UTILIZING LOCAL AND GLOBAL SIMILARITIES TO IMPROVE ACCURACY OF RECOMMENDER SYSTEMS

Dissertation submitted to Jawaharlal Nehru University, in partial fulfillment of the requirement for the award of the Degree of

MASTER OF TECHNOLOGY
In
COMPUTER SCIENCE AND TECHNOLOGY

By

DEEPA ANAND

Under the Guidance of
PROF. K. K. BHARADWAJ

SCHOOL OF COMPUTER & SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI -110067
JULY 2009

JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110067

CERTIFICATE

This is to certify that the dissertation entitled "**Utilizing Local and Global Similarities to Improve Accuracy of Recommender Systems**", being submitted by **Deepa Anand** to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, in partial fulfillment of the requirement for the award of the Degree of **Master of Technology in Computer Science and Technology**, is a bona fide work carried out by her under the guidance and supervision of **Prof. K. K. Bharadwaj**.

The matter embodied in the dissertation has not been submitted for the award of any other Degree or Diploma.

Deepa Anand
(Student)

16-07-09

Prof. K. K. Bharadwaj
(Supervisor)

Dean, SC&SS
Jawaharlal Nehru University
New Delhi-67
India.

Acknowledgement

It gives me immense pleasure in expressing my gratitude to all the people who have made this dissertation possible. First and foremost I would like to thank my supervisor Prof. K.K.Bharadwaj, who has taken more interest in my dissertation than myself. His interest and enthusiasm is infectious. Working with him has been an excellent learning experience. The dissertation would not have been possible without his encouragement, support and patient guidance.

I also gratefully acknowledge Dean, Prof. Parimala N. for providing the necessary infrastructure to carry out this dissertation. Thanks to my husband, V.Anand, who has been extremely understanding and supportive and just being there for me whenever I needed him. Thanks to my in-laws without whose support, this dissertation would have been impossible. Thanks to my parents for all their efforts in making me the person I am. And last but not the least, thanks to my little gems, Vishakha and Rishabh for making do without me, during the time I was working on this dissertation.

Deepa Anand

Abstract

The explosive growth of the web has led to the problem of 'information overload' - the overwhelming plethora of choices and options available to a user, often varying in quality. Recommender systems have emerged as a solution to this problem by filtering and presenting the users with information, products services etc. according to their tastes and preferences. The success of recommender systems is well illustrated by their applications in a variety of domains ranging from books, CDs, movies to vacations, mutual funds, radio stations etc. Collaborative Filtering is one of the most popular recommendation techniques, which recommends items to users, based on the preferences of users having similar tastes. In spite of their huge success they suffer from a range of problems, the most fundamental being the data sparsity. When the rating matrix is sparse, local similarity measures yield a poor neighborhood set thus affecting the recommendation quality. In such cases global similarity measures can be used to enrich the neighborhood set by considering transitive relationships among users even in the absence of any common experiences.

In this work we propose a recommender system framework utilizing both local and global similarities, taking into account not only the sparsity in the rating data, but also sparsity at the user level and the item level. Several sparsity measures, based on the active user and the item, are proposed in this work. Experimental results demonstrate that incorporating the user and item based sparsity measures in the weighting scheme for combining local and global similarities, outperform the fixed-α schemes, on accuracy of predicted ratings.

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter 1

INTRODUCTION

The advent of the internet and the ease of producing content, has led to an explosive growth of the amount of information available. Users are burdened with a staggering number of choices and options, and find it impossible to navigate through, and choose from. In a report, interestingly titled, "Dying for Knowledge", by Reuters, it was found that knowledge workers in Britain, faced with information overload, often suffered from increased anxiety and sleeplessness. This phenomenon is now recognized as a medical condition termed 'Information Fatigue Syndrome', coined by British psychologist Dr. David Lewis [Waddington, 1996].

The need for a solution to this abundance of information and the drive to bring the vendor and customer closer in e-commerce has led to popularity of web personalization.Web personalization can be described, as any action that makes the Web experience of a user personalized to the user's taste[Mobasher, et al, 2000]. Jeff Bezos, CEO of Amazon, expressed it as, "If I have 3 million customers on the Web, I should have 3 million stores on the Web" [Schafer et al, 2001].

Web recommender systems are the most notable application of Web Personalization.
They help users find the right information at the right time based on learnt user preferences. Research on recommender systems has been going on for more than a decade now, but with the increase in the number of e-commerce applications, online users, vendors and increasingly complex products and services, the demand for new intelligent recommendation techniques has also increased dramatically.

The following sections give a brief overview of recommender systems, various recommendation techniques, the various approaches to solving the data sparseness problem, the problem statement and the organization of the dissertation.

1.1 Recommender Systems

"The Web, they say, is leaving the era of search and entering one of discovery. What's the difference? Search is what you do when you're looking for something. Discovery is when something wonderful that you didn't know existed, or didn't know how to ask for, finds you." ***Jeffrey M. O'Brien in 'The race to create a "Smart" Google'***

The above quote captures the essence of what recommender systems are all about. Recommender systems are personalization tools which enable users to be presented information suiting his interests, which are novel, serendipitous and relevant, without being explicitly asked for. Users are, in general, inept at expressing their needs or may not have an exact idea of what they want. Recommender systems enable users to be presented items which they may not know of, thus supporting "discovery" rather than "search". In addition to reducing the search time for interesting items, they also enhance e-Commerce sales by converting browsers into buyers, increasing cross-sell and building consumers' loyalty [Scharer, et al 2001]. Recommender Systems have found their way into many entertainment and e-commerce web sites and not only help people find items of interest but also form communities of interest [Terveen & Hill, 2001]. Recommender systems have become ubiquitous, with their presence everywhere from recommending books (Amazon), CDs, music to recommending high risk products such as mutual funds and vacations.

More formally, the recommendation problem can be formulated as follows: Let C be the set of all users and let S be the set of all possible items that can be recommended, such as books, movies, or restaurants. Let u be a utility function that measures the usefulness of item s to user c, i.e., u : C X S -> R, where R is a totally ordered set. Then, for each user $c \in C$, we want to choose such item $s' \in S$ that maximizes the user's utility. More formally: $\forall c \in C,\ s'_c = \arg\max_{s' \in S} u(c,s)$ [Adomavicius & Tuzhilin, 2005].

Recommender systems exploit the ratings provided by users to different items to elicit user preferences. The rating can be explicit or implicit. These preferences are then used to generate recommendations.

1.2 Recommendation Methodologies

Recommender systems can be classified based on the sources of data on which the recommendation is based, the background data available, and the algorithm used to combine the input data and the background data to arrive at suggestions [Burke, 2002].

1.2.1 Collaborative Filtering

Collaborative Filtering [Goldberg et. al, 1992] is the automation of "word of mouth" [Shardanand &Patti, 1995], where opinions gleaned from people, who share similar tastes as the active user, is used in the decision making process. It is based on the assumption that users who have agreed in the past tend to agree in the future. The utility of an item for a particular user in collaborative recommender is computed by aggregating ratings of users "similar" to the active user. The items may then be arranged in descending order of their utility values in order to give the "top-N" recommendation to the user. The user profile consists of a vector of ratings for all the items in the system.

The following three steps are involved in recommendations using Collaborative Filtering

- **Preference elicitation:** Users generally state their preferences by rating items to express their like or dislike for a particular item.
- **Neighborhood Formation:** Based on the preferences expressed in the previous step the active user is matched with other users having similar tastes, to give a set of neighbors.
- **Prediction:** Ratings from all neighbors who have rated the item in question are aggregated to arrive at the predicted rating for the active user.

A different approach popularized by Amazon [Linden et.al. 2003] is the *item-based* CF, where associations among items are established using historical ratings information. The rating prediction for an item is made based on the ratings by the user for all "similar" items. Item-based CF performs better than its user-based counterpart when the number of items is relatively static [Sarwar et. al. 1998].

There are two broad approaches to the design of collaborative recommender systems: *memory-based* and *model-based.*

Memory Based CF

Memory-based algorithms are heuristics based algorithms, which utilize the entire rating history to arrive at predictions. These include the commonly implemented class of user-based and item-based CF methods. These algorithms work by finding a set of users who have the same tastes (neighbors) as that of the active user and who have experienced the item to be rated. Once a neighborhood set is formed, rating for the item in question is decided based on the ratings given by the neighbors. These algorithms are the most popular and are widely in use [Resnick et.al. 1994; Jameson et.al, 2003]. Memory based CF algorithms offer more accuracy since prediction computation happens in real-time and the effect of any new ratings is experienced instantaneously. However these algorithms don't scale well, since they are memory intensive. The online computation of recommendations especially proves to be a bottleneck in online recommender systems, with millions of customers and items and thousands of recommendations to be made per minute.

Model Based CF

Model-based recommender systems build a user-model in an off-line learning phase and then apply this model on-line for recommendation. User models can be generated using different machine learning techniques such as clustering [Kim & Ahn, 2008], decision trees, case-based reasoning, neural networks, and Bayesian networks [Manouselis &

Costopoulou, 2007]. Model-based CF algorithms are generally probabilistic in nature where the expected value of the rating is computed. Model-based recommender systems have lower memory requirement and a rapid recommendation process but lack the accuracy of the memory-based algorithms.

According to [Koren, 2008] the boundary between memory based and model based techniques is blurring with many memory based algorithms relying on rigorous models and some model based algorithms improving their accuracy by examining the entire rating database. Memory-based and model-based approaches have been combined in [Al-Shamri & Bharadwaj, 2008] using a fuzzy-genetic approach that retains the accuracy of memory-based CF and the scalability of model-based CF.

Collaborative Recommender Systems are popular due to the several advantages they offer, namely,

- Serendipitous or "out of the box" recommendations are possible.
- Filtering based on the quality and taste is made possible since humans are capable of analyzing such dimensions which is often hard for computers.
- These methods are domain free, since they do not rely on content-description and hence work equally well for news articles as for images

CF methods, however, have some limitations

- **New User problem:** A user needs to rate sufficiently large number of items in order to receive quality recommendations.
- **Latency:** The latency problem is defined as lack of sufficient user-ratings on items in an ACF system to provide enough ratings overlap for good recommendations. The "gray-sheep" problem is when a user has unique,

unusual tastes and hence does not have sufficient overlap with most users. The first few users to rate a new item, get little benefit from doing so, and this problem is known as "new-item" or the "early-rater" problem.

- **Sparsity**: As the number of items in a system increase in a recommender system, the most prolific of users rate only a small subset of items. Hence the overlap of items between any pair of users (for similarity computation) would be very less. This sparsity of the user-item matrix may lead to finding neighbors based on a small number of common items which may in turn affect the quality of recommendation. There have been various approaches to tackling the sparsity problem some of which are discussed in the next section.

- **Scalability**: In spite of more than a decade old research in the field of recommender systems, most systems do not scale well especially when dealing with large volumes of users and items. [Bell et al, 2007] propose novel algorithms for predicting user ratings of items by integrating complementary models that focus on patterns at different scales, which are accurate and scalable.

- **Loss of neighborhood transitivity**: Assume user u_a is very similar to user u_b and user u_b is very similar to user u_c, then it is possible that users u_a and u_c are similar via their transitive relation through user u_b, even if they have no rating overlap. But such a transitive relationship is not explored in the Collaborative Filtering systems.

1.2.2 Content Based Filtering

Content-based recommendation is an outgrowth and continuation of information filtering research and is based on the idea of recommendation as classification [Burke, 2005]. These systems recommend items similar to those the user preferred in the past. In content-based filtering the user preferences are elicited from the content description of items that the user has highly rated. The prediction for an unrated item is done on the basis of a match between the item's content and the user preference. Since content-based methods require a description of items, they generally work well in domains in which items have rich textual descriptions. An example of content-based recommendation is the personalized Google news.

There have been various approaches to learn the content-based profile of the user employing decision-trees, neural networks, Bayesian networks and clustering [Adomavicius & Tuzhilin, 2005; Burke, 2002]. Typically the item contents are described using the K most important keywords. The importance of a word in a particular document can be measured in a number of ways the most popular being the TF-IDF scores [T.Joachims, 1997].

Advantages of Content-based filtering methods:

- They are adaptive i.e. quality of recommendations improves with user feedback.
- They do not require any domain knowledge except for the content description.
- They do not suffer from the new item problem.
- They can provide explanations of recommended items by listing content-features that caused an item to be recommended.

Disadvantages of Content-based filtering methods:

- They suffer from cold-start problem, since the user needs to rate a sufficient number of items for the system to learn his preferences.

- They are unable to provide serendipitous recommendations i.e. there is no surprising item, all items recommended are similar to items already liked by the user.

- They are limited by the content-description of the item and are harder to apply in case of multimedia data where the automatic feature extraction is much harder.

1.2.3 Demographic Filtering

In demographic recommender systems, users are classified into classes based on their demographic attributes such as age, occupation, gender etc. and recommendations are then made based on these demographic classes. One of the most popular recommender systems is the Lifestyle Finder [Krulwich, 1997], which assigns users to one of 62 pre-existing clusters based on a few questions concerning the user's lifestyle. Recommendations are then made based on the cluster to which the user belongs. Another approach to collecting demographic information is through user's web pages [Pazzani, 1999]. Demographic filtering can be combined with other filtering techniques especially Collaborative filtering in order to overcome the drawbacks therein [Vosalis and Margaritis, 2007]. The main advantage of demographic filtering is that it does not rely on user's rating history thus avoids the new-user problem present in the above two methods. The major disadvantage is that there may be many users who don't fall in any demographic cluster.

1.2.4 Utility Based Recommender Systems

Unlike the two methods discussed above utility-based recommender systems do not attempt to construct a long term user profile. Suggestions are based on a computation of the utility of each item for the user for whom a utility function is stored [Manouselis & Costopoulou, 2007]. The utility function may be gathered using a dialogue between the system and the user, to infer which product features does the user emphasize on. For example to recommend a bike , Personalogic , an e-commerce site, asks the user how important the bike features are, such as frame durability, weight etc. Tête-à-tête is another example of a utility based recommender system. The advantage of utility based recommender system is that it can take into account non-product features such as product reliability, delivery date etc. [Burke, 2002].

1.2.5 Knowledge Based Recommender Systems

Knowledge based recommender systems use information about items and users in order to draw inferences about user requirements. They use functional knowledge i.e. how an item meets a particular user requirement to draw these inferences. Popular knowledge based recommender systems are the *recommender.com* and the Entrée system [Burke, 2002]. A recent knowledge based recommender system which is part of the Social Web is the StumbleUpon system (www.stumbleupon.com). Though knowledge based recommender systems can map user needs to products, it requires product domain knowledge which has to be stored and organized properly, thus requiring the services of a knowledge engineer. They however do not suffer from the drawbacks such as new-user or new-item problem.

1.3 Hybrid Recommender Systems

The different filtering techniques discussed above have their own strengths and shortcomings. Hybrid filtering systems combine different filtering techniques which complement each other to offer the benefits of all the methods. The different hybridization techniques as discussed in [Burke, 2002] are weighted, switching, mixed, feature combination, cascade, feature augmentation, or meta-level (see Table 1.1). Fab [Balabanovic & Shoham, 1997] uses a hybrid of collaborative and content based filtering. Collaborative filtering is hybridized with content based and demographic filtering in [Al-Shamri, Bharadwaj, 2008]. User's ratings are integrated with some content descriptions of the items to build a compact user model. The compact features so obtained are then combined with demographic features in order to find similar users.

Hybridization method	Description
Weighed	The scores(votes) of different techniques are merged together to produce a single recommendation . e.g. Entrée[Burke, 2002]
Switching	The system switches between techniques depending on the situation.
Mixed	Recommendations from different techniques are presented to the user at the same time.
Feature combination	Features from different recommendation data sources are thrown together into a single recommender system.
Cascade	One RS refines the recommendations given by another.
Feature augmentation	Output from one filtering technique is used as an input feature to another.
Meta□level	The user model learned by one RS is used as input as input to another.

Table 1.1 Hybridization Methods[Burke, 2002]

1.4 The Sparsity Problem in Collaborative Filtering

Collaborative filtering techniques are the most popular and effective as compared to the other filtering techniques. However sparsity of ratings data is one of the key challenges to these systems. In any recommender system, the number of ratings already obtained is usually very small compared to the number of ratings that need to be predicted [Adomavicius & Tuzhilin, 2005]. The user must rate a sufficient number of items in order to get a reasonable profile overlap in order for the system to establish his neighbors. The poor quality of recommendations at high sparsity levels, is established [Gřcar, et al. 2006]. Over the years a variety of solutions to this problem have been proposed.

Model-based approaches such as singular value decomposition can help lessen the effect of data sparsity by reducing the dimensionality of the space in which comparison takes place [Rosenstein & Lochbaum, 2000; Billsus & Pazzani, 1998]. [Sarwar et.al. 2000] apply Latent Semantic Indexing techniques in order to alleviate the sparsity problem.[Suryavanshi et, al, 2005] propose a two level model-based technique, employing relational fuzzy subtractive clustering as the first level modeling and then mining association rules within individual clusters, to produce recommendations which are scalable , accurate and less prone to effects of sparsity.

Hybridization of collaborative filtering with other filtering mechanisms is another way to help alleviate the sparsity problem. A global probabilistic model [Popescul et.al. , 2001] combining content-based and collaborative filtering methods use an extended Hofmann's aspect model to incorporate three-way co-occurrence data among users, items, and item content is shown to produce significantly better quality recommenders than k-nearest neighbors in the presence of sparsity. [Melville et.al. 2002] present a framework for combining content and collaboration which uses a content-based predictor to enhance existing user data, and then provides personalized suggestions through collaborative filtering.

Rules generated through Apriori algorithm [Sullivan et. al, 2002] have been used to address the sparsity problem by enabling similarity computation between pairs of users having no common ratings. A recursive algorithm [Zhang and Pu, 2007] is proposed, which allows nearest-neighbor users to join the prediction process even if they have not rated the given item. If a required rating value is not provided explicitly by the user, it is predicted recursively and then integrated into the prediction process. A similarity metric, Generalized Cosine Max, which does not need ratings of common items by users, is presented in [Anand et, al. 2007]. This similarity measure which uses item similarity within the calculation of user similarity is shown to work well when the rating data is sparse.

Trustworthiness of users is an important factor in social networks where everyone is free to create content and there is no central authority for quality control. Users thus act as assesses and explicitly specify the amount of trust they place on other user. Such trust statements can be gathered and organized as trust networks. Such trust networks can help form a group of trustworthy users. [Massa & Avesani, 2007] propose to replace the similarity computation step with the use of a trust metric, an algorithm able to propagate trust over the trust network and to estimate a trust weight that can be used in place of the similarity weight. Using trust as measure to find similar neighbors generally leads to a richer set of neighbors, especially when the data is sparse, since even if a user has only one trustworthy neighbor, using the trust propagation can enhance the set of neighbors (six degrees of separation theorem).

Global similarity measures enable computation of user similarities for users not sharing any common experiences. The ability to compute similarity between users who have not co-rated any items leads to a denser neighborhood set and thus helps overcome the problems arising due to sparsity in data. A graph-theoretic approach to collaborative filtering, which used the concepts of horting and predictability was proposed in [Aggarwal, et al, 1999]. This algorithm, which is a one of the key engines of Intelligent Recommendation Engine developed by IBM Research, enables users to participate in the prediction for an item even if they do not share common experiences with the active user.

A different approach of employing Markov-chain model of a random walk to compute user similarities/dissimilarities is proposed in [Fouss, et al, 2007]. Dissimilarities between users are computed using a bipartite graph which has users and items as nodes. A similar approach to computing item similarities using random walk model is introduced in ItemRank [Gori and Pucci 2007] and RandomWalk Recommender [Yıldırım & Krishnamoorthy, 2008].

A novel approach of combining predictions from locally similar neighbors, i.e. users who have co-rated items, and globally similar neighbors is proposed in [Luo, et al, 2008]. Two users are considered to be globally similar if they are connected through locally similar neighbors. It was established experimentally that when the data is very sparse, globally similar neighbors give better prediction, whereas in case of low sparsity predictions from local neighbors are better. The weighting of predictions from local/global neighbors as proposed by [Luo, et al, 2008] was static, i.e. the weight given to local/global predictions remained the same across all predictions. A promising extension to the above weighting scheme would be to consider the user/item level sparsity when weighting predictions. This is because intuitively, a user who has very few neighbors who have rated an item, and hence can contribute to the prediction, would have to rely on global neighbors to improve quality of recommendation.

In this dissertation we propose various sparsity measures based on the overall sparsity in the ratings data as well as sparsity based on the active user and the item whose rating needs to be predicted. The dependency of the sparsity measures on the user and item ensures that the local and global neighbors are weighed differently for every user and for every prediction. The experimental results support our theory and demonstrate that the proposed methods are superior to the fixed weighting scheme proposed in [Luo et al, 2008].

1.5 Organization of the Thesis

The rest of the dissertation is organized as follows. Chapter 2 discusses the various local and global similarity measures proposed in literature and present a framework combining local and global similarities. Different weighting scheme to combine predictions from local and global neighborhoods are proposed in Chapter 3. Chapter 4 gives details of various experiments conducted and the analysis of results so obtained. Finally, Chapter 5 presents the conclusion and points out some directions for future work.

Chapter 2

LOCAL AND GLOBAL SIMILARITIES

One of the important steps in collaborative filtering is the neighborhood formation step. Several similarity measures have been proposed in literature in order to identify users with similar inclinations. However most of the popular similarity measures proposed, base the similarity computation, on the local information available i.e. on ratings common to both users. Global similarity measures can complement local similarity measures in order to improve accuracy and coverage in sparse-data scenarios. This chapter presents an overview of the different local and global similarity measures proposed in literature and concludes with a framework for combining both in order to leverage the advantages of both approaches.

2.1 Local Similarity Measures

Similarity measurement between users plays an elemental role in both user-based and item-based algorithms. The most commonly used measurement techniques of the similarity between users are the Pearson Correlation [Resnick et al. 1994] and Vector Space Similarity [Breese et al., 1998] algorithms. Typically the similarity computation is based on finding the similarity between the rating vectors, containing ratings of items rated in common by both users.

Pearson Correlation coefficient defines similarity between users x and y as:

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)(r_{y,s} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)^2 \sum_{s \in S_{xy}} (r_{y,s} - r_y)^2}}, \qquad (2.1)$$

where S_{xy} is the set of items which users x and y have co-rated and $\bar{r}_x$ is the mean rating for user x.

Whereas Vector Space Similarity defines similarity as:

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} r_{x,s} r_{y,s}}{\sqrt{\sum_{s \in S_{xy}} r_{x,s}^{2} \sum_{s \in S_{xy}} r_{y,s}^{2}}}, \qquad (2.2)$$

When the preference information is binary i.e. like or don't like an item, then the Jaccard coefficient is used to measure user similarity. The Jaccard coefficient finds the ratio of number of items common to both users and the number of items present in at least one of the profiles.

$$sim(x,y) = \frac{|R_{x,i} \cap R_{y,i}|}{|R_{x,i} \cup R_{y,i}|}, \qquad (2.3)$$

where $R_{x,i}$ is the set of elements liked by user x

[Lathia et al, 2007] use concordance based methods to measure the association between a pair of users. A pair of rating s is concordant if the difference between each rating and the corresponding user's mean have he same sign. A pair of ratings is discordant if these differences have opposite sign. If a rating is equal to the user mean or if the item is not rated then the pair is tied. The method works by finding the number of concordant(C), discordant (D) and tied (T) pairs of ratings between two users. The measure of similarity using Somer's d is as defined:

$$sim(x,y) = \frac{C-D}{N-T}, \qquad (2.4)$$

where N is the total number of items

The adjusted-cosine similarity defined for item-based collaborative filtering takes into account the difference in rating scales between different users by subtracting the user average from the corresponding rating.

$$sim(i,j) = \frac{\sum_{u \in U} (R_{u,i} - \overline{R}_u)(R_{u,j} - \overline{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - R_u)^2} \sqrt{\sum_{u \in U} (R_{u,j} - R_u)^2}}, \tag{2.5}$$

[Candillier et. al. 2008] introduce several weighted similarity measures for user-based and item-based collaborative filtering. The method proposed uses jaccard similarity as a weighting scheme and combines it with other similarity measures such as Pearson correlation coefficient. Similarity measures, such as Pearson correlation coefficient, suffer from the drawback that two users may be very similar even if they only share appreciation on one attribute, which jaccard similarity successfully overcomes as it considers the quantity of overlap between the two users. Jaccard coefficient on the other hand doesn't take into account the difference of ratings between the vectors. The combined weighting scheme would offer the benefits of both similarity measures. So the weighted Pearson similarity between users x and y would be

$$sim(x,y) = Jaccard(x,y) * Pearson(x,y)\,. \tag{2.6}$$

2.2 Global Similarity Measures

Global similarity measures enable similarity computation between pairs of users who may not share any common experiences. Several algorithms utilizing predictions from global neighbors have been proposed in literature.

[Aggarwal et al, 1999] in their work, interestingly titled, "Horting hatches an egg", propose the concepts of horting and predictability. The graph consists of nodes representing users and a directed edge exists between user u_a and u_b if u_b can predict u_a. To predict the ratings for a particular item i_k by user u_a, assuming it has not been already rated by him, the shortest path from u_a is computed to a user say u_c who has rated i_k. The predicted rating for i_k by u_c is generated as a function of the path from u_a to u_c. The predicted ratings from all such users are then aggregated to get the final predicted rating. Here the predictability is asymmetric, i.e.: if user A predicts user B, it is not necessary that user B also predicts user A. Thus predictability can be seen as an asymmetric similarity measure .i.e. if A predicts B then A is similar to B from B's point of view. An approach to capturing transitive similarity between users, discovery hidden similarity (DHS), is proposed in [Lee et al, 2004]. The similarities are captured not only through movie ratings but also through user similarities.

A new technique of computing user similarities using a Markov-chain model of a random walk is introduced in [Fouss, et al, 2007]. A bipartite graph whose nodes are the users and items is used for similarity computation. Users and items are connected through edges if the user has experienced the item. The quantities, "average commute time" and "average first passage time" are used as similarity measure between two users. These quantities have the nice property of increasing when the number of paths connecting nodes increases and when the "lengths" of the paths decreases. Another random walk based approach, ItemRank [Gori and Pucci 2007], builds a correlation graph of items based on the calculation of correlation index between pairs of movies. The user preferences are then spread through the correlation graph, starting from the movies preferred by the user, in order to rank products according to expected user preference. A similar item-oriented algorithm is, RandomWalk Recommender[Yıldırım & Krishnamoorthy, 2008], that first infers transition probabilities between items based on their similarities and models finite length random walks on the item space to compute predictions.

A new collaborative filtering approach [Desrosiers & Karypis, 2008], computes global similarities between pairs of items and users, based on the solution to system of linear equations relating user similarities to item similarities. The new approach helps make accurate predictions in the presence of sparsity and also takes into account content-based similarities between users.

2.3 A Framework for Combining Local and Global Similarities

[Luo et al, 2008] proposed a novel method of combining predictions from local and global neighbors. The proposed method uses a Surprisal Vector based Similarity measure for deciding on local neighbors and maximin distance between users as the global similarity measure. The discussion in this section follows from [Luo et al, 2008].

2.3.1 Surprisal Vector Based Similarity

Several methods of computing local similarity have been discussed in the previous section. Most of the popular local similarity measures, such as Pearson correlation and Vector Space Similarity, have an important shortcoming namely; the ratings of all items are given equal credence. However it is true that some ratings for the same item carry more discriminative information than others. For example a best selling book would be rated highly by most users and hence two users cannot be deemed similar just because they give similar high rating to this book. However if the same best selling book, is rated poorly by a user, then this gives more information about the user (maybe he does not like the particular genre). This is illustrated with the example rating matrix shown in Fig 2.1.

The matrix has 8 users who have rated 4 movies in the scale 1-5. Dhoom II is a popular movie with most of the users giving high ratings. An examination of ratings for Dhoom II by users Ritu and Sheeja show that they give high ratings to the movie, but since most users are expected to rate the movie well, the ratings should not contribute much to the similarity between these two users. On the other hand, users Vishakha and Rishabh rate this popular movie poorly. Hence their ratings for the movie Dhoom II carry more information regarding their preference and hence should contribute more towards the similarity between the users.

	Dhoom II	RRCR	OSO	AALC
Ritu	5	1	4	
Sheeja	5		5	
Rishabh	1	4	2	
Mridula		1	5	2
Beena	5	2		1
Biju	5	2	4	1
Vishakha	1	5		3
Akshay	4	1	4	2

Table 2.1 : Ratings matrix

Surprisal-based Vector Similarity (SVS) is a similarity measure which takes into account the "surprisal" of a rating, which carries information about how different a particular rating is from the average attitude. The rating of each item is modeled as a Laplacian random variable *Laplace* (μ_i, b_i). The probability density function of the Laplacian random variable is

$$f(r \mid \mu, b) = \frac{1}{2b}\exp\left(\frac{-|r-\mu|}{b}\right) = \frac{1}{2b}\begin{cases}\exp\left(-\frac{\mu - r}{b}\right) & \text{if } r < \mu \\ \exp\left(-\frac{r-\mu}{b}\right) & \text{if } r > \mu\end{cases} \tag{2.7}$$

where μ and b are the location and scale parameters respectively. Given M ratings, independent and identically distributed samples $r_{1,i}, r_{2,i}, \ldots, r_{M,i}$, then using the maximum likelihood estimator, estimators of μi and bi are expressed as [Norton 1984]

$$\hat{\mu}_i = \frac{1}{M}\sum_{p=1}^{M} r_{p,i} \tag{2.8}$$

$$\hat{b}_i = \frac{1}{M}\sum_{p=1}^{M} | r_{p,i} - \hat{\mu}_i | \tag{2.9}$$

The quantity of information (surprisal) contained in a rating $r_{p,i}$, $I(r_{p,i})$ is defined as

$$I(r_{p,i}) = -\ln(f(r = r_{p,i} \mid \hat{\mu}_i, \hat{b}_i)) = \ln(2\hat{b}_i) + \frac{|r_{p,i} - \hat{\mu}_i|}{\hat{b}_i} \tag{2.10}$$

where $r_{p,i}$ is the rating of item 'i' by user 'p'. μ_i and b_i are the location and scale parameter, respectively. $\hat{b}_i$ and $\hat{\mu}_i$ are the maximum likelihood estimates of b_i and μ_i respectively.

Given the surprisal of all ratings, the user p's surprisal vector, S_p is defined as

$$\begin{aligned} S_p &= [s_{p,1}, \ldots, s_{p,N}]^T \\ &= [\operatorname{sgn}(r_{p,1} - \hat{\mu}_1) * I(r_{p,1}), \ldots, \operatorname{sgn}(r_{p,N} - \hat{\mu}_N) * I(r_{p,N})]^T, \qquad p = 1, \ldots, M \end{aligned}$$

where sgn ($r_{p,I} - \hat{\mu}_i$) represents whether the preference of user p is positive or negative with respect to the average attitude for the item.

The similarity calculation between two users is the vector space similarity between the users' surprisal vectors. This method, known as surprisal-based vector similarity (SVS), is defined as

$$sim_L(u_p,u_q) = \frac{\sum_{\{i|r_{p,i},r_{q,i}\neq 0\}} s_{p,i} * s_{q,i}}{\sqrt{\sum_{\{i|r_{p,i},r_{q,i}\neq 0\}} s_{p,i}^2} \cdot \sqrt{\sum_{\{i|r_{p,i},r_{q,i}\neq 0\}} s_{q,i}^2}} \qquad (2.11)$$

Significance weighting

Sometimes the similarity computation between two users is based on very few co-rated items, since there may be only a few items rated by both. To discount the high similarity between two users based on a small set of co-rated items, the following significance weighting scheme is proposed;

$$sim_L' = \frac{Min(|I_{u_p} \cap I_{u_q}|, \gamma)}{\gamma} sim_L(u_p,u_q), \qquad (2.12)$$

where $|I_{u_p} \cap I_{u_q}|$ is the number of items co-rated by users p and q. 'γ' is the minimum number of common items that needs to be rated in common by both users. If the number of items co-rated by p and q is less than 'γ' then the local similarity computed is discounted. This change avoids overestimating the similarities of users who have rated a few items identically, but may not have similar overall preferences [Luo et al, 2008]. This method is termed suprisal-based vector similarity with significance weighting.(SVSS)

2.3.2 Global Similarity

When the local neighborhood for the active user is very sparse then global neighbors can be used to make the neighborhood set rich. Under global user similarity, two users become more similar if they can be connected through a series of locally similar neighbors. The global neighbors for the active user are based on a user graph using local similarity as weight of the edges and then finding the maximin distance between the users.

User Graph

The user graph is defined as follows.

Def: (User graph) A user graph is an undirected weighted graph $G = (U, E)$, where

(a) U is the node set (each user is regarded as a node of the graph G);

(b) E is the edge set. Associated with each edge $e_{pq} \in E$, w_{pq} is a weight subject to

$w_{pq} > 0$, $w_{pq} = w_{qp}$.

The local user similarity is used as the weights of edges,

$$w_{pq} = \begin{cases} sim_L(u_p, u_q) & if\ sim_L(u_p, u_q) > 0, \\ 0 & else \end{cases}$$

Maximin distance on the user graph

Given a user graph $G = (U, E)$, a path from node u_p to u_q (u_p, $uq \in U$) is a sequence of links, $P_{pq} = (u_p, \ldots, u_i, \ldots, u_q)$, $up, ui, uq \in U$. If there are K paths between nodes u_p and u_q, these paths will be indicated as $P^1_{pq}, P^2_{pq}, \ldots, P^K_{pq}$. Given a path between u_p and u_q the minimal hop distance of these nodes along any path P^j_{pq} is defined as follows:

$$minimalhop_j(u_p, u_q) = \min_{u_i, u_{i+1} \subset P^j_{pq}} w_{i,i+1}, \forall u_i, u_{i+1} \in P^j_{pq}, 1 \le j \le k \quad (2.13)$$

The maximal value of the two nodes' minimal hop distance along any paths is called maximin distance of the two nodes.

$$maximinhop(u_p, u_q) = \max_{k=1,\dots,K} minimalhop_k(u_p, u_q)$$
$$= \max_{k=1,\dots,K}\left\{ \min_{u_i, u_{i+1} \subset P_{ij}^k} w_{i,i+1} \right\}, \quad \forall u_i, u_{i+1} \in P_{pq}^k. \tag{2.14}$$

The corresponding path is called as maximin path.

The global similarity of two users is defined as the maximin distance between them:

$sim_G(u_p, uq) = maximinhop(u_p, u_q)$.

Since all the edge weights are positive the global similarity between any pair of users is either zero or positive. It can also be shown that

$sim_G(u_p, u_q) \geq sim_L(u_p, u_q) \qquad \forall\ u_p, u_q \in U$

Example : Maximin distance

A user graph as per the local similarities given in table 2.1 is shown in Figure 2.2
In order to find the global similarity between user 1 and user 6, we note that there are three paths from user 1 to user 6 , $P_{16}^1 = 1-3-6$, $P_{16}^2 = 1-7-3-6$ and $P_{16}^3 = 1-7-5-6$

$minimalhop_1(1, 6) = \min(0.9, 0.5) = 0.5$

$minimalhop_2(1, 6) = \min(0.3, 0.7, 0.5) = 0.3$

$minimalhop_3(1, 6) = \min(0.3, 0.6, 0.8) = 0.3$

Hence the maximin distance between users 1 and 6 is given by

$Maximinhop(1.6) = \max(0.5, 0.3, 0.3) = 0.5$

$sim_G(u_p, u_q) = 0.5$

Therefore users 1 and 6 are globally similar to each other and they are connected via user 3.

	u1	u2	u3	u4	u5	u6	u7
u1	0	0.2	0.9	-0.2	-0.4	0	0.3
u2	0.2	0	0	0.5	0	-0.6	-0.9
u3	0.9	0	0	0	-0.8	0.5	0.7
u4	-0.2	0.5	0	0	0	-0.4	0
u5	-0.4	0	-0.8	0	0	0.8	0.6
u6	0	-0.6	0.5	-0.4	0.8	0	0
u7	0.3	-0.9	0.7	0	0.6	0	0

Table 2.2 : Similarities between pairs of users

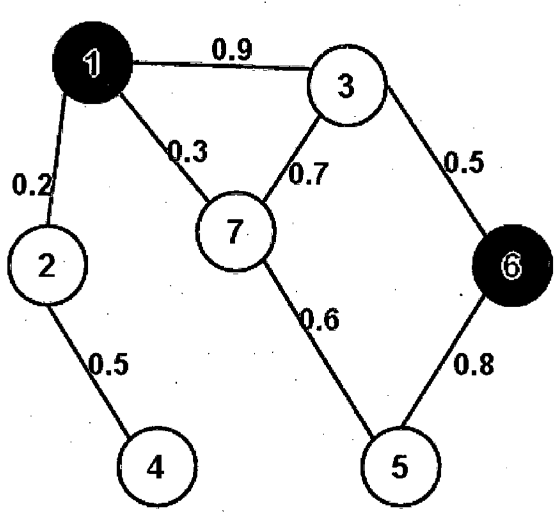

Figure 2.1 : User graph based on user similarities as per Table 2.2

2.3.3 Combining predictions from local and global neighbors

If the data set is dense enough then the neighborhood generated using local similarity measures is adequate to achieve high-quality recommendations. However when the sparsity in the dataset is high, and consequently the neighborhood set is poor, globally similar users can be used enrich the neighborhood set and thus enhance the prediction accuracy.

One approach is to combine the predictions generated by locally similar neighbors and globally similar neighbors. Prediction for an active user (u_a) is done by finding the k local nearest neighbors ($nn_L^k(u_a)$) and k global nearest neighbors ($nn_G^k(u_a)$). Both the neighborhoods are used for prediction.

$$\hat{r}_{a,i} = (1-\alpha)\frac{\sum_{u_k \in nn_L^k(u_a)} sim_L^l(u_k, u_a) r_{k,i}}{\sum_{u_k \in nn_L^k(u_a)} sim_L^l(u_k, u_a)} + \alpha \frac{\sum_{u_k \in nn_G^k(u_a)} sim_G(u_k, u_a) r_{k,i}}{\sum_{u_k \in nn_G^k(u_a)} sim_G(u_k, u_a)}$$

The parameter α decides the weightage given to global similarity. If $\alpha = 0$ then the prediction would be done using only local neighbors and if $\alpha = 1$ then only global neighbors would contribute to the prediction. Generally when the data is very sparse then the contribution of global neighbors should be more and similarly when the data is not very sparse then more importance needs to be given to local neighbors.

Chapter 3

WEIGHTING SCHEMES FOR COMBINING LOCAL AND GLOBAL SIMILARITIES

Chapter 2 focused on different local and global similarity measures. A framework utilizing both local and global neighbors by weighting the predictions from both sets of neighborhoods was also discussed. When the ratings matrix is dense then generally the local neighborhood set is rich enough to enable prediction for the active user, in which case the predictions from local neighborhood should be weighed more. However when the ratings matrix is sparse, the meager neighborhood set generated may lead to low quality recommendations, needing it to be enriched by the globally similar neighbors and thus more emphasis should be given to predictions from global neighbors. In the method discussed in Chapter 2, the weightage given to predictions from local and global neighbors was fixed and needed to be manually set depending on the data sparsity. In this chapter we propose some automatic weighting schemes for local and global similarities, which take into account not only the global sparsity in the ratings matrix, but also the sparsity at the user and item level.

3.1 Combining predictions from local and global neighborhoods

In the framework discussed in Chapter 2, combining contributions from local and global neighbors, the local and global predictions are combined as follows:

$$predR = (1-\alpha) * predR_l + \alpha * predR_G \tag{3.1}$$

where $predR_L$ is the predicted rating obtained through the local neighbors and $predR_G$ is the predicted rating obtained through the global neighbors. α is the weight given to prediction from the global neighborhood set. [Luo et al, 2008] empirically established the dependence of α, on the sparsity of the rating data. It was shown that with the increase in the sparsity of the ratings matrix, an increase in α led to more accurate predictions. This scheme of combining local and global similarities referred to as LS&GS in [Luo et al, 2008] will now onwards be referred to as **fixed- α** scheme.

The fixed- α scheme uses a constant value of α which needs to be set manually. However, it does not take into account the sparsity that may be present at the user level i.e. a rater who has rated very few items or whose interests share little with others. In this work we propose different weighting schemes to find the value of α taking into account not only the overall sparsity of user data but also the sparsity present at the user and item level. The "sparsity problem" in collaborative filtering refers to inability to find a sufficient quantity of good quality neighbors to aid in the prediction process due to insufficient overlap of ratings between the active user and his neighbors. This can happen when the ratings matrix is sparse, or the number of users participating is not large. Even when the data is dense enough to allow quality predictions for most users, some users may not have rated enough items or may have rated items not rated by most people, with the result that such users get poor quality predictions. Users whose local neighborhood set is sparse can thus be aided by using predictions from the global neighborhood set. Thus intuitively the value of α should depend on the user and the item whose rating is to be predicted. The proposed work introduces several formulae for α, which captures the various aspects of sparseness in the data.

3.1.1 Overall Sparsity Measure

The overall sparsity measure captures the level of sparsity in the entire rating matrix. This sparsity measure is universal i.e. the α computed is fixed for all users. The Overall Sparsity Measure is defined as

$$\textit{Overall Sparsity Measure} = 1 - \frac{nR}{nUsers * nItems} \tag{3.2}$$

where

nR - number of ratings by all users

$nUsers$ – total number of users in the system

$nItems$ – total number of items in the system

The overall sparsity is scaled down by a factor proportional to the number of participating users in the system.

For example the Overall Sparsity meaure for the matrix shown in Table 3.1 is

$$OS = 1 - \frac{8}{15} = 1 - 0.533 = 0.467$$

	Item 1	Item 2	Item 3
User 1	3		2
User 2		1	
User 3	4	2	
User 4	2		3
User 5		5	

Table 3.1 : Ratings matrix 1
(medium sparsity)

Figure 3.2 shows a scenario where the ratings matrix is very sparse. The Overall Sparsity measure in this case would be

$$OS' = 1 - \frac{3}{15} = 1 - 0.2 = 0.8$$

	Item 1	Item 2	Item 3
User 1	3		
User 2			
User 3		2	
User 4			3
User 5			

Table 3.2 : Ratings matrix 2
(high sparsity)

A dense ratings matrix scenario is illustrated by table 3.3. The Overall Sparsity measure in such a case would be

$$OS = 1 - \frac{12}{15} = 1 - 0.8 = 0.2$$

	Item 1	Item 2	Item 3
User 1	3	1	
User 2	2	3	3
User 3		2	5
User 4	2	2	3
User 5	4		5

Table 3.3 : Ratings matrix 3 (low sparsity)

3.1.2 User-Item specific Sparsity Measures

The overall sparsity measure introduced in the previous section is fixed for all users. However the size of the neighborhood set varies over the set of the users, where the number of neighbors depends on several factors, such as, the number of items the user has rated or the type of items (popular/unpopular) that the user has rated. When the local neighborhood is meager, global neighbors can contribute to improvement in accuracy of predictions. This means that the sparsity measure should take into account not only the active user, but also the item for which the rating needs to be predicted. Three sparsity measures, which capture sparsity at user-item level, are introduced in the following sections.

3.1.2.1 Local Global Ratio (LGR)

One simple measure is to find the ratio of number of local neighbors to the number of global neighbors who have rated the item. It is to be noted that the number of global neighbors always exceeds the number of local neighbors. This is due to the fact that $sim_G(x,y) \geq sim_L(x,y)$. The LGR for a user u and item i is defined as :

$$LGR(u,i) = \begin{cases} 1 - \dfrac{|L_{u,i}|}{|G_{u,i}|}, & if \ |Lu,i| < \eta \\ 0, & otherwise \end{cases} \tag{3.3}$$

where

$L_{u,i}$ - set of local neighbors of user ***u*** who have rated item ***i***

$G_{u,i}$ - set of global neighbors of user ***u*** who have rated item ***i***

η - Threshold value. If the number of local neighbors is above this threshold then the prediction will be based on local neighborhood.

Figure 3.2 shows the user graph constructed from the rating matrix given in Table 3.4. An edge between two users indicates that the users are locally similar. The user graph helps to get an idea of the local and global neighborhood set for an active user.

Example 1: If the rating for item 5 needs to be predicted for active user 7, then

$L_{7,5} = \{ 4 \}$

$G_{7,5} = \{ 1, 3, 4, 6, 11 \}$

Hence

$$LGR(7,5) = 1 - \frac{|L_{7,5}|}{|G_{7,5}|} = 1 - \frac{1}{5} = 1 - 0.2 = 0.8$$

Example 2: If the rating for item 5 needs to be predicted for active user 9, then

$L_{9,5} = \{ 1,3 \}$

$G_{9,5} = \{ 1, 3, 4, 6, 11 \}$

Hence

$$LGR(9,5) = 1 - \frac{|L_{9,5}|}{|G_{9,5}|} = 1 - \frac{2}{5} = 1 - 0.4 = 0.6$$

Example 3: If the rating for item 5 needs to be predicted for active user 8, then

$L_{8,5} = \{ 2, 5, 10\}$

$G_{8,5} = \{ 2, 5, 10\}$

Hence

$$LGR(8,5) = 1 - \frac{|L_{8,5}|}{|G_{8,5}|} = 1 - \frac{3}{3} = 1 - 1 = 0$$

Thus, in this last case, since the global neighborhood adds no new contributors to the already existing local set, the local neighborhood set suffices for the prediction.

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	4		5		2
User 2	1	4			3
User 3			4		1
User 4	5	1		4	2
User 5	2			1	4
User 6					2
User 7		1		5	
User 8			2	2	
User 9			4		
User 10		3	1		4
User 11	5				1

Table 3.4 : User Ratings matrix

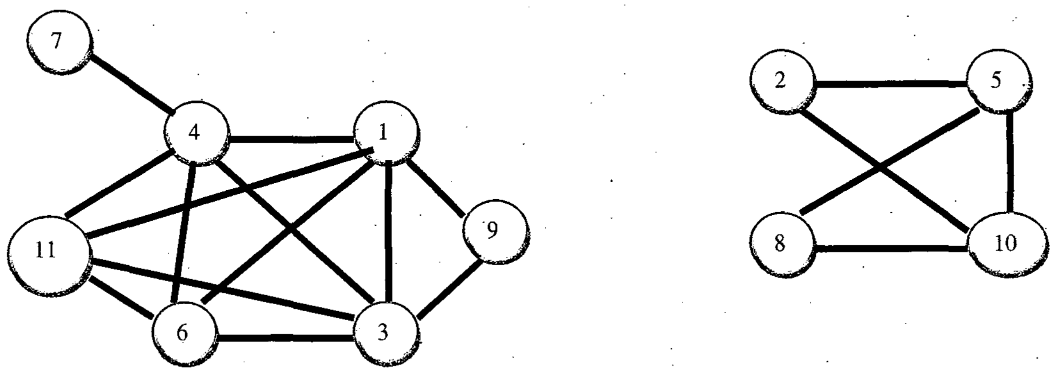

Figure 3.2 : User Graph Based on Table 3.4

3.1.2.2 User-Item Specific Sparsity Measure 1(UIS1)

The UIS1 measure bases the sparsity measurement on the ratio of number of local neighbors who have rated a particular item to the total number of people who have rated the item i.e. the UIS1 for a user u and item i is defined as :

$$UIS1(u,i) = \begin{cases} 1 - \dfrac{|L_{u,i}|}{|N_i|}, & if \ |L_{u,i}| < \eta \\ 0, & otherwise \end{cases} \tag{3.4}$$

where N_i - set of users who have rated item $\boldsymbol{i}$

Assume η= 4 in the examples below.

Example 1: UIS1 for user 7 when rating for item 5 is to be predicted

$L_{7,5}$ = { 4 }

N_5 = {1, 2, 3, 4, 5, 6, 10, 11}

$$UIS1(7,5) = 1 - \frac{|L_{7,5}|}{|N_5|} = 1 - \frac{1}{8} = 1 - 0.125 = 0.875$$

Example 2: For user 5 for item 2 the UIS1 measure would be

$L_{5,2} = \{ 2, 10\}$

$N_5 = \{ 2, 4, 7, 10 \}$

$$UIS1(5,2) = 1 - \frac{|L_{5,2}|}{|N_2|} = 1 - \frac{2}{4} = 1 - 0.5 = 0.5$$

Example 3: For user 6 for item 1 the UIS1 measure would be

$L_{5,2} = \{ 1, 4, 11\}$

$N_5 = \{ 1, 2, 4, 5, 11 \}$

$$UIS1(6,1) = 1 - \frac{|L_{6,1}|}{|N_1|} = 1 - \frac{3}{5} = 1 - 0.6 = 0.4$$

Example 4: For user 4 for item 5 the UIS1 measure would be

$L_{4,5} = \{1,3,6,7,11\}$

Since $|L_{4,5}| > \eta = 4$, hence the UIS1(4,5) = 0.

3.1.2.3 User-Item Specific Sparsity Measure 2(UIS2)

The UIS2 measure bases the sparsity measurement on the ratio of number of local neighbors who have rated a particular item to the total number of users in the local neighborhood set i.e. the UIS2 for a user u and item i is defined as :

$$UIS2(u,i) = \begin{cases} 1 - \frac{|L_{u,i}|}{|L_u|}, & if \ |L_{u,i}| < \eta \\ 0, & otherwise \end{cases} \tag{3.5}$$

where L_u - set of local neighbors for user ***u***

Example 1: If we wish to predict the rating for item 5 for user 1

$L_{15} = \{3, 4, 6, 11\}$

Since $|L_{1,5}| = \eta = 4$, hence the UIS2(1,5) = 0.

Example 2: For user 1 if we want to predict rating for item 2

$L_{12} = \{4\}$
$L_1 = \{3, 4, 6, 9, 11\}$

$$UIS2(1,2) = 1 - \frac{|L_{1,2}|}{|L_1|} = 1 - \frac{1}{5} = 1 - 0.2 = 0.8$$

3.2 Proposed Scheme

The proposed work introduces several schemes for weighting local and global similarities. These schemes are based on different sparsity measures, as discussed in the previous section, which capture various facets of sparsity, the overall sparsity and the sparsity at the user-item level. The main steps of the proposed recommender system framework are given below:

Step 1: Compute local user similarities

Compute the SVSS similarity between all pairs of users, using the formulas(2.11) and(2.12) as discussed in Chapter 2. Let $sim_L(x, y)$ refer to the local similarity between users x and y as computed in this step.

Step 2: Compute the global user similarities

Compute the global similarities between all pairs of users, by first constructing the user graph based on local similarities, and then finding the maximin distance between users, as described in Chapter 2. Let $sim_G(x, y)$ refer to the global similarity between users x and y as computed in this step.

Step 3: Obtain predicted ratings using local and global neighbors

The predicted rating for an item *i* for active user *u* is based on Resnick's prediction formula [Resnick et al, 1994].

$$pr_{i,k} = \bar{r}_i + \frac{\sum_{j \in N(i)} sim_{i,j} * (r_{j,k} - \bar{r}_j)}{\sum_{j \in N(i)} | sim_{i,j} |} \tag{3.6}$$

where $\bar{r}_i$ is the mean rating for user I, $sim_{i,j}$ is the similarity between users *i* and *j* and N(i) is the neighborhood of user *i*.

The above formula can be used to arrive at predictions from local neighborhood by setting sim_{ij} to $sim_L(i,j)$ and N(i) to the local neighborhood for user *i*. A similar method can be adopted to arrive at predictions from global neighborhood. Let $pr_{i,k}^L$ and $pr_{i,k}^G$ be the predicted ratings using local and global similarities respectively.

Step 4: Merge the local and global ratings

The predicted ratings from local and global neighborhoods are combined using the following formula:

$$pr_{i,k} = (1-\alpha) * pr_{i,k}^L + \alpha * pr_{i,k}^G$$

where α can be set to any of the sparsity measures as given below:

- Overall Sparsity Measure (OS) (section 3.1.1)
- User-item specific sparsity measures(LGR, UIS1, UIS2) (section 3.1.2)

Step 5: Compute MAE

For each item whose rating is to be predicted

- Compute the error i.e. $(pr_{i,k} - r_{i,k})$ for each item.

- Compute Mean Absolute Error(MAE) for active user u_i based on the formula:

$$MAE(u_i) = \frac{1}{|S_i|}\sum_{k=1}^{|S_i|} |\, pr_{i,k} - r_{i,k} | \tag{3.7}$$

where S_i is the cardinality of the test ratings set of user u_i.

- The total MAE over all the active users, N_T can be computed as

$$MAE = \frac{1}{N_T}\sum_{i=1}^{N_T} MAE(u_i) \tag{3.8}$$

MAE is used to compare the accuracy of different weighting schemes proposed with the fixed-α scheme.

Chapter 4

EXPERIMENTAL RESULTS

In this chapter we present the results of conducting experiments using the methods proposed in this work. The aim is to evaluate the performance of the various sparsity measures proposed in Chapter 3. The experiments are conducted comparing the proposed weighting scheme with the method proposed in [Luo et al, 2008]. The 100K MovieLens data set available at http://www.MovieLens.umn.edu, is used for the experiments.

4.1 MovieLens Dataset

MovieLens data sets were collected by the GroupLens Research Project at the University filtering algorithms.

The dataset comprises of 100,000 ratings, from 943 users on 1682 movies. All ratings follow the 1-bad, 2-average, 3-good, 4-very good, and 5-excellent numerical scale. Each user has rated at least 20 movies. Demographical information for each user such as age, gender, occupation and zip code are included for all users. Each movie is also associated with information about that movie, such as movie title, release date, video release date, and genre. The genre feature lists all genres to which the movie belongs. The various genres are Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-noir, Horror, Musical, Mystery, Romance, Sci-fi, Thriller, War, or Western. A single movie can belong to more than one genre. The different users' occupations are administrator, artist, doctor, educator, engineer, entertainment, executive, healthcare, homemaker, lawyer, librarian, marketing, none, other, programmer, retired, salesman, scientist, student, technician, and writer.

4.2 Experiments to compare the proposed weighting schemes with fixed-α schemes

Experiments are conducted to compare the predictive accuracy of each of the proposed weighting schemes with the fixed-α scheme. The various weighting schemes are compared against the fixed- α and best- α schemes under different configurations. These different configurations enable comparison between various schemes under different sparsity levels. In particular we conducted two experiments

(a) Comparison of overall MAE of the various proposed methods.
(b) Comparison of user-wise MAE for fixed- α, best- α and the User-Item Sparsity 2 measure.

4.2.1 Experimental Setup

The experiments are conducted under different configurations. A subset of 500 users is selected at random from among the 943 users and three different training-testing data sizes are used as (300-200), (200-300), and (100, 400) called MovieLens300, MovieLens200 and MovieLens100 respectively. The number of ratings, from the active users, which are used for constructing the user neighborhood, are varied from 10, 15 and 20. This gave a total of 9 configurations, M300G10, M300G15, M300G20, M200G10, M200G15, M200G20, M100G10, M100G15, M100G20. The training ratings are used as explicit ratings available and the test ratings are considered unavailable and hence need to be predicted. This protocol is similar to the experimental setup in [Luo et al, 2008]. The Given5 configuration in [Luo et al, 2008] did not result in significant improvement in accuracy of the proposed weighting schemes over the fixed-α scheme. This is because the proposed scheme uses Resnick's prediction formula[Resnick, 1994], for combining the predictions from the neighbors, which relies on the user ratings average in order to make predictions, and a small number of ratings from the active user does not give a good estimate of the user rating average. The Given5 configuration is replaced by the Given15 configuration.

4.2.2 Comparison of overall MAE

In order to test the effectiveness of employing the various weighting schemes in the local and global similarity framework we compare the overall MAE of the different weighting schemes with the fixed-α scheme. Two variations of the fixed-α scheme are tried out. In the first variation the weighting schemes are tested against the fixed- α scheme with α set to 0.5. In the second variation we test the proposed scheme with the fixed-α with α set to a value which results in the least MAE. The best α value is obtained empirically, by taking α values in the interval]0,1] in increments of 0.05. The α which gives the overall lowest MAE is the best α. The six experiments conducted are:

(i) Fixed-α ($\alpha = 0.5$)
(ii) Fixed-α with value of α which results in least MAE(Best-α)
(iii) Overall Sparsity(OS)
(iv) Local Global Ratio (LGR)
(v) User-Item Specific Sparsity Measure(UIS1)
(vi) User-Item Specific Sparsity Measure(UIS2)

The parameters in all the experiments are set to γ=30, k=35 and η=20. To compare the accuracy of the proposed methods to that of the fixed-α scheme we use the MAE as computed using formula (3.7) and (3.8). A smaller value of MAE indicates better accuracy.

Analysis of the Results

The results of the experiments are summarized in Table 4.1. The least MAE for each configuration is highlighted in bold. The results show that all the proposed weighting schemes outperform the fixed-α scheme under all configurations and in both scenarios. The user-item specific sparsity measures perform the best since they take into account sparsity at the user and item level.

		Fixed-α α = 0.5	Best-α	Overall Sparsity	L-G Ratio (LGR)	User-Item Sparsity 1 (UIS1)	User-Item Sparsity 2 (UIS2)
M300	Given 10	0.8129475	0.812543 (α = 0.4)	0.805147	0.804345	0.804531	**0.804027**
	Given 15	0.812891	0.812489 (α = 0.4)	0.789355	0.787897	0.788249	**0.785288**
	Given 20	0.806124	0.805686 (α = 0.4)	0.784821	0.782456	0.782061	**0.777601**
M200	Given 10	0.831347	0.826999 (α = 0.8)	0.814231	**0.808026**	0.813617	0.810119
	Given 15	0.835583	0.831019 (α = 0.9)	0.809451	0.803921	0.809280	**0.803878**
	Given 20	0.822953	0.820100 (α = 0.9)	0.781450	0.778220	0.782024	**0.776638**
M100	Given 10	0.890873	0.882813 (α = 1.0)	0.881902	**0.875070**	0.878655	0.879850
	Given 15	0.874861	0.860392 (α = 1.0)	0.853351	**0.843809**	0.848682	0.847794
	Given 20	0.861305	0.852570 (α = 1.0)	0.823898	**0.815129**	0.819083	0.816231

Table 4.1 : Fixed-α and Best- α versus Proposed Weighting schemes

4.2.3 Comparison of user-wise MAE

This experiment is conducted to find the number of users who get good quality prediction under the various weighting schemes. A good weighting scheme should not only lead to increased overall accuracy but should also increase the number of users receiving good quality prediction. Here we only report the results from the comparisons of UIS2 measure against the fixed-α and the best-α in the M200G20 configuration. User-wise MAE is the average MAE over all predictions for an active user and is computed using formula (3.7). Figures 4.1 through 4.6 show the comparison of user-wise MAE of UIS2 and the fixed-α and the best-α weighting schemes. It is to be noted here that there exist some users who have rated exactly 20 items and hence the set of items for which ratings prediction would be made, turns out to be null. Hence such users are not plotted on the graph. The MAE comparisons for the 300 active users are plotted in groups of 50 users. Table 4.2 shows the percentage of users for which UIS2 measure gives better predictions

	%age of users for whom user-item sparsity 2 is more accurate than fixed-α	**%age of users for whom user-item sparsity 2 is more accurate than best-α**
Users 1-50	61.7	61.7
Users 51-100	65.1	66.3
Users 101-150	68.75	66.75
Users 151-200	67.3	67.3
Users 201-250	65.3	67.3
Users 251-300	66	54

Table 4.2 : User-Item Sparsity 2 versus Fixed-α and Best-α(User-wise MAE)

Over all 300 users UIS2 measure gave more accurate predictions for 65.7% of the users when compared to the fixed-α scheme and better predictions for 63.9% of the users when compared with the best- α scheme.

MAE through fixed-α MAE through best-α MAE through User-Item Sparsity 2

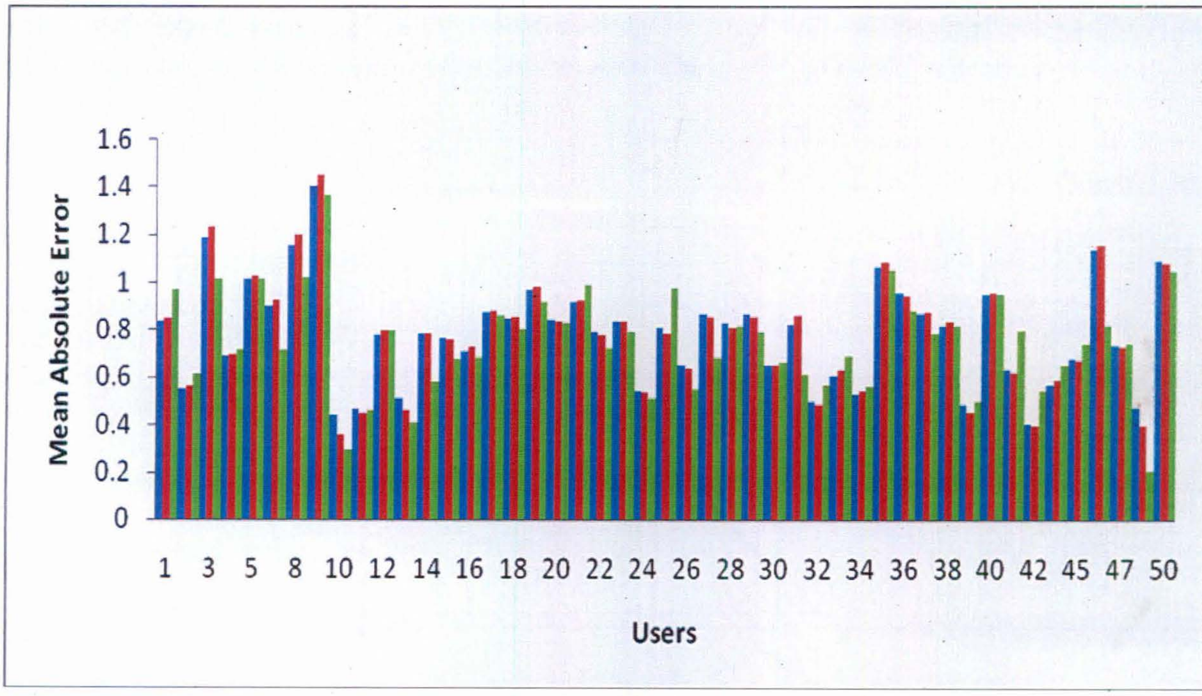

Figure 4.1 Mean Absolute Error for Users 1-50

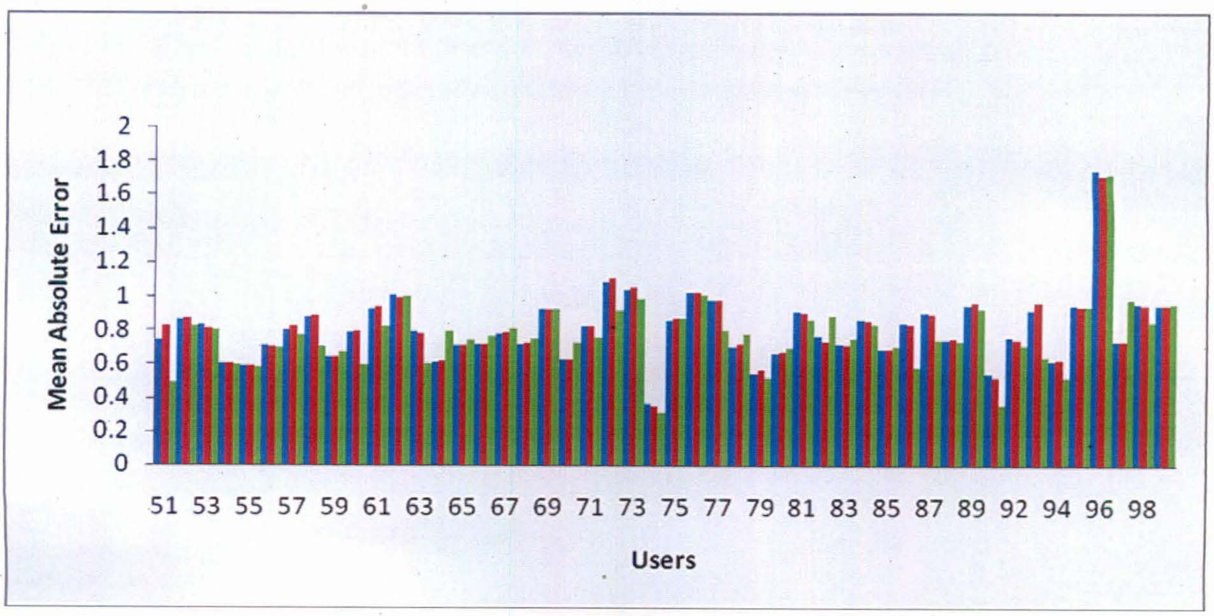

Figure 4.2 Mean Absolute Error for Users 51-100

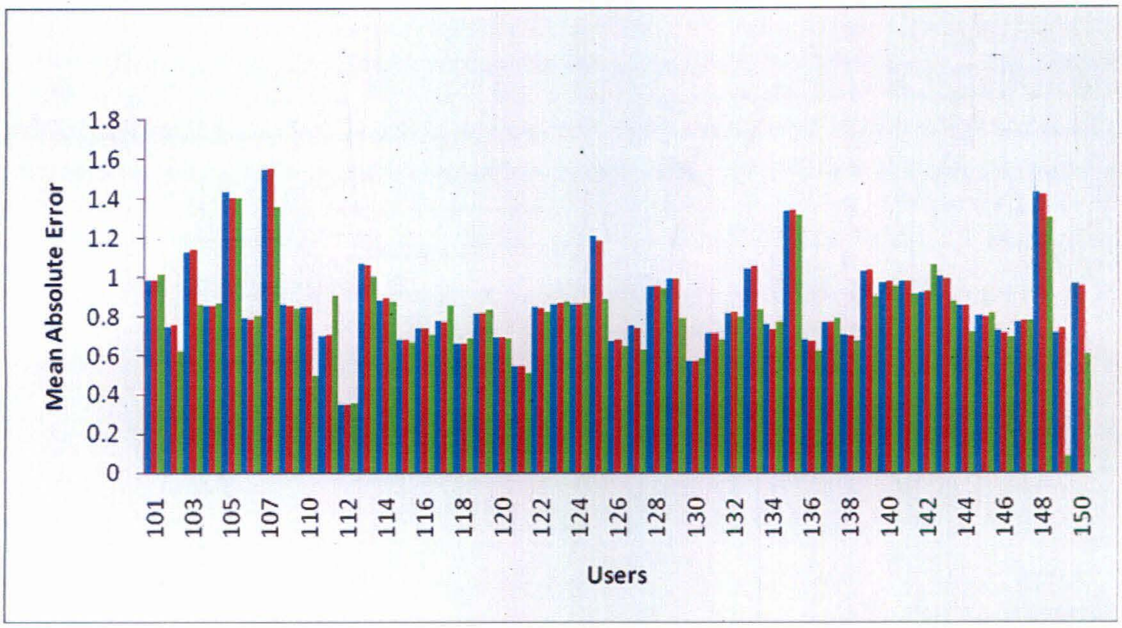

Figure 4.3 Mean Absolute Error for Users 101-150

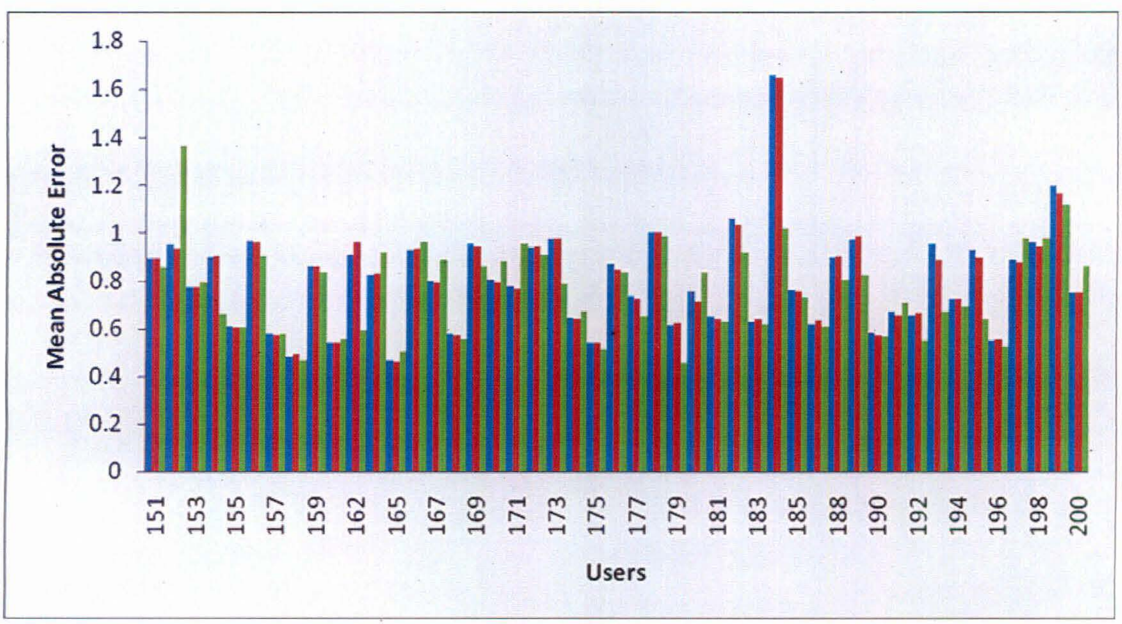

Figure 4.4 Mean Absolute Error for User 151-200

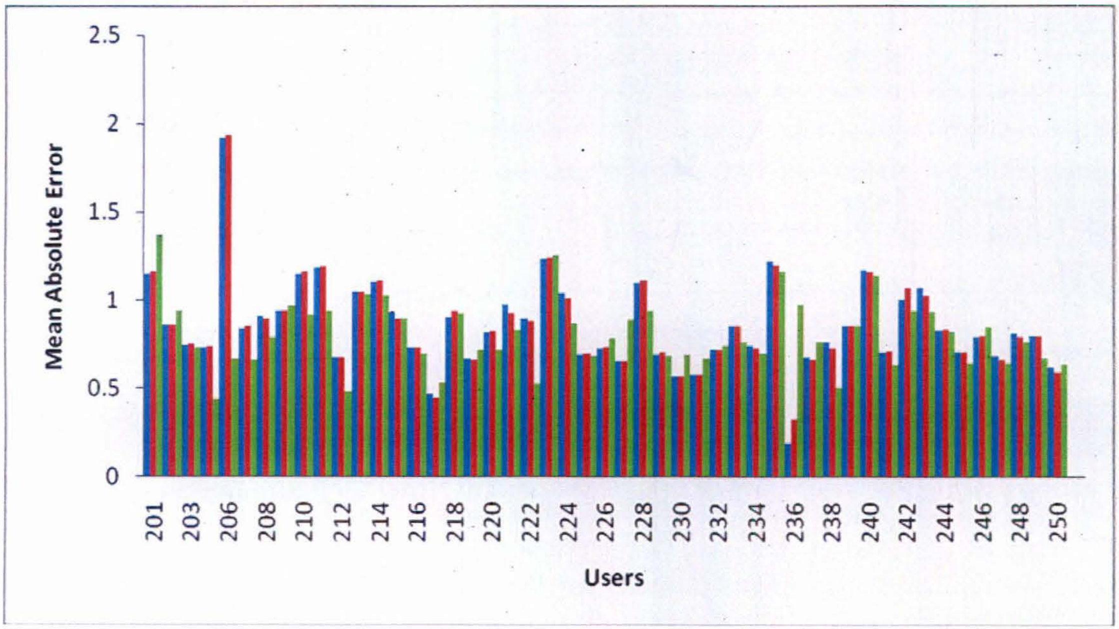

Figure 4.4 Mean Absolute Error for User 201-250

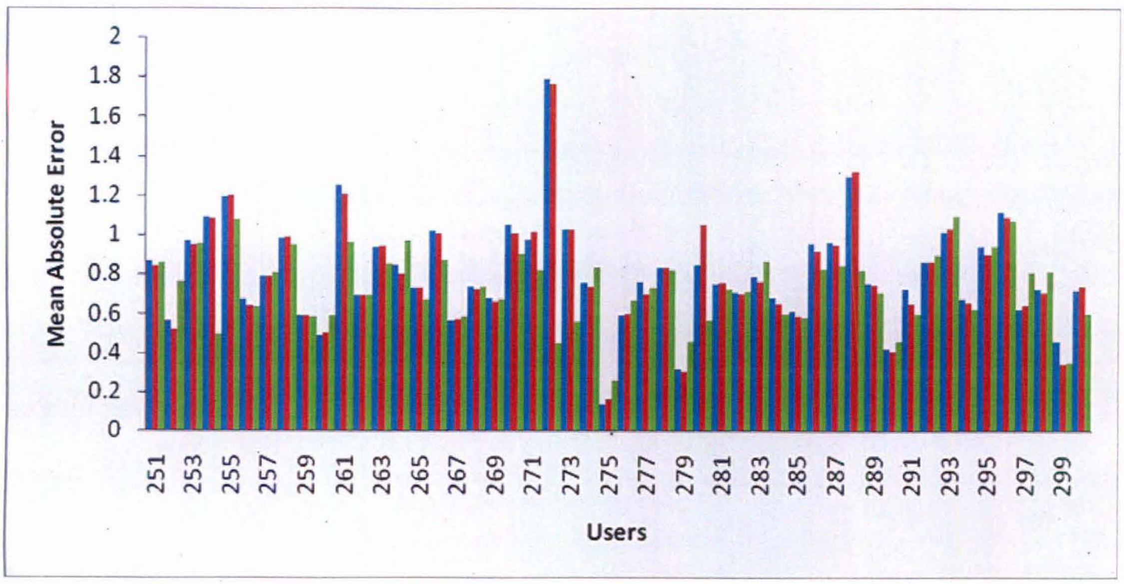

Figure 4.6 Mean Absolute Error for User 251-300

Chapter 5

CONCLUSION

This dissertation presents a collaborative filtering framework for combining local and global similarities with weighting schemes based on various sparsity levels in the data set. Different weighting schemes are proposed that take into account the overall sparsity as well as the sparsity at user and item level.

The proposed weighting schemes are compared with the fixed-α scheme of combining local and global neighbors. Experimental results show that the incorporation of the proposed weighting schemes lead to a significant improvement in prediction accuracy as compared to the fixed-α scheme.

Future Work

In the current work each of the different proposed sparsity measures provides us with an alternative to combine the local and global similarities. Further work needs to be done to derive an overall sparsity measure as a weighted sum of the proposed sparsity measures. Machine learning techniques, such as, evolutionary approaches may be exploited for automatically assigning appropriate weights to the various sparsity measures. One of the future research directions would be to explore alternative methods to derive the local and global similarities. Incorporation of Trust and Reputation concepts [Bharadwaj & Al-Shamri, 2009] in the proposed system also needs to be investigated.

REFERENCES

Adomavicius, G. and Tuzhilin, A.(2005), "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Trans. Knowledge and Data Engineering, vol. 17, no. 6, pp. 734–749.*

Aggarwal, C.C., Wolf, J.L., Wu, K-L., and Yu P.S.(1999), "Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering," *In Proceedings of the Fifth ACM SIG KDD Intelligent Conf. Knowledge Discovery and Data Mining, Aug. 1999.*

Al-Shamri, M.Y.H. & Bharadwaj, K.K. (2008), "Fuzzy-Genetic Approach to Recommender System Based on a Novel Hybrid User Model", *Expert Systems with Applications, Volume 35, Issue 3, pp. 1386-1399.*

Anand, S.S., Kearney, P., Shapcott, M.(2007), "Generating Semantically Enriched User Profiles for Web Personalization". *ACM Transactions on Internet Technologies, Vol. 7, no. 4, Article no.22.*

Balabanovic, M. and Shoham, Y.(1997), "Fab:Content-based, Collaborative Recommendation", *Communications of the ACM, Volume 40, no. 3, pp. 66-72*

Bell, R. M., Koren, Y. and Volinsky, C.(2007), "Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems", *Proc. 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 95-108.*

Bharadwaj, K.K. and Al-Shamri, M.Y.H. (2009), "Fuzzy Computational Models for Trust and Reputation systems",*Electronic. Commerce Research and Applications, Elsevier, Volume 8 , no. 1, pp. 37-47*

Billsus, D., and Pazzani, M.(1998), "Learning Collaborative Information Filters," *In Proceedings of the International Conference on Machine Learning, pp. 46-54*

Breese, J. S., Heckerman, D., & Kadie, C. (1998). "Empirical Analysis of Predictive Algorithms for Collaborative Filtering". *In Proceedings of the fourteenth annual conference on uncertainty in artificial intelligence, pp. 3–52.*

Burke, R. (2002), "Hybrid Recommender Systems: Survey and Experiments", *User Modeling and User-Adapted Interaction 12, Kluwer Academic Publishers, pp.331-370.*

Candillier, L., Meyer, F., and Fessant, F.(2008), "Designing Specific Weighted Similarity Measures to Improve Collaborative Filtering Systems", *In Proceedings of the 8th industrial conference on Advances in Data Mining: Medical Applications, E-Commerce, Marketing, and Theoretical Aspects, LNAI Volume 5077, pp. 242–255.*

Desrosiers, C., Karypis, G.(2008), " Solving the Sparsity Problem: Collaborative Filtering via Indirect Similarities", *Technical Report , University of Minnesota, TR 08-044.*

Fouss, F., Pirotte, A., Renders, J. M. & Saerens, M. (2007). "Random-walk Computation of Similarities Between Nodes of a Graph with Application to Collaborative Recommendation". *IEEE Transactions on Knowledge and Data Engineering, pp. 355–369.*

Goldberg, D., Nichols, D., Oki, B.M., and Terry, D.(1992), "Using Collaborative Filtering to Weave an Information Tapestry", *Communications of the ACM, Volume 35, Issue, pp. 61-70*

Gori, M., & Pucci, A. (2007). "ItemRank: A Random-Walk Based Scoring Algorithm for Recommender Engines". *In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 2766–2771.*

Gřcar, M., Mladenic, D., Fortuna, B., and Grobelnik, M.(2006) "Data Sparsity Issues in the Collaborative Filtering Framework", *Advances in Web Mining and Web Usage Analysis LNAI. Volume 4198, pp. 58-76.*

Jameson, A., Konstan, J., and Riedl, J.(2003), "AI Techniques for Personalised Recommendation.", *Tutorial SA1, Eighteenth International Conference on Artificial Intelligence.*

Joachims, T.(1997), "A probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization", *In Proceedings of the Fourteenth International Conference on Machine Leasrning,pp.143-151, San Francisc, CA. Morgan Kaufman.*

Kim, K., and Ahn, H. (2008). "A Recommender System Using GA K-means Clustering in an Online Shopping Market". *Expert Systems with Applications, Volume 34, Issue 2, pp. 1200–1209.*

Koren, Y.(2008), "Tutorial on Recent Progress in Collaborative Filtering", *In Proceedings of the 2008 ACM conference on Recommender systems(ACM Recsys'08), pp.333-334*

Krulwich, B.(1997), "Lifestyle Finder Intelligent User Profiling Using Large-Scale Demographic Data". *AI magazine, Volume 18, no. 2, pp. 37-46.*

Lathia, N., Hailes, S., and Capra, L.(2007). "Private Distributed Collaborative Filtering using Estimated Concordance Measures",. *In Proceedings of the 2007 ACM conference on Recommender systems, pp. 1-8.*

Lee, S., Yang, J., and Park, S.(2004),"Discovery of Hidden Similarity on Collaborative Filtering to Overcome Sparsity Problem", *Discovery Science, LNCS, Volume. 3245, pp. 396-402*

Linden, G., Smith, B., and York, J.(2003), "Amazon.com Recommendations:Item-to-Item Collaborative Filtering", *Internet Computing IEEE, Volume.7, Issue 1, pp.76-80*

Luo, H., Niu, C., Shen, R., Ullrich, C.(2008). "A Collaborative Filtering Framework based on both Local User Similarity and Global User Similarity", *Machine Learning, Springer, Volume 72, Number 3, pp. 231-245.*

Ma, H., King, I., & Lyu, M. R. (2007). "Effective Missing Data Prediction for Collaborative Filtering" In *SIGIR'07: proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval, pp.* 39–46, *New York: ACM.*

Manouselis, N., and Costopoulou, C. (2007), "Analysis and Classification of Multi-Criteria Recommender Systems", *World Wide Web: Internet and Web Information Systems, Volume 10 No. 4, pp. 415-441.*

Massa, P., Avesani, P. (2004), "Trust-aware Collaborative Filtering for Recommender Systems.", *Lecture Notes in Computer Science, Volume. 3290, pp. 492-508.*

Melville, P., Mooney, R., and Nagarajan, R.(2002), "Content-boosted Collaborative Filtering_for Improved Recommendations.", *In Proceedings of the Eighteenth National Conference on Articial Intelligence, pp.187-192.*

Mobasher, B., Cooley, R., & Srivastava, J. (2000). "Automatic Personalization Based on Web Usage Mining.", *Communications of the ACM, Volume 43, Issue 8, pp. 142-151.*

Norton, R. M. (1984). "The Double Exponential Distribution: Using Calculus to Find a Maximum Likelihood Estimator.", *The American Statistician, Volume 38, Issue. 2, pp. 135–136.*

Pazzani, M. (1999), "A Framework for Collaborative, Content-Based, and Demographic Filtering.",*Artificial Intelligence Review, Volume 13, Issue 5-6, pp. 393-408.*

Popescul, R., Ungar, L.H., Pennock, D.M., Lawrence, S.(2001), "Probabilistic Models for Unified Collaborative and Content-based Recommendation in Sparse-data Environments", *In Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence, pp. 437-444.*

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J.(1994), "Grouplens: An Open Architecture for Collaborative Filtering of Netnews", *In Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work, pp. 175-186.*

Rosenstein, M. and Lochbaum, C.(2000), "Recommending from Content: Preliminary Results from an E-Commerce Experiment.", *In: Proceedings of CHI'00: Conference on Human Factors in Computing, The Hague, Netherlands, pp.291-292.*

Sarwar, B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B., and Riedl, J.(1998), "Using Filtering Agents to Improve Prediction Quality in the Grouplens Research Collaborative Filtering system". *In Proceedings of ACM CSCW'98 Conference on Computer-Supported Cooperative Work,Social Filtering, Social Inuences, ACM Press, pp. 345-354.*

Sarwar, B. M., Karypis, G., Konstan, J., and Riedl, J..(2000), "Analysis of Recommender Algorithms for E-commerce", *In Proceedings of the 2nd ACM E-Commerce Conference(EC'00), Minneapolis, MN, pp. 158-167.*

Schafer, J.B. , Konstan, J.A., and Riedl, J.(2001), "E-Commerce Recommendation Applications", *Data Mining and Knowledge Discovery, Springer Verlag, Volume 5, Issue 1-2, pp. 115-153.*

Shardanand, U., and Maes, P.(1995), "Social information Filtering: Algorithms for Automating "Word of Mouth"", *In Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems, pp. 210-217.*

Suryavanshi, B.S.; Shiri, N.; Mudur, S.P., "Shankar Suryavanshi, Nematollaah Shiri, Sudhir P. Mudur,(2005), "Improving the Effectiveness of Model Based Recommender Systems for Highly Sparse and Noisy Web Usage Data", *In Proceedings. of 2005 IEEE/WIC/ACM International Conference on Web Intelligence, IEEE Press, pp.618 – 621*

O'Sullivan, D., Wilson, D., Smyth, B.(2002), "Using Collaborative Filtering Data in Case-Based Recommendation", *In Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society, AAAI, pp. 121-125.*

Terveen, L. & Hill, W. (2001), "Beyond Recommender Systems: Helping People help each other. ", *In HCI in the New Millennium, J. Carroll, Ed. Addison-Wesley.*

Vosalis, M.G., and Margaritis, K.G.(2007), "Using SVD and Demographic Data for the Enhancement of Generalized Collaborative Filtering. Information Sciences", *Information Sciences, Volume. 177, Issue 15, pp. 3017-3037.*

Waddington, P.(1996), "Dying for Information? A Report on the Effects of Information Overload in the UK and Worldwide", *Report, Reuters, United Kingdom.*

Yıldırım, H., Krishnamoorthy M.S.(2008), "A Random Walk Method for Alleviating the Sparsity Problem in Collaborative Filtering", *In Proceedings of the 2008 ACM conference on Recommender systems(RecSys'08), Lausanne, Switzerland., pp. 131-138.*

Zhang, J., Pu, P. (2007), "A Recursive Prediction Algorithm for Collaborative Filtering Recommender Systems", *In Proceedings of the 2007 ACM conference on Recommender systems(RecSys'07), Minneapolis, Minnesota, USA., pp. 57-64.*