute the similarity of an unseen item with the user profile based on the keyword, called the specifying keyword weight.

The term frequency/inverse document frequency (TF-IDF) measure is one of the best measures used for specifying keyword weight [10]. This measure is defined as follows: let N be the total number of documents that can be recommended to the users, and let k_i be the keyword that appears in n_i of the documents. Also, let $f_{i,j}$ be the number of times the keyword k_i is mentioned in the text of the document d_j. Then, $TF_{i,j}$ the term frequency of keyword k_i in document d_j is defined as:

$$TF_{i,j} = \frac{f_{i,j}}{max_z f_{z,j}} \tag{2.3}$$

where the maximum is computed over all keywords k_i which are mentioned in the text of the document d_j . If the keyword k_i does not appear in the document d_j , then $f_{z,j} = 0$.

The measure $TF_{i,j}$ gives more relevance to those keywords that appear more times in a specific document. However, the keyword can appear in many documents. Aim tends to be less useful to distinguish between a relevant document and a non-relevant one, the measure $TF_{i,j}$ is usually used in combination with the so-called inverse document frequency, $IDF_{i,j}$, is defined as:

$$IDF_i = \log \frac{N}{n_i} \tag{2.4}$$

Then, the TF-IDF weight for keyword k_i in document d_j is defined as :

$$w_{i,j} = TF_{i,j} \times IDF_i \tag{2.5}$$

The content of document d_j is defined as :

$$content(d_j) = (w_{1,j}, w_{2,j}, ..., w_{k,j}) \tag{2.6}$$

The utility function of document d_j to the user u_m is evaluated as the correlation between the vectors $d_j = content(d_i)$ and $u_m = userprofile(u_m)$ by the cosine of the angle between the vectors as:

$$f(u_m, d_j) = \cos(u_m, d_j) = \frac{u_m . d_j}{\|u_m\| . \|d_j\|} \tag{2.7}$$

A content-based message recommendation system was proposed by Krulwich & Burkey in 1997 [22], called InfoFinder, that learns user information interests from sets of messages, and other online documents that users have classified. Specifically, the system utilizes user-classified documents to build a search query strings for each of their personal categories, and executes these queries to regularly recommend users those new documents that match them. In order to build such queries, InfoFinder extracts semantically significant topic phrases from each document using several heuristics based on visually significant features, builds a decision tree with the identified phrases, and transforms the resulting decision tree into Boolean queries.

A personal news agent was introduced by Pazzani & Billsus [23, 24] in 1999-2000 that uses synthesized speech to read news stories to a user, called News Dude. These stories are recommended to the user according to separate models for short-term and long-term interests. User preferences are obtained by taking into account not only the user's ratings, but also the time they spent listening to the rated news readings. To determine the short-term recommendations, news stories are described in terms of TF-IDF vectors, which are compared with the cosine similarity measure, and are supplied to

a learning module based on the Nearest Neighbors (NN) algorithm. On the other hand, to establish the long-term recommendations, news stories are represented as Boolean feature vectors, where each feature indicates the presence or absence of a word, and are presented to a Bayesian learning module. According to the previous types of interests, the system also gives the user different explanations about the given recommendations.

2.2.2 Model-based

Model-based algorithms make predictions by first developing a model of the user ratings. The model building process is performed by different Machine Learning (ML) algorithms such as Bayesian networks, neural networks, clustering, and rule-based approaches. These techniques compute the weight of each item in the user's profile which denotes the importance of keyword k_i to the user.

Bayesian technique is worth being mentioned in many classification domains [10]. It is shown to be competitive with other complex approaches, especially in text categorization tasks. This technique can be used to classify unrated items into two classes C_1 (relevant) and C_2 (irrelevant). For example, in an article recommendation system using a content approach, the user profile contains preferences of the user such as the user is interested in business articles. These articles have terms such as market, stock, business, money, etc. The system computes the weight of the terms using methods that are mentioned in the item profile section. Each article is represented as a vector, and each vector contains such terms with their respective weights. In order to recommend an article to the user, the system uses the similarity measure such as cosine similarity or classification techniques such as a Naive Bayesian classifier to recommend an article with high weight for the user.

Making the “naive” assumption that features are independent given the class label. Naive Bayesian classifier is used to estimate the probability that an article belongs to a certain class C_1 (relevant) or C_2 (irrelevant) by giving a set of keywords $k_{1,j}, k_{2,j}, \ldots, k_{n,j}$ for that article:

$$P\left(C_i | k_{1,j}, k_{2,j} \ldots, k_{n,j}\right) \tag{2.8}$$

The probability of each class is computed to determine the most likely class of an example, and the example is assigned to the class with the highest probability. Although the assumption that features are independent once we know the class label of an item is not realistic in this domain, the Bayesian classifier has been shown to be optimal in many situations where this assumption does not hold and has been empirically shown to be competitive with more complex approaches in many others. Moreover, the Bayesian classifier is fast because its learning time is linear in the number of examples in the training data.

The utility function of the document d_j to the user u_m is defined as the ratio:

$$f(u_m, d_j) = \frac{P(C_1|k_{1,j}, k_{2,j} \ldots, k_{n,j})}{P(C_2|k_{1,j}, k_{2,j} \ldots, k_{n,j})} \tag{2.9}$$

Using the Bayes' rule:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \tag{2.10}$$

Equation (2.9) is transformed into:

$$f(u_m, d_j) = \frac{P(k_{1,j}, \ldots, k_{n,j}|C_1) \times P(C_1)}{P(k_{1,j}, \ldots, k_{n,j}|C_2) \times P(C_2)} \tag{2.11}$$

$P(C_1)$ and $P(C_2)$ are the same for all items in an article, so the expression (2.9) can be rewrite as:

$$f(u_m, d_j) \sim \frac{P(k_{1,j}, \ldots, k_{n,j}|C_1)}{P(k_{1,j}, \ldots, k_{n,j}|C_2)} \tag{2.12}$$

In [25], a Netnews filtering system called NewsWeeder that describes an article with a vector in which each component contains the number of occurrences a specific term appearing in the article. The system lets users rate their interest levels for the read articles in a 1-5 scale, and then learns their user profiles based on these ratings. Specifically, the system implements a Bayesian learning strategy based on the minimum description length principle, which takes into account a trade-off involving how to weight each term's importance, and how to decide which terms should be left out of the model

for not having enough discriminating power.

2.2.3 Limitations of Content-Based Approaches

Content-based recommendation systems have several limitations, which have been identified in the literature [10, 26, 27] are described as below:

- **Restricted content analysis**: Content-based recommendations are restricted by the features that are explicitly associated with the items to be recommended. For example, content-based movie recommendations can only be based on written materials about a movie: actors' names, plot summaries, genres, etc.

  The effectiveness of these techniques thus depends on the available descriptive data. Therefore, in order to have a sufficient set of features, the content should be either in a form that can be automatically parsed by a computer, or in a form in which the features can be manually extracted in an easy way. In many cases, these requirements are very difficult to fulfill. There are some domains that have an inherent difficulty to do the automatic feature extraction, and it is often not practical to assign features by hand. For instance, it is much harder to apply automatic feature extraction methods to multimedia data such as graphical images, video streams, and audio streams than it is for text content.

  On the other hand, if two items are represented by the same set of features, they are indistinguishable. For instance, since text documents are usually represented by their most important keywords, content-based systems cannot distinguish between a well-written text and a badly written one if they happen to use the same terms.

- **Over-Specialization**: A content-based approach tends to recommend items that are similar to the items rated before by the same user [10]. Content-based recommendation systems only retrieve items that score highly against a specific user profile. Tastes, interests or needs of other users that could enrich the recommendations are not taken into account. The content-based techniques cannot recommend items that are different from anything the user has seen before. Thus, for instance, a person with no experience in Spanish cuisine would

never receive recommendations for even the best Spanish restaurant in town. For example, a user who is interested in business articles will hardly receive a recommendation for an article in sports or technology [26].

- **Cold-start or New User Problem**: A user has to rate a sufficient number of items before a content-based recommendation system can really grasp his preferences, and present him with reliable recommendations. A new user having none or very few ratings may not be suggested any accurate recommendations. Thus, the system will not be able to provide accurate recommendations [10].

- **Portfolio effect or nondiversity problem**: in certain cases, items should not be recommended if they are too similar to something the user has already seen [27].

Table 2.3 summarizes the limitation of content-based recommendation techniques and solutions to address them.

Table 2.3: Common limitations of content-based approaches and possible solutions

Limitations	Possible solutions
Restricted content analysis	• Extract content features of the items through automatic or semi-automatic processes. • Prevent the occurrence of equal content descriptions for different items. • Add additional information based not only in specific content features but also in subjective human judgments (i.e., based on collaborative filtering features).
Over-Specialization	• Introduce some randomness in the recommendations. • Recommend items not directly related to the user profile, for example, considering correlated preferences of those people with similar tastes to the user (i.e., applying collaborative filtering mechanisms).
Cold-start	• Extend user preferences in cases where few ratings had been provided.
Portfolio effect	• Offer diversity in the recommendations according to related user • Filter out items not only if they are too different from the user's preferences, but also if they are too similar to something the user has seen before

2.3 Collaborative filtering approaches

The collaborative filtering algorithm is the most concerned approaches in a recommendation system, in which all users contribute with ratings based on their preferences. In the collaborative filtering, recommendations for the current user are produced by matching the user's ratings with ratings given by other users. The collaborative filtering recommendation algorithm [6] is different from other filtering technologies in that information is filtered by using evaluation instead of analysis, thus categorizing information based on the user's opinion of the information instead of the information itself. In addition, collaborative filtering stresses the concept of community by letting recommendations be a result of the opinions of the current user and other similar users . Recommendations for the current user are produced by matching the user's ratings with ratings given by other users [28, 29, 30]. These approaches will be described further, and their strengths and weaknesses will be addressed.

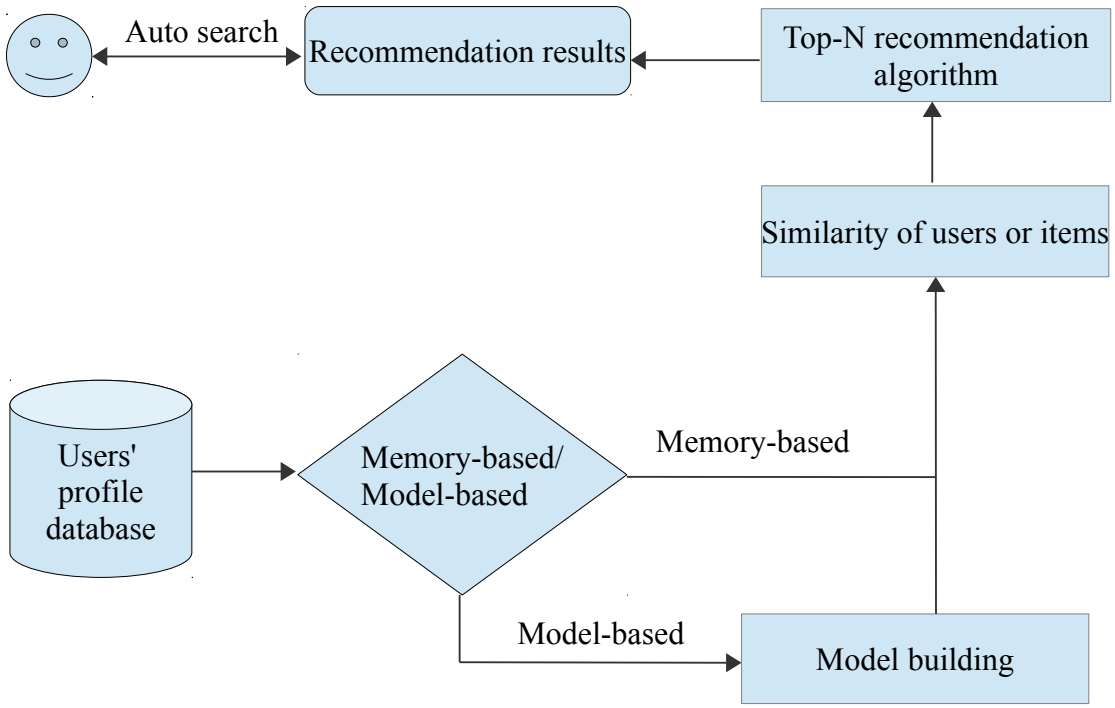

Figure 2.3: The collaborative filtering recommendation process

Figure 2.3 shows the typical collaborative filtering (CF) recommendation process. Collaborative Filtering systems are often classified as memory-based approach or model-based approach. The Memory-based approach makes rating predictions based on the

entire collection of previously rated items by the users. The model-based CF systems use the collection of ratings to learn a model, which is used to make predictions. Both of them use the user similarity or item similarity to predict the rating for the target user. Prediction is done by averaging the weight and the rating of the target user's neighbors by implementing Top-N recommendation algorithm.

2.3.1 Memory-based collaborative filtering

Memory-based approaches use the entire collection of the rated items in order to make recommendations or predictions. They can be divided into two approaches: User-based collaborative filtering approach and Item-based collaborative filtering approach. User-based CF approach compare the active user's ratings with those of other users to identify a group of similar people in such a way that the highest rated items of that group will be recommended to the active user. Item-based CF approaches, on the other hand, take each item of the active user's list of rated items, and recommend other items that seem to be similar to that item according to other users' ratings.

A) User-based collaborative filtering

User-based collaborative filtering is an algorithm based on the following assumption idea: People have similar preferences and interests; their preferences and interests are stable; we can predict their choice according to their past preferences. Because of the above assumptions, the user-based collaborative filtering algorithm is based on the comparison of one user's preference with other user's preference, to find his nearest neighbors, and according to his neighbor's interests or preferences to predict his interests or preferences.

User preferences are captured by observing the users' choices and/or ratings. Each choice or rating is stored in a user profile, creating histories of user in the form of action lists. To generate item suggestions, the recommendation algorithm correlates the target user's list of choices/ratings to the lists of every other user registered in the system, and selects the group of the most highly correlated users (i.e., the most "similar" users). Afterwards, the system creates a list of items chosen/rated by the identified like-minded users, and ranks this list by frequency and/or by rating. The most highly items are finally recommended to the target user. Hence, user-based collaborative filtering can be

implemented following three steps below.

The first step of user-based collaborative filtering algorithm is to obtain the user history profile, which can be represented as a rating matrix with each entry the rate of a user given to an item, for example as Table 2.4.

Table 2.4: User-Item rating matrix

User/ Item	I_1	I_2	I_3	I_4
U_1	4	?	5	5
U_2	4	2	1	-
U_3	3	-	2	4
U_4	4	4	-	-
U_5	2	1	3	5

A user-item rating matrix consists of a table where each row represents a user, each column represents a specific item, and the number at the intersection of a row and a column represents the user's rating value. The absence of a rating score at this intersection indicates that a user has not yet rated the item. The goal of user-based CF algorithm makes use of the entire user-item database to generate a prediction for the target user.

The second step of this approach is to calculate similarity between the target user a and the other users u who have both rated the same items. The computation of similarity aims to find their nearest neighbor set. Once neighbors of target user are formed, these systems use different algorithms to combine the preferences of neighbors to produce a prediction for the active user. There are some similarity measure methods [10] to compute $\text{sim}(a,u)$ such as:

- **Pearson correlation coefficient**: is the most widely used in the collaborative filtering algorithm. The similarity between the two users is measured by computing the Pearson correlation or coefficient-based on their rating vectors:

$$\text{sim}(a,u) = \frac{\sum_{i \in I} \left(r_{a,i} - \overline{r_a}\right)\left(r_{u,i} - \overline{r_u}\right)}{\sqrt{\sum_{i \in I} \left(r_{a,i} - \overline{r_a}\right)^2} \sqrt{\sum_{i \in I} \left(r_{u,i} - \overline{r_u}\right)^2}} \tag{2.13}$$

Where, $i \in I$ summations are over the items that both the users *a* and *u* have rated

$r_{a,i}, r_{u,i}$ are the rating of a user *a* and user *u* on item i, respectively

$\overline{r_a}, \overline{r_u}$ are the average rating of a user *a* and user *u* on all items, respectively.

- **Cosine-based user similarity**: the similarity between the two users *a* and *u* can be treating reach user as their rating vectors $\vec{r_a}, \vec{r_u}$ and computing the cosine of the angle formed by these vectors.

$$\text{sim}(a,u) = \cos(\vec{r_a}, \vec{r_u}) = \frac{\vec{r_a} \cdot \vec{r_u}}{\|\vec{r_a}\| . \|\vec{r_u}\|} = \frac{\sum_{i \in I} r_{a,i} \cdot r_{u,i}}{\sqrt{\sum_{i \in I} r_{a,i}^2} \cdot \sqrt{\sum_{i \in I} r_{u,i}^2}} \quad (2.14)$$

Where, $i \in I$ summations are over the items that both the users a and u have rated.

$r_{a,i}, r_{u,i}$ are the rating of a user a and user u on item i, respectively.

"**.**" denotes the dot-product of the two vectors.

The last step is to predict the items rating for the target user. To obtain predictions or recommendations is the most important step in a collaborative filtering system. User-based top-N recommendation algorithm is used to make predictions, in which, it firstly identify the k most similar users to the target user using the similarity computation results of the previous step. A subset of nearest neighbors (or k most similar users) of target user are chosen based on their similarity to target user. The set of nearest neighbors is a set of users which have the largest similarity with target user. After the k most similar users have been discovered, a weighted average of the ratings is used to generate predictions for the target user. A prediction for the active user *a* on a certain item *i* with k nearest neighbors can be done according to the following formula:

$$P_{a,i} = \overline{r_a} + \frac{\sum_{u=1}^{k} \text{sim}(a,u) * \left(r_{u,i} - \overline{r_u}\right)}{\sum_{u=1}^{h} |\text{sim}(a,u)|} \quad (2.15)$$

Where, a is target user, u is the nearest neighbors of a and i is an item.

r denotes rating of a user on an item and

$\bar{r}$ denotes the average ratings of all user on an item.

$\text{sim}(a,u)$ is the similarity between target user a and neighbor u.

For the simple example in Table 2.4, using the user-based CF algorithm to predict the rating for user U_1 on item I_2 like below:

- First step: calculate similarity between users. In this example, we use the Pearson correlation coefficient to compute user similarity. The similarity computation results are shown in Table 2.5. sim(1,2) denotes the similarity between U_1 and U_2 and it is is calculated as below:

$$\begin{aligned}\text{sim}(1,2) &= \frac{(r_{1,1}-\overline{r_1}).(r_{2,1}-\overline{r_2})+(r_{1,3}-\overline{r_1}).(r_{2,3}-\overline{r_2})}{\sqrt{(r_{1,1}-\overline{r_1})^2+(r_{1,3}-\overline{r_1})^2}.\sqrt{(r_{2,1}-\overline{r_2})^2+(r_{2,3}-\overline{r_2})^2}} \\ &= \frac{(4-4.5).(4-2.5)+(5-4.5).(1-2.5)}{\sqrt{(4-4.5)^2+(5-4.5)^2}.\sqrt{(4-2.5)^2+(1-2.5)^2}} = -0.9778\end{aligned}$$

Table 2.5: Similarity computation results between users

	U_1	U_2	U_3	U_4	U_5
U_1	1	-0.9778	0	0	0.6847
U_2	-0.9778	1	0.6247	0	-0.2411
U_3	0	0.6247	1	0	0.5930
U_4	0	0	0	1	0
U_5	0.6847	-0.2411	0.5930	0	1

- Second step: $P_{1,2}$ denotes the predicted rating of user U_1 on item I_2 and it

is calculated below:

$$P_{1,2} = \overline{r_1} + \frac{\sum_u \text{sim}(1,u) * \left(r_{u,2} - \overline{r_u}\right)}{\sum_u |\text{sim}(1,u)|}$$

$$P_{1,2} = \overline{r_1} + \frac{(r_{2,2} - \overline{r_2}).\text{sim}(1,2) + (r_{4,2} - \overline{r_4}).\text{sim}(1,4) + (r_{5,2} - \overline{r_5}).\text{sim}(1,5)}{|\text{sim}(1,2)| + |\text{sim}(1,4)| + |\text{sim}(1,5)|}$$

$$= 4.67 + \frac{(2-2.33).(-0.9778) + (4-4).0 + (1-2.75).0.6847}{1+0+0.6847}$$

$$= 4.15$$

In [31], a video recommendation system is presented. Upon client/server architecture, the system receives and sends an email to obtain user ratings and to provide video suggestions. User-based collaborative recommendations are shown to the users sorted by predicted ratings, and classified by video categories. The system also provides ranked lists with the most similar users, and gives recommendations to a group of users instead of to individual users.

The GroupLens project [32] is one of the most referenced CF works. Based on client/server architecture, the GroupLens system recommends Usenet news (Netnews) – a high volume discussion list service on the Internet. The short lifetime of Netnews, and the underlying sparsity of the rating matrices are the two main challenges addressed by GroupLens. On the system, users and Netnews are clustered based on the existing new groups, and implicit ratings are computed by measuring the time the users spend reading Netnews, and using "filterbots", i.e., programs that automatically process and rate documents.

B) Item-based collaborative filtering

An item-based collaborative filtering system suggests that a user U who likes item A, then U should be recommended item B if this item is found to be the most similar to item A . Like user-based approach, item-based approach recognizes patterns. However, instead of identifying patterns of similarity between user choices, it recognizes patterns of similarity between the items themselves. User preferences are captured in the same way as in user-based collaborative filtering – by observing the users' choices and/or ratings,

storing that information in user profiles, and creating lists of user actions.

Item-based CF techniques were developed to create recommendation systems with computation, lower costs than those relying on user-based CF. Item-based solutions do not have to inspect databases containing millions of users in real time in order to find users with similar tastes. Instead, they can pre-score items based on their ratings and/or attributes, and then make recommendations without incurring in a high computational load. More specifically, item-based techniques first analyze the user- item matrix to identify relationships between different items, and then use these relationships to indirectly compute recommendations for users.

The recommendations are generated by the following steps: firstly, the system finds similar items to the ones listed in the target user's profile, and then weights each similar item according to the ratings stored in that profile. Similar items can be defined as those which have closely matching attributes, or which have been highly rated by users who also like the items present in the target user's profile. And finally, the items with the highest average ratings are finally recommended to the target user by item-based top-N recommendation algorithm. The nearest neighbor set is identified by the k most similarities between items.

In general terms, item-based collaborative filtering looks at each item in the target user's list of chosen/rated items, and finds other items that seem to be "similar" to that item. There are some methods to measure the item similarity such as below:

- **Cosine-based item similarity**: measures the similarity between two items *i* and *j* by computing the cosine of the angle formed by their corresponding rating vectors $\vec{r}_i, \vec{r}_j$

$$\text{sim}(i, j) = \cos(\vec{r}_i, \vec{r}_j) = \frac{\vec{r}_i . \vec{r}_j}{\|\vec{r}_i\| . \|\vec{r}_j\|} = \frac{\sum_{u \in U} r_{i,u} . r_{j,u}}{\sqrt{\sum_{u \in U} r_{i,u}^2} . \sqrt{\sum_{u \in U} r_{j,u}^2}} \quad (2.16)$$

Where, $u \in U$ is the set of users that have rated both items *i* and *j*

$r_{i,u}$ and $r_{j,u}$ are the ratings of item *i* and item *j* on user *u*, respectively.

- **Correlation-based item similarity**: measures the item similarity between two items *i* and *j* by computing the Pearson correlation coefficient of their rating vectors $\vec{r}_i, \vec{r}_j$

$$\text{sim}(i,j) = \frac{\sum_{u \in U} \left(r_{i,u} - \overline{r_i}\right)\left(r_{j,u} - \overline{r_j}\right)}{\sqrt{\sum_{u \in U} \left(r_{i,u} - \overline{r_i}\right)^2} \sqrt{\sum_{u \in U} \left(r_{j,u} - \overline{r_j}\right)^2}} \quad (2.17)$$

Where, $u \in U$ is the set of users that have rated both items *i* and *j*

$r_{i,u}$ and $r_{j,u}$ are the ratings of item *i* and item *j* on user *u*, respectively

$\overline{r_i}, \overline{r_j}$ are the average ratings of item *i* and item *j* on all users, respectively.

- **Adjusted cosine item similarity**: the computation of the cosine-based item similarity (formula 2.16) has one drawback – the differences in rating scale between users are not taken into account. The adjusted cosine item similarity compensates for this by subtracting the corresponding user average rating $\overline{r_u}$ from each co-rated pair of items.

$$\text{sim}(i,j) = \frac{\sum_{u \in U} \left(r_{i,u} - \overline{r_u}\right)\left(r_{j,u} - \overline{r_u}\right)}{\sqrt{\sum_{u \in U} \left(r_{i,u} - \overline{r_u}\right)^2} \sqrt{\sum_{u \in U} \left(r_{j,u} - \overline{r_u}\right)^2}} \quad (2.18)$$

Where, $u \in U$ is the set of users that have rated both items *i* and *j*

$r_{i,u}$ and $r_{j,u}$ are the rating of item *i* and item *j* on user *u*, respectively

$\overline{r_u}$ is the average rating of user *u*.

Amazon.com is one of the famous e-commerce, which uses a recommendation system to suggest customer, shown in Figure 2.4. The first collaborative filtering systems reported in the literature followed a user-based approach. More recently, item-based collaborative filtering has gained momentum over the last years by virtue of

computational improvements in basic prediction algorithms. For cases where the number of users is much greater than the number of items, item-based CF computational performance has been shown to be superior in practice to user-based CF. Its success also extends to Amason.com commercial recommendation systems.

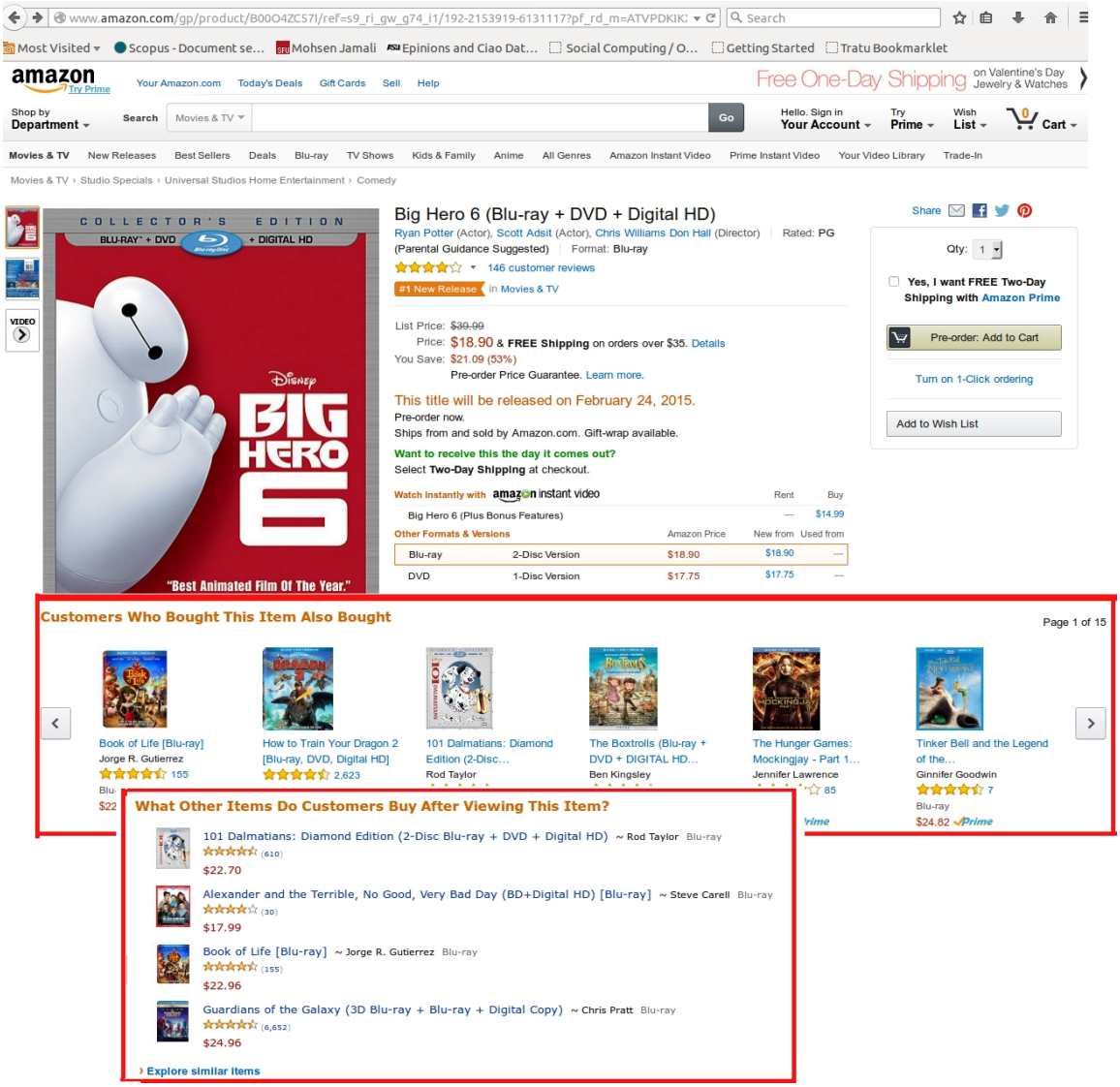

Figure 2.4: Amazon.com collaborative recommendation

2.3.2 Model-based collaborative filtering

The motivation behind model-based CF is that by building a model that reflects user preferences, some of the problems related to memory-based CF might be solved. This can be done by first compiling the complete data set into a descriptive model of users, items and ratings. This model can be built offline over several hours or days. Recommendations can then be computed by consulting the model.

The model- based approach uses the similarity of items to predict the rating of an item, instead of using the similarity of users. In model-based approach, prediction is done by averaging the ratings of similar items rated by the specific user [33]. Sorting is done according to dissimilarity, as in memory-based CF. The difference is that the column vectors (items) are sorted toward the specific item, and not as in memory- based CF, where row vectors are sorted toward the specific user. Sorting of the column vectors assures that the ratings for more similar items are weighted stronger.

The design and development of models (such as machine learning) can allow the system learn to recognize complex patterns based on the training data, and then make intelligent predictions for the collaborative filtering tasks for test data or real-world data, based on the learned models, clustering models, and dependency networks, have been investigated to solve the shortcoming of memory-based CF algorithms. Usually, classification algorithms can be used as CF models if the user ratings are categorical and regression models.

Model-based CF has several advantages over memory-based CF. First, the model-based approach may offer added values beyond its predictive capabilities, by highlighting certain correlations in the data. Second, memory requirements for the model are normally less than for storing the whole database. Third, predictions can be calculated quickly once the model is generated although the time complexity to compile the data into a model may be prohibitive, and adding one new data point may require a full recompilation. The resulting model of model-based CF systems is usually very small, fast and essentially as accurate as memory-based methods. Model-based methods may prove practical for environments in which user preferences change slowly with respect to the

time needed to build the model. Model-based methods, however, are not suitable for environments in which user preference models must be updated rapidly or frequently.

A Ringo recommendation system is proposed in [34] using collaborative filtering technique which makes recommendations of music albums and artists. One remarkable characteristic of Ringo is its initial user profile definition phase. When a user first enters the system, he is presented a list of 125 artists. The user rates those artists according to how much he likes listening to them. The list is formed in two parts. The first one is built on the most often rated artists, ensuring that the new user has the opportunity to rate artists which others have also rated so that there is some commonality in people's profiles. The second one is generated upon a random selection of items from the entire database, so that all artists and albums eventually end up getting a score at some point in the initial rating phases.

2.3.3 Limitations of collaborative filtering Approaches

Collaborative filtering recommendation systems have several limitations, which have been described as below:

- **New user or Cold star problem**: Collaborative filtering has the same problem as the content-based approach which is new users entering the system. In order to make recommendations to a user, the system needs to know the user's preferences from the ratings that the user makes. Since the new user is in the system, she/he has not rated items yet. Thus, the system will not be able to provide accurate recommendations [6, 10].
- **New item problem**: The systems should contain rated items in order to recommend some items to the users. When a new item enters the systems, the item has not rated by users yet. Collaborative filtering systems only rely on users' preferences to make recommendations and do not make use of content information of the existing items. Thus, until a new item is rated by a substantial number of users, the recommendation system is not able to recommend it. Hence, a recent item that has not yet obtained many ratings cannot be easily recommended. Therefore, the systems will not be able to recommend it to the users [6].

- **Sparsity**: Sparse is a major problem for the collaborative filtering approach. The total number of ratings is important in the recommendation system. In order to provide accurate recommendations by the recommendation systems, sufficient number of ratings should exist in the systems. For example, in movie recommending systems, there are many movies that have been rated by only a few people. The systems will rarely recommend these movies. The utility matrix (User × Item) that is used in the collaborative filtering approach will be sparse. Thus, providing accurate recommendations is challenging [6, 10].

- **Scalability**: In many practical collaborative filtering recommendation systems, the number of users and items increase rapidly in the system. Therefore, the system needs to provide more and complicated computational process, and this leads the computational resources going beyond the acceptable levels [6].

Table 2.6: Limitation of collaborative filtering approaches and possible solutions

Limitations	Possible solutions
New user or Cold star problem	• Use a hybrid recommendation technique combining content-based and collaborative information. • Attempt to determine the best items for a new user to rate, using information about item popularity, item entropy, user personalization, and combinations of the above.
New item problem	• Use a hybrid recommendation approach that considers both content-based and collaborative information during the recommendation processes
Sparsity	• Exploit user profile information when calculating user similarities. For example, two users could be considered similar not only if they rated the same items similarly, but also if they belong to the same demographic segment. • Apply dimensionality reduction techniques, such as Singular Value Decomposition (SVD), to reduce the dimensionality of sparse ratings matrices. • Use associative and inference rules, and related spreading activation algorithms to explore transitive associations among consumers and items.
Scalability	• Apply clustering techniques, such as K-means, to classify users or items into clusters.

Table 2.6 summarizes the limitation of collaborative filtering recommendation techniques

and possible solutions to address them.

2.4 Hybrid recommendation system

Content-based and collaborative filtering approaches have been widely used in commercial and research areas. But, they have many limitations mentioned in the previous sections. Therefore, the hybrid approach has been introduced to avoid the limitations of the content-based and the collaborative filtering approaches [7, 35, 36]. Hybrid recommendation system [7, 37] typically combines Content-based approach and collaborative filtering approach together to eliminate the limitations of pure approaches. Several recommendation systems combine two or more other approaches to gain better performance and eliminate some of the drawbacks of the pure recommendation systems approaches. Such as combining separate recommendation system [35,36]; adding content-based characteristics into the collaborative approach [35]; adding collaborative characteristics to the content-based approach [38]; developing a single unifying recommendation approach [39]. Besides, the hybrid recommendation system can be created by combining the dimensional reduction techniques [9, 40, 41]; classification techniques [42], etc. Some of them will be described further.

2.4.1 Combination of collaborative filtering and content-based approaches

Currently, many recommendation systems combine the collaborative filtering approach with some other approaches such as the content-based approach and the demographic approach. Combining the collaborative filtering and the content-based approaches is mostly used today in the industry. There are different ways to combine the collaborative filtering and the content-based approaches:

- Recommendation systems can be developed by implementing content-based and collaborative filtering methods separately and combining their predictions [43]
- Constructing a general unifying model that incorporates both content-based and collaborative filtering characteristics [39]
- Adding content-based characteristics into the collaborative filtering approach [35]
- Adding collaborative characteristics to the content-based approach [38]

Figure 2.5 shows an example of a hybrid user model recommendation system, which combines the content-based and the collaborative filtering approaches. Users' demographic information, which is added into the collaborative filtering approach, is content characteristics. Those content characteristics were used to build a hybrid user model. After that, the collaborative filtering prediction algorithm was applied to predict unknown ratings.

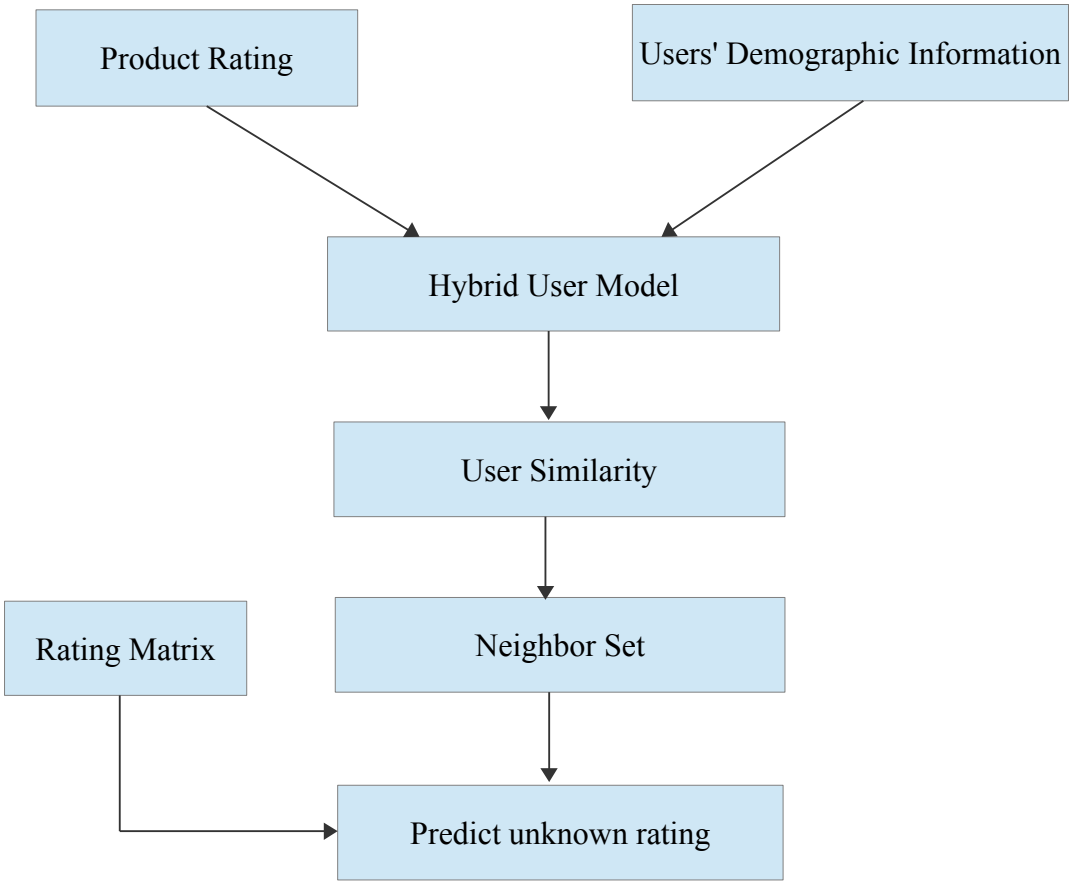

Figure 2.5: An example of hybrid user model recommendation system

In [44], an online newspaper recommendation system is presented, called P-Tango. This hybrid recommendation system combines content-based and collaborative filtering predictions by a weighted average. The content-based and collaborative weights are adjusted to be computed for each user and for each item according to the number of related ratings. Articles are described as a set of keywords and the newspaper sections they belong to. User profiles are divided into sections corresponding to the newspaper sections. Each profile section contains a set of explicit ratings and keywords given by the

user, and a list of implicit keywords which is populated by appending the keywords of the articles to which the user has given a high rating.

2.4.2 Other hybrid approaches

Besides the combination of the collaborative filtering and content-based approaches, hybrid recommendation systems [37] can be classified as:

- **Weighted hybrid recommendation systems**: these systems suggest items with aggregated scores that are computed by combining the results of the individual recommendation techniques to be combined. Those results are usually merged by linear combinations or vote consensus schemes.

  These methods have advantages that the different recommendation capabilities are incorporated in the recommendation process in a straightforward way. However, they have the implicit assumption that the relative value of the different techniques is more or less uniform across the space of items – which is not always true.

- **Switched hybrid recommendation systems**: these systems use some criterion to switch between recommendation techniques.

  These methods have advantages that the suggestions can be sensitive to the strengths and weakness of the constituent recommendation techniques.

- **Mixed hybrid recommendation systems**: these systems suggest the different recommendation techniques.

  These methods have advantages that they directly exploit the benefits of all own recommendation techniques.

- **Cascade hybrid recommendation systems**: these systems involve a staged sequential process: a first recommendation produces a coarse ranking of candidates, a second recommendation starts from the previously filtered list as the set of candidate items, and produces a refined set of final suggestions.

  These methods have advantages that they avoid employing the second, lower-priority technique on items that are well differentiated by the first technique, or

are sufficiently poorly-rated that they will never be recommended. By doing this, cascade recommendation achieve more computationally efficient recommendations than, for example, a weighted hybrid recommendation that has to apply all its techniques to all items.

- **Meta-level hybrid recommendation systems**: these systems combine two recommendation techniques by using the entire model generated by one (not the outputs) as the input for another.

  These methods have advantages that the learned model is a compressed representation of the user's interests, and the second recommendation step that follows can operate on this information-dense space more easily than on the initial raw data.

- **Hybrid recommendation systems based on feature augmentation**: a first recommendation technique produces a rating or classification of each item and then a second recommendation technique exploits the obtained information to enrich the inputs of its recommendation process.

  These methods have advantages that they offer a way to improve the performance of core recommendation techniques, enriching their inputs without modifying their internal model.

Google and Yahoo! are two famous website using a hybrid recommendation system. The performance of existing search engines has been often unsatisfactory in meeting users' information needs due to the enormous amount of returning information, and the fact that not all of these results are relevant or have an acceptable quality. The combination of content-based characteristics and other users' expert knowledge or search experience is a promising avenue for the implementation of a new generation of information retrieval systems. In the following, we describe several recommendation systems that could be considered as the first attempts to achieve the challenges of the so-called social information retrieval. Hybrid recommendation approaches have been mostly tested in experimental systems, and their success is increasingly being demonstrated in commercial applications.

In [26] a hybrid web page recommendation system is introduced, called Fab. In its content-based component, the text documents are represented with their most informative words, and are classified in a number of different topics. Content-based user profiles are defined according to the characteristics of the highest rated web pages for the different topics. The system uses a content-based approach in which items are rated by the user's content-based profile, and the most highly rated items are recommended to the user. This content-based approach together with a collaborative rating mechanism allow identifying emergent Communities of Interest (CoI), whereupon social interactions between like-minded people, are supported, and group as well as individual recommendations are automatically provided.

2.5 Recommendation system based on dimensionality reduction techniques

Dimensionality reduction is one of the most popular techniques which can deal with large sparsity problem of collaborative filtering algorithms. Because, these techniques enable us to get true dimensions of the data. Singular value decomposition (SVD) is one of the most concerned dimensional reduction techniques. An important property of SVD, which is particularly useful in recommendation systems, is that it can provide the best low-rank linear approximation of the original matrix. The SVD can be easily realized. However, due to its expensive computational cost, it has been considered inappropriate for practical applications involving massive data.

Given a user-item rating matrix $Z_{m\times p}$ with rank k, SVD can decompose $Z_{m\times p}$ into the product of three factors:

$$Z = U.S.V^T \tag{2.19}$$

Where, U is a $m\times r$ matrix whose are the orthogonal eigenvectors of ZZ^T

V is a $p\times r$ matrix whose are the orthogonal eigenvectors of $Z^T Z$

S is a $r\times r$ matrix whose is a diagonal matrix with r nonzero elements

Matrix S makes the effective dimensions of these three matrices $m\times r$, $r\times r$ and $p\times r$, respectively. The initial diagonal r elements $(s_{1,} s_{2,} ..., s_r)$ of S

have the property that $s_i \geq 0$ and $s_1 \geq s_2 \geq ... \geq s_r$. Based on the idea that the effect of small eigenvalues (and their eigenvectors) on a matrix-vector product is small, the low-rank approximation Z_d to Z, whose rank is at most d and which minimizes $\|Z - Z_d\|_F$, can be obtained by:

$$Z_d = U_d S_d V_d^T = U_d \sqrt{S_d} (V_d \sqrt{S_d})^T \tag{2.20}$$

Where, U_d is a $m \times d$ matrix whose columns are the first *d* columns of *U*

V_d is a $p \times d$ matrix whose columns are the first *d* columns of *V*

S_d is a $d \times d$ matrix which consists of *d* rows and *d* columns of *S*

The *i*-th row of $U_d \sqrt{S_d}$ can denote the preference model of u_i so that $(U_d \sqrt{S_d})_{ik}$ can represent the preference of user u_i for feature *k*. The *j*-th row of $V_d \sqrt{S_d}$ can denote the proportion model of v_j so that $(V_d \sqrt{S_d})_{jk}$ can represent the proportion of feature *k* on item v_j. Therefore, $(Z_d)_{ij}$ can be used for the predicted rating of user u_i on item v_j

$$(Z_d)_{ij} = (U_d \sqrt{S_d})^i ((V_d \sqrt{S_d})^j)^T \tag{2.21}$$

Input: raw user-item rating matrix $A_{m \times p}$

Output: predicted rating p_{ui} of user *u* on item *i*

- Fill raw rating matrix $A_{m \times p}$ into a dense matrix $B_{m \times p}$
- Normalize $B_{m \times p}$ as the matrix $Z_{m \times p}$ by *Z*-scores

$$Z_{ui} = \frac{B_{ui} - \overline{B^i}}{\sigma_i}, \text{ where } \overline{B^i} = \frac{1}{m} \sum_{u=1}^{m} B_{ui} \text{ and } \sigma^2 = \frac{1}{m-1} \sum_{u=1}^{m} (B_{ui} - \overline{B^i})^2$$

- Apply SVD to *Z*
- Acquire the appropriate low-rank approximation Z_d to *Z*
- Compute predicted rating p_{ui} according to $p_{ui} = \overline{B^i} + \sigma (Z_d)_{ui}$

Figure 2.6: SVD-based prediction algorithm

For example, given raw ratings in Table 2.7 and an active user u_2. Let predict the rating of active user u_2 on item i = 4, p_{24}. The middle value-based filling method, which uses the middle value in the rating range, is used to fill all unknown entries in the matrix. The value of d is set to 2. SVD will predict p_{24} as the following steps.

Table 2.7: Example of raw user-item rating matrix $A_{5\times 8}$

User/ item	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	2	-	5	1	-	3	-	-
u_2	1	3	2	-	4	-	5	-
u_3	-	-	4	2	4	2	-	5
u_4	2	2	4	2	5	-	-	-
u_5	3	-	5	1	3	1	-	4

- The first step: fill the raw rating matrix $A_{5\times 8}$ by the middle value-based filling method. Table 2.8 shows the dense matrix $B_{5\times 8}$

Table 2.8: Dense matrix $B_{5\times 8}$

User/ item	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	2	3	5	1	3	3	3	3
u_2	1	3	2	3	4	3	5	3
u_3	3	3	4	2	4	2	3	5
u_4	2	2	4	2	5	3	3	3
u_5	3	3	5	1	3	1	3	4

- The second step: calculate the mean value $\overline{B^i}$ and the standard deviation of the ratings σ_i and then normalize $B_{5\times 8}$ as the matrix $Z_{5\times 8}$ by Z-scores. Table 2.9 shows the results of $\overline{B^i}$ and σ_i calculation:

Table 2.9: $\overline{B^i}$ and σ_i *Calculation results*

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
$\overline{B^i}$	2.2	2.8	4	1.8	3.8	2.4	3.4	3.6
σ_i	0.84	0.45	1.22	0.84	0.84	0.89	0.89	0.89

Table 2.10 shows the normalized matrix $Z_{5\times8}$ as below:

Table 2.10: Normalized matrix $Z_{5\times8}$

User/ item	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	-0.239	0.447	0.8165	-0.956	-0.956	0.671	-0.447	-0.671
u_2	-1.434	0.447	-1.633	1.434	0.239	0.671	1.789	-0.671
u_3	0.956	0.447	0	0.239	0.239	-0.447	-0.447	1.565
u_4	-0.239	-1.789	0	0.239	1.434	0.671	-0.447	-0.761
u_5	0.956	0.447	0.816	-0.956	-0.956	-1.565	-0.447	0.447

- The third step: apply SVD on $Z_{5\times8}$. Table 2.11, Table 2.12 and Table 2.13 show the three decomposition matrix U, S and V^T with rank 5, respectively.

Table 2.11: Left eigenvectors matrix $U_{5\times5}$

-0.165	0.031	-0.747	-0.461	-0.447
0.719	-0.510	-0.005	0.150	-0.447
-0.262	-0.142	0.650	-0.536	-0.447
0.268	0.823	0.130	0.181	-0.447
-0.561	-0.202	-0.028	0.666	-0.447

Table 2.12: Eigenvalues matrix

4.211	0	0	0	0
0	2.2.731	0	0	0
0	0	2.351	0	0
0	0	0	1.134	0
0	0	0	0	0.000

Table 2.13: Right eigenvectors $V_{8\times5}^{T}$

-0.437	0.0725	0.319	-0.0218	0.069
-0.143	-0.674	-0.124	-0.357	-0.536
-0.419	0.254	-0.265	-0.069	-0.376
0.410	-0.148	0.391	-0.056	0.119
0.282	0.435	0.461	-0.024	-0.689
0.367	0.223	-0.282	-0.784	0.134
0.381	-0.417	-0.005	0.296	-0.134
-0.288	-0.199	0.605	-0.400	0.198

- The fourth step: with $d = 2$, obtain the appropriate low-rank approximation Z_2

Table 2.14: Low-rank approximation $Z_2 = U_{5\times2}.\, S_{2\times2}.\, V_{8\times2}^{T}$

0.311	0.042	0.313	-0.298	-0.159	-0.236	-0.301	0.183
-1.426	0.506	-1.625	1.449	0.249	0.801	1.738	-0.594
0.454	0.419	0.364	-0.394	-0.480	-0.492	-0.258	0.395
0.332	-1.677	0.096	0.130	1.298	0.918	-0.506	-0.774
0.994	0.709	0.851	-0.887	-0.907	-0.991	-0.671	0.790

- The fifth step: predict rating of active user u_2 on item $i = 4$,

$$p_{24} = \overline{B^4} + \sigma_4(Z_2)_{24} \approx 3$$

In [9], a personalized recommendation is proposed based on approximating the singular value decomposition, called ApproSVD. It firstly used a row-sampling approach to reconstruct massive data. The row-sampling approach rescales each row by an appropriate factor to form a relatively smaller matrix *C*. After that, they calculated the SVD of the reconstructed data *C*, which is a good approximation for the singular value decomposition of the original matrix. The algorithm normalized the right singular vectors of *C*. Finally, a matrix, which is good approximation to the original matrix, is returned. It is used to recommend unrated items to target users. ApproSVD method got high accuracy much faster than Drineas's LINEARTIMESVD algorithm [45] and the standard SVD algorithm, on large sparse dataset in particular. The ApproSVD prediction algorithm is described as below.

Input: a user-item matrix $A \in R^{m \times n}$, parameters $c, k \in Z^+$ satisfy $1 \le k \le c \le m$, sampling probabilities $\{p_i\}_{i=1}^{m}$ satisfy $p_i \ge 0$ and $\sum_{i=1}^{m} p_i = 1$.

Output: $H_k \in R^{n \times k}$ and $\sigma_t, t = 1, \ldots, k$.

1) For t =1 to c

   - Pick $i_t \in 1, \ldots, m$ under sampling probabilities $p_\alpha, \alpha = 1, \ldots, m$ (i_t denotes the row index of A)
   - Set $C^{(t)} = A^{(i_t)} / \sqrt{cp_{i_t}}$ (a row vector $A^{(i)}$ denotes the i-th row of A)

2) Construct C and compute its SVD; Now, $C = \sum_{t=1}^{c} \sigma_t y^{(t)} h^{(t)^T}$, where σ_t are the singular values of C and $h^{(t)}, t = 1, \ldots, c$ are its right singular vectors.

3) Keep the top k right singular vectors of C, namely $h^{(t)}, t = 1, \ldots, k$.

4) Return H_k, where $H_k^{(t)} = h^{(t)}$, and $\sigma_t, t = 1, .., k$

Compute $AH_k H_k^T$ to predict unknown ratings.

Figure 2.7: ApproSVD prediction algorithm

2.6 Accuracy measure of recommendation system

recommendation systems have been evaluated in many ways. Some evaluation metrics assess how close the ratings predicted by a recommendation system are to the actual ratings provided by the users. Other evaluation strategies take into account the frequency with which a recommendation system makes correct or incorrect decisions about whether an item is relevant for the user. Further, evaluation methods have been defined that quantify the ability of a recommendation algorithm to produce an ordering of the items that matches how the user would have ordered the same items according to his tastes.

In this research, we introduce some popular matrices, called accuracy matrices that have been used for the evaluation of recommendation systems. Accuracy metrics have been defined for two major tasks: The first one is to judge the accuracy of single predictions, and the second one is to evaluate the effectiveness of supporting users to obtain high- quality items. According to these tasks, accuracy metrics can be classified

into two main categories: predictive accuracy metrics and decision-support metrics.

Predictive accuracy metrics determines how close predicted ratings come to true ratings. They are particularly suited for tasks in which predictions are displayed along with the items. There are two most popular matrices in this category:

- ***Mean Absolute Error (MAE)***: A metric that measures the deviation of recommendations from their user specified values. For each rating prediction pair $\langle r_{u,i}, \overline{r_{u,i}} \rangle$. This metric treats the absolute error between them $\left| r_{u,i} - \overline{r_{u,i}} \right|$. The MAE is computed by first summing these absolute errors of the corresponding n rating predictions for all the m users, and then averaging the sum by the total number of users. A lower value of MAE corresponds to a higher accuracy of the recommendation.

$$MAE = \frac{\sum_{u=1}^{m} \sum_{i=1}^{n} \left| r_{u,i} - \overline{r_{u,i}} \right|}{mn} \tag{2.22}$$

  Where, $r_{u,i}$ is the true rating on a movie *i* by user *u.*

  $\overline{r_{u,i}}$ is the predicted rating.

- ***Root Mean Squared Error (RMSE)***: A metric that follows the same principle of MAE, but squaring the error before summing. Hence, large errors become much more pronounced than small ones.

$$RMSE = \sqrt{\frac{\sum_{u=1}^{m} \sum_{i=1}^{n} \left(r_{u,i} - \overline{r_{u,i}} \right)^2}{mn}} \tag{2.23}$$

Decision-support metrics determine how well a recommendation system can make predictions of high-relevance items. They are particularly suitable for evaluating top-n recommendation lists: users only take care about errors for highly ranked items. Prediction errors for low ranked items are unimportant, since users have no interest in them anyway. These metrics include classic Information Retrieval measures such as:

- ***Precision**:* A metric that represents the probability that an item recommended as relevant is truly relevant. It is defined as the ratio of items correctly predicted as relevant among all the items selected.

$$Precision = \frac{TR}{TR + FR} \tag{2.24}$$

Where, TR is the number of true relevant predictions.

FR is the number of false relevant predictions.

- ***Recall***: A metric that represents the probability that a relevant item will be recommended as relevant. It is defined as the ratio of items correctly predicted as relevant among all the items known to be relevant.

$$Recall = \frac{TR}{TR + FN} \tag{2.25}$$

Where, TR is the number of true relevant predictions

FN is the number of false non-relevant predictions.

CHAPTER 3 AN ENHANCED RECOMMENDATION SYSTEM USING USERS' LATENT RELATIONSHIPS WEIGHTING UTILIZATION (CF-ULRW)

In this chapter, the first part of the proposed method is presented and discussed. It is researched based on the utilizing the latent relationships of users to enhance the user-based collaborative filtering, called CF-ULRW. The CF-ULRW aims to deal with the large sparse data problem to improve prediction accuracy and reduce the time complexity of recommendation algorithm. In the experiment of the first part, the effectiveness of the CF-ULRW algorithm is compared with the user- based collaborative filtering [6] and the rowsampling approximating singular value decomposition (ApproSVD) [9] which were introduced in chapter 2.

3.1 Background research

The user-based or item-based collaborative filtering techniques are used, simple and intuitive to generate individual suggestions [46]. [47] proposed a hybrid predictive algorithm with smoothing (HPAS), it smoothed the unrated data by pre-computing the item's similarities and then built the predictive model based on both users' similarity and items' similarity. [48] presented a hybrid collaborative filtering based on the fusion of user-based algorithm and item-based algorithm with the control factor. [26, 49] combined collaborative filtering with content-based filtering to resolve the sparsity problem.

Several recommendation techniques have been proposed to deal with the sparsity problem. One of the most popular techniques is matrix factorization. Sarwar [8] applied singular value decomposition (SVD) to reduce the dimensions of a sparse rating matrix and eliminate rating matrix sparsity. Zeinab et al. [50] proposed to fill unrated value with user median, item median, the total median of ratings, and then singular value decomposition approach were implemented on preprocessing data for predictions. In [51], principal component analysis (PCA) was applied to the resulting dense subsets of the ratings' matrix. Yin et al. [52] used nonnegative matrix factorization (NMF) for recommendations, and proved that the dimension reducing techniques with non-negative constraints are more effective than others.

Matrix factorization can be combined with other techniques to improve an

accuracy of recommendation systems. [53, 54] presented SVD-based collaborative filtering algorithm. SVD was used to obtain a low-dimensional matrix which form is a low-rank estimate of the original rating-matrix. Therefore, the unknown ratings in the original rating-matrix was predicted by the corresponding entries in this low-rank estimate aims to fit the training data. Finally, collaborative filtering was applied to fitted training data without unknown ratings to make recommendations. In [55], particle swarm optimization (PSO) was applied for the SVD-free latent semantic index to obtain the optimal number of dimensions. In [56], the matrix factorization model was combined with social network analysis (SNA) to evidence possible social influence aim to make a social group recommendation system.

Besides, matrix factorization also can be used to analyze contextual information. [57] adopted free-formatted text-based tags into the traditional 2-Dimensional SVD approach and analyzed the effect of different tag similar techniques to the 3- Dimensional SVD recommendation performance. In [58], singular value decomposition was applied for extracting the most significant features corresponding to each entity.

3.2 Contributions

The goal of this chapter is to deal with large sparse data to improve prediction accuracy and system performance in a recommendation system. In order to reach this goal, the first contribution of CF-ULRW method is the reconstructive large sparse data matrix A by column-sampling approach. It extracts a constant number of columns of original user-item rating matrix A, scales the columns appropriately to form a relatively smaller matrix C, and then computes the singular value decomposition (SVD) of C. After that, the algorithm normalizes the left singular vectors of C. These vectors are used to obtain the best low-rank latent relationships of the user's data matrix. Finally, the second contribution of our proposed method is measurable users' latent relationships weighting values for rating prediction algorithm by Pearson correlation coefficient.

3.3 Users' latent relationships

SVD algorithm has an important attribute that enables us to get the true dimensionality of the original data, which is a user-item matrix A with rank r. The small singular values are less effective than larger ones, and the singular values are corresponding with the

singular vectors. Consequently, the low-rank approximation can be obtained by keeping the top k largest singular values of A (k << r). The assumption of this approach is that there are latent relationships between users and items, which affect the rating of a user for a given item. Specifically, it is assumed that there is a set of k factors, which determine how a user rates an item, and these factors can be captured by the rank-k SVD. The left singular vectors enable us to get the best low-rank latent relationships of users, and the right singular vectors enable us to get the best low-rank latent relationships of items. Our work focuses on the left singular vectors. However, SVD has been considered inappropriate for various practical applications involving massive data because SVD requires expensive computation and weakest performances for large sparse matrices. Therefore, our proposed method has been used a new approximating SVD instated of standard SVD to obtain low-rank latent relationships of the user's data matrix.

For reducing the dimension of the matrix, a row-sampling approach of Xun, Jing, Guang and Yanchun [14] named ApproSVD algorithm gave solution with higher accuracy than the standard SVD algorithm and LinearTimeSVD [16], on large sparse datasets. Different from above, we adopt to sample columns from a user-item rating matrix and then extract users' latent relationships by column-sampling approximating SVD algorithm. These users' latent relationships are used to measure users' latent relationships weighting values in rating prediction algorithm. The column-sampling approximating SVD algorithm enables us to get the best low-rank users' latent relationship data matrix with lower computational cost.

3.4 CF-ULRW algorithm description

Figure 3.1 shows the workflow of the CF-ULRW algorithm. The CR-ULRW algorithm description is given as below: Give a user-item matrix $A \in R^{mxn}$, where the users are the rows and the items are the columns.

Firstly, the matrix A was reconstructed by sampling c columns, rescale each column by an appropriate factor to form a relatively smaller matrix $C \in R^{mxc}$. The columns are chosen randomly without considering sampling probabilities. The sampling probabilities $\{p_i\}_1^n$ are given by:

$$p_i = nnz\left(A^{(i)}\right) / nnz(A), i = 1,...,n \tag{3.1}$$

where, $A^{(i)}$ denotes the number of nonzero element in the i^{th} column of A

nnz(A) denotes the number of nonzero element in the matrix A.

We choose the i^{th} column of A which has the largest sampling probability p_i, and then choose the j^{th} column of A which has the second largest sampling probability p_j, keep on until picking enough c columns of A.

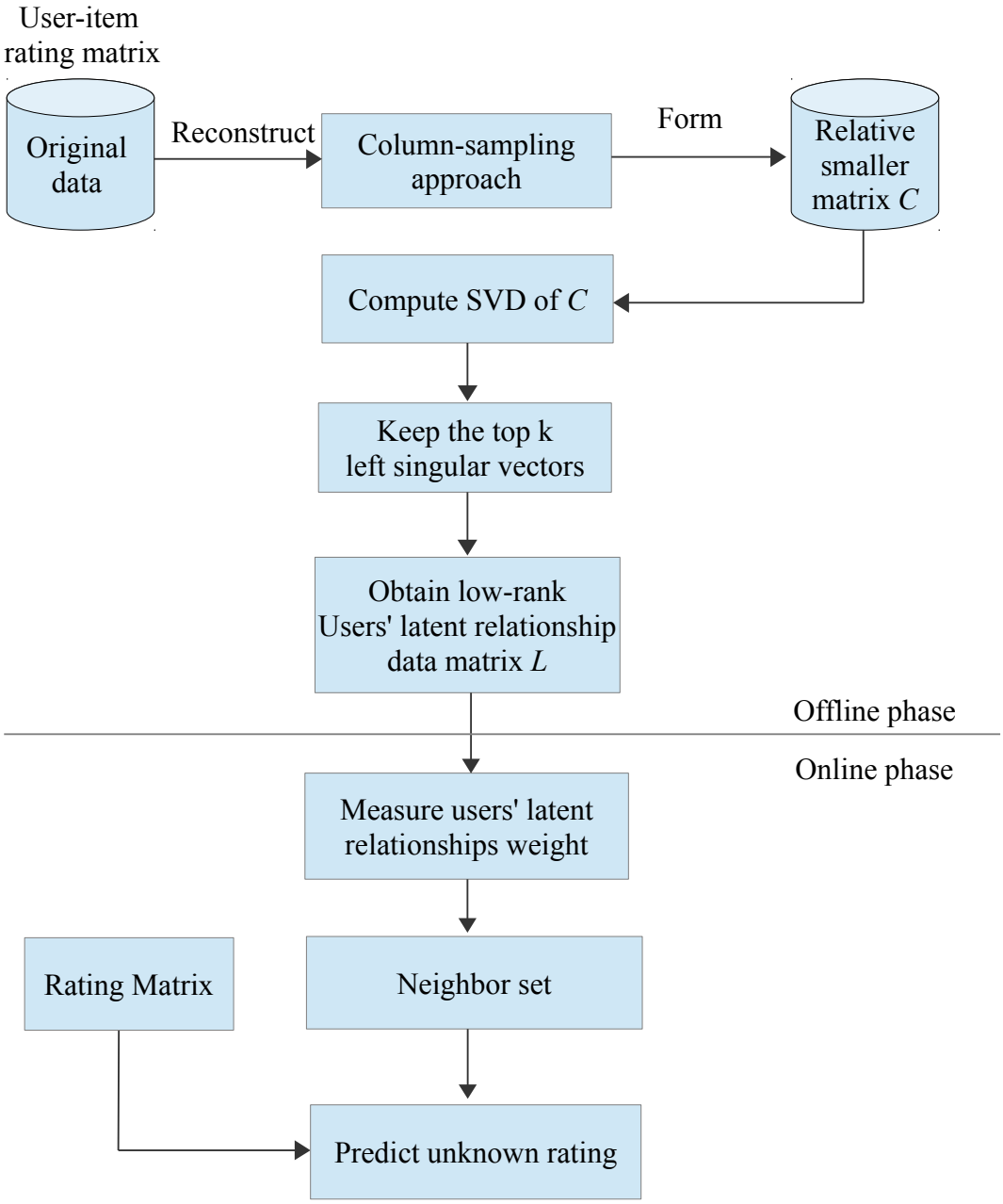

Figure 3.1: The workflow of the CF-ULRW algorithm

According to the sampling order, matrix C was constructed by reducing the scale of each column.

$$C^{(t)} = A^{(i_t)} / \sqrt{cp_{i_t}} \tag{3.2}$$

Where, a column vector $A^{(i)}$ denotes the *i*-th column of A.

Secondly, we calculate the singular values and corresponding left singular vectors of the matrix C as below:

$$C = \sum_{t=1}^{c} \sigma_t y^{(t)} h^{(t)T} \tag{3.3}$$

Where, σ_t are the singular values of C

$y^{(t)}$ are the left singular vectors of C,

$h^{(t)T}$ are the transpose of the right singular vectors of C , $t = 1,...,c$.

The left singular vectors of C are a good approximation to left singular vectors of A, and they are used to obtain the best low-rank users' latent relationship data matrix.

Thirdly, keeping the top k largest singular values and corresponding left singular vectors as $S_k \in R^{kxk}$ and $U_k \in R^{mxk}$, respectively. Fourthly, multiply two matrices U_k and $\sqrt{S_k}$ to obtain low-rank users' latent relationship data matrix $L_k \in R^{mxk}$.

$$L_k = U_k \times \sqrt{S_k} \tag{3.4}$$

Fifthly, users' latent relationships weight is measured by Pearson correlation coefficient. The Pearson correlation coefficient is an effective method to measure the strength of association between users.

$$L(u,v) = \frac{\sum_{i=1}^{k} \left(r_{u,i} - \overline{r_u}\right)\left(r_{v,i} - \overline{r_v}\right)}{\sqrt{\sum_{i=1}^{k} \left(r_{u,i} - \overline{r_u}\right)^2 \sum_{i=1}^{k} \left(r_{v,i} - \overline{r_v}\right)^2}} \tag{3.5}$$

Where, *u* and *v* denote users in users' latent relationship data matrix L_k

r denotes the latent relationship value of a user on an item

$\bar{r}$ denotes the average of the latent relationship values of all user on the item in L_k.

And finally, rating of target user is predicted based on the average of users' latent relationships weight of top *h* the nearest neighbors' target user. The number of nearest neighbors *h* is chosen randomly.

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{h} L(a,u) \times (r_{u,i} - \bar{r}_u)}{\sum_{u=1}^{h} |L(u,a)|} \tag{3.6}$$

Where, *a* is target user, *u* is the nearest neighbors of *a* and *i* is an item.

r denotes rating of a user on an item

$\bar{r}$ denotes the average ratings of all users on an item in *A*

L(a,u) is the latent relationships weight between user *a* and *u*

3.5 Experimentation

3.5.1 Dataset

The experimentation is experimented on MovieLens dataset [32] which is provided by MovieLens group. MovieLens is one of the most popular sparse datasets. The dataset consists of 100,000 ratings from 943 users on 1682 movies. The ratings range from 1 to 5, where the best films are rated 5 and the worst films are rated 1. The dataset is divided into a training set and a test set. The sparsity of data is calculated as below:

$$sparsity = 1 - \frac{100000}{943\text{x}1682} = 0.937$$

Mean Absolute Error (MAE) is used to evaluate the efficiency. A lower value of

the MAE corresponds to the higher accuracy of the recommendation. The experimental methods are performed on training set to make predictions for the test set.

3.5.2 Experimental results

The CF-ULRW method utilizes the users' latent relationships to measure the similarity between users. User similarity is the key role in the accuracy of user-based filtering recommendation system. In the user-based filtering method, it is obtained from training set with high computational cost (m*n), where m is the number of users and n is the number of items. Besides, it has low accuracy because training set is too sparse. The users' latent relationship matrix is obtained by the meaningful information extraction from training set reconstruction. The users' latent relationship matrix has a small size (m*k), where m is the number of users, and k is the top largest singular values of training matrix reconstruction (k << n). The user similarity matrix is calculated with low computational cost and overcome the large sparse data problem by using the users' latent relationship matrix, which makes our proposed method get a good result.

In the experiment, our experiment is separated into two parts. The first part, we do experiment to compare our proposed method called CF-ULRW with the row-sampling approximating SVD called ApproSVD. There are two main objectives: the first one is to estimate the optimal parameter of the proposed method; the second one is to prove that our proposed method can deal with the large spare data problems to improve prediction accuracy. The second part, we do experiment to compare CF-ULRW with the user-based collaborative filtering called CF by using the optimal parameter, which is estimated from the first part. The objective of the second part is to show that our proposed method can improve performance system as well as the prediction accuracy.

First, we show the prediction accuracy comparison between CF-ULRW and ApproSVD. We implement with 80% training set, and 20% testing set and choose the number of samples for matrix C, as 100, 200, 300, 400, 500 and 600 and the dimension of the eigenvector is 10. The nearest neighbor parameter of the proposed method is set h = 50. From the Table 3.1, MAE values of our proposed method are smaller than ApproSVD in all cases. The MAE of the CF-ULRW method has slightly changed with the increase of c for the same k. It means that our proposed method still gets a high

accuracy with the low value of the pair (c, k), then it can reduce the computational cost of the system.

Table 3.1: Mean Absolute Error at different sample c under the same dimensional k

c	k	CF-ULRW	ApproSVD
100	10	0.7792	1.0269
200	10	0.7773	1.0891
300	10	0.7777	1.1212
400	10	0.7778	1.1773
500	10	0.7770	1.200
600	10	0.7772	1.2142

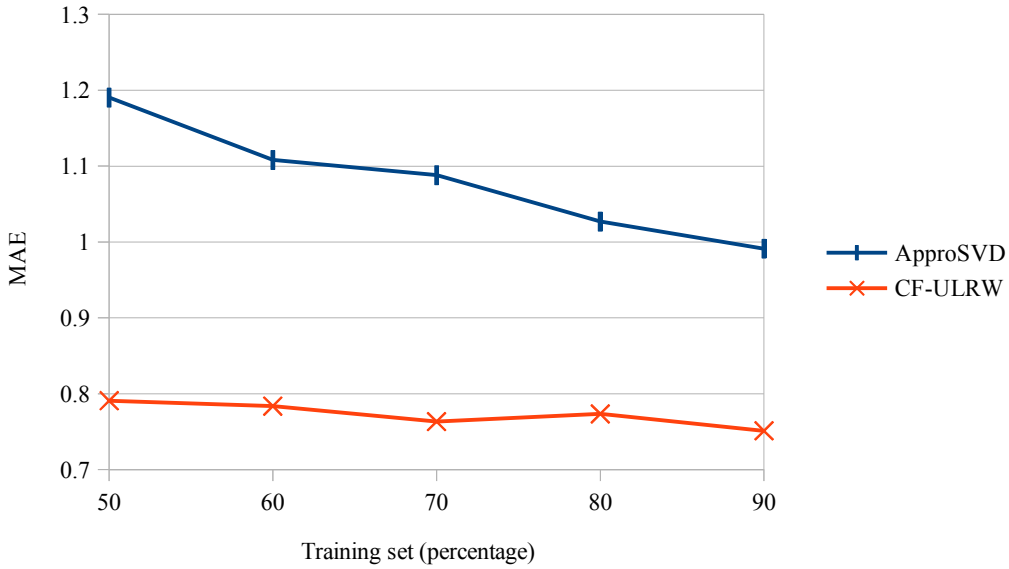

Figure 3.2: Mean Absolute Error at different training set

After that, we choose the optimal parameter pair (c, k) of ApproSVD, equals to (100, 10), and we change the training set equals to 50%, 60%, 70%, 80% and 90%. The nearest neighbor parameter of the CF-ULRW method is set h = 50. In Fig. 3.2, MAE of the CF-ULRW method is smaller than ApproSVD, even we have chosen the optimal parameter of ApproSVD.

To achieve that, the CF-ULRW method gets better results than ApproSVD. We use the same parameters as above, but we set the number of samples, equals to 600 and the dimension of the eigenvector is changed to k = 100, 200, 400, and 600. In Table 3.2,

we can show that the CF-ULRW method outperforms than ApproSVD, even we increase a value of k to make MAE of ApproSVD decrease. However, the large value of k will make the system consume very high time computation.

Table 3.2: Mean Absolute Error at the same sample c and different dimensional k

c	k	CF-ULRW	ApproSVD
600	100	0.7849	1.0016
600	200	0.7910	0.8928
600	400	0.7924	0.8216
600	600	0.7912	0.8075

In the second part, we show the prediction accuracy comparison between CF-ULRW and CF. We implement with 80% training set and 20% testing set and set the parameter pair (c, k) = (200, 10), and we change set of the nearest neighbor parameter *h* = 10, 20, 30, 40, 50, and 60. Both use the same Pearson correlation coefficient method to measure weighting values. In Figure 3.3, MAE of the proposed method CF-ULRW is smaller than CF. The time complexity of CF is O(n) for weighting calculation, and the time complexity of CF-ULRW is O(k) for weighting calculation. With k << n, our proposed CF-ULRW still gets higher accuracy than CF. It can prove that our proposed method CF-ULRW has time complexity to compute Pearson correlation similarity less than CF.

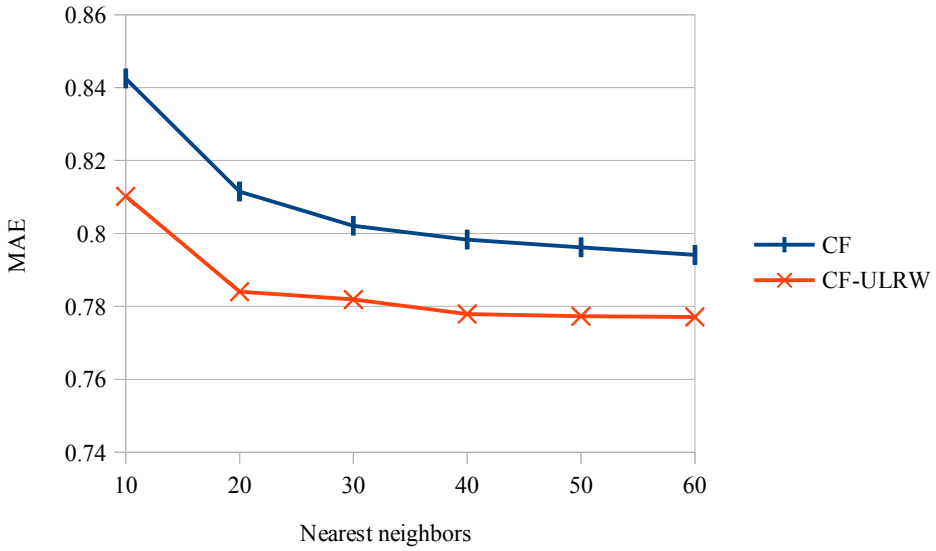

Figure 3.3: Mean Absolute Error at different set of the nearest neighbor h

In Figure 3.4, we choose the optimal parameter pair (c, k) = (200, 10) and change the training set equals to 50%, 60%, 70%, 80% and 90%. The nearest neighbor parameter is set h = 50. The MAE of our proposed method is smaller than CF algorithm. It can prove that our proposed method can improve the prediction accuracy more than CF.

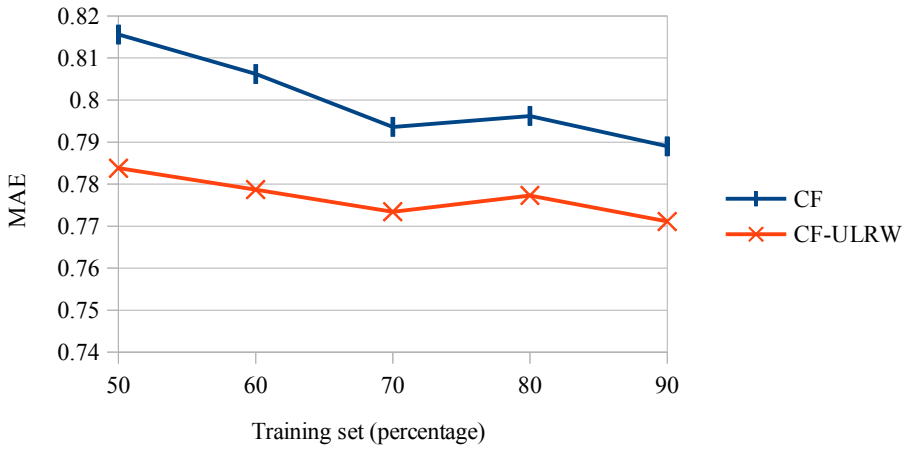

Figure 3.4: Mean Absolute Error at different training set

3.6 Discussion

In this chapter, the first part of the proposed method is introduced. It is known as an enhanced user-based collaborative filtering with users' latent relationships weighting utilization, CF-ULRW. From the result of the experiments, the CF-ULRW algorithm outperforms other methods. The CF-ULRW algorithm successfully deals with the large sparse data problem and improves prediction accuracy as well as system performance. Users' latent relationships are extracted with lowest computation cost by column-sampling approximating SVD algorithm. Pearson correlation coefficient is used to measure users' latent relationships weighting values for rating prediction algorithm. The experimental results have demonstrated that the CF-ULRW algorithm has a higher prediction accuracy than row-sampling approximating SVD algorithm and user-based collaborative filtering algorithm. Otherwise, a user-based collaborative filtering and the CF-ULRW algorithm use the same Pearson correlation coefficient method to measure weighting values, but the CF-ULRW algorithm saves time for weighting calculation much more than a user-based collaborative filtering because users' latent relationship data matrix has a lower dimension.

CHAPTER 4 AN ENHANCED RECOMMENDATION SYSTEM USING REVIEW HELPFULNESS FEATURES AND REVIEW HELPFULNESS TRUST

In this chapter, the second and the third part of the proposed method are presented and discussed. The second part is a hybrid recommendation system using review helpfulness features, called RHF-CF. Review helpfulness features will be measured by using review rating and then it used to construct the hybrid model. The RHF-CF algorithm constructs the hybrid model in the offline phase and calculates the recommendation results in online phase. The RHF-CF algorithm aims to deal with the scalability and sparse data problem and decrease time consuming.

Aim increases the accuracy of a recommendation system, an improvement is implemented with RHF-CF by using a review helpfulness trust, which is known as the third part of the proposed method. The improved algorithm is called RHT-CF. In the RHT-CF algorithm, review helpfulness trust values will be added for calculating the recommendation results in the online phase of the RHF-CF algorithm.

In the experiment, the results of the RHF-CF algorithm will be compared with collaborative filtering based on hybrid user model (HUM-CF) [59], user-based collaborative filtering (UB-CF) [6] which are introduced in chapter 2. The results of the RHT-CF algorithm will be compared with RHF-CF algorithm to prove the effect of trust values.

4.1 Background research

Presently, when consumers buy a product in the e-commerce system, they always consider about reviews of other consumers for a product besides product rating information because there are many attackers who aim to fake a product rating information. It causes the inefficiency system. To deal with this problem, there is additional information called the review rating. It determines which reviews are helpful. Review helpfulness rating plays an important role in the consumer's decision.

There have been many kind of research which study model-based collaborative filtering to deal with the scalability problems. The hybrid user model is constructed by

using user-item rating, item description, and user demographic information [6]. User demographic information is used in the recommendation for improving the rating prediction accuracy [6].

In [60], a user model is constructed by using user-item rating matrix, user demographic and time information. It provided a better recommendation at a specific time. On the other way, some research [61, 62] used user-based collaborative filtering with time weight, which combined user-item rating and time weighting to predict the recommendation result.

The user-based or item-based collaborative filtering techniques are used, simple and intuitive to generate individual suggestions [46]. A hybrid predictive algorithm with smoothing [47] is proposed to smooth the unrated data by pre-computing the item's similarity and then build the predictive model based on both users' similarity and items' similarity. Reference [48] presented a hybrid collaborative filtering based on the fusion of user-based algorithm and item-based algorithm with the control factor and [26] combined collaborative filtering with contend-based to resolve the sparsity problem.

Besides, there are other techniques; e.g. Trust-based filtering recommendation system [63, 64, 65]. The trust-based recommendation can make recommendations as long as a new user connects to a large enough component of the trust network. Trust network improves the coverage of recommendations. The level of trust between users will be represented by trust matrix.

4.2 Review rating

The review rating of consumers is very important information of the products that will affect users in the online communities. It appears on some websites such as Epinions.com, determines the helpfulness of reviews. The review rating is calculated from the average rating of a review, which users have given for evaluation, in the range of 1 to 5. The high rating means the most users in the system, considered that this review is very helpful for the item. Each product rating has a review rating that shows the helpfulness of product rating. Figure 4.1 shows an example of review rating on the website Epinions.com.

J. Otterbacher [66] examined the nature of "helpfulness". The examination proved the meaning of helpfulness and the quality of information in the reviews. This research also analyzed five quality dimensions that are reviewers' reputations, topical relevancy, ease of understanding, believability and objectivity. The experimental result found these dimensions are related to the helpfulness scores.

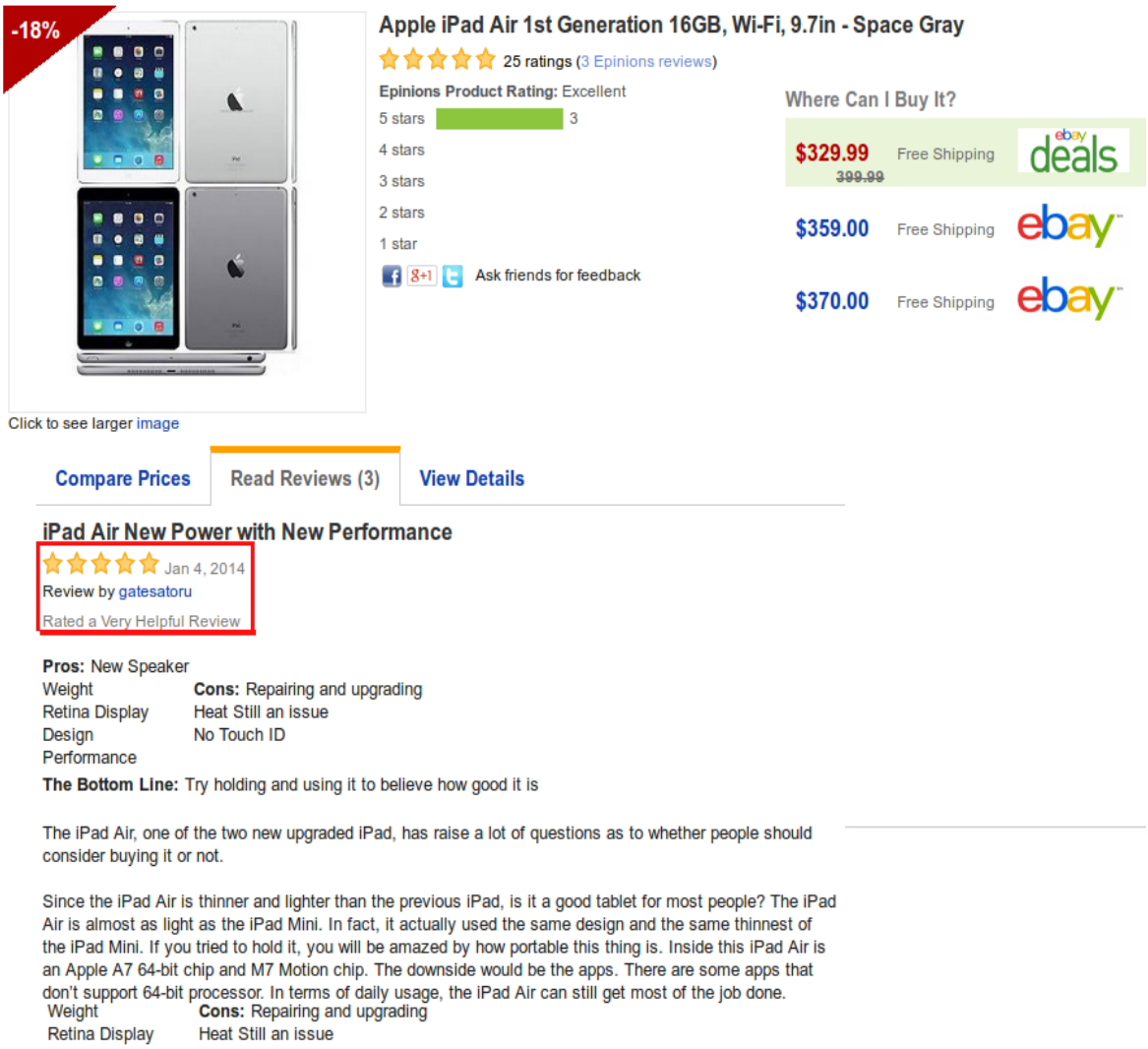

Figure 4.1: Example of review rating on Epinions.com

4.3 Contributions

This chapter is present a novel hybrid recommendation method to deal with scalability, sparse data problem aim to improve prediction accuracy and performance system. To achieve that, there are two main contributions in this chapter. The first contribution is the

extraction of review helpfulness features from review rating. After that, these attributes will be used to build the hybrid model. The hybrid model is for computing the similarity between users to make recommendation results. Aim to improve the quality of the first contribution, the second contribution is created. In the second contribution, review helpfulness trust is measured by using a review rating too. The trust values are added into user similarity computation which helps to get an optimal weight vector or more accurate user similarity. Weight or similarity computation is an important work in a recommendation system based on collaborative filtering. They Weight or similarity decides the quality of a recommendation system, in terms of accuracy.

4.4 Review helpfulness features extraction

4.4.1 Extraction formula

Review helpfulness features are extracted from review ratings and item features. Given a user-item review rating matrix R^{mxn}. The features F_k are defined as the set of categories of items. The review helpfulness features are known as the relative feature scores [59]. It is measured following these steps below:

- *Total Rating:* the sum of ratings that the user i grades for all the items, which is expressed as $TR(i)$.

$$TR(i)=\sum_{j\in T_i} R_{i,j} \tag{4.1}$$

  Here, T_i stands for the set composed of items have been graded by the user i
  $R_{i,j}$ is the review rating of the user i to an item j

- *Feature Rating:* the sum of effective ratings that user i grades for the feature k, which is expressed as *FR(i,k).*

$$FR(i,k)=\sum_{j\in F_k\subset T_i, R\geq\frac{v}{2}} R_{i,j} \tag{4.2}$$

  Here, F_k is the set of items in feature k

v is the maximum rating in the given system. The ratings are not less than *v/2* is defined as effective rating.

- *Feature Frequency:* the total times of effective ratings that user i grades for the feature k, which is expressed as *FF(i,k)*.

$$FF(i,k)=\sum_{j\in F_k\subset T_i}\delta_p(R_{i,j}),p\in\left\{x\,|\,s\geq\frac{v}{2}\right\} \tag{4.3}$$

$$\delta_p(R_{i,j}) = \begin{cases}1 & if \quad p=R_{i,j}\\ 0 & if \quad p\neq R_{i,j}\end{cases}$$

- *Relative Feature Rating:* Relative Feature Rating that user i grades for feature k, expressed as *RFR(i,k)*, is defined as the ratio of Feature Rating *FR(i,k)* and Total Rating *TR(i,k)*.

$$RFR(i,k)=\frac{FR(i,k)}{TR(i)} \tag{4.4}$$

- *Relative Feature Frequency:* Relative Feature Frequency that user i grades for feature k, expressed as *RFF(i,k)*, is defined as the ratio of Feature Frequency *FF(i,k*) and *TF(i)*.

$$RFF(i,k)=\frac{FF(i,k)}{TF(i)},TF(i)=(T_i) \tag{4.5}$$

Here, *TF(i)* is the total times of ratings graded by user i.

- *Modified Relative Feature Frequency:* The preference of a user for an item is reflected by the rating value. However, in the definition of *RFF*, the degree of preference cannot be reflected, for it gives all effective ratings the same weight value. Modified Relative Feature Frequency is given to overcome the disadvantages of *RFF*, and it is modified as follows:

$$MRFF(i,k)=\frac{\sum_{j\in F_k\subset T_i} w_p\times\delta_p(R_{i,j})}{\frac{v}{2}\times TF(i)} \tag{4.6}$$

$$p\in\left\{x\,|\,x\geq\frac{v}{2}\right\}\wedge w_p=p-\frac{v}{2}+1$$

- *Relative Feature Score:* Relative Feature Score that user i takes for feature k, expressed as *RFS(i,k)*, is defined as:

$$RFS(i,k)=\frac{2\times Max\times RFR(i,k)\times MRFF(i,k)}{RFR(i,k)+MRFF(i,k)} \tag{4.7}$$

RFS is the harmonicmean of *RFR* and *MRFF*. *RFS* values range from 0 to *Max*, which is suitable for systems based on ratings. *Max* is the maximum possible rating value in the given system. This formula is used to measure review helpfulness features. RFS of review ratings is review helpfulness features.

4.4.2 Example of review helpfulness features extraction

Table 4.1 gives an example of the data gotten from Epinions dataset to explain the concepts defined in section 4.3.1. The first and the second columns are user and item, respectively. The third column is the categories which are represented as features. In this example has two categories (1 and 2) corresponding two features, called ***F1*** and ***F2***. The fourth column is review rating.

Table 4.1: Example of original data

User	Item	Feature (category)	Review rating
1	1	1	5
1	2	1	5
1	3	2	4
2	1	1	5
2	2	1	3
2	3	2	5
2	4	2	4

In order to get RFS, we firstly compute *RFR* and *MRFF* by formula (4.4) and (4.6), respectively. *RFR-Helpfulness* denotes *RFR* of review rating. *MRFF-Helpfulness* denotes *MRFF* of review rating. The results show in Table 4.2

RFR-Helpfulness and *MRFF-Helpfulness* are obtained from review ratings. The example for *RFR-Helpfulness* and *MRFF-Helpfulness* of user 1 on feature F1:

$$RFR\text{-}Helpfulness(1,F1)=\frac{FR(1,F1)}{TR(1)}=\frac{5+5}{5+5+4}=0.7143$$

$$MRFF\text{-}Helpfulness(1,F1)=\frac{(5-\frac{5}{2}+1).1+(5-\frac{5}{2}+1).1}{\frac{5}{2}.3}=0.9333$$

Table 4.2: User-Category matrix computed RFR and MRFF of review rating

	Users	**Features (categories)**	
		F1	**F2**
RFR-Helpfulness	1	0.7143	0.2857
	2	0.4706	0.5294
MRFF-Helpfulness	1	0.9333	0.3333
	2	0.5000	0.6000

Table 4.3 shows the result of *RFS* computation, it is computed by formula (4.7). The example for *RFS-Helpfulness* of user 1 on feature F1:

$$RFS\text{-}Helpfulness(1,F1)=\frac{2\times Max\times RFR\text{-}Helpfulness(1,F1)\times MRFF\text{-}Helpfulness(1,F1)}{RFR\text{-}Helpfulness(1,F1)+MRFF\text{-}Helpfulness(1,F1)}$$

$$RFS\text{-}Helpfulness\ (1,F1)\ =\ \frac{2\times 5\times 0.7143\times 0.9333}{0.7143\ +\ 0.9333}=4.0462$$

Table 4.3: User-Category matrix computed RFS of review rating

	Users	Features (categories)	
		F1	**F2**
RFS-Helpfulness	1	4.0462	1.5385
	2	2.4242	2.8125

4.5 Algorithm description: an enhanced recommendation system using review helpfulness features (RHF-CF)

Figure 4.2 shows the workflow of the RHF-CF algorithm. It split the computation into the offline phase and the online phase. Item features are classified by using product categories. In the offline phase, review helpfulness features and product preference features are extracted from review rating and product rating, respectively. After that, the hybrid model is constructed on the basis of the combination product preference features set and review helpfulness features set. In the online phase, the hybrid model is used to calculate users' similarity which aims to estimate neighbor set for predicting task. Top-N nearest neighbor algorithm is applied to make recommendation results.

4.5.1 offline phase

There are two main tasks in this phase:

- Extract review helpfulness features and product preference features: Product preference features are extracted from a product rating by the same way with the review helpfulness features in section 4.4. After that, these features are used to construct the hybrid model.

- Construct a hybrid model: the hybrid model is constructed by using a combination of review helpfulness features and product preference features. The hybrid model is for computing the similarity between users that is measured by the similarity in preference (RFS-Preference) and the similarity in opinion (RFS-Helpfulness).

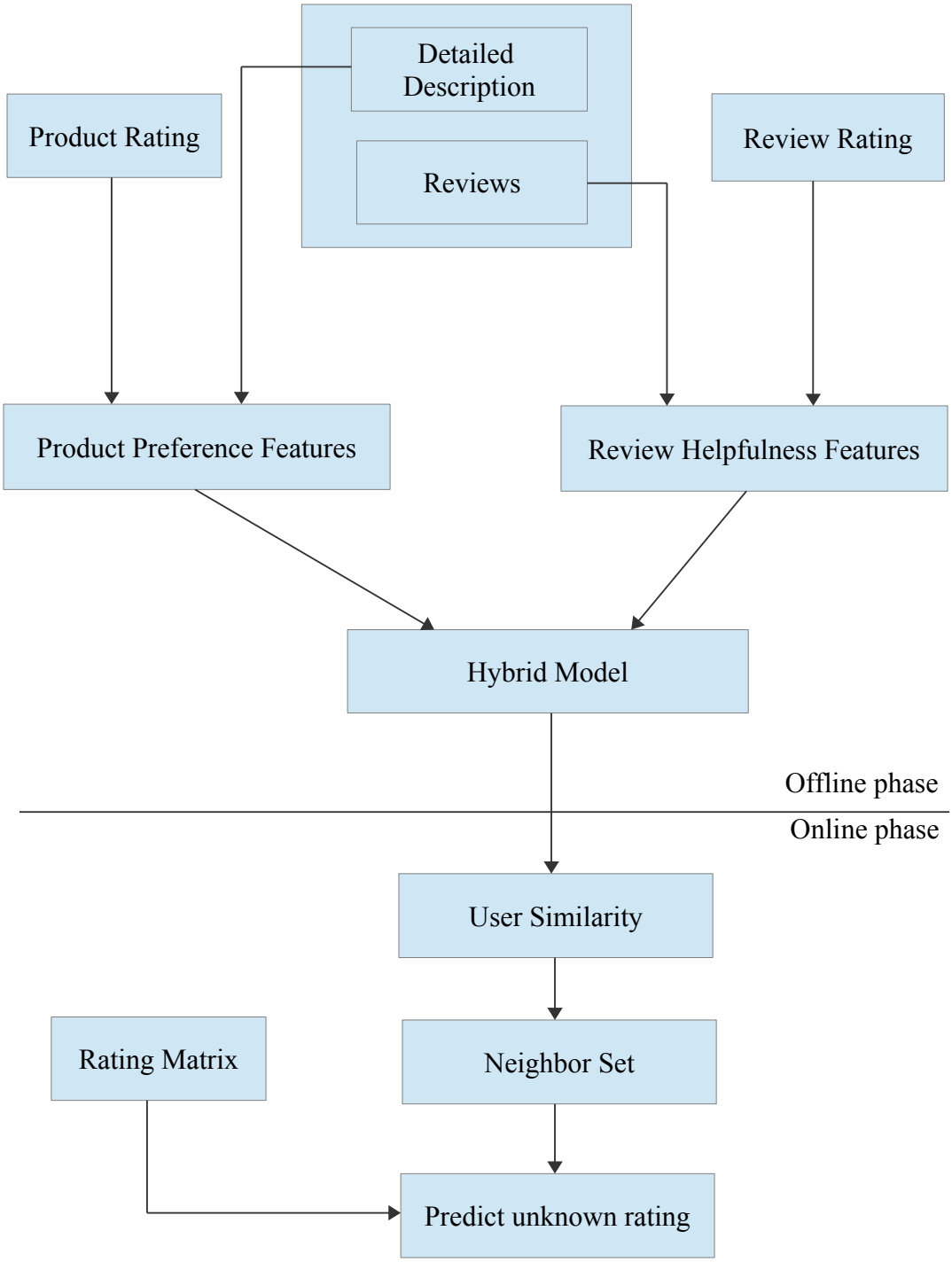

Figure 4.2: Workflow of the RHF-CF algorithm

Example of hybrid model construction:

- Table 4.4 is an example of the data from Epnions dataset. The first and the second columns are user and item, respectively. The third column is the categories which represent as features. There are two categories as 1 and 2, corresponding two features, called F1 and F2. The fourth and fifth columns are product rating and review rating, respectively.

Table 4.4: An example of original data

User	Item	Feature (category)	Product rating	Review rating
1	1	1	4	5
1	2	1	3	5
1	3	2	5	4
2	1	1	4	5
2	2	1	4	3
2	3	2	5	5
2	4	2	4	4

- Review helpfulness features and product preference features are extracted by applying the formulas in section 4.4. RFS-Helpfulness denotes for review helpfulness features and RFS-Preference denotes for product preference features. Table 4.5 shows the results of RFS computation for all users.

Table 4.5: User-category matrix computed RFS of review rating and product rating

	Users	Features (categories)	
		F1	F2
RFS-Helpfulness	1	4.0462	1.5385
	2	2.4242	2.8125
RFS-Preference	1	2.7861	2.2013
	2	2.4242	2.8125

- Hybrid model construction: Table 4.6 is constructed by using a combination of RFS-Preference features and RFS-Helpfulness features, called the hybrid model.

RFS-Preference features are product preference features which are obtained from RFS of product rating and item features. RFS-Helpfulness features are the review helpfulness features which are obtained from RFS of review rating and item features.

Table 4.6: Hybrid model

User	***RFS-Helpfulness***			***RFS-Preference***		
	F1	***…***	***Fp***	***F1***	***…***	***Fp***
U1	H1(1)	…	H1(p)	H1(p+1)	…	H1(p+p)
...						...
Um	Hm(1)	…	Hm(p)	Hm(p+1)	…	Hm(p+p)

4.5.2 Online phase

There are two main tasks in this phase: the first task computes the similarity of users to find the nearest neighbor of the target user. Neighbors are users who have high similarity to target user. And then, the second task is rating prediction for the target users based on similarity between the target user and the neighbor set.

- Computing similarity of users: there are many kinds of method to measure similarity between users. For example, Pearson correlation coefficient [10], Cosine vector [10], Mean squared difference [34], Relevant similarity [59]. Relevant similarity considered the best method for the hybrid model [59]. After the hybrid model is constructed in Table 4.6, the relevant similarity is applied for computing the similarity between users. The formula is as follows:

$$S_{u,v} = \frac{\sum_{k=1}^{p+p} \left| \left(H_{u,k} - \overline{H}_{u,k}\right) . \left(H_{v,k} - \overline{H}_{v,k}\right) \right|}{\sqrt{\sum_{k=1}^{p+p} \left(H_{u,k} - \overline{H}_{u,k}\right)^2} . \sqrt{\sum_{k=1}^{p+p} \left(H_{v,k} - \overline{H}_{v,k}\right)^2}} \quad (4.8)$$

Where, u and v are users

$H_{u,k}$ and $H_{v,k}$ are values of from the hybrid model of user u, v for feature k

$\overline{H}_{u,k}$ and $\overline{H}_{v,k}$ are average values from the hybrid model of user u, v for feature k.

- Predicted rating: after the user similarity matrix is obtained, Top-N nearest neighbor method is applied to find nearest neighbors of the target user. The prediction formula is as follows:

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u \in N} S_{a,u}\left(r_{u,i} - \bar{r}_u\right)}{\sum_{u \in N} S_{a,u}} \tag{4.9}$$

Where, $P_{a,i}$ is the predicted rating of target user a on a item i.

N is the nearest neighbors set of a target user a

$\bar{r}_a$, $\bar{r}_u$ is the average rating of target user a and user u, respectively

$r_{u,i}$ is the rating of user u to item i

4.6 Algorithm description: an improvable recommendation accuracy using review helpfulness trust (RHT-CF)

Figure 4.3 shows an improvement of the RFH-CF algorithm using review helpfulness trust, called RHT-CF. Weight or similarity computation is an important work in a recommendation system based on collaborative filtering. The purpose of the improvement is to find an optimal weight vector, which helps to get more accurate user similarity. Based on that ideal, an improvable algorithm RHT-CF is implemented by using a review helpfulness trust. The improvement is implemented in both phase. In the offline phase, review helpfulness trust is estimated from review rating. After that, these trust values are used to compute a user similarity in online phase, which is used in the rating prediction algorithm. Top-N nearest neighbor algorithm is still used to make recommendation results. The effect of the improved algorithm is proved via experimental results, in terms of accuracy.

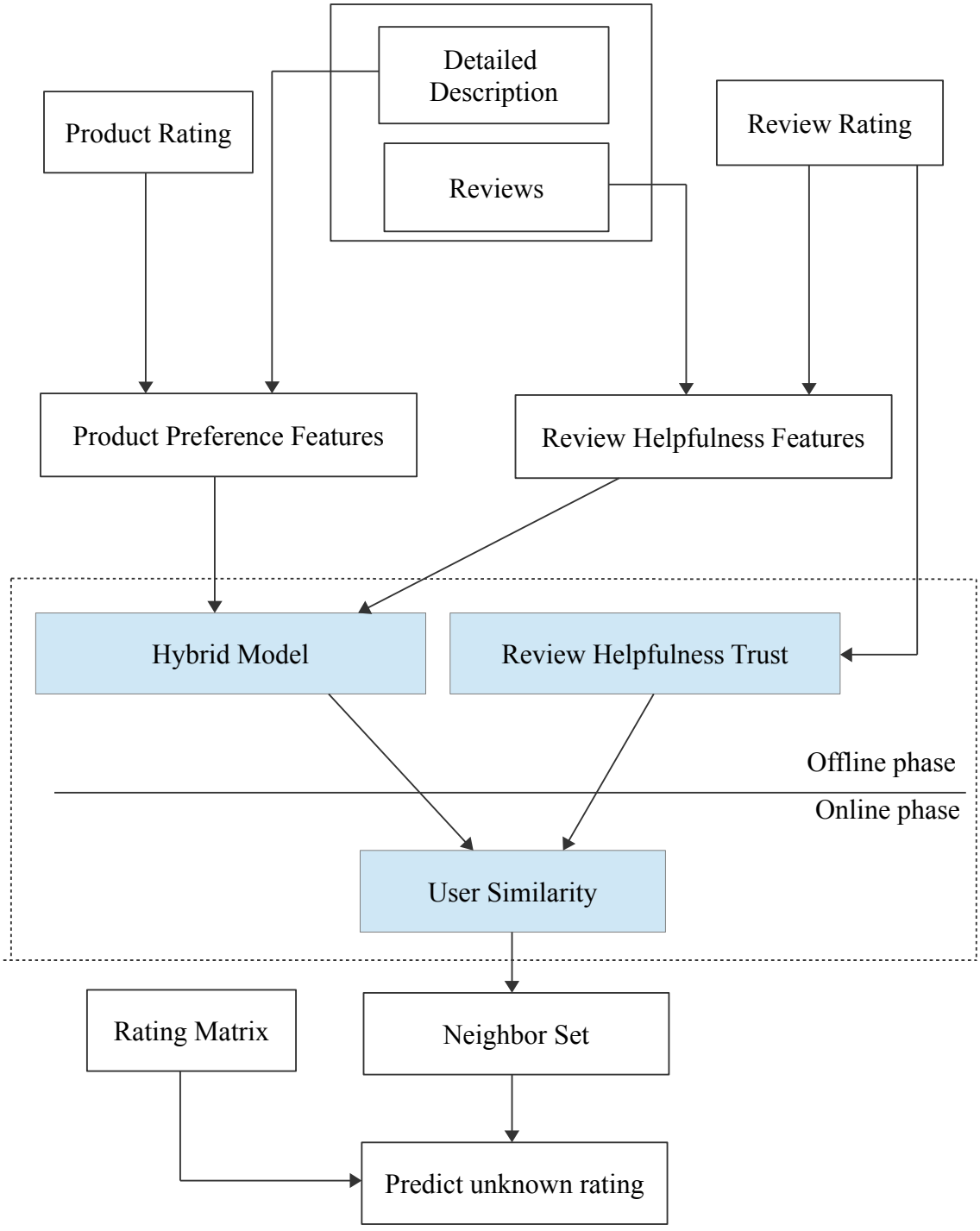

Figure 4.3: Workflow of RHT-CF algorithm

4.6.1 Review helpfulness trust computation

Review helpfulness trust shows the reliability between users, which is computed by using review rating. It is used in user similarity computation in the next section. It is calculated via three steps as below:

- Normalized average review rating in each category for each user:

$$FR(u,k)=\frac{\sum_{i\in F_k\subset T_u} R_{u,i}}{(T_u)\times max(T_u)}, \tag{4.10}$$

Where, F_k is a set of items have been graded by user *u* in feature *k*

T_u is a set of items have been graded by user *u*

$R_{u,i}$ is the rating of user u on item *i*.

- Taste set: a set which shows the interest of users on all categories. The value is 1 or 0, which is classified by normalized average review rating above.

$$H(u,k) = \begin{cases} 1 \ \ if \ \ FR(u,k) \geq \alpha \\ 0 \ \ if \ \ otherwise \end{cases} \tag{4.11}$$

Where, *u* denotes user

k denotes the category

α is the threshold of taste set.

- Review helpfulness trust computation: trust between users is computed by Jaccard coefficient, which compares the taste set. The trust value of user *u* and *v* is defined as the proportion of the number of intersections and the number of unions of H_u and H_v .

$$T_{u,v} = \frac{|H_u \cap H_v|}{|H_u \cup H_v|} \tag{4.12}$$

Where, H_u and H_v is taste set of the user *u* and *v*, respectively.

$T_{u,v}$ is trust between user *u* and *v*

4.6.2 User similarity computation

A new method is applied to calculate user similarity, which review helpfulness trust is added into user similarity computation. The relevant similarity formula (4.8) is modified as follows:

$$S_{u,v} = \frac{\sum_{k=1}^{p+p} T_{u,v} \times \left| \left(H_{u,k} - \overline{H}_{u,k}\right) . \left(H_{v,k} - \overline{H}_{v,k}\right) \right|}{\sqrt{\sum_{k=1}^{p+p} \left(H_{u,k} - \overline{H}_{u,k}\right)^2} . \sqrt{\sum_{k=1}^{p+p} \left(H_{v,k} - \overline{H}_{v,k}\right)^2}} \tag{4.13}$$

Where, *u* and *v* are users

$T_{u,v}$ is review helpfulness trust value between user *u* and *v*

$H_{u,k}$ and $H_{v,k}$ are values of a hybrid model of user u and v for feature k

$\overline{H}_{u,k}$ and $\overline{H}_{v,k}$ are average values from the hybrid model of user u, v of feature k

4.7 Experimentation

Two experiments will be implemented as below:

- First experiment: RHF-CF is experimenting in the first experiment. After that, it is compared with User-based collaborative filtering algorithm [6] and Collaborative filtering based on hybrid user model [59], called UB-CF and HUM-CF, respectively.

- Second experiment: RHT-CF is experimenting in the second experiment. After that it is compared with the result of the RHF-CF in the first experiment.

4.7.1 Dataset

The experimentation is experimented on Epinions dataset. The data were collected by graduate students of Arizona State University in May 2011 [67]. The dataset contains 922,262 ratings (1-5 scales) from 22,166 users on 296,227 items, which are divided into 27 categories. We extract a subset of 1317 users who have at least 100 ratings for items. Hence, the extracted dataset contains 245,042 ratings from 1317 users on 54,985 items.

The extracted dataset is divided into two parts as a training set and a testing set. The experiment is performed on training set to make a prediction for the test set. Mean Absolute Error (MAE) is used to evaluate the efficiency. It is used to measure the closeness of predicted ratings to the true ratings.

4.7.2 Experimental results

For the first experiment, Figure 4.4 shows the result with the different number of neighbors set, and the training set equals 80%. It can show that the RHF-CF algorithm is outperform than others. We choose the number of neighbors equals to 10 that is the lowest number before the accuracy is not changed for all methods. After that, we do the experiment with the different size of training set from 50% to 90%. The result is shown in Figure 4.5 that can substantiate that our proposed method gets higher prediction accuracy than another two algorithms.

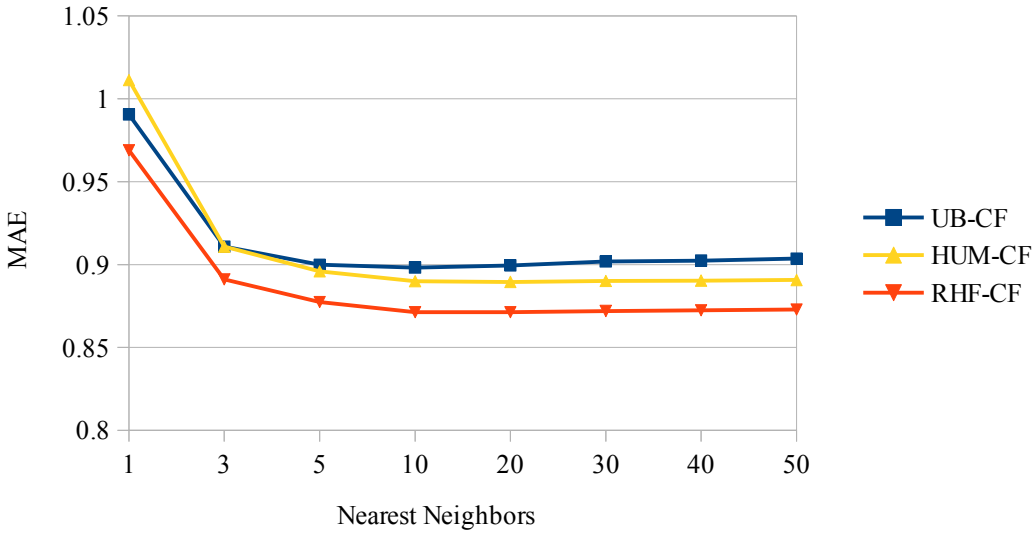

Figure 4.4: Comparison at different number of neighbor set

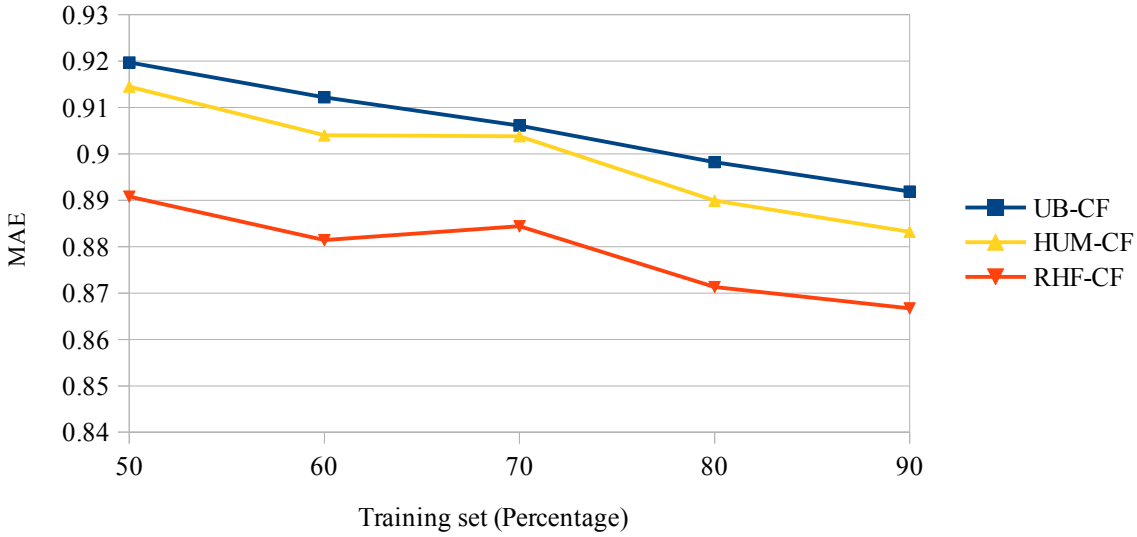

Figure 4.5: Comparison at different size of training set and neighbor =10

For the second experiment, we chose the training set, equals 80% and the testing set equals 20%. The different number of the neighbor set is 1, 3, 5, 10, 20, 30, 40, 50. The results are shown in Figure 4.6. It can show that the RHT-CF algorithm outperforms than RHF-CF. To achieve that, the RHT-CF algorithm gets better results than others overall cases. Therefore, we set the number of neighbors equal to 10 that is the lowest number of neighbors before the accuracy is not changed for all methods. After that, we did the experiments with the different size of training set from 50% to 90%. In Figure 4.7, it can prove that the RHT-CF algorithm gets a higher prediction accuracy than RHF-CF.

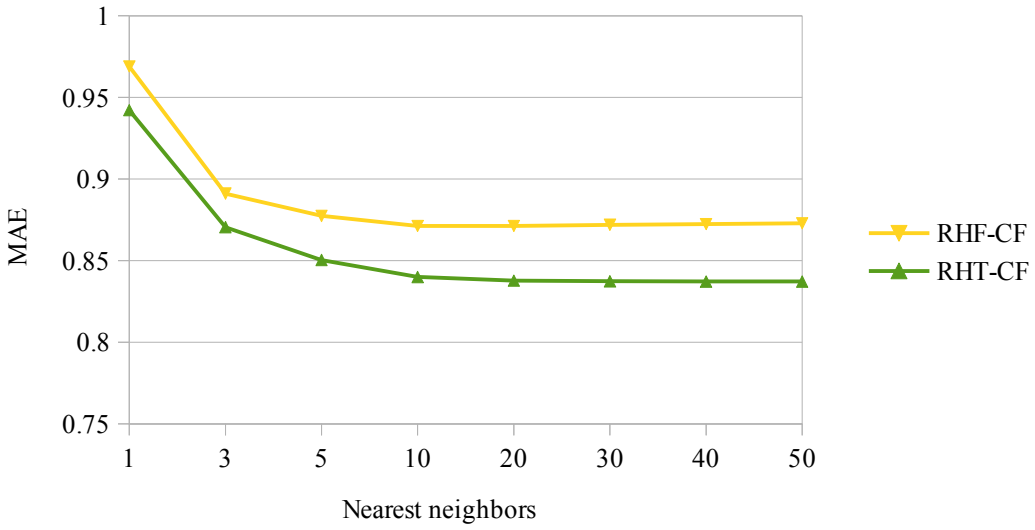

Figure 4.6: Comparison at different number of neighbor set

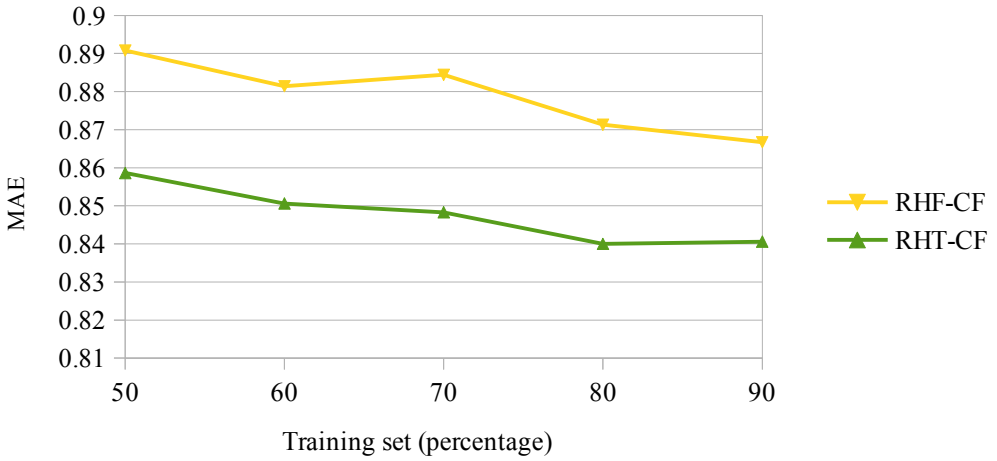

Figure 4.7: Comparison at different size of training set and neighbor =10

4.8 Discussion

In this chapter, the second part of the proposed method is known as a hybrid recommendation system with review helpfulness features and then the second part of the proposed method is an improved algorithm by using a review helpfulness trust. The experiment is divided into two parts. The first part implements a hybrid recommendation system using review helpfulness features. The second part implements an improvement algorithm using review helpfulness trusts. The results showed that this research achieved the aim to deal with the scalability problems and improves the prediction accuracy.

In the first experiment, the hybrid model is constructed by using the combination of similarity in opinion and similarity in preference. At the online phase, the similarity of users is computed based on the constructed hybrid model. When we compare the size of the original data matrix of user-based collaborative filtering with the matrix of the RHF-CF algorithm, we found that the size of the original data matrix is cut down from m*n to m*(p+p), where m is the number of users, n is the number of items, and p is the number of features, especially p << n. Therefore, the similarity computation cost in the RHF-CF algorithm is less than user-based collaborative filtering technique.

In the second experiment, the review helpfulness trust is estimated by Jaccard coefficient from review rating at the offline phase. At the online phase, the similarity of users is computed based on the constructed hybrid model and review helpfulness trust. The review helpfulness trust is added into the user similarity computation which helps to get a more accurate user similarity. The experiment results show that the improvement algorithm RHT-CF outperforms than other algorithms.

CHAPTER 5 CONCLUSION AND FUTURE WORK

5.1 Comparison of the three factors utilization

In this section, the comparison of the three factors utilization is presented. The extractions of three factors are implemented in the offline phase, so these extractive processes are not an effective performance of recommendation generation. We just need to compare the three factors utilization in the online phase. There are two main steps in the online phase: user similarity computation and predicted unknown rating. Because there are no differences in the predicted unknown rating step, this section will focus on the comparison of similarity computation of user between three factors utilization, in terms of computational complexity. Table 5.1 shows the comparison of user similarity computation.

Table 5.1: Comparison of user similarity computation

<table>
<tr><th></th><th>CF</th><th>CF-ULRW
(Users' relationship)</th><th>RHF-CF
(Features)</th><th>RHT-CF
(Trust)</th></tr>
<tr><td rowspan="2">Computational complexity</td><td>O(m*n)</td><td>O(m*k)</td><td>O(m*(p+p))</td><td>O(m*(p+p))</td></tr>
<tr><td>m: users
n: items</td><td>k: top largest singular value and it is a variant variable</td><td colspan="2">p: number of features and it is invariant variable.</td></tr>
<tr><td rowspan="2">In real application</td><td colspan="4">p<k<<n. So user similarity computation cost of the first part is not as good as the second and the third part.</td></tr>
<tr><td colspan="2">Easily consist and apply to all available databases</td><td colspan="2">Variable p is not always available. The existence of p depends on the product content.</td></tr>
<tr><td>Conclude</td><td colspan="4">The third part should be used to make a recommendation system more than the others if the review rating and product features are available in the database.</td></tr>
</table>

In the first part, the users' latent relationships are used Pearson correlation coefficient to compute the similarity between users. This user similarity matrix is cut down from (m*n) to (m*k), where m is the number of users, n is the number of items and k is the top largest singular values of training matrix reconstruction. From the experiment, we got k << n.

The review helpfulness feature and the review helpfulness trust, which are respectively introduced in the second and the third parts, are combined in the user similarity computation formula. This user similarity matrix is cut down from (m*n) to (m*(p+p)), where m is the number of users, n is the number of items and p is the number of product features.

In a database, p, is an invariant variable because it is the number of product features, and k is a variant variable because it is randomly selected based on data dimensionality reduction. In the real data, $p << n$. So we can conclude that $p < k$ in the real application. So the computation cost of user similarity in the first part is not as good as the second and the third part. However, the review helpfulness features and the review helpfulness trust are extracted from review rating. Information of review rating and the variable p are not always available. The existence of the variable p depends on the product content. The first part, which is implemented with the only information about product rating, is easily consisted and applied to all available databases. The second part and the third part should be used to make a recommendation system more than the first part if the review rating and product features are available in the database.

5.2 Conclusion and future work

Based on the results of the various experiments, we can conclude that our proposed method outperform other methods with the same dataset, in terms of accuracy. The first part of the proposed algorithm is an enhanced user-based collaborative filtering with users' latent relationships weighting, called CF-ULRW. The second part of the proposed method is an enhanced recommendation system by using review helpfulness features, called RHF. The third part of the proposed method is an improvement of the second part by adding review helpfulness trusts, called RHT-CF. The users' latent relationships are extracted from product rating. The review helpfulness features and the review helpfulness trusts are extracted from review rating.

The CF-ULRW method can deal with the sparse data problem as well as improve the performance system. This is possible due to the low computation cost of this method, which is more efficient compared to the user-based collaborative filtering (CF) and the rowsampling approximating singular value decomposition algorithm (ApproSVD). The

column-sampling approximating singular value decomposition method, which is investigated in CF-ULRW method, is a good method to normalize the left singular vectors. The left singular vectors are used to obtain the best low-rank approximating user data matrix, called users' latent relationships. Users' latent relationships weighting, which is used in the predicted rating process, is measured by the Pearson correlation efficiency with lowest computation cost. It helps to improve the quality of user similarity computation which plays an important role in an accurate prediction. Via the experiment results, we can demonstrate that users' latent relationships weighting utilization is an effective factor to improve the accurate prediction as well as the performance of a recommendation system.

RHF-CF method, which is investigated by using review helpfulness features, can deal with scalability problem and improve the prediction accuracy. RHF-CF method constructs a hybrid model by using the combination of similarity in opinion and similarity in preference. The hybrid model is constructed in offline phase, so it helps to reduce computational cost as well as improve the performance of recommendation system. The prediction is implemented based on the hybrid model on the online phase with low computation cost. The experiment results demonstrate that RHF-CF method outperforms the user-based collaborative filtering algorithm and the hybrid user model collaborative filtering algorithm.

RHT-CF method, which is investigated by using review helpfulness trusts, is used to improve the quality of RHF-CF method, in terms of accuracy. Review helpfulness trusts are added to the user similarity computation. This addition increases the accuracy of user similarity computation as well as the accuracy of prediction.

For the future work, we plan to use a classification technique with the second proposed method. The classification techniques will be applied to the hybrid model to reduce the data matrix size. Besides, it is easy to add new features into the hybrid model with low computational cost.

REFERENCES

[1] A. Gulli and A. Signorini. “The indexable web is more than 11.5 billion pages.” In *Proceedings of 14th International World Wide Web Conference*, Chiba, Japan, 2005. pp. 902-903

[2] Internet usage statistics. “The big picture of the world Internet users and population status.” *http://www.internetworldstats.com/stats.htm*

[3] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. GroupLens. “An Open Architecture for Collaborative Filtering of Netnews.” *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 1994

[4] P.Resnick and H.R. Varian. “recommendation systems. “volume 40, *ACM Press* , 1997.

[5] M.J. Pazzani, D. Billsus. “Content-Based Recommendation Systems.” Springer-Verlag, *Berlin Heidelberg*, 2007

[6] X. Su, T.M. Khoshgoftaar. “A Survey of Collaborative Filtering Techniques.” *Hindawi Publishing Corporation Advances in Artificial Intelligence,* Volume 2009.

[7] Gunawardana, C. Meek. “A Unified Approach to Building Hybrid recommendation Systems.” *ACM conference on recommendation system*, pp. 117-124, 2009.

[8] Sarwar, B. Karapis, G. Konstan, J. Riedl. “Application of dimensionality reduction in recommendation system-a case study.” *Proceedings of the WebKDD 2000 Web Mining for E-Commerce Workshop at ACM SIGKDD*, pp. 421–428. ACM, 2000.

[9] Zhou, X., He, J., Huang, G., Zhang, Y. “A personalized recommendation algorithm based on approximating the singular value decomposition (ApproSVD).” *In: IEEE/WIC/ACM*, 2012.

[10] G. Adomavicius, A.Tuzhilin. “Toward the next generation of recommendation systems: A survey of the state-of-the-art and possible extensions.” *Knowledge and Data Engineering, IEEE Transactions on*, 17(6), 734-749 , 2005.

[11] H. Lieberman. Letizia. “An Agent That Assists Web Browsing.” *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995.

[12] R. Meteren and M. Someren. “Using Content-Based Filtering for Recommendation.”

[13] A. Jennings and H. Higuchi. “A personal news service based on a user model neural network.” 1992.

[14] Goldberg, K. Roeder, T. Gupta, D. Perkins, C. “Eigenstate: A Constant Time Collaborative Filtering Algorithm.” *Information Retrieval 4(2*), pp. 133–151, 2001.

[15] N. Belkin and B. W. Croft. “Information filtering and information retrieval: two sides of the same coin.” *ACM Press*, 1992.

[16] N. Bendakir, E.,A . “Using association rules for course recommendation.” *In Proceedings of the AAAI Workshop on Educational Data Mining*, pp. 31-40, 2006.

[17] T. Chellatamilan, R. SURESH. “An e-Learning Recommendation System using Association Rule Mining Technique.” *European Journal of Scientific Research Vol. 64 No*, 2, pp. 330-339 , 2011.

[18] X. Amatriain. “recommendation systems.” *In MLSS’14 Pittsburgh*, 2014.

[19] J. Beel, S. Langer, M. Genzmehr, B. Gipp, and A Nürnberger. “A comparative analysis of offline and online evaluations and discussion of research paper recommendation system evaluation.” *In ACM, editor, RepSys ’13 Proceedings of the International Workshop on Reproducibility and Replication in recommendation Systems Evaluation*, p.p 7–14, 2013

[20] Ricci, F. Rokach, L. Shapira, B. Kantor, and P.B. “recommendation Systems Handbook.” *Springer*, 2011.

[21] M.P. Robillard, W. Maalej, R.J. Walker, and T. Zimmermann. “Recommendation Systems in Software Engineering.” *Springer*, 2014.

[22] B. Krulwich, C. Burkey. “The Infofinder Agent: Learning user interests through heuristic phrase extraction.” *IEEE Intelligent Systems and Their Applications*, pp. 22-27, 1997.

[23] D. Billsus, M.J Pazzani. “A personal news agent that talks learns and explains.” *Proceedings of the 3rd International Conference on Autonomous Agents*, pp. 268-275. Seattle, WA, USA, 1999.

[24] D. Billsus, M.J Pazzani. “User modeling for adaptive news access.” *User Modeling and User-Adapted Interaction*, pp. 147-180, 2000.

[25] K. Lang. “Newsweeder: learning to filter net news.” *Proceedings of the 12th International Conference on Machine Learning (ML 1995*). Tahoe City, CA, USA, pp. 331-339, 1995.

[26] M. Balabanovi, Y.Shoham. “Fab: content-based, collaborative recommendation.” *Communications of the ACM,* pp. 66-72, 1997.

[27] Burke. “Hybrid recommendation Systems: Survey and Experiments. User Modeling and User-Adapted Interaction,” pp. 331-370, 2002.

[28] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. “Using collaborative filtering to weave an information tapestry.” volume 35. *ACM Press*, pp. 61-70 , 1992.

[29] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. “An algorithmic framework for performing collaborative filtering.” *ACM Press*, 1999 .

[30] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. “GroupLens: An Open Architecture for Collaborative Filtering of Net news.” *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 1994 .

[31] G. Hill, L. Stead, M. Rosenstein. “Recommending and evaluating choices in a virtual community of use.” *Proceedings of the 13th International Conference on Human Factors in Computing Systems*, pp. 194-201, 1995.

[32] GroupLens Research. “Movielens dataset.” http://grouplens.org/datasets/movielens/

[33] M. Deshpande and G. Karypis. “Item-based top-N recommendation algorithms.” *ACM Trans.*, 2004.

[34] U. Shardanand, P. Maes. “Social information filtering: algorithms for automating 'word of mouth'.” *Proc. Conf. Human factors in computing systems*, 1995.

[35] Michael J. Pazzani. “A Framework for Collaborative, Content-Based and Demographic Filtering.” *Artificial Intelligence Review*, 1999 .

[36] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. “Combining Content-Based and Collaborative Filters in an Online Newspaper,” 1999.

[37] R. Burke. “Hybrid recommendation systems: survey and experiments.” *User Modeling and User-Adapted Interaction*, pp. 331-370 , 2002.

[38] I. Soboroff and C. Nicholas. “Combining content and collaboration in text filtering,” 1999 .

[39] C. Basu, H. Hirsh, and W. W. Cohen. “Recommendation as Classification: Using Social and Content-Based Information in Recommendation.” *AAI/IAAI*, 1998.

[40] Osmanlı, O.N., Toroslu, I.H. “Using Tag Similarity in SVD-Based Recommendation Systems.” *The 5th International Conference on Application of Information and Communication Technologies*, pp. 1–4. IEEE, 2011).

[41] Sarwar, B. Karapis, G. Konstan, J. Riedl. “Application of dimensionality reduction in recommendation system-a case study.” *Proceedings of the WebKDD 2000 Web Mining for E-Commerce Workshop at ACM SIGKDD*, pp. 421–428. ACM, 2000.

[42] C. Basu, H. Hirsh, and W. W. Cohen. “Recommendation as Classification: Using Social and Content-Based Information in Recommendation.” *AAI/IAAI*, 1998.

[43] C. Christakou, S. Vrettos, A. Stafylopatis. “A hybrid movie recommendation system based on neural networks.” *International Journal on Artificial Intelligence Tools*, p.p 771-792., 2007.

[44] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, M. Sartin. “Combining content-based and collaborative filters in an online newspaper.” *Proceedings of the SIGIR Workshop on recommendation Systems: Algorithms and Evaluation* , 1999.

[45] M.W. Drineas, P. Kannan, R. Mahoney. “Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix.” p.p 158-183., 2006.

[46] Schafer, J.B. Frankowski, D. Herlocker, J. Sen. “Collaborative filtering recommendation systems.” *LNCS*, vol. 4321, pp. 291–324. Springer, Heidelberg, 2007 .

[47] Hu, R. Lu. “A Hybrid User and Item-based Collaborative Filtering with Smoothing on Sparse Data.” *The 16th International Conference on Artificial Reality and Telexistence Workshops*, pp. 184–189. IEEE CS Press, 2006.

[48] Chen, Y. Yu. “A Hybrid collaborative Filtering Algorithm Based on User-Item.” *International Conference on Computational and Information Science,* pp. 618–621. CPS , 2010.

[49] Ziegler, C.N. Lausen, G. Schmidt-Thieme. “Taxonomy-driven computation of product recommendations.” *Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management*, pp. 406–415. ACM, New York, 2004.

[50] Zeinab, S. Mansoor, R. Mahdi. “New Algorithm for recommendation Systems based on Singular Value Decomposition.” *International eConference on Computer and Knowledge Engineering*, pp. 86–91. IEEE, 2013.

[51] Goldberg, K. Roeder, T. Gupta, D. Perkins. “Eigenstate: A Constant Time Collaborative Filtering Algorithm.” *Information Retrieval 4(2)*, 133–151, 2001.

[52] Yin, C. Peng. “A Careful Assessment of Recommendation Algorithms related to dimension reduction techniques.” *Knowledge-Based Systems 27*, pp. 407–423, 2012.

[53] Ren, Y. Gong. “A Collaborative Filtering Recommendation Algorithm Based on SVD smoothing.” *The 3rd International Symposium on Intelligent Information Technology Application (IITA 2009)*, vol. 2, pp. 530–532. IEEE, 2009.

[54] Qilong, B. Xiaoyong, L. Zhongying. “Clustering Collaborative Filtering Recommendation System Based on SVD algorithm.” *IEEE 4th International Conference on Software Engineering and Service Science*, pp. 963–967. IEEE, 2013.

[55] Abdelwahab, A. Sekiya, H., Matsuba, I. Horiuchi, Y. Kuroiwa, S. Nishida. “An efficient collaborative filtering algorithm using SVD-free Latent Semantic Indexing and particle swarm optimization.” *International Conference on Natural Language Processing and Knowledge Engineering*, pp. 1–4. IEEE, 2009.

[56] S. Christensen, I. Schiaffino. “Matrix Factorization in Social Group recommendation Systems.” *The 12th Mexican International Conference on Artificial Intelligence, pp. 10–16. IEEE*, 2013.

[57] I.H. Osmanlı, O.N. Toroslu. “Using Tag Similarity in SVD-Based Recommendation Systems.” *The 5th International Conference on Application of Information and Communication Technologies*, pp. 1–4. IEEE, 2011.

[58] S. Gupta, R. Jain, A. Rana, S. Singh. “Contextual information based recommendation system using Singular Value Decomposition.” *IEEE* (2013) 978-1-4673-6217-7/13 .

[59] S. Qian, W. Xianhu Y. Min. “Collaborative filtering recommendation algorithms based on hybrid user model.” *International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, 2010.

[60] T.K. Farman, U. Ghulam, S. Sung, L. Yun, K.P. Kyeong, M. Jin. “Hybrid recommendation system with temporal information.” *ICOIN*, 2012, pp.421-425.

[61] D. Chen, D. Xu. “A collaborative filtering recommendation based on user profile weight and time weight.” *Computational Intelligence and Software Engineering (CiSE)*, 2009 .

[62] H.Z. Yang, L. Li. “An enhanced collaborative filtering algorithm based on time weight.” *International Symposium on Information Engineering and Electronic Commerce*, 2009.

[63] J. Wang, J. Yin, Y. Lie, C. Huang. “Trust-based collaborative filtering.” *In Proc. 8Th Int. Fuzzy Systems and Knowledge Discovery*, pp. 2650-2654, 2011.

[64] P. Massa, P. Avesani. “Trust-aware recommendation systems.” *In Proc. recommendation systems*, pp. 17-24, 2007.

[65] S. Ray, A. Mahanti. “Improving prediction accuracy in trust-aware recommendation systems.” I*n Proc. 43Rd Int. System Sciences*, pp. 1-9, 2010.

[66] J. Otterbacher. “Helpfulness in online communities: a measure of message quality.” *In proceedings of the 2009 ACM CHI*, April 7, 2009, pp.955-964

[67] Jiliang Tang. “Product Review Datasets: Epinions and Ciao.” http://www.public.asu.edu/~jtang20/datasetcode/truststudy.htm

APPENDIX A: LIST OF PAPERS PUBLISHED

1. T.T. Thuan, P. Sutheera, "An enhanced user-based collaborative filtering recommendation system using the users' latent relationships weighting utilization", *Asia Simulation Conference (AsiaSim), CCIS 474*, pp. 153-163, Kitakyushu, Japan, 26 Oct – 28 Oct 2014.

2. T.T. Thuan, P. Sutheera, "Hybrid recommendation system with review helpfulness features," *IEEE -TENCON*, Bangkok, Thailand, 22 Oct – 25 Oct 2014.

3. T.T. Thuan, P. Sutheera, "An enhanced hybrid recommendation system with review helpfulness trust," *Regional conference on computer and information engineering (RCCIE*), pp. 172 – 177, Yogyakarta, Indonesia, 06 Oct- 08 Oct 2014.

AUTHOR BIOGRHAPY

Personal Information

Name	To Thi Thuan
Sex	Female
Nationality	Vietnamese
Date of Birth	07th April 1987
Place of Birth	Hanoi, Vietnam

Education

Bachelor Degree

Project	Information Extractions
Department	Department of Information System
Faculty	School of Information and Communication Technology
University	Hanoi University of Science and Technology, Vietnam
Duration	2005-2010

Master Degree

Thesis	An Enhanced Hybrid Recommendation System Based on Dimensionality Reduction Techniques
Field of Study	Computing in Engineering System
College	International college
University	King Mongkut's Institute of Technology Ladkrabang
Duration	2013 - 2015