MACQUARIE University SYDNEY·AUSTRALIA

A Peer-based Social Relationship Enhanced Recommendation Model

by

Youliang Zhong

A thesis submitted in fulfillment for the degree of Doctor of Philosophy

in the

Department of Computing

Faculty of Science and Engineering

Supervisor: Prof. Jian Yang

September 2015

Declaration of Authorship

I, YOULIANG ZHONG, declare that this thesis titled, 'A PEER-BASED SOCIAL RELATIONSHIP ENHANCED RECOMMENDATION MODEL' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"In a group of three people, there is always something I can learn from. Choose to follow the strengths of others, use the shortcomings to reflect upon ourselves."

Confucius, 551-479 B.C.

Abstract

Recommender systems have been developed to address the "information overload" issue by providing users with potentially useful products or services. While most of the current systems are continuing to deal with globally collected large number of users and items, little attention is paid to the situations where users ask for recommendations through a limited number of personal social circles.

Social networks, on the other hand, are one of the most popular channels through which people share and exchange information. Many studies in recent years have shown that incorporating social relations and interactions into recommender systems will significantly improve recommendation qualities. To make superior recommendations from peers, we are especially interested in developing a recommendation model that emulates the natural recommendation style in real life.

In this thesis, we propose a peer-based social recommendation model that imitates the natural recommendation process in social networks. The model forms neighborhoods from peers across social circles, through which the peers participate in the recommendation process by propagating requests and responses in a relay fashion. Furthermore, we develop a machine learning method to measure tie strengths among the peers in a social network, based on various social relationships. Generally speaking, the stronger the tie strength between two individuals, the more similar interests they may commonly share. Furthermore, the learned tie strengths are then incorporated in recommendation process to increase the accuracy and relevance of the recommendations.

We have conducted comprehensive experiments by using the real datasets from popular social media services. The evaluation results demonstrate that our proposed recommendation model outperforms other popular and state-of-the-art recommendation methods in terms of widely accepted evaluation metrics.

Acknowledgements

This thesis would not have been possible without the support of many people, as well as the grants of Macquarie University Research Excellence Scholarship scheme and an Australian Research Council Linkage project (LP120200231).

First of all, I would like to thank my supervisor, Professor Jian Yang, for her guidance and support throughout the last four years, especially in the most difficult moments of my research. I really appreciate the way she patiently encouraged me to explore new paths throughout this research, and offered suggestions and corrections. Also, many thanks to Dr. Weiliang Zhao, my co-supervisor, for his always-timely support and for being able to ask me those questions which triggered me to think about the alternatives when looking at the subjects at hand.

I also owe a lot to the professors and researchers in the Department of Computing, in particular, I would like to show my deep gratitude to Prof. Mehmet A. Orgun, A/Prof. Yan Wang, Dr. Lan Du and my fellow PhD student Xiaoming Zheng, who made insightful and valuable comments about my research through stimulating and interesting discussions. I sincerely thank all the people at the Department of Computing, Faculty of Science and Engineering, ScienceIT Team and Library at Macquarie University for their warm support and help.

Last, but not least, I am indebted to all the members of my family for their love, support and encouragement throughout my PhD study. Without them this work would have never been realized.

Publications based on This Thesis

[1] Y. Zhong, W. Zhao, and J. Yang. Personal-hosting restful web services for social network based recommendation. Service-Oriented Computing (SOC) 2011, pages 661-668, 2011 [145].

[2] Y. Zhong, W. Zhao, J. Yang, and L. Xu. Peer-based relay scheme of collaborative filtering for research literature. COOPERATIVE INFORMATION SYSTEMS (CoopIS) 2011, pages 321-328, 2011 [146].

[3] Y. Zhong, L. Du, and J. Yang. Learning social relationship strength via matrix co-factorization with multiple kernels. In Web Information Systems Engineering (WISE) 2013, pages 15-28. Springer, 2013 [143].

[4] Y. Zhong, X. Zheng, J. Yang, M. A. Orgun, and Y. Wang. Kpmcf: A learning model for measuring social relationship strength. In Web Information Systems Engineering (WISE) 2013, pages 519-522. Springer, 2013 [147].

[5] Y. Zhong, J. Yang, and R. Nugroho. Incorporating Tie Strength in Robust Social Recommendation. In Big Data, 2015 IEEE International Conference on. IEEE, 2015, pp. 63-70. [144].

Contents

List of Figures

List of Tables

Abbreviations

AUC	**A**rea **U**nder **C**urve
CF	**C**ollaborative **F**iltering
CMF	**C**ollective **M**atrix **F**actorization
CN	**C**onte**N**t-based filtering
CII	**C**ommon **I**nterested **I**tem
LVM	**L**atent **V**ariable **M**odels
MAE	**M**ean **A**bsolute **E**rror
MF	**M**atrix **F**actorization
MFSR	**M**atrix **F**actorization based **S**ocial **R**ecommendation
PCC	**P**earson **C**orrelation **C**oefficient
PRC	**P**recision-versus-**R**ecall **C**urve
RMSE	**R**oot **M**ean **S**quared **E**rror
ROC	**R**eceiver **O**perating **C**haracteristic

To my wife and my great family . . .

Chapter 1

Introduction

This thesis presents a novel recommendation model: a peer-based recommendation model incorporated with social relationship strength. In this chapter, we highlight the principal elements of the research in this thesis. We first provide an overview of the research problem, then describe the main challenges confronting the peer-based recommendation model, and finally summarize the major contributions of the research. We show the organization of this thesis in the last section.

1.1 Research Overview

Recommendation is a natural social process in everyday life. People receive recommendations in various ways such as spoken words, travel guides, letters of reference, classified advertisement, purpose-specific surveys, and so on. Recommender systems are those intelligent computer programs which help people to examine available information in computer systems in order to find potentially interesting and favorable products or services - movies, books, music, restaurants, home appliances, and so on.

In order to analyze a large amount of information, recommender systems mainly employ two types of filtering approaches: *Content-based Filtering (CN)* and *Collaborative Filtering (CF)* [3]. The *CN* approach creates profiles for items or users to describe their characteristics. For instance, a profile of a piece of music may include its attributes such as debut date, composer, genre, album, playing time, number of listeners, and so on. A profile of a user may contain age, gender, nationality, education and employment history. These profiles are then used by the recommender systems to find the items which may match users' preferences.

In contrast to *CN*, the *CF* approach relies on users' past behavior or experiences, for instance users' ratings over items, without the need for explicit profiles of users or items. The fundamental assumption of *CF* is that, if user u and user v give similar ratings over a number of items, the two users are considered to have similar preferences, and consequently the two users may take similar actions for other items [44].

The *CF* approach can be further categorized into two groups: memory-based and model-based methods [20]. Memory-based *CF* methods directly use user's ratings to calculate the similarity between the users or items, and make predictions according to these similarity values. While model-based *CF* methods firstly develop certain mathematical models, and then apply the models to available rating data to predict those items which have not yet been rated.

Of the advanced model-based *CF* methods, *Matrix Factorization (MF)* represents a type of state-of-the-art modeling techniques, providing superior accuracy and flexibility than other model-based recommendation methods [67]. In its basic form, matrix factorization decomposes a matrix ("target matrix"), for instance a user-item rating relation, into two or more "latent matrices" representing the latent features of the users and items, such that the matrix product ("approximate matrix") of the latent matrices will approximate to the original target matrix. A basic approach for using matrix factorization for recommendation tasks is to learn the missing values in the "target matrix" from the "approximate matrix". Usually, a high correspondence between the features users and items leads to a recommendation.

Generally speaking, to make recommendations for a particular user (we often refer to a user seeking recommendations as an "active user"), a typical *CF* method first selects other users having similar tastes with the active user by analyzing the rating patterns in computer systems. The selected users form a recommendation-neighborhood in the context of collaborative filtering. From this neighborhood, the *CF* method collects all the items rated by the neighbors. Finally, the method chooses those items which are highly appreciated by the neighbors but not yet experienced by the active user. Because the *CF* approach makes recommendations based solely on users' rating patterns without the need for knowledge of either items or users, it has been widely adopted.

There are many issues related to *CF* recommendation methods. Of them, the "data-sparsity" issue is one of the most well-known problems. Generally, in most of the user-item relations, the number of observed ratings will be far less than the multiplication outcome of the numbers of users and items. For instance, the well-cited Netflix data set has a 99% sparsity [17].

The "data-sparsity" issue also appears when new users or new items are added to *CF* recommender systems, being called a "cold-start" problem as well. Because the new users or new items are mostly associated with no ratings, it is difficult for the *CF* recommender systems to make decisions for them. Besides these well-known issues, when we take social connections and interactions into consideration, there are several more challenging problems to be addressed: small-sized neighborhood, types of networks, and rich social information.

The key to a successful *CF* recommender system is to form a reasonably small-sized neighborhood. This is in fact a two-fold dilemma. On the one hand, a small size of neighborhood means less computation load. On the other hand, however, a small size may lose some of the potentially interesting items. Therefore, the size of the neighborhood is sensitive to the performance of recommender systems. This will become more severe when the number of the users and items explosively grows. For instance, the aforementioned Netflix dataset includes over 100 million ratings from more than 480 thousand users for nearly 18 thousand movies [17]. How to find a reasonably small-sized neighborhood from a huge numbers of users and items becomes a practical issue for all *CF* methods.

Next, recall that the *CF* approach relies only on users' ratings, therefore, the conventional *CF* methods hardly take social information into account. The studies in social science point to the phenomenon that people have similar tastes essentially because they may have similar personal or social characteristics, such as age, gender, location, education, jobs, friends, associations, and so on. This is called “homophily in social networks” [88]. In recent years, many studies have proposed to include social information in recommender systems along with rating data. Most of the current studies do so on an ad-hoc basis, for instance friend-relationships, locations, tags, or the like. We believe that an integrated form of various relationships should be developed so that recommender systems can make use of rich social information.

Furthermore, recent studies suggest that the recommendation-neighborhoods can be categorized into two basic types: similarity- and familiarity-networks [50, 108]. The former includes those users whose social activities are found to overlap with other users' behavior, while the latter consists of those users whom other users have known. Intuitively, for the purpose of seeking recommendations, a user often prefers self-organized social networks with whom the user was already familiar or from whom the user often receives advice. In a survey conducted by Guy et al. [50], the familiarity-network group showed proportions of 45.7% of the items being interested versus 37.1% being not-interested in the recommended items; whereas, the similarity-network group exhibited proportions of 38.7% and 48.2%, respectively. These outcomes demonstrate “the superiority of the familiarity-network as a basis for recommendations” [50]. A challenging task here is to formulate the construction process of such a familiarity-network in computer systems.

To address the above issues, the peer-based recommendation process in daily life could be a promising solution. In such a process, the personal social circles are typical familiarity-networks, each of which has a limited number of friends or peers. When a user seeks recommendations in such a setting, he/she will mostly ask for help from these peers, and the peers will respond to the recommendation request by either utilizing their local resources or forwarding the request to their circles. Owing to the nature of social circles, in many circumstances users are more willing to share their personal information within local circles than over public platforms.

Owing to the advent of social media services in recent years, it becomes now possible and necessary to build recommender systems following the same approach as we make and receive recommendations in real life. Firstly, the online friend-relationships have become increasingly popular, which form the basis of familiarity-networks. As shown in the study by Lampe, Ellison and Steinfeld [73], the average number of "Facebook friends" was 201 in 2006, and grew to 333 in 2008, increasing about 65% in two years. Moreover, a recent survey [104] in 2012 reported that, 62% of the adults worldwide were using social media services, the numbers of online friends per user ranged from 29 to 481 in various countries.

Besides the online friend-relationships, a new feature of the "Social Circle" has been introduced in most popular social media services in recent years. For instance, the "Circles" of *Google+* [1] and the "Moments (Friend Circle)" of *WeChat* [2] allow users to assign classmates, family members, and colleagues to different groups. This is a typical form of familiarity-networks. Likewise, *Facebook* encourages users to assign his/her friends into distinct "Groups" [3] for more finely

[1]https://plus.google.com/
[2]http://www.wechat.com/
[3]https://www.facebook.com/about/groups

grained information-sharing than a single friend list. In the case of *Twitter*, users are allowed to organize *followees* into distinct “Lists” [4] such that users are able to see the tweets posted from the followees with particular concerns.

Nevertheless, it is not a straightforward task to build such models that emulate the recommendation process in real life. There are a number of challenges:

- First, in the recommendation process of real life, most users have only a limited number of friends or peers. For the purpose of forming recommendation neighborhoods, simply considering all the friends or peers may not make sense to deliver high quality recommendations. It is still a challenging task to form reasonably sized neighborhoods from the available information in computer systems.
- Next, most of the current filtering algorithms are based on globally collected data, for instance all users’ ratings or preferences over all the items. However, in the case of personal social circles, most users keep their data locally and prefer the data to be only accessed by their peers. This private nature of local data makes it difficult to collect and process users’ data by a global approach. To deal with such a situation, a demanding job will be developing a specific filtering algorithm to collaborate peers through social circles.
- Also, in real life, the recommendations for an active user are mostly contributed by either the peers in the user’s direct social circles, or those in the peers’ circles. On many occasions, the active user refines the recommended items based on the social relationship strengths between the active user and

[4]https://twitter.com/(username)/lists

the peers. This directly brings in two challenges: (i) how to capture an integrated weight of various social relationships? (ii) how to measure the weights or strengths?

- Last, supposing we are able to measure the strengths of the social relationships among the users in a social network, the subsequent task is to employ the strengths in recommender systems. Although there are many studies that include particular social information in recommender systems, there remain a number of issues such as "robustness" that indicates the ability to deal with the outliers and imbalance of data. It is still a demanding task to develop and evaluate a robust recommendation method to incorporate the relationship strengths in recommender systems.

In this thesis, we propose a peer-based social recommendation model to address the aforementioned challenges. In particular, we develop a relay-based neighborhood formation method, allowing active users to find the peers who can contribute recommendations. We also design a set of efficient peer-based collaborative filtering algorithms that propagate recommendation requests and responses through social circles.

Utilizing the state-of-the-art *Matrix Factorization (MF)* technique, we develop a machine learning method to measure the tie strengths among the users in a social network from demographic and interactive information. This tie strength represents the integrated weight of combined social relationships. Subsequently, we incorporate the tie strengths in recommendation process to improve recommendation qualities.

We have conducted comprehensive experiments by using the real datasets from popular social media services *Flickr*, *Epinions.com* and *Last.fm.* The evaluation results demonstrate that our proposed recommendation model outperforms both popular *CF* algorithms *Pearson CF* and *Slope One*, and certain state-of-the-art social recommendation methods *KPMF*, *SoRec* and *SR2*, in terms of widely-accepted evaluation metrics: *Precision-versus-Recall Curve (PRC)*, *Mean Absolute Error (MAE)* and *Root Mean Squared Error (RMSE).*

1.2 Research Challenges

In this section, we describe three major challenges in our research: making recommendations within social circles, measuring tie strength, and incorporating tie strength into a robust recommendation process.

1.2.1 Making Recommendations with Social Circles

Recommender systems are the computer programs that select potentially interesting items for users by analyzing the information in computer systems about the users and items. Taking the widely adopted *CF* approach as an example, a conventional *CF* method takes the following steps to produce recommendations:

(i) based on an active user's information and certain *similarity definitions*, the method searches all the users with their preferences say ratings over the items available in a computer system, in order to find those users who have the similar tastes to the active user;

(ii) using certain *prediction algorithms*, the method predicts the ratings over the items which have not been experienced by the active user, and finally selects only the highly scored items as recommendations;

(iii) the above *similarity definitions* and *prediction algorithms* are defined based on users' rating patterns from a global perspective [3, 62].

However, such a conventional method may not directly match the peer-based nature of recommendation in our daily life, where each user has one or more social circles with a limited number of peers, and so do the peers. Usually, the recommendations are contributed by the peers in a parallel manner based on their locally held data. That means the recommendation process should be essentially based on a local perspective using a collaborative approach. Furthermore, the active user often refines the feedback from the peers according to the weights of the relationships between the active user and the peers. This suggests that the recommendation process in real life often takes social information into consideration.

The *CF* methods aiming to address the above issues can be grouped into two types: *Familiarity-based Collaborative Filtering* and *Decentralized Collaborative Filtering*. Of the first group, most studies built up recommendation neighborhoods by searching the peers through a personal social network, and then predicted the rankings of the items rated by the neighbors using conventional *CF* algorithms [16, 50, 51, 66].

Owing to the neighborhood formation from personal social networks, these studies especially improved the quality of personalized recommendations. However, most studies in this group manipulate data using a globally collecting and processing approach. This approach may meet difficulties when dealing with personal social circles where the users prefer the data to be propagated only by their peers. Also, most of these studies did not fully exploit the rich social relationships between the active user and his/her peers in a social network.

The studies in the group of *Decentralized Collaborative Filtering* were basically applying distributed computing techniques to the recommendation process [11, 30, 53, 89, 128, 132, 136]. The term "decentralized" here indicates a direct opposition to "centralized" computing of *CF* methods, which maintain data and conduct filtering in a single computer. Such a centralized architecture consequently faces a number of problems such as data sparsity and system scalability when the numbers of users and items dramatically increase. In contrast to "centralized computing", most studies of *Decentralized Collaborative Filtering* deal with the data scattered throughout a network of computers and perform information filtering in each node of the network.

While some proposals in this group merely provided distributed data storage functionality [53, 136], certain studies of *Peer-to-Peer (P2P)* recommendation models emulated the peer-based recommendation process in daily life [11, 30]. For instance, the *P2Prec* system proposed by Draidi, Pacitti and Kemme [30] was built up on a *P2P* network for large-scale document sharing by using *LDA* technique. Here *LDA* stands for *Latent Dirichlet Allocation*, a generative probabilistic method used for deriving topics from documents [18].

The *P2Prec* system of Draidi et al. [30] starts from a bootstrap server, which runs an *LDA* program for periodically new-coming documents from all the peers, and also maintains a matrix of global topic distribution. In the meantime, each peer in the system executes a local *LDA* program for inferring local topics of the peer's local documents, using the global topics as prior knowledge. The local topics are then used to characterize each peer by interests-in-topics. With these interests, each user update its neighborhood of the most relevant peers by periodically *gossiping* with other peers in the network. When the neighborhood becomes stable, the user then makes recommendations by using a certain form of general *CF* algorithms.

Although the *P2Prec* system did allow every peer to work on its local data and to communicate with other peers using distributed *gossip* protocol, the design of the "bootstrap server" somehow becomes a bottleneck of distributed computing. Also, the concept of "interests-in-topics" has limitations in dealing with additional social information in the recommendation model.

In summation, to make recommendations for social networks with social circles, there is a need to answer the following questions: (i) how to form quality recommendation neighborhoods from social circles? (ii) how to work with the data locally kept by the peers in a distributed fashion? (iii) how to take account of the rich social relationships among the peers in a social network?

1.2.2 Measuring Tie Strength

The previous subsection describes the challenges to neighborhood formation from social circles. The concept of "neighborhood" in recommender systems means a collection of other users having similar tastes or interests. This concept directly coincides with the "homophily" and "tie strength" studied in Sociology. Homophily is described as a "principle that a contact between similar people occurs at a higher rate than among dissimilar people" [88], while the tie strength means users' overall attitudes towards others in a social network [47].

In particular, the studies in the literature investigate three types of homophily-related characteristics: demographic, psychological and social. The demographic characteristics include age, sex, race/ethnicity, education and religion. The examples of psychological characteristics are intelligence, attitudes and aspirations. Social characteristics comprise occupation, social class, network positions, behavior, and the like [88].

It is worth noting that the literature on "homophily in social networks" also noted that peer groups were an important source of influence on people's behavior. In particular, the association of some characteristics between a user and its peers could be used as the evidence to measure the tie strength among the users in a social network [25, 31].

The early studies on homophily targeted at small social groups such as school children, college students and small urban neighborhoods. Then, the objects being studied were scaled up to schools, communities, or even the US population as a whole. In recent years, the studies of homophily concentrated on the organizational contexts of networks, such as organizations, work force, web pages, and so on. [88].

While the studies on homophily focused on the significance of various individual characteristics, the studies of tie strength paid attention to the overall attitudes of users towards others in social networks. In particular, the tie strength is an abstract and integrated concept, which was primarily defined by Granovetter as "a combination of the amount of time, the emotional intensity, the intimacy (mutual confiding), and the reciprocal services which characterize the ties" [47].

In line with the study by Granovetter [47], much work has been done to analyze the predictive factors of tie strength. The study by Marsden and Campbell [85] pioneered the development of tie strength measurement among the users in social networks by using multiple indicator techniques. In particular, this study defined two types of variables with tie strength: indicators and predictors. The indicators were observed measures of social network ties, including closeness, duration, frequency, breadth of discussion topics, and confiding, whereas the predictors were unobserved aspects related to tie strength, such as kinship, co-worker, and neighbor status. Using a regression-styled approach, this study designed a linear function between tie strength and predictors, and a set of linear functions associating the predictors with the indicators. The major contribution of this study is a comprehensive analysis of the significance of certain observed indicators of tie strength. For instance, it suggested a measure of "closeness" or emotional intensity was the best indicator of tie strength.

Similarly, the study by Gilbert and Karahalios [42] used a statistical method to analyze 74 variables of tie strength in seven major dimensions, based on a survey over 35 participants with 2,184 rated Facebook friendships. The seven dimensions were intensity, intimacy, duration, reciprocal services, structural, emotional support and social distance. Likewise, the study by Kahanda and Neville [61] proposed a method for using 50 features to predict *link strength*, a synonym of tie strength, among the users in online social networks. These 50 features were categorized into four groups: attribute-based, topological, transactional and network-transactional. Both studies made similar findings to those of Marsden and Campbell [85].

Owing to its importance and usability, tie strength has been applied to a wide range of application domains, such as emotional health [113], professional social networks [69, 70], economic outcomes [49], mobile communication [103, 141], online social behavior analysis [9, 13, 60, 135, 141], criminal networks anatomy [28, 115], and so on.

All the above studies defined or measured the tie strength directly based on manifest variables, which were either observed or well-known ones. However, tie strength is in fact an abstract and integrated form of various social relationships or ties. Therefore, we believe that *Latent Variable Models* are the promising techniques for measuring tie strength, where tie strength could be the projection of some latent variables that are somehow unknown or potential ones though possibly validated by facts, for example, peoples political tendency and business feeling against particular events [12]. In other words, a major challenge of measuring tie strength is to capture and quantify these latent variables.

Recently, a number of studies are proposed to measure tie strength by means of latent variables [135, 142, 149]. The study by Xiang et al. [135] is a typical case. This study considered tie strength as a latent effect of "profile similarities", and also assumed that the tie strength would impact users' interactions. Consequently, this study modeled tie strength as the conditional probability of a Gaussian distribution given profile similarities, and also the probability distribution of social interactions. When apply this model, one needs to prepare two specific data matrices: "profile similarities" and "user interactions", compiled from users' demographic and interaction information.

Apart from the study by Xiang et al. [135], other studies on tie strength either exploit a single type of data for instance users' interactions, or use a single method to handle diverse form of information. To perform these methods, one needs to pre-compile diverse information into a single relation table. This may be difficult in many cases, for example, users' education information may have only three levels, but the ethnicity information may comprise more than one hundred countries or areas.

In order to precisely measure tie strength in social networks, a critical task is to directly infer tie strength from heterogeneous social information, including users' demographic and social interaction information. The former includes age, gender, location, education, skills, jobs, and the like; and the latter contains posting comments, setting tags, sharing reviews, and so on.

1.2.3 Incorporating Tie Strength in Recommendation

Provided that we have successfully measured the tie strengths from rich social information, the next challenging task is to incorporate the tie strength into recommender systems. Many studies have been proposed to include social information in recommendation process. Most of these studies are based on variant *Matrix Factorization* techniques [137], which we call Matrix Factorization based Social Recommendation or *MFSR* methods. From the perspective of technology, these methods can be categorized into three major approaches: *Social Ensemble* [57, 82, 84, 102, 138], *Kernelized Matrix Factorization* [4, 148], and *Collective Matrix Factorization* [35, 78, 83, 118, 119, 139].

A common rationality behind the methods of *Social Ensemble* is that the observed ratings of a user should represent not only the user's own taste but also the preferences of his/her friends. For this reason, these methods model the latent variables by various combinations of the preferences of both the users and the friends.

On the other hand, *Kernelized Matrix Factorization* improves recommendation qualities by assigning kernels to the priors of latent feature vectors. Certain studies reported great achievements of using kernelized methods for image processing and recommendation, however, in our experiments, for instance, of *KPMF* model [148] we find the *Kernelized Matrix Factorization* approach always stay at the middle positions for metrics measures in most evaluation cases.

Instead of treating social information as "side information" along with the single rating matrix, *Collective Matrix Factorization* twistingly factorizes two or more target matrices in order to affect the distinct latent variables by each other. Taking the *SoRec* model of [83] as an example, along with an existing rating matrix, the *SoRec* constructs a specific "social network matrix" for side information, of which each element represents the local authority and hub values between two users. Then, the *SoRec* performs matrix factorization simultaneously over the two target matrices.

Although the aforementioned three types of *MFSR* methods have achieved great successes in various circumstances, several issues are remaining. In particular, we found that, through our experiments with joining tie strength, most of the existing methods received good values of *Mean Absolute Error* and *Root Mean Squared Error* but poor measures of *Precision-versus-Recall*. That means these methods might have good quality of proximity but poor character of relevance [10].

This might be contributed by the fact that most of these methods applied a Gaussian probabilistic approach to matrix factorization, in which the objective function is derived from Gaussian priors, that makes it equivalent to a squared loss in factorization process. The squared loss has shown sensitive to the outliers of data, that is considered as a "robustness" issue of matrix factorization [1]. There are several proposals to deal with this issue in general matrix factorization models [72, 133]. However, we have not seen any reports on the improvement in relation to robust *MFSR* methods, especially when incorporated with tie strength.

1.3 Main Contributions

To address the challenges stated in the previous section, in this thesis, we propose a peer-based social relationship enhanced recommendation model. The model consists of three major parts, i.e., a peer-based collaborative filtering method, a measuring method for tie strength, and a robust matrix factorization method incorporated with tie strength. The main contributions of this thesis can be summarized as follows.

- We develop a peer-based collaborative filtering method. The proposed method follows the recommendation style in everyday life, where people find potentially interesting items from peers through personal social circles. The method fulfills the filtering task by allowing peers to propagate recommendation requests and responses in a relay fashion.

- We propose a kernelized probabilistic matrix factorization method to measure the tie strengths among the peers in a social network. The method first learns kernel matrices from users' social interactions, then infers users' latent features by executing matrix factorization over users' profiles, and lastly calculates the tie strength from users' latent features.

- We build a matrix factorization based social recommendation method, incorporated with tie strengths. This method also introduces a robust mechanism in the factorization process.

- We have conducted comprehensive experiments over the real datasets from actual social media services, including *Flickr*, *Epinions.com* and *Last.fm*.

We evaluate our proposed model with two conventional *CF* algorithms i.e., *Pearson CF* and *Slope One*, and three state-of-the-art matrix-based social recommendation methods i.e., *SoRec*, *SR2* and *KPMF*.

Our evaluation results show that the proposed model outperforms the other recommendation methods in most cases in terms of well-accepted metrics: *Mean Absolute Error (MAE)*, *Root Mean Squared Error (RMSE)* and *Precision-versus-Recall Curve (PRC)*. Our evaluation also reveals that tie strength does make significant impact to the qualities of recommender systems.

- During the research for this thesis, we developed an evaluation system for the purposes of experiment and evaluation. This evaluation system is equipped with a number of functions such as measuring tie strength from multiple relations, making recommendations using selected algorithms, evaluating the performance of specific methods, drawing graphical presentation of various metrics, and so on.

1.4 Thesis Organization

The rest of this thesis is organized as follows.

In Chapter 2, we provide a comprehensive review of recommender systems, especially focusing on two important categories of model-based collaborative filtering methods: *Regression Models* and *Latent Variable Models*. Our proposed peer-based filtering method belongs to *Regression models*, and our measuring method for tie strength is a *Latent Variable model*. Furthermore, we explore the studies on three related topics: Familiarity based Neighborhood Formation, Tie Strength Measurement, and Matrix Factorization based Social Recommendation.

Chapters 3, 4 and 5 are the core elements of the research in this thesis, describing three important aspects of our proposed recommendation model: peer-based filtering, tie strength measurement, and incorporation of tie strength. All three chapters are presented in a similar style. In each chapter, the first section introduces the motivation of the work. Then, two or three sections are contributed to elaborate the details of algorithms. After the algorithms, the experiment settings and evaluation results are shown in the following two sections. At last, each chapter is finished with a section of comparison and summary.

In particular, Chapter 3 presents a peer-based filtering method *CoRec*, which emulates the recommendation process in real life. We elaborate two key aspects of the method: a relay-based process and a variety of peer-based filtering algorithms.

In Chapter 4, we propose a collective matrix factorization method for measuring tie strength, named as *KPMCF*. We firstly review the studies of tie strength. Then, we present our method in three steps: (i) learning kernels from social interactions; (ii) inferring users' latent features by factoring users' profiles; and (iii) calculating pair-wise tie strengths from users' latent features.

In connection with the techniques proposed in Chapter 3 and Chapter 4, Chapter 5 develops a robust matrix factorization method, named as *TieRec*, for incorporating tie strength in recommendation process. This chapter includes comprehensive experiments on *TieRec* in comparison with other state-of-the-art recommendation methods. The evaluation results demonstrate that the proposed *TieRec* outperforms other methods in most evaluation cases, in terms of *Root Mean Squared Error* and *Precision-versus-Recall* measures.

Finally, Chapter 6 makes brief concluding remarks and also discusses the potential work in the near future.

The Chapter of *Appendix* describes an evaluation system developed during the research work of this thesis. This chapter includes the outline of system modules, graphic user interface, and major functions. The comprehensive functionality of the evaluation system makes it possible to upgrade to a common tool for evaluating social recommendation methods.

Chapter 2

Background

In this chapter we present the background of recommender systems. This chapter is organized as follows:

The following section introduces the notations to be used throughout the thesis. Section 2.2 provides an outline of recommender systems, including two major approaches: *Content-based filtering* and *Collaborative filtering.* In this section, we particularly explore the popular collaborative filtering methods in two important categories, i.e., *Regression Models* and *Latent Variable Models.* In fact, the methods studied in this thesis mainly belong to these two categories.

In section 2.3, we briefly review three closely related topics: Familiarity based Neighborhood Formation, Tie Strength Measurement, and Matrix Factorization based Social Recommendation.

At last, section 2.4 recapitulates the research work of the thesis, showing our position within underlying methodology.

2.1 Notation

In this section, we provide the notations used throughout this thesis, including those for variables and matrices, probability and statistics, as well as plate models.

2.1.1 Variables and Matrices

- Scalars are denoted by lower-case Roman or Greek letters when used to express variables, for instance y, λ. When used as constants, the scalars are denoted by upper-case Roman letters in order to differentiate from other variables, for instance M or N for matrix dimensionality.

- Vectors are denoted by upper-case Roman or Greek letters: V, Λ. A vector of length n can be denoted as $V = (v_1, v_2, ..., v_n)$. Vectors are assumed to be row-vectors. When a vector is used as an element of a matrix, then the notation of matrices is applied.

- Matrices are denoted by capital-italic Roman or Greek letters, and sometimes shown with superscript, for instance M, Ψ, or Ψ^t. The rows, columns and elements of a matrix are denoted by corresponding lower-case letters with subscripts. For example, m_i or $m_{i:}$ stands for the i-th row, $m_{:j}$ for the j-th column of matrix M, whereas m_{ij} stands for an element of M at row i and column j and $(\psi^t)_{ij}$ for that of Ψ^t.

- The set of real numbers is denoted as $\mathbb{R}$. Real coordinate spaces over N coordinates are denoted by $\mathbb{R}^N$. In the case of matrices, real coordinate spaces on $M \times N$ dimensions are denoted as $\mathbb{R}^{M \times N}$.

- A norm is denoted by $||\cdot||$, likewise, the Euclidean norm on $\mathbb{R}^N$ is denoted as $||\cdot||_2$. The Frobenius norm on $\mathbb{R}^{M\times N}$ is denoted as $||\cdot||_F$, and is defined as

$$||X||_F = (\sum_{i=1}^{M}\sum_{j=1}^{N} x_{ij}^2)^{1/2} \tag{2.1}$$

- The gradient of a function f is denoted by ∂f. The gradient of a function f with respect to its variable x is denoted as $\frac{\partial f}{\partial x}$.

2.1.2 Probability and Statistics

- Random variables are denoted in the same way as those of Variables and Matrices (section 2.1.1). In particular, we refer to $\mathcal{X}$ as a dataset consisting of a set of records, each being associated with a set of n random variables $x_1, \cdots, x_n$.

- A draw from a random variable x with probability density function $f(\theta)$ is denoted as:

$$x \sim f(\theta) \tag{2.2}$$

where θ represents the parameters of the probability density function f.

- A simple distribution of a single random variable and a joint distribution on multiple random variables are expressed as follows:

$$p(x|\theta) \; or \; p(x;\theta) \quad \text{(simple distribution)} \tag{2.3}$$

$$p(x_1, \cdots, x_n|\theta) \; or \; p(x_1, \cdots, x_n;\theta) \quad \text{(joint distribution)} \tag{2.4}$$

  where θ represents the parameters of the distribution.

- The joint distribution of a set of random variables can also be expressed as the product of all the probability distributions of the variables given their corresponding parent variables:

$$p(x_1, \cdots, x_n|\theta) = \prod_{i=1}^{n} p(x_i|x_{\pi i};\theta_i) \tag{2.5}$$

  where $x_{\pi i}$ is the parent variable of x_i in the manner in which the conditional probability of x_i is conditioned on $x_{\pi i}$. Moreover, θ_i is the parameter of the corresponding conditional probabilities.

- The above joint distribution can be used to define a likelihood function, where $\mathcal{X}$ can be considered as a set of fixed training data, and the likelihood function assigns a value to each possible θ for $\mathcal{X}$.

  Moreover, for a given set of a training data, the problem of choosing the parameters which maximize the likelihood function is called *Maximum Likelihood Estimation (MLE)* (Equation 2.7), that is essentially a point estimation of the parameters.

$$l(\theta) = p(\mathcal{X}|\theta) = p(x_1, \cdots, x_n|\theta) \tag{2.6}$$

$$\theta_{MLE} = \text{argmax } \ l(\theta) \ = \text{argmax } \ p(\mathcal{X}|\theta) \tag{2.7}$$

- Because a log-normalizer can reduce a high-dimensional integration to a low-dimensional one, the problem of *Maximum Likelihood Estimation (MLE)* can be equivalent to the problem of minimizing a log-summation form by applying logarithm to the likelihood function. Equation 2.8 shows such a log-likelihood function ($ll(\theta)$).

  Such an estimation can be simplified provided that each $p(x_i|x_{\pi i};\theta_i)$ is a certain type of a distribution, that depends only on the parent variable $x_{\pi i}$ and parameter θ_i:

$$\begin{aligned} ll(\theta) &= -log\ p(x_1, \cdots, x_n|\theta) \\ &= -log \prod_{i=1}^{n} p(x_i|x_{\pi i};\theta_i) \\ &= -(\sum_{i=1}^{n} log\ p(x_i|x_{\pi i};\theta_i)) \end{aligned} \tag{2.8}$$

- Also, the parameters θ can be determined by *Bayesian Inference*, which solves the posterior distribution of the parameters conditioned on the training data by using Bayes theorem:

$$p(\theta|\mathcal{X}) = \frac{p(\mathcal{X}|\theta)p(\theta)}{p(\mathcal{X})} \tag{2.9}$$

where $p(\theta)$ is called the prior probability distribution, representing the prior belief in various conditions of parameter space. On the other hand, $p(\mathcal{X}|\theta)$ is called the posterior distribution, which is a combination of the prior belief with the likelihood.

- To learn the posterior distribution, one can compute the training data by calculating an integral over a potentially large parameter space.

$$p(\mathcal{X}) = \int p(\mathcal{X}|\theta)p(\theta)\mathrm{d}\theta \tag{2.10}$$

For the purposes of making recommendations, we need to predict the likelihood of new test data $\mathcal{X}^{new}$ given training data $\mathcal{X}$. This is an integral as follows.

$$p(\mathcal{X}^{new}|\mathcal{X}) = \int p(\mathcal{X}^{new}|\theta)p(\theta|\mathcal{X})\mathrm{x}\theta \tag{2.11}$$

- As the above integral (Equation 2.11) is usually difficult to calculate, we can use Monte Carlo methods to approximate it. *Monte Carlo* methods are a broad class of algorithms which perform repeated and random samplings to obtain results. In the case of Equation 2.11, it can be approximated by using the following *Monte Carlo* estimation:

$$p(\mathcal{X}^{new}|\mathcal{X}) \approx \frac{1}{S}\sum_{s=1}^{S} p(\mathcal{X}^{new}|\theta^s) \tag{2.12}$$

$$where\ \theta^s \sim p(\theta|\mathcal{X})$$

where $\mathcal{X}^{new}$ represents the new test data, $\mathcal{X}$ the training data, and S the size of parameter space.

2.1.3 Plate Models

Plate models are used to represent the repeated structures in probabilistic graphical modeling, where the nodes stand for variables and the arc for the dependencies between the variables. Probabilistic graphical modeling is often used to define the mathematical forms for joint or conditional probability distributions between variables [22, 96].

A plate indicates that the inside variables are repeated. The subscripts of the variables are the indexes for repetition, and the frequency of the repetition is usually denoted at the corner of a plate. If a variable is embedded in two or more plates, that means it involves multiple layers of repetitions.

Figure 2.1 and Figure 2.2 demonstrate how plats are used to represent the repetition of variables. In these two figures, the already known variables are represented by shading nodes, while those unknown or to-be-learned variables are shown as clear nodes. Usually, it is optional to draw circles surrounding hyper-parameters, which are the variables defined by the users prior to learning process.

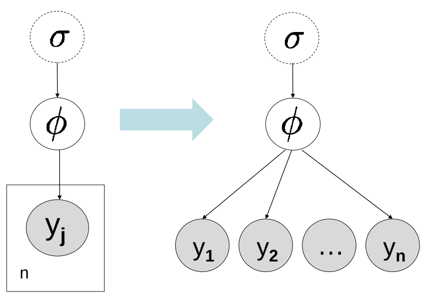


Figure 2.1: Single plate

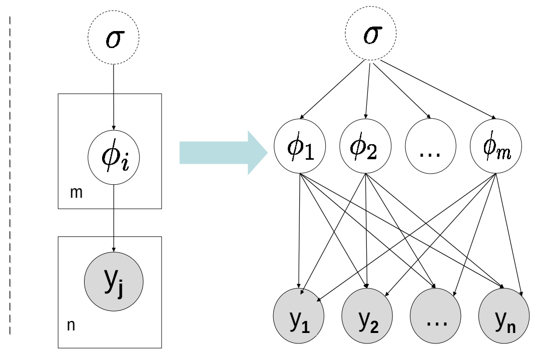


Figure 2.2: Hierarchical plates

In the left-hand image of Figure 2.1, a plate surrounding variable y_j with a number n at the corner. This indicates that variable y has n-times of its values. In another word, one ϕ is associated with n-times y values. This can be alternatively represented by an expanding image at the right-hand of the figure, where all the values of $y_1, y_2, \cdots, y_n$ are explicitly depicted. In contrast to y, either the variable ϕ or the hyper-parameter σ still represents a single value in the expanded image.

Figure 2.2 shows a more complicated case of hierarchical plates. In the left-hand image of Figure 2.2, the variable ϕ is also surrounded by a plate with repetition times of m, and this variable ϕ is linking to an n-times-repeated variable y. The expanded image at right-hand clearly shows the complicated relations among the values of σ, ϕ and y: one σ is linking to m-times ϕ values, while each ϕ value is associated with n-times y values.

2.2 Recommender Systems in General

With the surge in the popularity of Web technology, people constantly use social media services, collaboration tools, and wikis to search, accumulate and acquire new knowledge, as well as to share the information with their friends and colleagues. The Web technology also allows users to be informed of the latest updates from their social connections. However, the abundance of information on the internet becomes a bottleneck of communication.

To address the "information overload" problem, recommender systems have been widely adopted in various web systems. On one hand, the recommender systems help users find personalized recommendations that suit users' tastes. On the other hand, these systems assist businesses to discover unprecedented opportunities for special needs from customers. Nowadays, recommender systems have become an indispensable part of our life.

Recommender systems are intelligent computer programs that are able to filter the large amount of information in computer systems for users' needs. Although such systems have been available for long time, it is only since the mid-1990's that the study of recommender systems has emerged as an independent research area. The problem being studied is the rating estimation for those items which have not yet been experienced by the users, based on existing rating structures. Theoretically, recommender systems are rooted from various research fields, such as information retrieval [111], forecasting theories [7], management science [98], cognitive science [107], approximation theory [105], machine learning [125, 134], and so on.

2.2.1 Content-based and Collaborative Approaches

Generally speaking, the underlying methodology of various recommender systems can be divided into two approaches: content-based filtering (*CN*) [14] and collaborative filtering (*CF*) [106, 117]. In the *CN* approach, every user and item is associated with a profile describing its characteristics. For example, a user's profile may include age, gender, nationality, location and education. Similarly, a profile of a piece of music may contain its attributes such as debut date, composer, genre, album, performance time, number of listeners. These profiles are then used by the recommender systems to select the items matching individual user's requests according to certain similarity definitions.

A well-known *CN* recommender system is the *Music Genome* system, used for Internet radio service *Pandora.com* [1]. In the system, each song is given scores based on hundreds of distinct characteristics, which include not only the musical attributes of music but also other significant qualities that help to understand listeners' preferences.

The *CN* approach has many limitations such as restricted feature extraction, the "New User" problem and over-specialization. First, in order to create the profiles, all the content regarding users and items must be in a text-based form so that they can be "automatically and directly read" by computer systems. However, it is often difficult to automatically and directly derive text descriptions from some forms of data, for instance, images, audio and video streams.

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

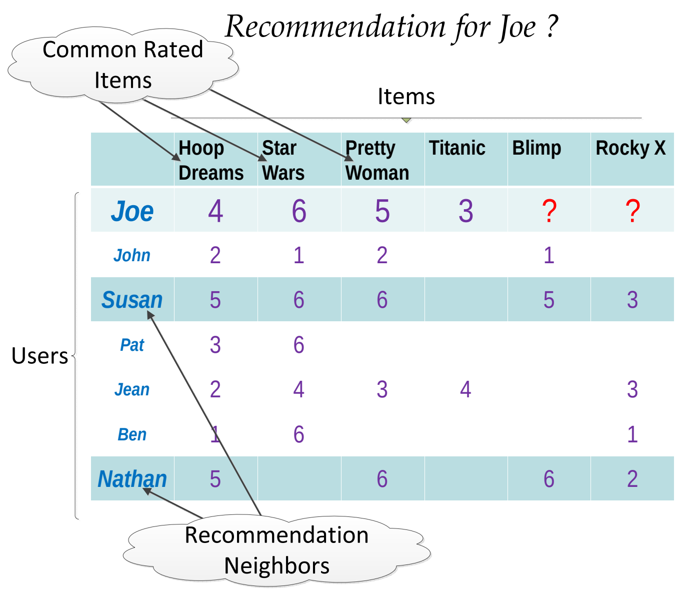
Another limitation occurs in the situation of "New User". When a new user comes to a *CN*-based recommender system, the system has too little knowledge to create the user's profile. Therefore, the system can not make accurate recommendations for the user until a comprehensive profile is established.

The problem of over-specialization appears in two respects. On one hand, the *CN*-based systems mostly cannot recommend certain "fresh" items for a user that are different from the user's profile, because new items are selected by matching the existing profiles of users and items. On the other hand, these systems may recommend so many items similar to those items which a user has already experienced; for instance a number of different articles for a single event but having similar profiles to existing ones holding by the user.

Differing from the *CN* approach, the *CF* approach basically recommends those items which are highly appreciated by other people having similar tastes. Coined by the developers of the first recommender system *Tapestry* [43], the *CF* approach was developed based on a common assumption: if two users u and v have similar rating patterns over a number of items, then these two users may have overlapping preferences for other items as well [44]. Some of the successful commercial deployment of *CF* approach includes Amazon [77], TiVo [5] and Netflix [17].

Figure 2.3 illustrates how a *CF* recommendation method works. As shown, provided that there are seven users and six items (movies), all the users give ratings over certain movies, ranging from one to six (one indicates the most-disliked, six signifies the most-liked). The task is to recommend new movies for Joe. This becomes a problem of predicting the scores for not-yet-seen movies *Blimp* and

Rocky X. Firstly, according to the ratings patterns, the *CF* method may select users Susan and Nathan as Joe's recommendation neighborhood because all the three people have a similar rating pattern - giving higher ratings to movies *Hoop Dreams*, *Star Wars* and *Pretty Woman*. Then, the CF method predicts the ratings over the remaining two movies from the ratings given by Susan and Nathan. For simplicity, supposing the prediction takes an average value of the existing ratings given by Susan and Nathan, thus a rating of 5.5 for *Blimp* ($5.5 = (5 + 6)/2$), and 2.5 for *Rocky X* ($2.5 = (3 + 2)/2$) are derived. Obviously, the movie *Blimp* attain a higher rating (5.5) than *Rocky X* (2.5); accordingly, the movie *Blimp* would be recommended for Joe.



	Hoop Dreams	Star Wars	Pretty Woman	Titanic	Blimp	Rocky X
Joe	4	6	5	3	?	?
John	2	1	2		1	
Susan	5	6	6		5	3
Pat	3	6				
Jean	2	4	3	4		3
Ben	1	6				1
Nathan	5		6		6	2

FIGURE 2.3: Illustration of collaborative filtering approach

A major attraction of the *CF* approach is that it purely relies on users' past experiences or observed ratings, without the need to know all the profiles of users and/or items. Generally, *CF* approach achieves higher predictive accuracy than *CN* approach. Nevertheless, the *CF* approach also suffers from a number of issues, such as *Cold-Start* and *Scalability*. The *Cold-Start* issue is also known as the *New*

User/Item problem, similar to that with *CN* approach. Because the *CF* approach solely rely on users' existing ratings to make recommendations, *CF* approach may not work well with new users or new items which are associated with very few ratings.

The *Scalability* problem occurs when the numbers of the users and items substantially grow. For example, with tens of millions of users and/or items, a conventional *CF* recommender system would suffer serious scalability difficulties when it needs to immediately react to online requests for instant recommendations.

To address the above issues, many methods of *CF* approach have been studied. These methods can be further categorized into memory-based and model-based methods [20, 62]. Generally, the memory-based *CF* methods make predictions based on the entire collection of existing ratings; while model-based *CF* methods make recommendations based on certain mathematics models.

2.2.2 Memory-based and Model-based Methods

In memory-based methods, every user has a "neighborhood" consisting of other users having similar interests. For a user seeking recommendations (called an "active user"), by identifying a neighborhood of the user, the unknown rating for a not-yet-experienced item can be computed as an aggregate of the ratings for that item given by the neighbors.

To form the recommendation neighborhoods of users, *Similarity* is an important concept for memory-based *CF* methods. Major similarity definitions are used in memory-based methods, including correlation-based similarity such as *Pearson Correlation Similarity* [106].

The *Pearson Correlation Similarity* calculates the degree to which two entities, say users, are linearly related with each other [106]. Formally, a *Pearson Correlation Similarity* between two users u and v can be described as follows:

$$pcs(u,v) = \frac{\sum_{i \in I}(r_{ui} - \bar{r_u})(r_{vi} - \bar{r_v})}{\sqrt{\sum_{i \in I}(r_{ui} - \bar{r_u})^2}\sqrt{\sum_{i \in I}(r_{vi} - \bar{r_v})^2}} \tag{2.13}$$

where I stands for the set of the items rated by both users u and v; r_{ui} represents the rating given by user u for item i, and r_{vi} by user v for item i; and $\bar{r_u}$ and $\bar{r_v}$ are the average rating values for user u and v respectively.

In memory-based methods, after having constructed a recommendation neighborhood for an active user, the most important subsequent task is to predict the ratings for "potential items", and to produce a recommendation list for the user. Here the potential items mean those items which are highly appreciated by the neighbors but not yet experienced by the active user. A memory-based method involves prediction and recommendation computation. The major algorithms of prediction computation include *Weighted Sum* [106] and *Weighted Average* [112]. Most memory-based *CF* methods employ *Top-N* algorithm or its variants for recommendation computation.

Although the memory-based *CF* methods have many advantages such as being easy to implement and easy to add new data, they also have several limitations. First, because memory-based *CF* methods rely on similarities across all relevant users over all items, it is difficult to pre-compute all the similarities for online queries. Also, because of the similarity computation, memory-based *CF* methods often require all users, including the active user, to have a minimum number of ratings.

To overcome the shortcomings of memory-based *CF* methods, many model-based *CF* methods have been proposed. In recent years, the model-based *CF* methods have become the mainstream of recommender systems. Generally speaking, the model-based *CF* methods take two steps to make recommendations. First, to develop a model from observed rating data; and second, to estimate the scores for the items in question by utilizing the model. Such a model is usually a data mining or machine learning method.

In this thesis, we are especially interested in two popular groups of model-based *CF* methods: Regression Models and Latent Variable Models. In brief, the methods of regression models develop models by learning the relationships between predicted ratings and some other information, for instance, the characteristics of items. Whereas, the methods of latent variable models work on such a presumption that the observed variables in certain circumstances can be projected by some latent variables or unobserved factors. In the next two subsections, we give a short review of these two groups.

2.2.2.1 Regression Models

Regression Models or Regression Analysis is a well-established statistics and probabilistic field, which can be traced back nearly two centuries or earlier, and has been actively applied in almost all scientific disciplines [34]. The studies of *Regression Models* analyze the relationships between a set of explanatory variables and a set of variables of primary interest. Usually the explanatory variables are also called independent variables, and the variables of primary interest are called dependent variables.

One real life example of *Regression Models* can be an analysis of the relationship between a firm's investment in promotion (independent variables) and its sales achievement (dependent variables). The question here is, given the historical data of promotion and sales, how can we predict the potential sales depending on the investment in future promotion?

Although *Regression Models* have a large number of variations, most popular models used in recommender systems belong to *Simple Linear Regression Model*, that basically deals with one independent variable and one dependent variable, and the relationship between them is a linear function. A typical *Simple Linear Regression Model* can be formally described as follows:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \; where \; (i = 1, \cdots, n) \tag{2.14}$$

where x_i is an independent variable, and y_i a dependent variable; β_0 the population Y-intercept, and β_1 a population slope coefficient; ϵ_i is a random error term assumed that $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$.

The establishment of a model like Equation 2.14 is to find the unknown parameters β_0 and β_1. For example, they can be estimated according to the measure of Least

Squares (LS):

$$LS(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2 \tag{2.15}$$

$$\text{for given data } \ (y_i, x_i) \ , \ i = 1 \ \cdots \ n$$

In this subsection, we take a popular collaborative filter algorithm *Slope One* as an example to show how a *CF* system predicts ratings for potential items by means of *Regression Models*. Proposed by Lemire and Maclachlan, *Slope One* [76] is one of the most popular model-based *CF* methods, based on a simplest form of linear function: $f(x) = x + b$.

Owing to its natures of "easy to implement" and "reasonable accuracy", *Slope One* has been widely used as building blocks in a number of recommender systems [21, 39, 40, 59, 63, 91, 94, 124], as well as certain widely-spreading open-source libraries such as *Apache Mahout* [2] and *Easyrec* [3].

The basic principle behind *Slope One* is the "popularity differential" between items for users. This can be illustrated by a toy example in Figure 2.4. Consider two users A and B, and two items I and J, User A rates item I as 1 and item J as 1.5, whereas, user B assigns a rating of 2 to item I. As we observe that user A assigns a higher rating to item J than item I $(1.5 > 1)$, we could assume that user B would also give item J a higher rating then item I, say, $2 + 0.5 = 2.5$ provided that the two users somehow share similar interests.

[2]https://mahout.apache.org/, last access on 17/03/2014
[3]http://www.easyrec.org/, last access on 17/03/2014

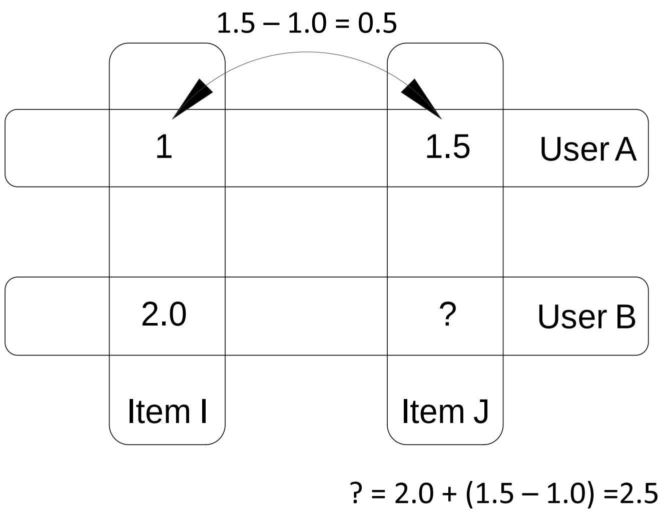


FIGURE 2.4: Illustration of the concept of popularity differential

The *Slope One* algorithm includes a family of schemes. The basic scheme can be considered as a direct implementation of the "popularity differential" concept. Let χ be the set of all the ratings in a training dataset. For any user u, r_{ui} stands for a rating given by user u to item i. Furthermore, for any two items i and j, $S_{ji}(\chi)$ is defined as a subset of χ, including all the ratings r_{ui} and r_{uj} if a user u has rated both items i and j (such a user is denoted as $r_u \in S_{ji}(\chi)$). Also, $T(u)$ is defined as the set of the items rated by a user u.

To formalize the *Slope One* algorithm, an average deviation of item i with respect to item j, denoted by dev_{ji}, is defined, based on which the prediction $\widehat{r}_{uj}$ for a user u for item j can be defined as follows:

$$\widehat{r}_{uj} = \overline{r}_u + \frac{1}{|R_j|} \sum_{i \in R_j} dev_{ji} \tag{2.16}$$

$$where\ dev_{ji} = \sum_{r_u \in S_{ji}(\chi)} \frac{r_{uj} - r_{ui}}{|S_{ji}(\chi)|} \tag{2.17}$$

where $\overline{r}_u$ represents the average rating of user u, R_j is the set of all the other items that satisfy $|S_{ji}(\chi)| > 0$ other than i.

Most of the *CF* methods of *Regression Models* claim to be easy to implement, efficient at query time, and working well for dense data. There are limitations when applying them to sparse data, and as such, the accuracy of recommendations will be severely downgraded [23]. To address this issue, we develop a peer-based *CF* method of *Regression Models* so that reduce the impact of data sparsity. Moreover, we investigate the feasibility of employing *Latent Variable Models* techniques in our recommendation model. The basic concepts and techniques of *Latent Variable Models* are briefly discussed in the next subsection.

2.2.2.2 Latent Variable Models

The study of *Latent Variable Models (LVM)* has a long history, that can be traced from the early part of the last century [12, 33, 123]. The idea of latent variables came from the fact that people who performed well in certain mental ability tests also tended to do well in other things. This led to the thought that the scores of an individual person might be the manifestations of some underlying general ability, called general intelligence. The general intelligence can be considered as the latent variables of test scores. It is the latent variables which are supposed to combine in some way to produce the actual performance in tests.

LVM has been used in a wide range of domains, including education testing, psychology, biology, economy, data mining, image and signal processing, topic modeling, and so on [12, 131]. When apply *LVM* to recommender systems, the "latent variables" often mean the unobserved factors which may affect users' rating preferences. Some examples of unobserved factors are: a movie's nature of serious or escapism, users' attitude towards females or males. The task of predicting users' potential ratings becomes a task of inferring these unobserved factors from the observed ratings. The inferred factors are then used to project the users' possible ratings over not-yet-experienced items. Owing to this reason, the term *Latent Variable Models* is always interchangeably used with *Latent Factor Models* in recommender systems.

As one of the most popular realizations of *LVM*, Matrix Factorization has shown great promise for improving recommender systems in terms of good scalability and high accuracy. In 2006 and 2007, the studies by Funk [38] and Bell and Koren [15] took leading positions in Netflix prize competition [17, 67]. Since then, matrix factorization has become one of the most successful and popular techniques for recommender systems [67, 120, 126].

A general view of *Matrix Factorization* used for recommender systems can be described as follows. Matrix factorization is a process of decomposing a matrix ("target matrix") into two or more "latent matrices", each with a far lower dimensionality than that of the target matrix, such that the matrix product ("approximate matrix") of the derived latent matrices will approximately get back to the original target matrix. Based on the assumption that the latent matrices reflect users' preferences and items' nature respectively in a low-dimension latent space, a basic approach of making prediction is to learn the missing values in "target matrix" from "approximate matrix". Usually, a high correspondence between the factors of users and items leads to a highly potential recommendation.

The above description of matrix factorization can be depicted by Figure 2.5. The target matrix is the matrix $R \in \mathbb{R}^{M \times N}$ representing the ratings for all the users over the movies. Two latent matrices are $\Phi \in \mathbb{R}^{M \times K}$ and $\Psi \in \mathbb{R}^{N \times K}$, representing users' preferences and movie's nature respectively, where ϕ_i and ψ_j are k-dimension vectors, k is the number of latent factors, $k << NorM$.

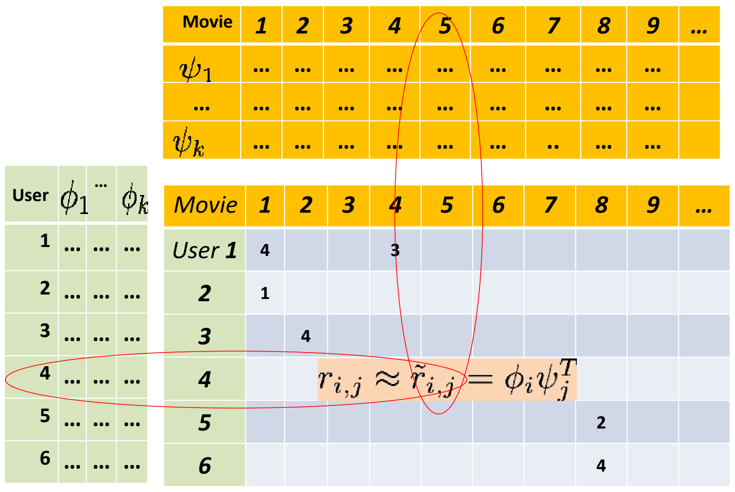


FIGURE 2.5: Matrix factorization over user-movie matrix

As shown in Figure 2.5, the product of Φ and Ψ serves the potential ratings for all the users over all the movies. Formally,

$$R \approx \widetilde{R} = \Phi * \Psi^T \tag{2.18}$$

$$\textit{that is, } r_{ij} \approx \widetilde{r_{ij}} = \phi_{i:} * \psi_{j:}^T$$

The latent matrices are also called parameters or models. The task of Matrix Factorization is exactly to infer these latent matrices, this is usually done by matrix decomposition - a well-defined mathematics problem. There are a number of existing matrix decomposition algorithms, for instance, *Singular Value Decomposition (SVD)* [93]. The definition of SVD is that, for any matrix $A \in \mathbb{R}^{M \times N}$ there exist two orthogonal matrices $U \in \mathbb{R}^{M \times M}$ and $V \in \mathbb{R}^{N \times N}$ such that

$$A = U \Sigma V^T \tag{2.19}$$

where Σ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_{min(N,M)}$ at its diagonal.

SVD has become a basic building block of many applications, such as noise-removal for imaging and sound processing, dimensionality reducing in large scale datasets, latent semantic analysis, and so on. However, when directly apply SVD to recommendation applications, it faces several difficulties. First, conventional SVD can only work well on dense datasets. However, most dataset in recommender systems are very sparse. For instance, the sparsity of well-cited Netflix Prize dataset is high up to 99% [17]. Second, SVD is basically used to minimize squared Frobenius norm, that may not be appropriate to support a sound explanation of the underlying latent factors in recommender systems.

An alternative way of factoring a matrix is to directly model the observed ratings, in the meantime employing regularized terms to avoid over-fitting. In particular, as the performance of a model is usually measured by *Root Mean Squared Error (RMSE)* on test data, it is very likely to learn the latent matrices Φ and Ψ by minimizing the squared difference between a target matrix and its approximate matrix, expressed by the following *Objective Function*:

$$L(R, \Phi, \Psi) = \sum_{(ij)\in K} (r_{ij} - \phi_i \psi_j^T)^2 + \lambda(\|\Phi\| + \|\Psi\|)^2 \tag{2.20}$$

where R represents the user-item matrix of observed ratings, and r_{ij} represents the rating given by user i for item j. K is the set of all the pairs of observed ratings with regard to users and items. The second part of the object function is a regularized term.

As the original *LVM* took a statistical point of view on latent variables, it is natural to assume that the latent variables may follow certain statistical distributions [12]. Based on this assumption, the study by Salakhutdinov and Mnih [110] proposed a matrix factorization model using a probabilistic approach, called *Probabilistic Matrix Factorization (PMF).* In the *PMF* model, the latent variables are supposed drawn from certain probability distributions:

- for each user, draw a user's features u_i from a Gaussian distribution $\mathcal{N}(u_i|0, \sigma_U^2 \boldsymbol{I})$; and
- for each item, draw an item's nature v_j from a Gaussian distribution $\mathcal{N}(v_j|0, \sigma_V^2 \boldsymbol{I})$;
- furthermore, each rating $r(i, j)$ is treated as the posterior distribution over the user's features and item's nature, that is, draw $r(i, j)$ from $\mathcal{N}(r_{ij}|u_i * v_j^T, \sigma_R^2)$.

Formally, let the matrix $R \in \mathbb{R}^{M\times N}$ represent the ratings given by N *users* for M *items*, and $U \in \mathbb{R}^{M\times D}$ and $V \in \mathbb{R}^{N\times D}$ stand for the latent features of *User* and

Item such that $R \approx UV^T$, where D is the dimensionality of latent matrices. To predict the missing values in R, a conditional distribution is set over the observed ratings R, and two zero-mean Gaussian priors on the latent matrices U and V respectively.

$$p(R|U, V, \sigma_R) = \prod_{i=1}^{N} \prod_{j=1}^{M} [\mathcal{N}(r_{ij}|u_i^T v_j, \sigma_R^2)]^{I_{ij}} \tag{2.21}$$

$$p(U|\sigma_U^2) = \prod_{i=1}^{N} \mathcal{N}(u_i|0, \sigma_U^2 \boldsymbol{I}) \tag{2.22}$$

$$p(V|\sigma_V^2) = \prod_{j=1}^{M} \mathcal{N}(v_j|0, \sigma_V^2 \boldsymbol{I}) \tag{2.23}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ stands for the probability density function of Gaussian distribution; and I_{ij} for an indicator function, that equals to one if user i gives a rating for item j, otherwise becomes zero.

Now, the problem of predicting the ratings for potential items becomes how to find the posterior distribution $r(i, j)$ given the distributions of latent variables u_i and v_j. This can be depicted by a plate model shown in Figure 2.6.

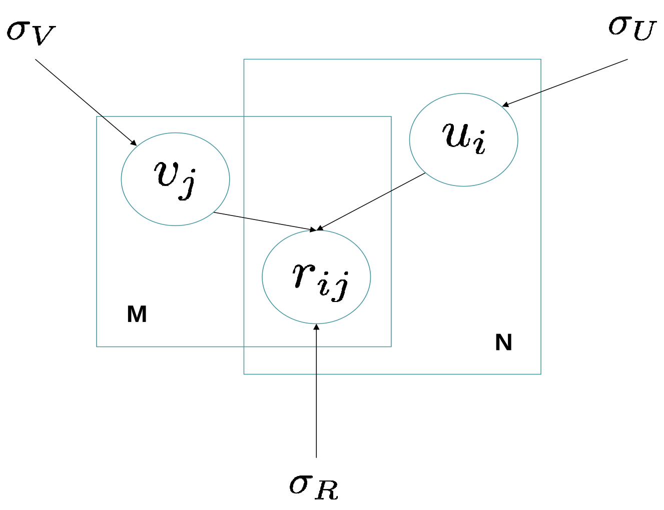


FIGURE 2.6: Graphical model of Probabilistic Matrix Factorization

The log-posteriors over the latent features U and V are given by

$$
\begin{aligned}
& log\ p(U, V|R, \sigma_R^2, \sigma_U^2, \sigma_V^2) \qquad (2.24)\\
&= log\ p(R|U, V, \sigma_R^2) + log\ p(U|\sigma_U^2) + log\ p(V|\sigma_V^2)\\
&= -\frac{1}{2\sigma_R^2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(r_{ij} - u_i^T v_j)^2\\
&- \frac{1}{2\sigma_U^2}\sum_{i=1}^{N} u_i^T u_i - \frac{1}{2\sigma_V^2}\sum_{j=1}^{M} v_j^T v_j\\
&- \frac{1}{2}((\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij})log\ \sigma_R^2 + ND\ log\ \sigma_U^2 + MD\ log\ \sigma_V^2) + C
\end{aligned}
$$

where C is a constant independent of U and V.

Inferring this model is equivalent to minimizing a sum-of-squared-error objective function with quadratic regularization terms as follows,

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(r_{ij} - u_i^T v_j)^2 + \sum_{i=1}^{N}\lambda_U\|u_i\|_F^2 + \sum_{j=1}^{M}\lambda_V\|v_j\|_F^2 \qquad (2.25)$$

where $\lambda_U = \frac{\sigma_R^2}{2\sigma_U^2}$, $\lambda_V = \frac{\sigma_R^2}{2\sigma_V^2}$, and $\|.\|_F^2$ denotes the Frobenius norm.

A local minimum of the above objective function (Equation 2.25) can be learned by executing a gradient descent algorithm on the latent variables u_i and v_j, as described in the following derivatives.

$$\frac{\partial E}{\partial u_i} = \sum_{j=1}^{n} I_{ij}(u_i^T v_j - r_{ij})v_j + \lambda_U u_i \tag{2.26}$$

$$\frac{\partial E}{\partial v_j} = \sum_{i=1}^{m} I_{ij}(u_i^T v_j - r_{ij})u_i + \lambda_V v_j \tag{2.27}$$

Similar to the *PMF*, many successful matrix factorization based recommender systems have been proposed by using probabilistic and Bayesian approaches [97, 120, 126]. In order to incorporate social connections and interactions, matrix factorization based recommender systems are facing two major challenges. The first challenge is to include as much more social information as possible. And the second one is to handle the sensitivity to data noise and outliers. The second issue is mainly caused by the squared error in objective functions (known as L2 LOSS) in matrix factorization models. A number of studies have reported that the L2 LOSS lacks robustness under certain circumstances where severe noise and outliers occur in data matrices, and several solutions to general matrix factorization models are proposed. However, to introduce “robustness” in matrix factorization based social recommendation methods is still a challenging task.

In the next section, we will dig into three interesting topics in the study of recommender systems: (i) familiarity based neighborhood formation, (ii) tie strength measurement, and (iii) matrix factorization based social recommendation. For each topic, we explore several representative methods and show how the limitations or issues are appearing in existing studies.

2.3 Related Topics

This section shortly reviews the work in three interesting topics which are closely related to the research in this thesis. The first topic is regarding neighborhood formation, one of the most core techniques of *CF* methods. The second topic is tie strength measurement. This topic is hardly comprehensively discussed in the mainstream of recommender systems, we will discuss it in detail. The last topic is for matrix factorization based social recommendation, focusing on the incorporation of social information in recommendation process.

2.3.1 Familiarity based Neighborhood Formation

One of the key techniques of *CF* methods is the construction of recommendation neighborhoods, from which the potentially interesting items are selected out. In recent years, certain studies on recommender systems categorized the neighborhoods into two main types: similarity-networks and familiarity-networks [50]. A similarity-network contains those people whose social activities overlap others. By contrast, the familiarity-network of a user includes those people whom the user knows well or with whom the user has a social relationship [50, 108].

The studies by Ruffo et al. [108] and Guy et al. [50] suggest that, for the purpose of seeking recommendations, a user often prefers familiarity-networks consisting of the people with whom the user is already familiar, or from whom the user often receives advice. The experiment by Guy et al. revealed that, for the recommended items, the group of familiarity-networks showed 45.7% interesting versus 37.1% not-interesting. However, the similarity-network group manifested a ratio of 38.7% interesting versus 48.2% not-interesting [50].

The above result coincides with the general knowledge in real life, where people mostly receive recommendations from peers in personal social circles. Figure 2.7 (reproduced from [150]) illustrates two different approaches of neighborhood formation: similarity-networks (left-hand image) and familiarity-networks (right-hand image).

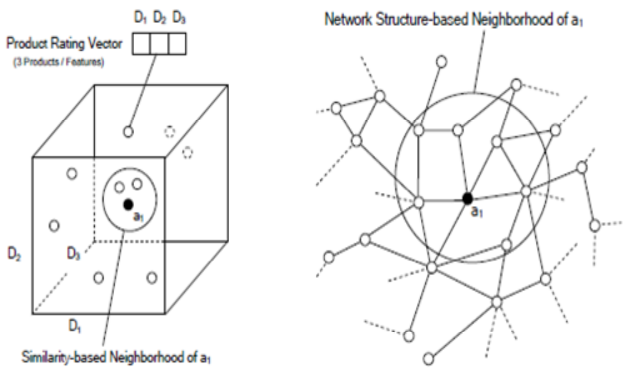


FIGURE 2.7: Neighborhood by similarity- and familiarity-networks

Obtaining useful recommendations through familiarity-networks matches the concept of *Small-World* theory [32, 95, 99, 101] in social science. The *Small-World* theory speculates that, almost any pair of people in the world can be connected to one another by a chain of intermediate acquaintances in a typical chain length of six people [29, 127]. Moreover, an empirical study by Killworth et al. [64] revealed that the average *Small-World* path length was 3.23, based on an analysis of 10,920 shortest path connections between 105 people. This phenomenon is also informally referred to as “six degrees of separation” [99].

Learning from the *Small-World* phenomenon, we believe that it is possible to produce quality recommendations by working through a user’s social circles and allowing every peer on the network actively participate the recommendation process. This is quite different from the conventional neighborhood formation methods, in which recommendation process waits to start until the whole neighborhood gets ready by incessantly collecting a huge amount of data regarding users and items.

The study by Ben-Shimon et al. [16] is a typical example of forming recommendation neighborhoods directly from social networks. The study builds up a neighborhood from an active user’s personal social network using a *Breadth-First Search (BFS)* algorithm [100], then calculates the rankings of media items based on the ratings given by other peers in the neighborhood. The ranking formula penalizes the distances between users. This ranking method may have a strong bias towards a user’s direct friends. As the result, a user has more direct friends would receive higher rankings than other users who have less direct friends. Besides, this ranking method deals with only binary ratings (positive or negative), as such, it will be difficult to work in general situations with numerical rating values.

A number of studies build their recommendation neighborhoods on top of existing peer-to-peer networking topologies [11, 30, 108]. The study by Ruffo, Schifanella and Ghiringello [108] proposed a *preference network*, which was a self-organized and interest-based cluster on top of a *Peer-to-Peer (P2P)* network. To construct such a *preference network*, this study defines a neighborhood by a two-layer structure of social relationships: one is *the contacts* of the user, and the other is *the contacts of the contacts* of the user. Based on these two sets, *the list of friends* was formed for making recommendations.

Similarly, both of the *PREGO* system by Baraglia et al. [11] and *P2Prec* system by Draidi et al. [30] imitate the peer-based recommendation approach in real life. A common approach of these systems is to set up preference information over the items, for example topics-of-interests [30]. Then, each user periodically gossips with other peers to find neighbors having similar interests. Once the neighborhood is stabilized, every peer makes recommendations using general *CF* algorithms.

Although these peer-to-peer systems work in on-line situations, they are still following the conventional neighborhood formation method. As such, the recommendation process cannot start until a great amount of data of users and items have been collected. In addition, these peer-to-peer systems directly utilize existing networking protocols such as Distributed Hash Table (DHT) for communication. That prevents the recommender systems providing flexible communication interfaces for complicated tasks, for instance, incorporating rich social information into recommendation.

For all the aforementioned methods, another important issue is rating-style-adjustment. We often see that some users in one circle are in favor of high ratings for their movies. However, the users in another circle may prefer to vote all movies at low ratings. When a user has a number of peers in different circles, directly aggregating other users' ratings may lose some high quality movies owing to different rating styles. In the case of *PREGO* and *P2Prec* systems, the rankings are prepared via directly *gossiping* with other peers, it would become difficult to adjust the diverse rating styles in different social circles.

Sum the above studies up, there remain a number of issues in familiarity based neighborhoods formation. Most of these issues will be addressed in our proposed Peer-based Collaborative Filtering method in Chapter 3.

2.3.2 Tie Strength Measurement

Social relationship strength is one of the most important research topics in social network analysis, it measures how strong or weak the relationships are among the users in a social network. In the literature, social relationship strength is also referred to *Tie Strength.* A theory of "The Strength of Weak Ties" was initially introduced in [47, 48]. In line with the theory, [85] discussed the quantitative measurement of social relationship strength using multiple indicator techniques.

Previous research of social relationship strength mainly focused on users' face-to-face communication, where one node often connected at most tens of other nodes; and the relationship strengths were mostly of descriptive or binary forms such as being friends or not, strong or weak. However, the situation on the web has been significantly changed. Nowadays, an average user may have several hundreds of connections, and some individuals have much more than usual [129]. In addition, the measurement of tie strength has become sensitive to fine granularity instead of a simple binary form.

The study of Tie Strength emphasizes users' overall attitudes towards other people in a social network. As depicted in Figure 2.8, the tie strength is an abstract and integrated measurement of the weights of combined social attributes and relationships among the users. As illustrated in the figure, the strength between two persons depends on various factors, such as location, education, job, ethnicity, common actions on shopping and tweeting, etc.

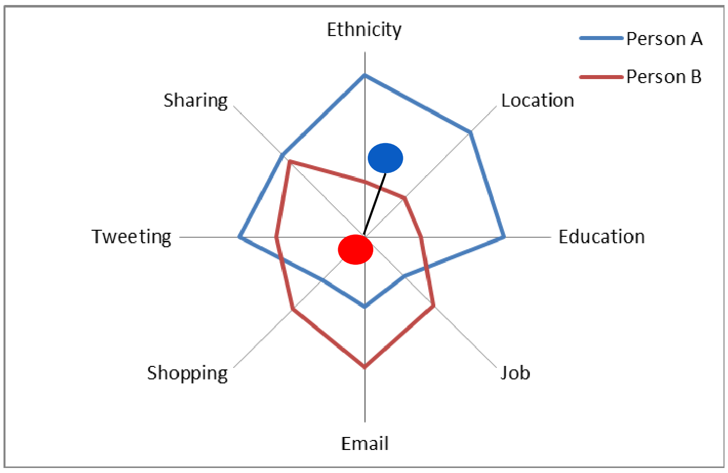


FIGURE 2.8: Integrated measurement of users' overall attitudes

A primary definition of Tie Strength is "a combination of the amount of time, the emotional intensity, the intimacy (mutual confiding), and the reciprocal services which characterize the ties" in the study by Granovetter [47]. This study differentiated "strong ties" from "weak ties". The former meant the close relationships for instance family, trusted friends and certain personalized groups; the latter indicated such a relationship which constituted a "local bridge" to link the others in a social system that are otherwise disconnected. From the perspective of making recommendations, we pay more attention to the strength of strong ties.

Generally speaking, strong ties tend to bond similar people to each other, and these similar people prefer to cluster together such that they are mutually connected [47]. In other words, the studies of strong ties focus more on the similarity or homophily in flat social circles, based on various characteristics and relationships such as age, gender, friendship, work, advice, support, information transfer, kinship [88]. This is especially useful in recommendation applications. Therefore, in this these the strength of strong ties is measured on symmetrical and transitive concepts. For example, users A and B are of ages at 20', while user C is of age 60'. From the homophily principle, users A and B may have stronger tie strength than those of A and C and B and C. Provided that the tie strength between A and B (as well as between B and A) is 0.8, then those of A and C and B and C may be 0.3. Furthermore, because A is a close peer of B (0.8), and B has a looser relation with C (0.3). Consider both the direct and indirect links between A-C and A-B-C, we may assign the tie strength between A and C as 0.54 (0.3 + (0.8 x 0.3)).

In line with Granovetter's study [47], much work has been done to analyze various factors or aspects of tie strength [42, 61, 85]. The work by Marsden and Campbell [85] pioneered the concept of "predictive factors" of tie strength. This study defines two types of variables for measuring tie strength: indicators and predictors. The indicators are observed measures of social network ties, including closeness, duration, frequency, breadth of discussion topics, and confiding. Whereas, the predictors are unobserved aspects related to tie strength, such as kinship, co-worker and neighbor status. Utilizing these variables, tie strength is then defined by a linear function of the predictors, and the predictors can be solved by a set of linear functions associated with indicators. This study concluded that a measure of “closeness” was the best indicator for tie strength.

Similar to Marsden and Campbell’s study [85] but with significant scale up, the study by Gilbert and Karahalios [42] statistically analyzed 74 variables in seven major dimensions, based on a survey over 35 participants with 2,184 rated Facebook friendships. Of the seven dimensions (intensity, intimacy, duration, reciprocal services, structural, emotional support and social distance), this study found that, the intimacy dimension accounted for 32.8% of the predictive capacity for tie strength, the biggest capacity compared to other dimensions. This result coincided with the main findings by Marsden and Campbell [85]. The study by Gilbert and Karahalios [42] also made novel findings which had not been well-understood in prior work, such as the significance of certain predictive variables in structural dimension.

Likewise, the study by Kahanda and Neville [61] used 50 features to predict *link strength*, a synonym of tie strength, among the users in online social networks. These 50 features are categorized into four groups: attribute-based, topological, transactional and network-transactional. The study conducted an investigation over randomly selected Facebook users with 8,766 linked friends. This study applied three supervised learning algorithms to classify the link strength: logistic regression (LR), bagged decision trees (BDT) and naive Bayesian classifiers (NBC) [134]. The findings of the study include two outcomes. First, this study found that the features of network-transactional group had the largest impact on the overall performance of *link strength* measurement, accounting for 12 of top 15 most-impact-variables. Second, it concluded that the bagged decision trees (BDT) performed the best, at overall ROC/AUC results.

In the meantime, many studies have been conducted to apply tie strength to a wide range of application domains. While certain studies focused on “weak ties”, the majority of the studies show more interests in “strong ties”. Intuitively, the stronger the tie connecting two individuals, the more similar their behavior. For instance, the study by Krackhardt [69] reported that strong ties helped organizations to deal with crises. The investigation by Schaefer, Coyne and Lazarus [113] demonstrated that emotional support from strong ties such as trusted friends or family would be important to health and well-being. Moreover, the study by Ding et al. [28] proposed a method to identify strong ties in a criminal network environment.

A common feature of the above studies is that, tie strength is measured directly based on manifest variables, which are either observed or well-known ones. Just recently, certain studies have shown two new trends: (i) exploiting comprehensive information from online social media services, including users’ interaction and demographic data; and (ii) employing latent variable models to develop measuring methods. In particular, the studies by Xiang et al. [135] Zhao et al. [142] and Zhuang et al. [149] attract our great interest.

The study by Xiang et al. [135] performs matrix factorization over users’ both interaction activities and profile similarities. This study develops a probability model based on the assumption that the tie strength is a “hidden effect” of users’ profiles, and is also the “hidden cause” of users’ interactions. In this study, each of the demographic and interaction data is compiled into a single table. Then, the proposed model infers tie strength from the two tables. In fact, users' social attributes and relationships mostly have diverse forms, to pre-compile all information in a single table may risk the loss of some important features. For example, it will be not easy to combine users' education and ethnicity information in a single table.

Based on the similar assumption with [135], the study by Zhuang et al. [149] exploits users' interactions to measure tie strength. This study calculates a number of users' similarities based on common actions or interaction. For instance, it defines the similarities of mutual comments, common groups, mutual friends, etc. The model learns each type of similarity by a single kernel, and then combines the multiple kernels into a single one using a *Kernel Alignment* technique. The combined kernel is then used to measure the tie strengths among the users. This study provides a good approach of capturing users' interactions for measuring tie strength. Unfortunately, it does not explore the way of dealing with users' profile information.

The framework proposed by Zhao et al. [142] predicts relationship strength on various activity fields by calculating a joint distribution of *profile strength* and *interaction strength*. The so-called *interaction strength* was determined by a *relatedness* value of the contents of messages. This is basically a semantic approach, being limited in document-based environments.

In Chapter 4, we will address some of the current issues occurring in online social media environments.

2.3.3 Matrix Factorization based Social Recommendation

Inspired by the success of matrix factorization for general recommender systems, a substantial amount of research has been conducted to incorporate social information into matrix factorization [137]. The matrix factorization based social recommendation methods are called *MFSR* methods in this thesis. All *MFSR* methods commonly assume that, a user's preference is mostly affected by the user's social connections and/or interactions. This assumption coincides with the principle of "homophily in social networks" studied in sociology [88].

The study by McPherson, Smith-Lovin and Cook [88] associated the homophily with three types of characteristics: the demographic, psychological and social. The demographic attributes include age, sex, race/ethnicity, education and religion. The psychological ones contain intelligence, attitudes and aspirations. At last, the social characteristics involve occupation and social class, network positions, and behavior. This homophily principle suggests that people tend to build connections with other people having similar characteristics. Generally speaking, the stronger the relationship, the higher the likelihood that more interactions occur between the people.

From the technology perspective, a common approach of *MFSR* methods is treating social information as "side information" along with the basic rating or preference data. Recent studies of *MFSR* methods can be categorized into three groups. That is, *Social Ensemble* [57, 82, 84, 102, 138], *Collective Matrix Factorization* [35, 78, 83, 118, 119, 139], and *Kernelized Matrix Factorization* [4, 148].

A common rationality held by the studies of *Social Ensemble* group is that the observed ratings of a user should represent the preferences of his/her trust friends. For this reason, the study by Ma, King and Lyu [82] models a conditional distribution over the observed ratings by a linear combination of the favors of all the friends. Going one step further, the study by Jamali and Ester [57] makes use of propagation of trustworthiness. Similar to the above two studies in [57, 82], the study by Yang, Steck and Liu [138] develops a concept of "Trust Circles". This model infers the recommendations by the same way as that of [57], but using only the observed ratings in a particular category with trust friends.

The studies of *Social Ensemble* approach claimed work well in certain circumstances. When design an ensemble method, most studies use linear combination method. In addition, our experiments find that, for instance, the *SR2* by Ma et al. [84] achieves quite good accuracy measures (Root Mean Squared Error (RMSE)), however, obtain low relevance (Precision-versus-Recall Curve (PRC)) in some evaluation cases.

Collective Matrix Factorization (CMF), or *Matrix Co-Factorization* in certain literature, has received great attention in last few years. The application domains of *CMF* include co-clustering [80, 81], image processing [119], signal process [116, 140], algorithm improvement [65, 79], and of course social recommendation [35, 56, 78, 83, 119, 139].

When applying *CMF* to social recommendation, a common approach is to factorize two or more target matrices with shared entities such that the factorization can simultaneously infer latent variables for multiple entities. This is based the idea that the latent variables will be twistingly affected by each other when being simultaneously factorized. In the case of *SoRec* model by Ma et al. [83], this model constructs a specific "social network matrix" for social relationships, of which every element is calculated from the local authority and local hub value between two users. Then, *SoRec* performs collective matrix factorization over the rating matrix and the social matrix. Similarly, the study by Fang and Si [35] also co-factorizes two target matrices, one implicit feedback matrix and one resource information matrix, where the entity *item* participates in both matrices.

Taking *SoRec* as a representative of *CMF* methods, our experiments reveal that, while *SoRec* performs superior for relevance (Precision-versus-Recall Curve (PRC)), it gains slightly low accuracy (Root Mean Squared Error (RMSE)) in most evaluation cases.

In recent years, *Kernel learning* [55, 114] has become increasingly popular, used in various applications such as Bayesian inference, computational biology and link analysis. Some recent studies employed kernel methods in matrix factorization to improve recommendation qualities [2, 4, 148]. The *PMA* model by Agovic, Banerjee and Chatterjee [4] utilizes kernels to capture the covariance for rows and columns of a target matrix, then additively combines the kernels to generate an "approximate matrix" for prediction.

Similarly, the *KPMF* model proposed by Zhou et al. [148] assigns kernels to the priors of latent matrices, and infers the prediction from the matrix product of the latent matrices. Despite the fact that kernel learning effectively captures the similarities between the nodes in a graph, our experiments show that *KPMF* method receives medium scores in both metrics of accuracy and relevance. This may indicate that assigning kernels as a prior to users' latent features may not effectively handle the outliers of rating data in certain circumstances.

In our experiments on *MFSR* methods, we also find that nearly all the existing *MFSR* methods receive good or reasonable values of *Root Mean Squared Error*, but show poor performance in *Precision-versus-Recall* measurement in most evaluation cases. This result suggests that these methods may have good quality of proximity but poor character of relevance [10].

Though there are many factors affecting the poor relevance of *MFSR* methods, we find that a "robust" approach improves the quality of relevance. For this reason, we deploy the L1 LOSS technique proposed by Wang et al. [133]. Owing to this technique, our proposed method achieves higher quality of relevance than all other comparing *MFSR* methods. In Chapter 5 (Recommendation by Incorporating Tie Strength), our new design of a robust matrix factorization method with tie strength incorporated, as well as the detailed comparison between our method and other *MFSR* methods will be elaborately discussed.

2.4 Quick Recap

Through the discussion so far, we understand that although much work has been done for developing various recommender systems, there has been little attention paid to the situations where users are seeking recommendations through their social circles. Furthermore, few studies have been conducted for measuring tie strength from rich social relationships. To address these issues, in this thesis, we propose a peer-based recommendation model incorporated with social relationship strength - *A Peer-based Social Relationship Enhanced Recommendation Model*. Figure 2.9 shows the position of the research in this thesis within the underlying techniques.

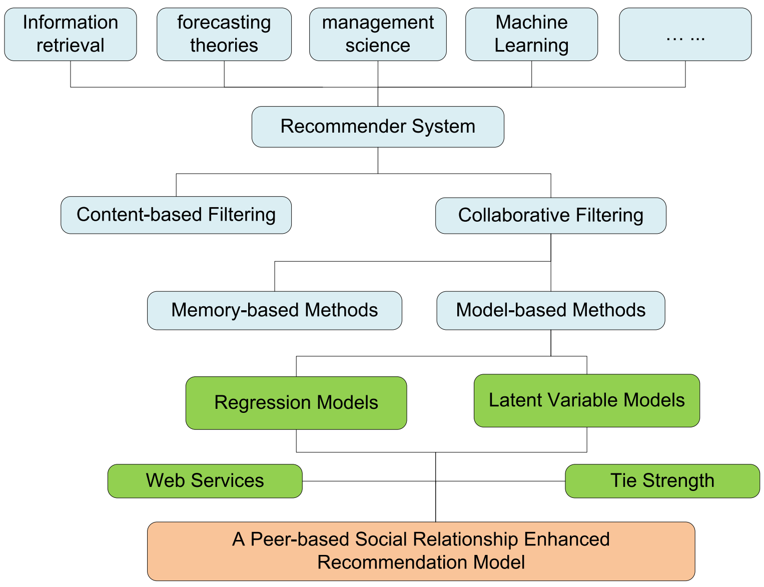


FIGURE 2.9: The position of the research in this thesis

As shown, the research of this thesis is basically built upon two major *CF* models: *Regression Models* and *Latent Variable Models*. Firstly, the proposed model includes a variety of peer-based filtering algorithms of *Regression Models*, which help to form reasonably small-sized recommendation neighborhoods from social circles. Moreover, web services are employed in the proposed model in order to provide comprehensive communication among the peers in a relay process.

Secondly, we believe that incorporating tie strength in recommendation processes will greatly improve recommendation qualities. To this end, we design a measuring method to estimate users' tie strength from various social relationships. Furthermore, we develop a robust matrix factorization based method that incorporates tie strength in recommendation process. Both the tie strength measuring method and robust recommendation method are under the umbrella of *Latent Variables Models*.

The major contributions of this thesis can be summarized as follows:

- We propose a relay-based collaborative filtering method *CoRec*, which emulates the recommendation process in everyday life. The *CoRec* method includes a relay process and a set of relay-based filtering algorithms. This method allows peers in a social network to propagate recommendation requests and responses over the social networks. The proposed *CoRec* method is especially suitable for the situations where people are seeking recommendations through their social circles.

- We design a measuring method to estimate users' tie strength based on various social relationships, named as *KPMCF*. The proposed measuring method uses kernel learning to capture users' interaction relationships, and performs collective matrix factorization to infer users' latent social attitudes. The latent social attitudes are then used to calculate users' tie strengths.

- We develop a robust matrix factorization method *TieRec*. This method employs L1 LOSS to strengthen the robustness of matrix factorization. In the meanwhile, it incorporates users' tie strengths into factorization process.

All the three parts, i.e., *CoRec*, *KPMCF* and *TieRec*, are integrated together to form a *Peer-based Social Relationship Enhanced Recommendation Model.*

- We have conducted comprehensive experiments for every method of the proposed model, using the real datasets from popular social media services. The evaluation results demonstrate that the proposed recommendation model outperforms representative state-of-the-art recommendation methods. Our experiments also prove that the tie strength does play an important role in recommender systems.

In the following three chapters, we will elaborately discuss the three key methods of our proposed model. Chapter 3 presents a peer-based collaborative filtering method *CoRec*; Chapter 4 discusses how to measure users' tie strength using *KPMCF*; and Chapter 5 develops a robust matrix factorization method *TieRec* with tie strength incorporated.

Chapter 3

Peer-based Collaborative Filtering

In this chapter, we propose a peer-based collaborative filtering method, named as *CoRec*, that imitates people's natural information-collecting process in everyday life. This method is especially suitable for the situations where people are seeking recommendations through their social circles. Different from conventional *CF* methods which need to globally collect a great amount of data, the *CoRec* method produces recommendations for every user through local social circles. It works well even if when some users have only a few peers.

This chapter is organized as follows: The following section illustrates the information filtering process in real life. Sections 3.2 and 3.3 elaborate the proposed *CoRec* method. In particular, we discuss the process and system aspects in section 3.2, and describe the filtering algorithms in section 3.3. After the method, section

3.4 and 3.5 shows the experiment settings and evaluation in comparison with other advanced *CF* methods. Lastly, the comparison and summary of *CoRec* are given in section 3.6.

3.1 Motivating Example

Learning from real life, we realize that a social fabric formed from various social circles is potentially able to provide comprehensive information for users' various requests. Consider the scenario of literature search, where a researcher wants to expand his/her collections of a specific topic through personal social circles. The researcher may ask for help from his/her peers who are presumed to hold some relevant articles. People who are asked can then pass along the request to their associates, and the associates may continue to forward the request to other associates. When some peers in the request-chain are able to offer recommendations, they will mostly provide feedback with those articles that are highly rated by themselves or by their peers. This scenario shows a natural way of asking for recommendations in everyday life.

In this thesis, we propose a peer-based filtering method *CoRec*, that mimics the information filtering practice in real life. Figure 3.1 illustrates the concept of the *CoRec* method. In the figure, each user is supposed to maintain a collection of items and associated ratings. For instance, user *Alex* possesses a list of items "*a, c, e*", with associated ratings "*8, 7, 3*"; he also has three friends: *John, Peter* and *Eddy*. *Alex* has common item *c* with *John*, and *e* with *Peter*, but has nothing in common with *Eddy*. We designate an item such as *c* or *e* as *Common Interested Item*, and the friends who commonly have these items as co-peers. Therefore, *Alex* and *Peter* are the co-peers of *Alex*.

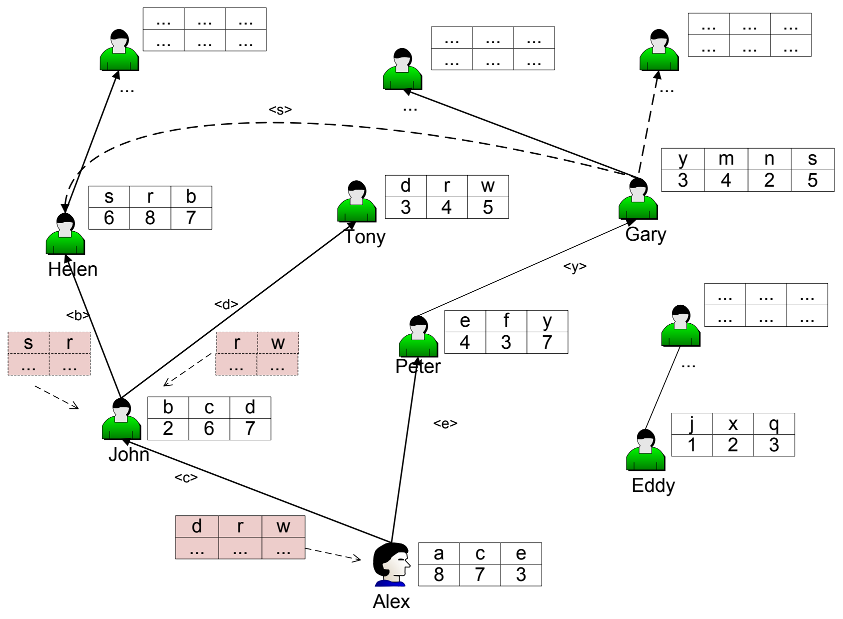


FIGURE 3.1: Concept diagram of peer-based collaborative filtering

When user *Alex* wants to expand his collection, he sends a request to his co-peers *John* and *Peter*; similarly *John* can forward the request to his co-peer *Helen* and *Tony*, and so forth. For a given request, the initial requester and all his successive co-peers will form a request-response structure, and every user in the structure will produce recommendations on his own. Finally, after receiving responses from his co-peers *John* and *Peter*, *Alex* aggregates and filters the recommended items to create a final recommendation list for himself.

3.2 Relay Process via Social Networks

In this section, we present the relay process of *CoRec* which go through a user's social circles. Firstly, we describe the social network settings on which the *CoRec* method is based. Then, depict the process flow of the method. Lastly, we propose a Web Services design to realize the communication among the peers in the relay process.

3.2.1 Social Network Settings

Social networks here refer to the networks of people connected through the internet, helping people make friends and share information. We assume that each user in a social network keeps a list of social relations or "friends", with which a peer to peer network is formed. Every user maintains a collection of items with associated ratings, and possibly with some other preference data, for instance, tags. To summarize the social network settings used in this thesis, we define the following notations:

- A social network is defined as a directed graph $G = (V, E, I)$, where V stands for a set of vertices, representing the users in question; E for the set of the directed edges over V; and I for the set of items rated by some users in V.
- For any user $v \in V$, he/she maintains three sets: items $I(v)$, ratings $R(v)$ and friends $F(v)$. Here $I(v) = \{i_1 \dots i_N\}$ stands for the set of the items rated by the user; $R(v) = \{r_{vi} \mid i \in I(v)\}$ for the set of the associated ratings given

by user v over the items $I(v)$, where r_{vi} is a rating given by v to i. Finally, $F(v) = \{u \mid \langle v, u\rangle \in E\}$ stands for the set of friends of user v.

To conduct recommendation in a social network graph G, each active user v_0 can construct a recommendation neighborhood. This neighborhood is initiated the active user by asking for recommendations of a specific topic. The recommendation request will be sent to some of his/her friends who show common interests with regard to the topic. We call these friends *co-peers*. When a co-peer receives a recommendation request from one of his/her friends, the co-peer may either directly make recommendations for the friend, or forward the request to other social relations and aggregate their recommendations for feedback. The propagation process may keep going until certain constraints are met. We name such a neighborhood as a *Co-Peer Graph (CPG)*, denoted $CPG_{v_0} = (v_0, V', E', I')$ as follows:

- Let v_0 be an active user, having his/her items $I(v_0)$, ratings $R(v_0)$ and friends $F(v_0)$. The active user initiates a recommendation request to his/her friends.
- For two users $u, v \in V$, suppose that u is a friend of v. Let $I_{(u,v)} = I(u) \cap I(v)$, refer to the *Common-Interested-Items (CII)* between u and v. Users u and v are called *co-peers* if $I_{(u,v)} \neq \emptyset$. That is, there is at least one item in which both u and v are interested.
- For any user $v \in V$, those co-peers that forward a request to v are called the inbound co-peers of v, and consequently the directed edges from the inbound co-peers to v are added to *CPG*. These inbound co-peers are denoted as $IN(v) = \{w | v \in F(w), I_{(w,v)} \neq \emptyset\}$. For the active user v_0, we set $IN(v_0) = \emptyset$.

- Those co-peers to whom user v forwards a request are called the outbound co-peers of v, denoted as $OT(v)$, $OT(v) = \{u \mid u \in F(v), I_{(v,u)} \neq \emptyset\}$. All the directed edges from v to the outbound co-peers are also added to CPG. Furthermore, the set of the outbound co-peers who are especially interested in a particular item i is represented by $OT_i(v)$.

  When make actual recommendation in social circles, provided that user A asks user B for recommending books, user B may forward the request to user C and D, but would not forward back to user A. For this purpose, we request that $OT(v) \cap IN(v) = \emptyset$ for a given user v.

- From the stand point of the active user v_0, a *relay depth level* is defined as the path length by which a recommendation request travels from v_0 to a certain outbound co-peer. Therefore, all the outbound co-peers of v_0 are named as the *1*-level co-peers, expressed as V^1. Here "k-level" stands for "the level of relay depth k".

Now, the construction of the *Co-Peer Graph* of user v_0 or CPG_{v_0} starts from the active user v_0 and proceeds to its *1*-level co-peers in a breadth-first fashion.

$$Let\ V^1 = \{v|v \in OT(v_0)\}$$

$$E^1 = \{\langle v_0, v\rangle | v \in V^1\}$$

$$I^1 = \{i|i \in I(v),\ v \in V^1\}$$

Subsequently, all the *2*-level co-peers are those users who are the outbound co-peers of the *1*-level peers, represented by V^2.

$$\begin{aligned}
&Let\ V^2 = \{v|v \in \cup_{u\in V^1} OT(u)\}\\
&E^2 = \{\langle u, v\rangle|u \in V^1\ and\ v \in V^2\}\\
&I^2 = \{i|i \in I(v),\ v \in V^2\}
\end{aligned}$$

In general, all the k-level co-peers for $k > 2$ are those users who are the outbound co-peers of *k-1*-level ones, represented by V^k.

$$\begin{aligned}
&Let\ V^k = \{v|v \in \cup_{u\in V^{k-1}} OT(u)\}\\
&E^k = \{\langle u, v\rangle|u \in V^{k-1}\ and\ v \in V^k\}\\
&I^k = \{i|i \in I(v),\ v \in V^k\}
\end{aligned}$$

When the construction process terminates, the *Co-Peer Graph* of user v_0 is obtained as follows:

$$\begin{aligned}
&CPG_{v_0} = (v_0, V^{'}, E^{'}, I^{'}) \qquad (3.1)\\
&where\ V^{'} = \bigcup_k V^k\ E^{'} = \bigcup_k E^k\ I^{'} = \bigcup_k I^k
\end{aligned}$$

The above definition of *CPG* represents the situation in online environments where users propagate information through their personal social networks. It is worth mentioning that a *CPG* is not simply a duplicate of a personal social network, although it is a subset of the graph representing the social network rooted from an active user. In particular, the peers in a *CPG* are the active user's direct or indirect social relations, who share common interests with each other. The edges of the *CPG* are dynamically formed during the recommendation process, based on the active user's preference data, as well as the decisions made by the co-peers.

The *CPG* has a distinct advantage of constructing a more effective recommendation neighborhood than those formed in conventional *CF* methods. According to the definition of *CPG*, the peers in the neighborhood formed by *CPG* may have no directly co-rated items with an active user, or are not the direct friends of the active user. This increases the opportunities of gathering more co-peers and more potential items, and as such, to ensure that all users can obtain quality recommendations even if some users may have very few co-peers. We strongly believe that the concept of *CPG* does match the general practice of making recommendation in real life.

We notice that the above definition of *CPG* may implicate some potential issues as those occurring in online circumstances. For instance, a user might receive repeated requests in a single request-response chain, or even a loop of requests and responses. Also a recommendation request might be repeatedly forwarded over a dynamically growing network without proper ending. These issues can be resolved by using certain control features deployed in relay process flow, such as *maximum-relay-depth*, *due-date-time*, and a *cache-of-executed-requests*.

Based on the above social network settings, in the next subsection we define the relay process flow of the *CoRec* method.

3.2.2 Sequence Flow of Relay Process

In connection with the multi-level neighborhood *CPG* form from social circles, we design a relay process for making recommendation. Figure 3.2 shows the sequence diagram of the relay process, consisting of five important methods: ***initiate***, ***relay***, ***receive***, ***wait***, and ***recommend***.

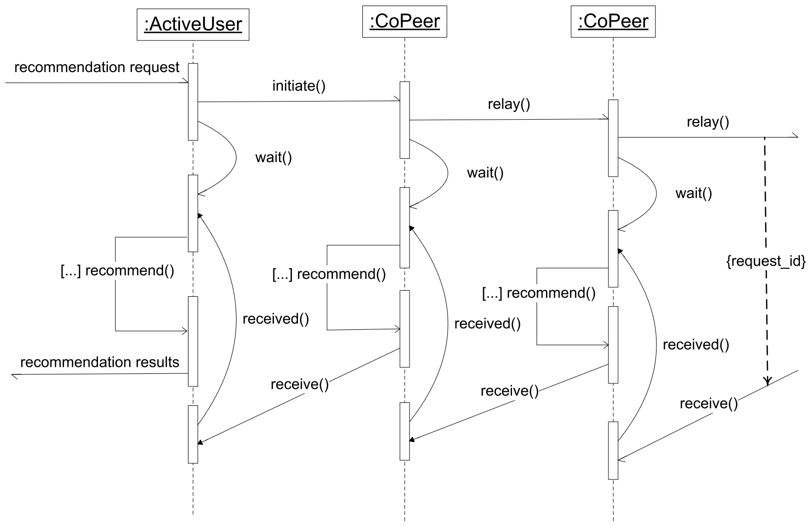


FIGURE 3.2: The sequence diagram of relay process

- ***initiate***: To initiate a recommendation request, an active user (*ActiveUser*) sends an *initial request* to his/her outbound co-peers (*CoPeer*) by calling the ***initiate*** method. In the meantime, the user forks a parallel process ***wait*** to control the relay process. The *initial request* consists of the active user's preference data including his/her ratings $R(v_0)$. The *initial request* also includes user-defined constraints such as *maximum-relay-depth* and *due-date-time* which are used to control the relay process.

- ***relay***: When an outbound co-peer (*CoPeer*) receives a request from his/her inbound co-peer, he/she calls ***relay*** method to send a *consequent request* to his/her outbound co-peers (*CoPeer*), and also creates a parallel process ***wait*** in the meantime.

  A *consequent request* has two parts. One part is the *initial request* issued by the active user, and the other is the preference data of the inbound co-peer calling ***relay*** method. When a user receives a *consequent request*, the user forwards only the *initial request* part to his/her outbound co-peers. By doing so, only the active user's preference data is forwarded throughout the whole *CPG*.

- ***receive***: While both the ***initiate*** and ***relay*** methods serve the function of sending requests, the ***receive*** method collects responses from the outbound co-peers at the next relay depth level. In the ***receive*** method the co-peer keeps a *cache-of-executed-requests* in order to avoid repeated requests or request-response loops. Whenever a response arrives, the ***receive*** method reports to the ***wait*** process.

- ***wait***: The ***wait*** method executes control functionality to ensure a successful process cycle. When some control constraints (*maximum-relay-depth*, *due-date-time*, *cache-of-executed-requests*, etc) are reached, for instance if a user is at *maximum-relay-depth*, the ***wait*** method will directly invoke the ***recommend*** method, instead of waiting for responses from outbound co-peers.

- ***recommend***: This ***recommend*** method represents the procedure of predicting items for inbound co-peers. In general, nearly all the users in a *CPG* perform three functions:

(a) *Prediction*: to predict the ratings of the potentially interesting items, based on the preferences of all the related users, including the active user, inbound co-peer, and the peer who is making predictions;

(b) *Aggregation*: to consolidate the ratings of those items which are recommended by a number of outbound co-peers;

(c) *Filtering*: to select the predicted items according to certain decision rules such as a threshold of the ratings for qualified items.

After *Filtering*, the ***recommend*** method sends the selected items to the corresponding inbound co-peer by invoking the ***receive*** method. In the case of active user, the ***recommend*** method will produce a final recommendation list.

We will elaborate the three functions of ***recommend*** method in section 3.3. Before this, we present a realization of the relay process using Web Services in next subsection.

3.2.3 Relay via Web Services

In order to implement the communication among the peers in the above relay process, we design a *RESTful Web Services* in *CoRec*. This design equips every user with a dedicated personal-hosting RESTful web services engine, that will perform the peer-based filtering algorithms via a gossip-styled communication. To demonstrate the feasibility of the design, we have developed a simulation system on the top of *tornado*, a Python web framework and asynchronous networking library [1]. In this section, we highlight the workflow of the web services.

Web Services (WS) has been widely adopted as one of the promising integration technologies because of its advanced features of loose-coupling and interoperability. Conceptually, a web service is a software component provided through one or more network-accessible endpoints, by which service provider and service consumer exchange requests and responses in the form of self-contained documents [46].

The core technology of *Web Services (WS)* is based on *Simple Object Access Protocol (SOAP)* and *Web Services Description Language (WSDL)*, which we call *SOAP-WS* [6]. While this *SOAP-WS* effectively delivers interoperability among heterogeneous systems, *REpresentational State Transfer Web Services* (hereafter call *REST-WS*) [37] explores great potential for communication in a wide range of computing facilities, as long as the facilities satisfy existing industrial standards, such as *Hypertext Transfer Protocol (HTTP)* and *Extensible Markup Language (XML)*. Consider the situation of peer-based relay process, each co-peer works in his/her own environment. For this reason, we select *REST-WS* as the communication platform of the relay process.

The peer-based workflow of *REST-WS* is shown in Figure 3.3, involving three important parts: *Tornado*, a *REST* server that provides peers with an http-server and http-client for communication; *Recomm-Engine*, the recommendation engine

[1]http://ww.tornadoweb.org/

responsible for performing ***recommend*** method; and *MongoDB*, the database for storing the resources.

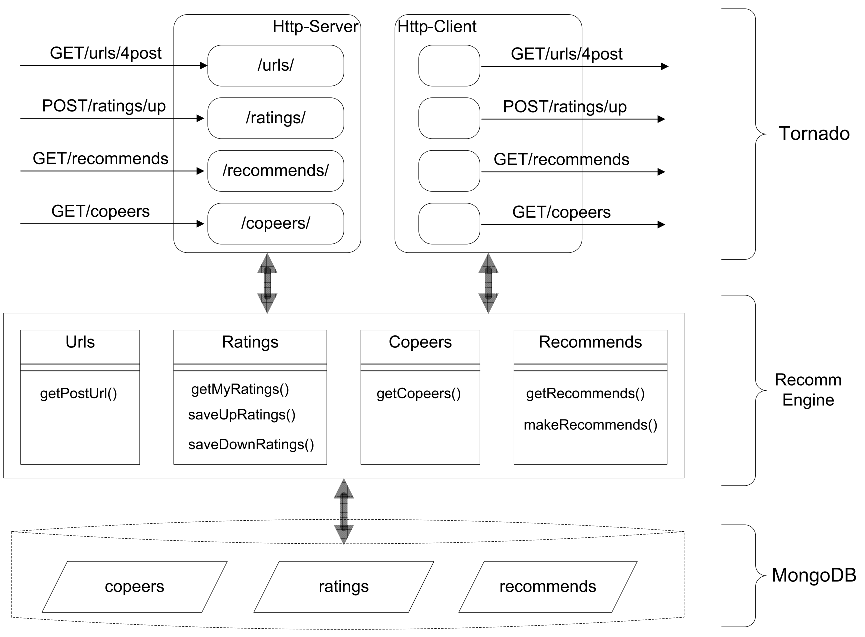


FIGURE 3.3: The *REST-WS* workflow in a simulation system

There are four key endpoints in the system used to propagate the ratings and recommendations: *GET/urls/4post*, *POST/ratings/up*, *GET/recommends*, and *GET/copeers*. Every peer can either receive or request the invocations of these endpoints as an Http-server or an Http-client. The workflow of these endpoints is described as follows:

- *GET/urls/4post*: This is the initial message flow between a user and one of his/her social relations when the user invites the peer to join a recommendation process. If the peer is able to do so, then the user will receive a positive response, with a hyperlink for posting the users' rating data, otherwise, the user will receive a negative answer with a NULL for the hyperlink.

- *POST/ratings/up*: The user will post his/her ratings to the peer. The peer will check the incoming information, and decide whether they become co-peers or not. In the case of a positive answer, the reply from the peer to the user will include a hyperlink for getting recommendations.

- *GET/recommends/*: This endpoint represents an asynchronous communication, which requires a certain amount of time for the co-peer to reply to the recommendation requests and responses. Usually, a co-peer receiving this call will initiate further *REST* workflow with his/her social relations via an Http-client.

  The response to this endpoint includes recommended items with predicted ratings, as well as the hyperlinks to the other peers who made the recommendations.

- *GET/copeers/*: This endpoint is used for obtaining further information from the co-peers who made certain recommendations in the relay process.

In this section, we have discussed the process and system aspects of the *CoRec* method. The core filtering algorithms in the recommendation engine will be discussed in the next section.

3.3 Peer-based Filtering Algorithms

This section contributes to the details of the three key functions of the ***recommend*** method in *CoRec*: *Prediction*,*Aggregation* and *Filtering*. For the purposes of description, we assume three users involve in a recommendation process: w, v and u. Suppose that user w is an inbound co-peer of v, and user u an outbound co-peer of v; and consequently, user v is an outbound co-peer of w, and meanwhile an inbound co-peer of u. Recall the motivating example, we can image that user *John* can be v between Alex (w) and *Helen* (u), where Alex sends a recommendation request to *John*, and *John* forwards the request to *Helen*. From the point view of sending back recommendation response, *Helen* sends a response to *John*, and subsequently *John* sends a response to *Alex*.

3.3.1 Prediction

The purpose of *CoRec* is to recommend new items for active users. To this end, the *Prediction* function needs to estimate the scores of the items which are not experienced by active users but highly rated by other users. Then, the items which are given higher scores (hereafter called as "potential items") are selected for further filtering. In a relay-based process, such potential items are limited to those items which should be highly recommended by any of the co-peers of an initial active user but are not rated by the active user. As user v has an outbound co-peer u, user v obtains extra items $I^r(v)$ from u, in addition to his/her own rated items $I(v)$. So, the potential items held by user v will become $I^p(v) = (I(v) + I^r(v)) \setminus I(v_0)$.

In the relay process, when user v predicts the items through user w, user v needs to take w's preference into account. Using a linear model of $f(x) = x + b$, the best predictions for w from v can be inferred by minimizing the *Mean Square Error*

$E = \sum_{j \in I_{(v,w)}} (r_{vi} + b - r_{wi})^2$. That means, the constant b must be the average difference between the ratings of the *Common-Interested-Items* of users v and w. For a potential item $i \in I^p(v)$, $\widehat{r}^{v}_{wi}$ denotes the rating given by *CoRec* system for user w from user v on item i, refer to "predicted rating". Whereas, the rating r_{vi} and r_{wi} are the ratings given by users w and v for item i before the system running, refer to "original rating".

The concept of the above *Minimizing Mean Square Error* is basically working for those items which are not the *Common-Interested-Items* between users v and w. However, owing to the nature of relay-based process, a potential item for an active user may be also the *Common-Interested-Items* between two co-peers (none of them is active user). We may also want to recommend such items if they are highly-appreciated by these co-peers. To not miss these items for the active user, we develop two prediction formulas for the corresponding situations.

The following Equation 3.2 is for those items which are not the *Common-Interested-Items* between users v and w:

$$\begin{aligned} \widehat{r}^{v}_{wi} &= r_{vi} + \frac{1}{|I_{(v,w)}|} \sum_{j \in I_{(v,w)}} (r_{wj} - r_{vi}) \\ &= \frac{1}{|I_{(v,w)}|} \sum_{j \in I_{(v,w)}} (r^*_{vi} - r^*_{vj}) + \overline{r}^{v}_{w} \end{aligned} \tag{3.2}$$

where $\overline{r}^{v}_{w} = \frac{1}{|I_{(v,w)}|} \sum_{j \in I_{(v,w)}} r_{wj}$, standing for the average rating of the *Common-Interested-Items* between user v and w but given by w; r^*_{vi} and r^*_{vj} represent either an original rating or an aggregated rating held by v; and r_{wj} is the original rating given by w for item j.

And, the following prediction formula (Equation 3.3) is especially used for the potential items for the active user, but these items are also the *Common-Interested-Items* between users v and w:

$$\widehat{r}^{v}_{wi} = (r^{*}_{vi} - \overline{r}_v) + \overline{r}_w \tag{3.3}$$

$$where\ i \in I^{p}(v) \cap I_{(v,w)}$$

where $\overline{r}_v$ and $\overline{r}_w$ are the corresponding average ratings of users v and w respectively.

To illustrate the *Prediction* function (Equation 3.2), let's pick up a scenario in motivating example where *Tony* is making predictions for *John*. *Tony* rates three items (d, r, w) on $(3, 4, 5)$ respectively. Of these items, items d is a *Common-Interested-Item*, which is rated by *John* at 7. So, the average rating of the *Common-Interested-Items* between *Tony* and *John* is $(7 + 3)/2 = 5$.

If *Tony* wants to predict the ratings of items r and w for *John*, he firstly calculates the derivations of item d with respect to r and w. The derivations are $(item\ r) : 4-3 = 1$ and $(item\ w) : 5-3 = 2$ respectively. And then, the derivations are added on top of the average rating of *Common-Interested-Items* for prediction. That is, *Tony* recommends the items r and w for *John* at ratings of $(item\ r) : 1 + 5 = 6$ and $(item\ w) : 2 + 5 = 7$ respectively.

3.3.2 Aggregation

During the course of peer-based process, a user in the process may receive a same item from multiple outbound co-peers, each co-peers sends its predicted rating over the item. In order to make prediction for an inbound co-peer of the user, the user needs to consolidate all the ratings for such items. Consider a scenario of user *John*, he obtains a recommended item r from both *Helen* and *Tony*. In another word, one item r now has two ratings. In that case, *John* needs to aggregate the ratings of the item r in order to make predictions for his inbound co-peer *Alex*.

The *Aggregation* function is essentially the problem of *Preference Aggregation* [121] in the domain of *Social Choice* [8], where a vast number of aggregation algorithms have been studied [24]. In this these we focus on the mechanism of a peer-based collaborative filtering method with relay process. All the main algorithms should be adaptive for various applications. Therefore, instead of using some existing aggregation algorithms, we prefer to design our own that is explicitly expressed by two most direct and relevant variables derived in the *Prediction* function: the predicated ratings and the number of the co-peers who contribute the potential items.

Let $I^r(v)$ be the items recommended by all the outbound co-peers of v after completing a filtering function (to be discussed in section 3.3.3). For an item i in $I^r(v)$, it might be recommended by multiple outbound co-peers. To retain the information, we compute a vote-count $C_u(i)$ for the accumulated number of users who have already recommended item i through to user u. If item i is only rated by a single user u, we define $C_u(i) = 1$.

In the following discussion, r_{vi} stands for the rating given by user v for item i; $\widetilde{r}_{vi}$ for the aggregated rating by user v for item i; and $\widehat{r}^u_{vi}$ for the predicted rating for user v for item i recommended by a peer user u (to be discussed in the next subsection). The aggregated rating can be calculated by the following Equation 3.4:

$$\widetilde{r}_{vi} = \frac{1}{C_v(i)} \sum_{u \in \chi_i} \widehat{r}^{u}_{vi} * C_u(i) \tag{3.4}$$

$$where\ C_v(i) = \sum_{u \in \chi_i} C_u(i)$$

where χ_i represents $OT_i(v) \cup \{v\}$. Along with the aggregated rating, the vote-count of item i through to v is also aggregated.

The *Aggregation* function can be explained by continuing the same scenario in the *Prediction* function, where *Tony* recommends items r and w for *John* at ratings of 6 and 7 respectively. Using the same *Prediction* function, *Helen* recommends the items r and s for *John* at ratings of 4 and 6 respectively. Of the three items w, r and s, item r is recommended by both *Tony* and *Helen*, while w is recommended only by *Tony*, and s only by *Helen*. According the Equation 3.4, *John* aggregates the three items by (i) $item\ r : (6+6)/2 = 6$, (ii) $item\ w : 7/1 = 7$, and (iii) $item s : 4/1 = 4$. These ratings of w, r and s will become the source to be used by *John* in the *Prediction* function for his inbound co-peer *Alex*.

3.3.3 Filtering

In order to select highly appreciated items for inbound co-peers, all outbound co-peers perform a *Filtering* function after *Aggregation*. The filtering rules should also be adaptive, for instance in favor of some users' rating styles. On principle, so-called "highly appreciated items" of a user are those which are given higher ratings by the user. However, in the circumstance of relay-based recommendation, users in different circles usually have different ratings styles even in a single scale for instance from 1 to 10. Some users may prefer to give preferred items with higher rates, say 8 or 9; while some others like to rate items with lower scores, for instance 3 or 4. To ensure that no discrimination is taking against the potential items owing to diverse rating styles, we adopt a "top-N nearest items" policy in the filtering process. That is, we will select only top-N items which have close rating style to that of every corresponding inbound co-peer. By doing so, a potential item would have a relatively higher rate than other items for every co-peer.

Usually a user's rating style is basically represented by the average rating value of the user. Therefore, when user v selects items for user w, the filtering function can be described by the following Equation 3.5:

$$I_v^f(w) = \{i | i \in I^p(v) \ and \ |\widehat{r}_{wi}^v - \overline{r}_w^v| < \alpha\} \tag{3.5}$$

where $I_v^f(w)$ represents the selected "nearest top-N" items recommended for user w by user v; other notations have the same meaning as in Equation 3.2. In the formula α is a parameter affecting recommendation results. A smaller α may produce recommendations having more similar rating style but less relevant items to active users, whereas a larger α may bring in more relevant items but with less similar rating style to active users. We will discuss this in section 3.5.

After filtering, the selected items are sent to the previous inbound co-peer for aggregation purposes (section 3.3.2), where $I^r(v)$ are the items recommended by all the outbound co-peers for user v. In other words, $I^r(v) = \cup_{u \in OT(v)} I_u^f(v)$.

By iteratively applying the aforementioned Equations 3.4, 3.2 and 3.5, an active user v_0 obtains a set of recommended items with predicted ratings. From the items and ratings, the active user will perform further aggregation and filtering, and produce his/her final recommendation list $\langle I^f(v_0),\ R^f(v_0)\rangle$, where $I^f(v_0)$ is derived from the filtering function over the aggregated ratings, and $R^f(v_0) = \{\widetilde{r}_{v_0 i} | i \in I^f(v_0)\}$, and $\widetilde{r}_{v_0 i}$ is the aggregated rating after the filtering process.

Most of the users in a *CPG* execute the above three functions, excepting two cases. The first exception occurs when some users hit the control constraints. For example, when a user knows that he/she has reached *maximum-relay-depth*, then, this user will directly make predictions on his/her own items without waiting for the recommendations from outbound co-peers.

Another case is for active users. Because active users have no inbound co-peers, they do need to perform *Prediction* function. But all active users need to perform *Filtering* function in order to produce high quality recommendations.

The activity diagram in Figure 3.4 illustrates how an item 10095 in the *Last.fm* dataset is recommended following the peer-based filtering algorithms. The item 10095 is rated by user 7118 and user 4289, as well as the active user 63. If the item is hidden from the active user, the users 7118 and 4289 will recommend the item to user 63 through relevant co-peers.

In the last two sections, we present the details of the proposed *CoRec* method, including peer-based relay process and peer-based filtering algorithms. In the

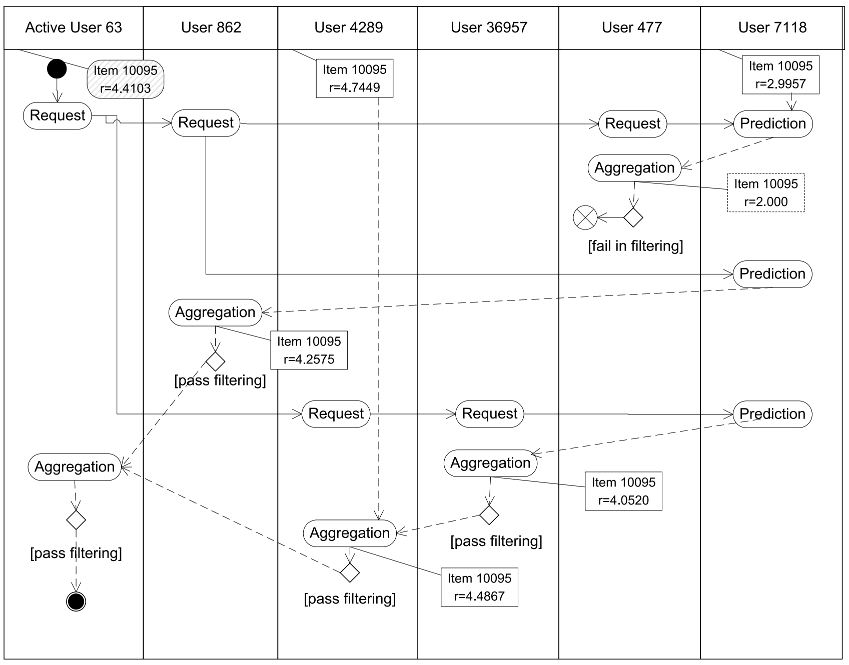


Figure 3.4: An example of peer-based filtering process

following two sections, we will show the experiments of proposed method. In particular, Section 3.4 describes the settings of the experiments, and Section 3.5 discusses the evaluation results in comparison with other advanced *CF* methods.

3.4 Experiment Settings

In this section, we give the experiment settings for our studies in this thesis, including the datasets for experiments, the algorithms for comparison and the metrics being evaluated. A part of this section will be repeated in chapters 4 and 5 as well.

3.4.1 Dataset Description

In the study of the *CoRec* method, we have conducted comprehensive experiments over two real datasets from actual social media services *Last.fm* [2] and *Epinions.com* [3]. The original datasets are provided by the studies of [66] (*Last.fm* dataset) and [86] (*Epinions.com* dataset) respectively.

Table 3.1 highlights the datasets from *Last.fm* and *Epinions.com. Last.fm* allows users to create profiles and augment them with the music tracks that they listen to. Whereas, *Epinions.com* encourages users to contribute reviews on a variety of products. The main tasks of the experiments are to recommend the potential music or products for users.

Table 3.1: *Last.fm* and *Epinions.com* datasets

Source	user	items	ratings	items/user	friends/user	tags
Last.fm	3,148	30,520	802,963	222	3.5	8,296
Epinions.com	40,163	139,738	664,823	16.5	14.3	NA

As shown in Table 3.1, the *Last.fm* dataset comprises about 3,000 users, 30,000 items and 800,000 ratings. Each user in this dataset on average rates 200 and more items and has 3.5 friends. This dataset also includes more than 8,000 tags. On average, each user assigns 21 tags and each tag is attached to 14 items. The *Last.fm* dataset includes a "play-count" of the music tracks to which users have listened. The values of 'play-count' range from 1 to 7,939. To use the "play-count" values as rating scores in the experiments, we apply a logarithmic function to these play-count values.

[2]http://www.last.fm
[3]http://www.epinions.com/

For the dataset of *Epinions.com*, the numbers of users and items in this dataset are about 10 and 5 times larger respectively than those of *Last.fm*. This dataset has about 665,000 ratings, which are given by integer values ranging from 1 to 5. Any pair of users who hold a trust value of 1 are treated as friends in the experiments. In contrast to the *Last.fm* dataset, each user in *Epinions.com* dataset on average has a smaller number of rating items (16.5) but more friends (14.3) (whereas 200 items and 3.5 friends for each user in the *Last.fm* dataset).

3.4.2 Methods for Comparison

Using the foregoing datasets, we have compared the proposed *CoRec* method with certain popular collaborative filtering and state-of-the-art recommendation methods. Of the general collaborative filtering models, we select *Pearson CF (Pearson)* and *Slope One*, two typical memory-based and model-based *CF* methods respectively.

- *Pearson CF* is one of the most well-known memory-based CF algorithms [106]. A major appeal of *Pearson CF* is its simplicity such that the *Pearson CF* is always used as a building block in many other recommender systems. The following Equation 3.6 is the formula of *Pearson CF* used in the experiments.

$$\widehat{r}_{vi} = \overline{r}_v + \frac{\sum_{u \in U}(r_{ui} - \overline{r}_u)w_{vu}}{\sum_{u \in U} w_{vu}} \tag{3.6}$$

where $\widehat{r}_{vi}$ stands for the predicted rating on item i for user v; while r_{ui} represents the rating on item i given by user u; $\overline{r}_v$ and $\overline{r}_u$ represent the mean ratings of users v and u respectively; and $u \in U$ stands for the neighborhood of user v.

- *Slope One* is one of the popular model-based CF algorithms [76], which includes a family of algorithms. We use *Weighted Slope One Scheme* (Equation 3.7) in the experiments as follows:

$$\widehat{r}_{uj}^{ws1} = \frac{\sum_{i \in T^j(u)} (dev_{ji} + r_{ui}) C_{ji}}{\sum_{i \in T^j(u)} C_{ji}} \tag{3.7}$$
$$where\ dev_{ji} = \sum_{r_u \in S_{ji}(\chi)} \frac{r_{uj} - r_{ui}}{|S_{ji}(\chi)|}$$

where $T^j(u) = T(u) - \{j\}$, the set of the items rated by the user u except item j, $C_{ji} = |S_{ji}(\chi)|$.

3.4.3 Evaluation Metrics

In the experiments and evaluations, we mainly employ three types of metrics: *Precision-versus-Recall Curve (PRC)*, *Mean Absolute Error and Root Mean Squared Error (MAE and RMSE)*. Each is outlined as follows.

PRC

Precision-versus-Recall Curve (PRC) is a widely accepted metric to evaluate the performance of information retrieval algorithms. Here *Precision* is defined as the fraction of the retrieved relevant items in the answer set, whereas *Recall* is the fraction of the retrieved relevant items of the whole set of the relevant items [10]. In the experiments, the "relevant items" of an active user are defined as those items rated by the user.

Furthermore, we measure *Average PRC* of all active users in all experiments. In the following discussion, the term *PRC* refers to the *Average PRC*. In the following Equation 3.8, where A stands for the set of *retrieved items*, R for the whole *relevant items*, and R_a for the intersection of A and R (*retrieved relevant items*). Furthermore, $P_p^u(r)$ is the precision at recall level r for the p-th experiments for user u; N_p and N_u are the numbers of the experiments and the users respectively. As such, $\overline{P}(r)$ represents the *Average PRC* value at recall level r in all N_p experiments and for all N_u users.

$$
\begin{aligned}
&\mathit{Precision} = \frac{|R_a|}{|A|} \quad \mathit{Recall} = \frac{|R_a|}{|R|} \\
&\mathit{and\ Average\ PRC}: \ \overline{P}(r) = \sum_{u=1}^{N_u} \sum_{p=1}^{N_p} \frac{P_p^u(r)}{N_p}
\end{aligned}
\tag{3.8}
$$

Figure 3.5 shows an example of a *PRC* Curve, with the *Precision* as the Y-coordinate and *Recall* as the X-coordinate. If we exemplify with an analogy, namely the *PRC* plot with web suffering, the curve represents the relevant items appearing on the retrieved web pages. On one hand, the lower the *Recall* value (start from 0%), the higher likelihood the retrieved relevant items appear on earlier pages. On the other hand, the higher the *Precision* value, the more retrieved relevant items on the pages.

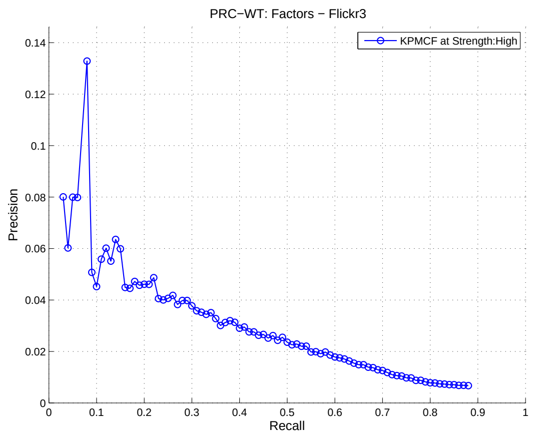


FIGURE 3.5: A sample PRC curve

Mean Absolute Error - MAE

While the *Precision-versus-Recall (PRC)* curve is used to measure the precision-recall performance of various algorithms, *Mean Absolute Error (MAE)* is commonly adopted in the collaborative filtering community to evaluate the accuracy of algorithms [54]. MAE calculates the errors between the predicted ratings generated by recommendation methods and the ratings made by users. In the experiments, we use *Normalized Mean Absolute Error (NMAE)* described in the following Equation 3.9:

$$NMAE = \frac{1}{|V|} \sum_{v \in V} \frac{MAE_v}{r_v^{max} - r_v^{min}} \tag{3.9}$$

$$where\ MAE_v = \frac{1}{|T(v)|} \sum_{j \in T(v)} |\widehat{r}_{vj} - r_{vj}|$$

where V stands for all the users in the test dataset, r_v^{max} and r_v^{min} for the minimum and maximum values of the original ratings given by each user, $T(v)$ for those recommended items which are originally rated by the user, $\widehat{r}_{vj}$ for the predicted rating given by a recommendation method for user v over item j, and r_{vj} for the original rating made by user v on item j.

Based on the above settings, we conduct experiments to evaluate the performance of the proposed *CoRec* method. The following section gives the details of the evaluation.

3.5 Evaluation

In the experiments, we compared the proposed model *CoRec* with the baseline algorithms: *Pearson CF (Pearson)*, *weighted Slope One (wS1)* on the same datasets from *Last.fm* and *Epinions.com*. In particular, we performed experiments on 829 users of the *Last.fm* dataset, where each user had two to eight co-peers; and 1,360 users of the *Epinions.com* dataset with five to 25 co-peers. Every user is treated as an active user once engaged in the experiments. Using the average values of the recommendations for these users, we then calculated the evaluation metrics described in section 3.4.3.

3.5.1 Parameter Setting

In the experiments, we have tried the relay depth levels at two, three, four and five. The recommendation results shown that the experiment of deploying relay depth at three achieved the best performance with regards to the above metrics. When using filtering formula (Equation 3.5), we picked up at most one thousand items every time to form a *nearest top-N* list.

The parameter α in Equation 3.5 affects the balance between the rating styles and relevant items of recommendations. Our experiments show that, when the α was assigned a smaller value, for instance 0.5, the recommendation results showed a quite close rating style to the active user but with a relatively less relevant items. In contrast, a larger value of α, for example 1.0, often brought more relevant items in the final recommendations but with less similar rating styles to active users. To achieve a reasonable balance, we assigned 0.65 to α for both datasets in the experiments.

3.5.2 Precision and Accuracy

Figures 3.6 and 3.7 show all the *PRC* curves of the evaluation algorithms over *Last.fm* and *Epinions.com* datasets. As a whole, the *PRC* of *CoRec* outperforms the other three methods in most of the recall levels. In particular, for the *Last.fm* dataset (Figure 3.6), the *PRC curve* of CoRec starts from a high precision of 0.11 at the 10% recall level, while both *wS1* (*weighted Slope One*) and *Pearson* (*Pearson CF*) start from 0.02. In the case of the *Epinions.com* dataset (Figure 3.7), the *PRC* of *CoRec* shows over 0.085 precision at the 10% recall level, while the corresponding precision of *wS1* is at 0.05 and that of *Pearson* is at 0.03.

As to the Accuracy measurement, Table 3.2 shows that the *MAE* of *CoRec* stay at the leading level of the three methods (0.09159 for *Last.fm* dataset and 0.2279 for *Epinions.com* dataset). The greatly superior *PRC* of *CoRec* indicates that the

CoRec model does recommend more relevant items than the other three methods. This can be attributed by the highly social-relevant co-peers in the *Co-Peer Graphs*.

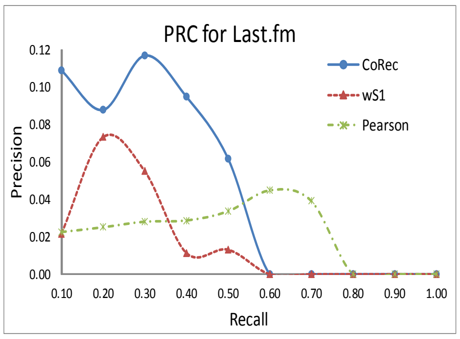


FIGURE 3.6: Last.fm

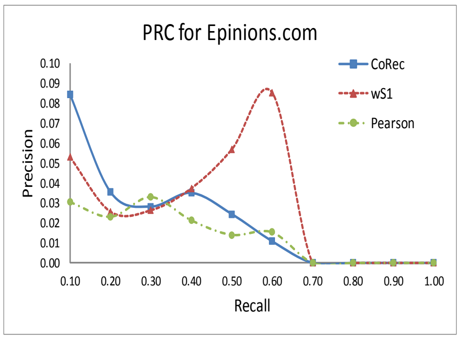


FIGURE 3.7: Epinions.com

TABLE 3.2: NMAE values

Dataset \ Algorithm	CoRec	wS1	Pearson
Last.fm	0.09159	0.0927	0.2928
Epinions.com	0.2279	0.2462	0.2771

It is observed that the PRC curve of *wS1* in *Epinions.com* rises after the recall level 40%, and reaches a peak value of over 0.08 precision at the 60% recall level. That result shows a high rate of the relevant items recommended by the *wS1* algorithm, however it also suggests that these relevant items are recommended at the lower-end of the top-N lists. In contrast, the *PRC* of *CoRec* always leads high precisions at the early recall levels, which intuitively demonstrates that the *CoRec* method recommends relevant items at the higher-end of the top-N recommendation results.

3.5.3 Personalized Rating Style

To examine how the recommendations made by *CoRec* are close to users' personal rating styles, we prepared two datasets with different rating styles for a same user (63) in *Last.fm* data. In particular, the user 63 gets the average rating of 3.6658 in the first dataset, and of 6.6879 in the second dataset. In the meantime, we kept the same friendship relations for the two datasets. We carried out two recommendation processes over the two datasets with distinct rating styles.

Two plots in Figure 3.8 visually display the distinct rating styles of the two recommendation results, with each plot taking the top-20 recommended items. The left-hand image shows the recommended rating distribution for the first rating setting, an average rating of 3.6658. And, the right-hand image shows the second rating setting, an average rating of 6.6879. These two plots show that recommendations made by *CoRec* are always close to a user' individual rating style, regardless of the changes in the setting of the user's rating.

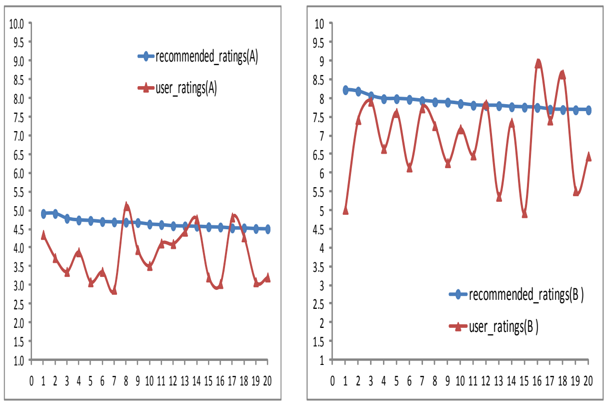


FIGURE 3.8: Personalized rating styles

3.5.4 Evaluation on Non-Response

When a user sends a request to his/her friends for recommendations in online circumstances, it is difficult to expect that all the friends will respond to him/her.

This is also true for the *CoRec* model in which the recommendations are co-produced by the co-peers in a multi-level neighborhood. To investigate the behavior of non-responses, we conducted an experiment of all the 829 users in the *Last.fm* dataset at different *response-rates* by randomly hiding some of the co-peers in a *CPG*. Here the response-rate was defined as the percentage of the responses from outbound co-peers.

Figure 3.9 shows four (4) *PRC* curves of the 829 users at varied response-rates, 100%, 75%, 50% and 33% respectively. As shown, all the *PRC curves* are close to the style of 100% response-rate. In particular, the *PRC* curve of 75% response-rate fully matches that of 100% response-rate, and even achieves superior precision at some recall levels. On the other hand, the *PRC* curves of both 50% and 33% response-rates are lower at most recall levels. In particular, most of the *PRC* values of the 33% response-rates are about 3-5% lower than those of the 100% response-rates. These curves suggest that, the *CoRec* model should work properly in online situations with varied response-rates, although too few responses will decrease the recommendation performance in terms of the precision-recall metric.

3.5.5 Evaluation on Relay Depths

Intuitively, in a propagation process, the deeper the relay depth, the more the co-peers, and the more the co-peers, the more the recommended items. The question to be asked, is do more recommended items mean more relevant items for active users? To examine the correlation between relay depths and recommendation

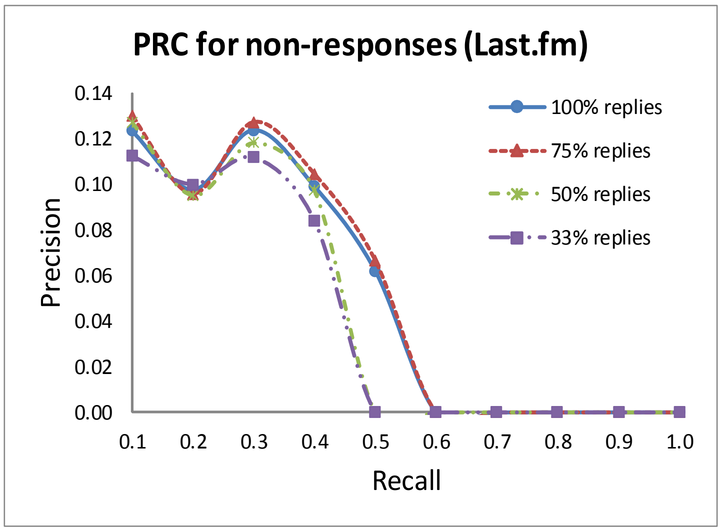


FIGURE 3.9: The effect of non-responses

quality, we conducted experiments on the same *Last.fm* dataset at various relay depths. Figure 3.10 shows four (4) *PRC* curves of all the 829 users at relay paths from two to five. As shown, the *PRC* curves of different relay paths show distinctly different precision values at most recall levels. Of them, the *PRC* curves of relay depths two and three have better performance than those of relay depths four and five.

We have discovered that, when the relay depth increases, the precision values of the corresponding *PRC* tend to decrease, especially at the recall levels between 10% and 30%. This can be interpreted as more co-peers at deeper relay depth levels in a *CPG* probably contribute more items, however, that may not correspondingly increase the number of relevant items for the active user of the *CPG*. Consequently, too many irrelevant items recommended to the user, impair the *Precision* of the recommendations instead.

So far, we present the methodology aspects of the proposed *CoRec* method. We

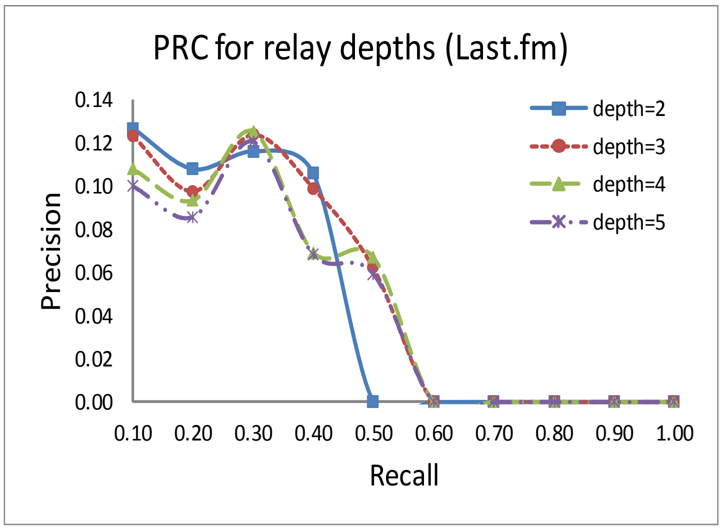


FIGURE 3.10: The impact of relay depths

also demonstrate the experiments of the method, and discuss the evaluations results. In the following section, we review those existing studies which are closely related to the *CoRec* method.

3.6 Comparison and Summary

In this section, we compare our proposed *CoRec* method with two typical *CF* methods: *Pearson CF* and *Slope One*.

While *Pearson CF* is a classic memory-based *CF* method, *Slope One* is an effective model-based one. Both methods are based on the conventional approach of neighborhood formation. In order to obtain quality recommendations, they request a great amount of data, usually tens of thousands or even millions of users and ratings. In contrast, the peer-based *CoRec* allows every user to obtain quality recommendations by asking for help from his/her peers in a Co-Peer Graph (*CPG*).

The *CPG* of each active user is in fact a small-sized and dynamic familiarity based social network. For every peer in the *CPG*, the number of all his/her inbound and outbound co-peers is small, ranging from tens to hundreds. The *CPG* does not request to globally gather all the peers with all their ratings in a central place. In *CoRec*, the whole filtering process is distributed in individual peers, each peer works on a small amount of data, only tens or hundreds of ratings. The relay-based approach makes it possible for *CoRec* to work in distributed online environments.

Owing to the peer-based filtering algorithms, *CoRec* also has the ability to adjust rating styles in different social circles. We often see that a group of users like to give high ratings for their preferred movies. Whereas, the users in another circle may prefer to vote all the movies at low ratings. When an active user has a number of peers participating in different circles, directly aggregating other peers's ratings should not be an appropriate way, as that would probably miss some highly appreciated items by other users because of different rating styles. To address this issue, *CoRec* let every peer aggregate and predict ratings on one's own, locally adjusting rating styles before making final decision.

With the advent of social networking services and mobile computing in recent years, much work has been done to develop recommender systems with social information incorporated. The majority of these systems extract various similarities among the users from social information, such as tags, comments, trustworthiness, sharing documents, etc. Then, they exploit the similarities to make recommendations as side information along with the rating data [51, 66, 71, 150]. However, most of these systems lack a method of dynamic neighborhood formation and a mechanism for adjusting rating style.

There are very few studies which directly utilize personal social networks to form a recommendation neighborhood. The studies in [16] and [41] are two typical examples. For instance, the method proposed by Ben-Shimon et al. [16] constructs a recommendation neighborhood by travel over users' personal social networks, then collect all the rating information from the neighbors. A ranking is calculated by a linear formula over the collected ratings of all other users. The ranking formula is penalized by the distances between an active user and other users. And, the ratings in the study are of binary values (positive or negative). Because the formula penalizes the distances between users, such a ranking formula has a strong bias towards a user's direct friends. As a result of this, a user has more direct friends may receive higher rankings of all the items than other users who have less direct friends. Besides, the method of [16] deals with only binary ratings, whereas, the *CoRec* works with numerical values - a more general condition in recommender systems.

The proposed *CoRec* method can also be categorized in the group of peer-to-peer distributed collaborative filtering systems, such as the studies in [11, 30, 108]. A basic approach of these methods is to set up preference information over the items, for example "topics-of-interests" [30]. Then, each user periodically *gossips* with other peers to find the neighbors having similar interests. Once a user's neighborhood is stabilized, the user makes recommendations by using general *CF* algorithms.

Although the peer-to-peer collaborative filtering systems work in on-line situations, they are still taking the conventional approach of neighborhood formation. That makes these system need a great amount of data for quality recommendations. Moreover, because all these systems collect data through direct communication between an active user and all other peers, it is difficult to adjust diverse rating styles in different social circles.

In addition, existing peer-to-peer filtering systems are built on top of the strict networking protocols such as Distributed Hash Table (DHT). In contrast, the *CoRec* method employs universal *REST-WS* design, which provides flexible interfaces for comprehensive communication among users. The *REST-WS* design in *CoRec* helps to incorporate rich social information into recommender systems.

In summary, we propose a relay-based collaborative filtering method *CoRec*, which emulates the information filtering process in everyday life. The *CoRec* method includes a relay process and a set of peer-based filtering algorithms, allowing peers to propagate recommendation requests and responses over the social networks. The proposed *CoRec* method is especially suitable for the situations where people are seeking recommendations through their social circles. In addition, the embedded *REST-WS* design in *CoRec* makes it easy to work in truly distributed online environments.

Incorporating social relationships is an effective way to increase recommendation qualities. In the next two chapters, we will discuss this topic from two important aspects: (i) measuring tie strengths among the users in a social network (Chapter 4), and (ii) incorporating the tie strengths into recommender systems (Chapter 5).

Chapter 4

Measuring Tie Strength

With the increasing popularity of social networking services in recent years, a great amount of information becomes available on the Web. Nowadays, it is possible to derive social relationships from various forms of resources such as tags, comments, documents, pictures, etc. This brings a new challenge and also a great opportunity for the study of social relationship strength.

The studies in social science have suggested that tie strength is one of the best choices to represent the abstract and integrated form of social relationship strength. In this chapter, we present a novel method for measuring tie strength among the users in a social network, named as *KPMCF*.

This chapter is organized as follows: In next section 4.1, we briefly review the studies of tie strength measurement. The following three sections describe three subsequent steps of the proposed method: section 4.2 models various social relationships by means of kernel learning techniques; section 4.3 factorizes multiple profiles into users' latent social attitudes, jointly with the kernel of interactions; and section 4.4 calculates the tie strength by using the inferred latent variables. After the three-step-description, sections 4.5 and 4.6 discuss the experiment and evaluation on the proposed method. At last, we sum up this chapter in the last section.

4.1 Social Relationship Strength

The studies of Tie Strength emphasize users' overall attitudes towards other people in a social network. The tie strength is an abstract and integrated form of various social relationships among the users. A primary definition of Tie Strength was defined by Granovetter [47] as "a combination of the amount of time, the emotional intensity, the intimacy (mutual confiding), and the reciprocal services which characterize the ties".

In line with Granovetter's study [47], much work has been done to analyze various factors for tie strength measurement [42, 61, 85]. The work of Marsden and Campbell [85] pioneered the concept of "indicators and predictive factors" of tie strength. In particular, this study defines two types of variables with tie strength: indicators and predictors. The indicators are observed measures of social network ties, including closeness, duration, frequency, breadth of discussion topics, and confiding, whereas, the predictors are unobserved aspects of tie strength, such as kinship, co-worker and neighbor status.

Similar to the above study [85] but with significant scale up, the study by Gilbert and Karahalios [42] used a statistical method to analyze 74 variables in seven major dimensions, including intensity, intimacy, duration, reciprocal services, structural, emotional support and social distance, of tie strength. Of the seven dimensions, this study found that the intimacy dimension accounted for more than 30% of the predictive capacity for tie strength - the biggest capacity compared to other dimensions. These finding coincide with the main findings by Marsden and Campbell [85]. Basically, all these studies were based on the relationships derived from face-to-face communication.

The widely spreading social media and mobile services provide great opportunities for analyzing and measuring user's online behavior related to social ties. Many studies have been reported to measure the tie strength in Mobile Phone networks [141], micro-blog platform Twitter [9], professional network *LinkedIn* [135], music sharing services *Last.fm* [13], global social network *Facebook* [60], and so on.

A common feature of the above studies is that most of them calculate tie strength directly based on manifest variables, which are either observed or well-known ones. In fact, tie strength is an abstract and integrated form of combined social relationships. In other words, the factors of tie strength should be underlying latent variables. Therefore, to infer these latent variables of tie strength is the key to measuring tie strength.

In recent years, a number of methods [135, 142, 149] are proposed for measuring tie strength using state-of-the-art techniques such as *Latent Variable Models (LVM).*

Although these studies have achieved great success, several issues are to be addressed. The first issue is how to integrate the diverse forms of social information. All the methods in [135, 142, 149] uses a single table to represent either characteristics and behaviors. This will be difficult for varied datasets with quite different sizes. For example, in the case of *Flickr* test dataset, the number of Labels is low at 42 while that of Tags is 654. To simply put these information in a single table will risk the outliers and imbalance of the data.

Another issue is how to integrate two important parts of users' information: profile characteristics and interaction behaviors? The former basically represent users' static and long term information such as gender, nationality and education, while the latter usually reflect users' short term intention, for instance, users' shopping time and place in a particular duration show their tendency during that time. That is, the latter can be considered as the effect of the former. Consequently, tie strength measurement should take both types of the information into consideration, but with the characteristics at a higher priority. However, the studies of [142, 149] treat all these information equally when they work on the two types of information. Moreover, the study of [135] employs only the information of interactions.

To address the above two issues, we develop two tactics in our proposed method. On one hand, to handle the diverse forms of social information, we deal with two types of information by two different techniques: using kernel learning to handle interaction tables having a same size, whereas employing collective matrix factorization to process profile matrices with different sizes. On the other hand, in the matrix factorization process we impose the kernel of interactions to the latent variables of profiles. By doing so, we are able to infer the tie strength on both characteristics and behaviors, meanwhile keeping the characteristics at a higher priority than behaviors.

In the next three sections, we present a three-step method, named as *KPMCF* to measure tie strength from rich social information. These three subsequent steps are: (i) modeling social relationships by means of kernel methods (section 4.2), (ii) learning users' latent social attitudes using collective matrix factorization over both characteristics and behaviors (section 4.3), and (iii) calculating tie strength from the latent social attitudes (section 4.4).

4.2 Modeling Social Relationships

Nowadays, social networking and social media services have become an indispensable part of everyday life. People use these services to enjoy interest groups, to forward messages and photos, to post comments and tags, and so on. Using these services, one is able to collect rich and diverse forms of social information from various sources, such as tags, comments, documents, pictures, etc. As tie strength is an integrated form of users' profile characteristics and interaction behaviors, one important task of measuring tie strength is to handle the diverse forms of social relationships, as many as possible.

In this section, we propose a kernel learning method to represent varied social relationships in a combined form. Firstly, we illustrate the rich information derived from an actual dataset. Next, we outline a kernel method of representing user's relationship. At last, we combine the multiple relationships in a single kernel matrix, which will be used for learning users' latent social attitudes in next section.

4.2.1 Users' Characteristics and Behaviors

Users' profile characteristics and interaction behaviors have heterogeneous forms, including users' demographic and interaction information. Some examples of the information are location, education, jobs, friend-assignment, posting comments, writing tags, sharing reviews, and so on. Most of these pieces of information can be mathematically represented by matrix.

Matrix is a mathematical form of an entities-attributes data block, that is also called a "relation". Generally, a matrix is a rectangular array of numbers, symbols, or expressions, arranged in rows and columns. Every entity is represent by a row, and every attribute by a column. Each entry of the matrix is called an element, which stores a value. Generally speaking, a relation of users' characteristics will be such a matrix that the rows represent users but the columns the attributes. On the other hand, a relation of users' interactions with other people can be represent by a square matrix where both rows and columns are users. Consider a matrix of users' friendship, the rows of the matrix represent the users themselves, and the columns represent the friends of these users. Each element in the matrix is a value of 1 or 0, indicating whether a user accepts a person in the columns as his/her friend (1) or not (0).

In particular, in order to measure the tie strength among the users in a social network, we need to extract users' various social information from existing dataset. In the following, we illustrate the users' relations derived from a real dataset of *Flickr* [1].

Flickr helps users post and share photos with other people, it also encourages users to make contact lists, join interesting groups, and add comments on photos. The users of *Flickr* have formed extensive social networks, which involve a great amount of information with regard to users' profile characteristics and interaction behavior. Especially, the dataset we are using in this study includes a wide range of users' activities, such as a user's friends, interesting groups, location (country) of posting photos, the time of posting, label of photos, tags of photos, etc. Based on these information, we are able to derive a number of explicit and implicit relationships among users.

Actually, we extract seven (7) matrices from the dataset. Of them, three are used as profile relationships: *User-Country (P^c)*, *User-Label (P^b)* and *User-Tag (P^t)*, and another four as interaction relationships: *User-Friend (X^f)*, *Co-Tag (X^t)*, *Co-Label (X^b)* and *Co-Group (X^g)*. The details of these matrices are specified as follows:

User-Country (P^c)- A User-Country matrix P^c associates users with countries, which are converted from the locations in the original dataset. The rows of P^c stands for users, and the columns for countries from which users post photos. As

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

one user may post many photos in a same country, we set each entry p^c_{ij} by the number of the photos posted by a user i from a country j.

User-Label (P^b)- Labeling in Flickr is essentially a facility of classification provided by image sources, therefore the labels in some degree indicate users' preferences for photos. The number of the labels in *Flickr* dataset is about 50, defined by *Flickr*. Of the user-label matrix P^b, the rows stand for users, and the columns for labels. Each entry p^b_{ij} is set to the count of a same label j received by a user i.

User-Tag (P^t)- In Flickr, each author is able to assign tags to his/her photos, thus the tags can be read as a sign of users' intention. We calculate a *tag-vector* for each user in a tag space developed by *Bag-of-Words* model [10]. In other words, each row in P^t specifies a user's preferences for tags. The number of the tags are really huge comparing to those of labels. In our test dataset, it reaches at 27,250. To conduct our evaluation, we have filtered out meaningless tags such as a single character or punctuations, as well as those tags which occur only one time. Finally, the number of the tags used in experiments dropped to 654.

User-Friend (X^f)- The User-Friend matrix X^f is an interaction matrix, of which both the rows and columns are users. Friends often share many photos, so an entry x^f_{ik} is set by the numbers of the photos posted by both users i and k. Instinctively, the more shared photos, the stronger the relationship of the two users.

Co-Tag (X^t)- While the User-Friend matrix specifies explicit connections, the Co-Tag matrix X^t represents implicit interactions among users by observing how many of the same tags are posted by user-pairs. Each entry of X^t is set by the accumulated count of the same tags posted by the corresponding pair.

Co-Label (X^b)- Similar to Co-Tag matrix X^t, the Co-Label matrix X^b also represents the implicit interactions among users. Each entry of X^b is the accumulated count of the common labels posted by two corresponding users of the same photos.

Co-Group (X^g)- Intuitively, if two users join the same groups, then the two users probably have strong social ties. The Co-Group matrix X^g is constructed by specifying the numbers of the same groups in which two users post their photos.

For further discussion in the following subsection and sections, we denote the *User-Label* matrix P^b as P to represent a profile relationship, and refer $PR = \{P^c, P^b, P^t\}$ to all the profile relationships, and N_p to the size of PR. Similarly, we denote the *User-Friend* matrix X^f as X to stand for a interaction relationship, and $XR = \{X^f, X^t, X^g, X^b\}$ for all the interaction relationships, N_x for the size of XR.

Thus, given the various profile and interaction matrices, we need to deal with them in a uniformed way. To this end, we introduce a technique of Kernel Learning for integrating interaction matrices in the next subsection.

4.2.2 Kernel Learning

Kernel learning [55, 114] offers a natural framework to study the relationships between structured objects. In machine learning, one of the most important tasks is to learn the general relationships between objects. To this end, one need to transform the raw representation of data into feature vectors. Many machine learning algorithms perform the transformation by using mapping functions, which are mostly very complicated. However, kernel methods can achieve the transformation by taking only the covariance structure of the raw data.

Using specific transformation mechanism, kernel methods can map raw data from a limited dimensional space into a much higher or theoretically infinite dimensional space without the need of explicit computation over the mapping. In recent years, kernel methods have been successfully studied in a wide range of applications such as hand-writing recognition, text classification, biological computation, face detection, temporal prediction, etc [114].

As graph is one of the most general representation of discrete metric data, the study of graph kernels have received great interests by researchers. In particular, the study by Smola and Kondor [122] proposed a family of graph kernels based on the theory of graph Laplacian. Of the proposed kernels, Regularized Laplacian kernel is a simple but effective form to measure the similarities between the nodes in a graph. As defined in the previous subsection, because all the social relationships are metric data, we prefer to make use of Regularized Laplacian kernels for representing social relationships.

Let $X \in \mathbb{R}^{N_U \times N_U}$ stand for one of the users' interaction graphs, and $K^X = (k(i,k))_{ik}$ for a Regularized Laplacian kernel matrix [74] that specifies the similarities between any pair of two users i and k with regard to W, where N_U is the number of users involved in X. The kernel matrix K^X is constructed by the following steps [122].

1. Normalize X to ensure every entry in X is not negative:

$$\forall x_{ik} \in X, x_{ik} \geq 0 \; where \; i, k \in \{1, ..., N_U\} \tag{4.1}$$

2. Define a *Normalized Laplacian* $\tilde{L}^X$ for the graph represented by X:

$$\tilde{L}^X = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} \; where \; L = D - X \tag{4.2}$$

$$D \text{ is a degree matrix with } d_{ii} = \sum_{k=1}^{N_U} x_{ik}, \; i = 1, ..., N_U$$

3. Construct a Regularized Laplacian kernel matrix K^X as follows, where $\gamma > 0$ is a constant:

$$K^X = (\mathbf{I} + \gamma \tilde{L}^X)^{-1} \tag{4.3}$$

Once we are able to represent each interaction relationship by a corresponding kernel matrix, the next step is to combine them into a uniformed structure, discussed in next subsection.

4.2.3 Kernels of Social Interactions

In this subsection, we construct a single kernel to represent the multiple interaction relationships, at this moment, all the relationships can be expressed by a

Regularized Laplacian kernel matrix as described in the previous subsection.

Let $X \in \mathbb{R}^{N_u \times N_u}$ stand for the user-friend interaction matrix, then a Regularized Laplacian kernel matrix $K = (k(i,j))_{ij}$ can be used to stipulate the correlation between any pair of two users i and j participating in X, where N_u is the number of the users in X.

So far, we have constructed one individual kernel K^k for each interaction relationship X^k, where $k \in \{f, t, g, c\}$ representing the interaction relationships $X^k \in \{X^f, X^t, X^g, X^c\}$. To well match the natures of multiple relationships, we generate a combined kernel K_U by a linear combination of these individual kernels. Figure 4.1 shows the integration of the kernels from multiple social interactions, following the plate notation of the graph models [22] (section 2.1).

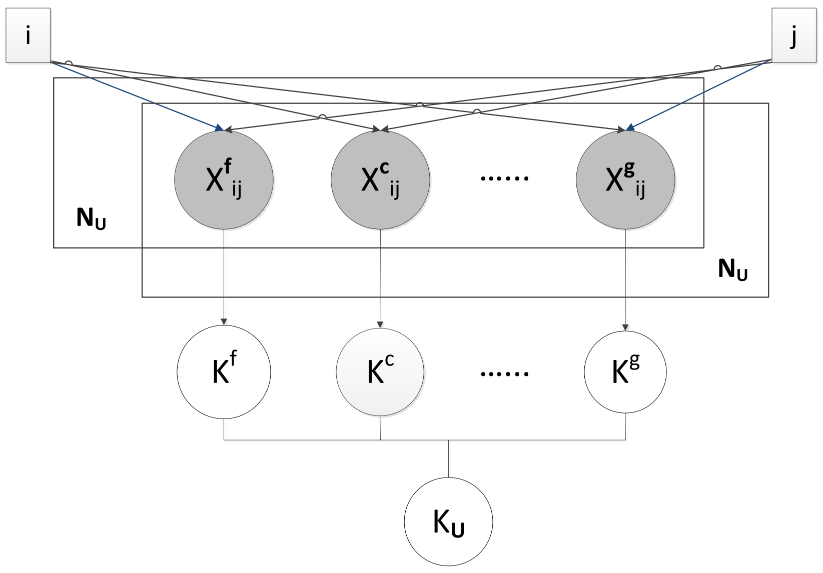


FIGURE 4.1: Multiple kernels of social interactions

Consider a collection of users, we make a linear combination K_U of all the kernels K^k, where $k \in \{f, t, g, c\}$. Intuitively, an entry $(K_U)_{ij}$ specifies the overall correlation between two users i and j with regard to all the given interaction relationships:

$$K_U = (k(i,j))_{ij},\ k(i,j) = \frac{1}{N_k} \sum_{k \in \{f,t,g,c\}} \alpha^k (K^k)_{ij} \tag{4.4}$$

$$where\ i,j \in \{1,...,N_u\},\ k = |\{f,t,g,c\}|\ and\ \sum \alpha^k = 1$$

Besides the set of *users*, we also have other three profile relationships P^c, P^b and P^t. For all these profile relationships, we denote L as the representative of them, where $L \in \{C, B, T\}$. Now, we simply set a diagonal kernel $K_L = \sigma_L^2 \mathbf{I}$ for $L \in \{C, B, T\}$, where $\mathbf{I}$ is an identity matrix.

The above Equation 4.4 is a simple form in order to evaluate the concept of measuring tie strength. There are certain studies regarding the optimal combination of multiple kernels [26, 27, 45], and we will leave the detailed study to future work.

In the following section, the above kernels K_U and K_L are jointly used with users' profile matrices to learn users' latent variables, which represent the users' social preferences.

4.3 Factorizing User Profiles

As we understand that, tie strength is an abstract and integrated form of users' characteristics and behaviors. That is, tie strength is an integration of multiple sources. Furthermore, tie strength is not an explicit measurement such as weight and length, it should be a measurement based on some sort of Latent Social

Attitudes, for example, people's political tendency and business feeling against a particular event. The study of social science has speculated that the behavior of an individual person might be the manifestations projected by some underlying latent social attitudes [47, 88]. The task of measuring tie strength is the problem of finding and calculating these latent social attitudes.

Recent studies indicated that users' profile characteristics affect their behaviors on the web, and exploiting social information could improve the qualities of social network analysis, such as recommendation task [83]. Motivated by these studies, we believe that an integrated approach is needed to deal with profile and interaction relations with diverse forms. To this end, we propose a collective matrix factorization method to learn users' underlying social attitudes.

This section contributes to the details of the proposed method. Firstly, we outline the concept of collective matrix factorization. Then, we present a graphical representation of the method. At last we elaborate the inferring algorithms.

4.3.1 Collective Matrix Factorization

Our goal is to learn users' latent social attitudes from multiple social relations. In order to deal with the multiple relations with diverse natures, *Collective Matrix Factorization (CMF)* is a natural choice. Generally, the technique of matrix factorization is to factorize a single matrix (target matrix) into two dimension-reduced matrices (latent matrices) which represent the latent variables behind the target matrix. As such, the single matrix factorization [2] limits itself to deal with only one target matrix. In the situations where multiple relations or target matrices need to be analyzed, the single matrix factorization becomes powerless.

In order to deal with multiple target matrices, a number of studies *Collective Matrix Factorization (CMF)* have been proposed such as the studies of [78, 118, 119], though some different names are used in the literature. All of these studies are based on the assumption that the latent variables will be twistingly affected each other within the course of collective factorization.

Figure 4.2 (reproduced from [118]) shows a typical *CMF* model proposed by Singh [118]. In this figure, only two target matrices are used, but this does not affect the generality of the concept of *CMF* for dealing with multiple target matrices.

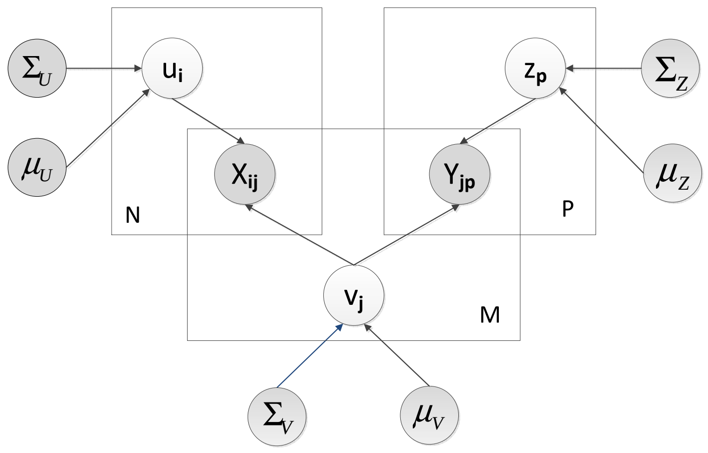


FIGURE 4.2: Graphical model of *CMF*

In Figure 4.2, X and Y are two target matrices. X represents a relationship between two entities: *User* and *Movie*. Similarly, Y stands for that of *Movie* and

[2]we in this thesis call general matrix factorization as "Single Matrix Factorization" in order to differentiate from "Collective Matrix Factorization" when needed.

Actor. A special feature of *CMF* is that one entity is participating in both target matrices X and Y. In this example, it is *Movie*.

When *CMF* is performed, it generates three latent variables: V, U and Z. Of the three latent variables, U represents the latent behavior of *User* in X, and Z indicates the latent behavior of *Actor* in Y. However, the latent variable V stands for the latent behavior of *Movie* in both target matrices X and Y. In addition, variables $\Sigma_U, \mu_U, \Sigma_V, \mu_V, \Sigma_Z, \mu_Z$ are the super-parameters of the corresponding latent variables.

4.3.2 Graphical Model

Based on the concept of *CMF*, we develop a model to learn users' latent social attitudes from their social relationships. In the case of *Flickr* dataset, we have user's profile matrices P^c, P^b, P^t, representing User-Country, User-Label and User-Tag relationships.

Different from the typical *CMF* model depicted in Figure 4.2, we replace the super-parameters with kernels as the priors of latent variables. Particularly, we set the prior of users' latent variables by the users' combined interaction kernel K_U, and the prior of other latent variables of profile relationships by corresponding kernel K_L.

Apply the above consideration into account, we design the graphical model of the learning model as shown in Figure 4.3, following the plate notation of graphical models [22].

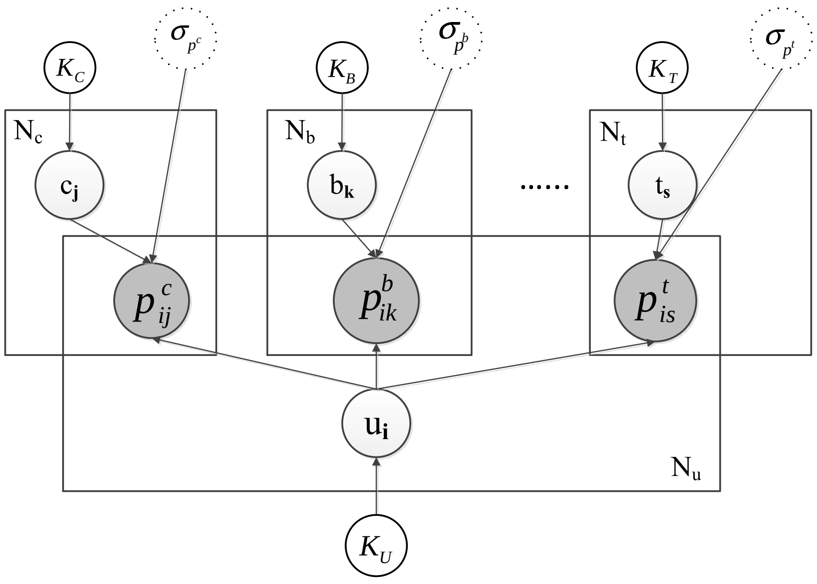


FIGURE 4.3: Graphical representation of learning model

As shown in the graphical model, the proposed *KPMCF* model is essentially a matrix co-factorization process, with kernels for the priors of latent vectors. In the middle of the picture, p^c_{ij}, p^b_{ik} and p^t_{is} stand for the observed values in profile matrices $\{P^c, P^b, P^t\}$, whereas c_j, b_k, t_s and u_i stand for the latent vectors of *countries*, *labels*, *tags* and *users* respectively. At the corners of the plates, N_c, N_b, N_t and N_u specify the dimensions of the corresponding latent vectors.

Out of the plates are input parameters. σ_{p^c}, σ_{p^b} and σ_{p^t} representing the variances of Gaussian distributions to be set over the profile matrices. On the other hand, K_C, K_B, K_T and K_U are the kernels for the priors of corresponding latent vectors. In particular, K_U is a combination of multiple kernels derived from interaction matrices: $user - friend$ (X^f), $mutual - tag$ (X^t), $mutual - group$ (X^g) and $mutual - comment$ (X^c).

This graphic model is translated to a collective matrix factorization algorithm in the following subsection.

4.3.3 Inferring Latent Variables

According to the above graphic model, we now perform a co-factorization over multiple profile matrices P^c, P^b and P^t. For each profile matrix $P \in \{P^c, P^b, P^t\}$. We set the likelihood over these profiles given latent variable U of *users* and other latent variables C, B, T of corresponding profile relationships P^c, P^b, P^t. In the meantime, we assign the kernel K_U as the prior to latent variable U, and K_L to the priors of other latent variables C, B, T respectively. These probability settings can be expressed by the following formulas:

$$p(P|U, L, \sigma_P^2) = \prod_{i=1}^{N_u} \prod_{j=1}^{N_l} [\mathcal{N}(p_{ij}|u_{i:}l_{j:}^T, \sigma_P^2)]^{I(i,j)} \tag{4.5}$$

$$p(U|K_U) = \prod_{d=1}^{N_d} \mathcal{GP}(u_{:d}|0, K_U) \quad p(L|K_L) = \prod_{d=1}^{N_d} \mathcal{GP}(l_{:d}|0, K_L) \tag{4.6}$$

Applying Bayesian inference to the above distributions, the log-posterior over U and L can be formulated as follows:

$$\begin{aligned} log\ p(U, L|P, \sigma^2, K_U, K_L) = & -\frac{1}{2\sigma^2} \sum_{i=1}^{N_u} \sum_{j=1}^{N_b} I(i,j)(p_{ij} - u_{i:}l_{j:}^T)^2 \qquad (4.7) \\ & - \frac{1}{2} \sum_{d=1}^{N_d} u_{:d}^T K_U^{-1} u_{:d} - \frac{1}{2} \sum_{d=1}^{N_d} l_{:d}^T K_L^{-1} l_{:d} \\ & - A\ log(\sigma^2) - \frac{N_d}{2}(log(|K_U|) + log(|K_L|)) + C \end{aligned}$$

where A is the number of non-missing entries in P, $|K_U|$ and $|K_L|$ are the determinants of K_U and K_L, and C an independent constant.

Maximizing the log-posterior over U and L is equivalent to minimizing the following object function $\tilde{E}$:

$$\tilde{E} = \sum_{P\in PR}^{P\in\mathbb{R}^{N_u\times N_l}} \frac{1}{2\sigma_P^2}\sum_{i=1}^{N_u}\sum_{j=1}^{N_l} I(i,j)(p_{ij} - u_{i:}l_{j:}^T)^2 \\ + \frac{1}{2}\sum_{N_p=|PR|}\sum_{d=1}^{N_d} u_{:d}^T K_U^{-1}u_{:d} + \frac{1}{2}\sum_{L\in\mathbb{R}^{N_l\times N_d}}\sum_{d=1}^{N_d} l_{:d}^T K_L^{-1}l_{:d} \tag{4.8}$$

A local minimum of the objective function $\tilde{E}$ can be found by executing a gradient descent program over each row of U and $L \in \{C, B, T\}$ alternatively:

$$\frac{\partial\tilde{E}}{\partial u_{id}} = \sum_{P\in PR}^{P\in\mathbb{R}^{N_u\times N_l}} (\frac{1}{\sigma_P^2}\sum_{j=1}^{N_l} I(i,j)(u_{i:}l_{j:}^T - p_{ij})l_{dj} + \epsilon(u)^T K_U^{-1}u_{:d}) \tag{4.9}$$

$$\frac{\partial\tilde{E}}{\partial l_{jd}} = \sum_{P\in PR}^{P\in\mathbb{R}^{N_u\times N_l}} (\frac{1}{\sigma_P^2}\sum_{i=1}^{N_u} I(i,j)(u_{i:}l_{j:}^T - p_{ij})u_{di} + \epsilon(l)^T K_L^{-1}l_{:d}) \tag{4.10}$$

where $\epsilon(k)$ denotes a k-dimensional unit vector with the k-th component being one and others being zero.

Finally, the latent variables u_i and l_j can then be updated iteratively until certain conditions are satisfied. The update equations are as follows:

$$u_i^{t+1} = u_i^t - \eta\frac{\partial E}{\partial u_i} \tag{4.11}$$

$$l_j^{t+1} = l_j^t - \eta\frac{\partial E}{\partial l_j} \tag{4.12}$$

where η is a parameter of a learning rate.

4.4 Calculating Tie Strength

Having users' latent social attitudes inferred from varied social information, we in this section calculate users' tie strength by means of their latent social attitudes. Doing this is based on the *Principle of Homophily in social networks.* This principle suggests people tend to build connections with other people having similar characteristics. Generally speaking, the stronger the relationship among the people, the higher the likelihood that more interactions occur among them [47, 88].

As discussed in section 2.3.2, we in this these focus on strong ties, the strength of which is measured with symmetrical and transitive properties. The symmetry means the tie strength between two users are equal in either direction. Furthermore, in a social circle users' tie strengths are potentially transitive. Provided that both users u and v have strong tie strength with user w, consequently, users u and v will mostly have stronger tie strength than the situation where no tie exists with user w. Based on this assumption, we calculate users' tie strengths by accumulating the tie strengths with commonly related other users. The detailed formula is as follows.

First, by applying *Pearson Correlation Coefficient (pcc)* function to the latent matrix $U \in \mathbb{R}^{N_u \times N_d}$, we calculate the pair-wise similarities between all the pairs of the users in U. Then, for every pair of two users i and j, we find out the other users who interact with both users, named as "mutual peers" and denoted as M^{ij}. Finally, we sum the similarities between the two users and the similarities between the pair and these "mutual peers" to obtain the social relationship strength s_{ij}

between i and j. Formally:

$$\begin{aligned} s_{ij} &= pcc(i,j) + \sum_{m \in M^{ij}} (pcc(i,m) + pcc(m,j)) \\ where\ pcc(x,y) &= \frac{\sum_{d=1}^{N_d}(u_{xd} - \bar{u_x})(u_{yd} - \bar{u_y})}{\sqrt{\sum_{d=1}^{N_d}(u_{xd} - \bar{u_x})^2 \sum_{d=1}^{N_d}(u_{yd} - \bar{u_y})^2}} \end{aligned} \tag{4.13}$$

The values calculated from the Equation 4.13 can be used as the tie strength among the users in a social network, which represents the users' whole attitudes towards other users in the network. In the following two sections, we demonstrate how the measured tie strengths can be used in certain social applications, and how the tie strengths represent a closer relationship then other affinities among users.

4.5 Experiment Settings

4.5.1 Dataset Description

We have conducted experiments of the proposed *KPMCF* method over a dataset outlined in Table 4.1. The *Flickr-PASCAL* dataset is sourced from the study by McAuley and Leskovec [87], which in fact includes four datasets: *Flickr-CLEF*, *Flickr-MIR*, *Flickr-NUS* and *Flickr-PASCAL*.

From the data source *Flickr-PASCAL*, we extracted seven relations for experiments (named as *Flickr3* dataset): User-group, User-Location (Country), User-Label, User-Friend, Co-Tag, Co-Group and Co-Label. Of them, the first three relations were treated as users' profile relations, whereas the last four were treated as interaction relations. The experimental tasks are to recommend the might-be-interested groups for users.

Table 4.1: *Flickr-PASCAL* data source and relations

Flickr	Photos	Users	Groups	Tags	Locations	Label	Friends
Source	10,189	8,698	6,951	27,250	1,222	50	N/A
Relations		1,324	562	654	62	42	1324

4.5.2 Evaluation Metrics

Receiver Operating Characteristics - ROC

In the experiments, we evaluate a case of how the tie strength can be used to classify friendships in a existing dataset. To compare the performance of the classification, we employ a classification metric: *Receiver Operating Characteristics (ROC)* [19, 36]. *ROC* specifically deals with classification accuracy, as well as "sensitivity" and "specificity" [92].

Consider a classification problem with two classes, where each instance is mapped to one element of $\{p(ositive), n(egative)\}$ of the two class labels. Given a classifier, there are four possible outcomes for any instance: true positive (TP), false negative (FN), true negative (TN) and false positive (TP). Subsequently, we can estimate the *true positive rate* (tp_rate) and *false positive rate* (fp_rate) as follows.

$$
\begin{aligned}
tp\_rate &= \frac{\#TP}{\#TP + \#FN} \\
fp\_rate &= \frac{\#FP}{\#FP + \#TN}
\end{aligned}
\tag{4.14}
$$

With the *tp_rate* plotted on the Y axis and *fp_rate* on the X axis, an *ROC* graph can be drawn, which depicts the relative tradeoffs between benefits (true positives) and costs (false positives); that is, the classification accuracy of the classifier. To compare the performance of classifiers, an alternative method is to calculate a single scalar value for the *Area Under the (ROC) Curve*, denoted as *AUC* [19]. In the experiments, both *ROC* and *AUC* are taken into account when used for evaluation.

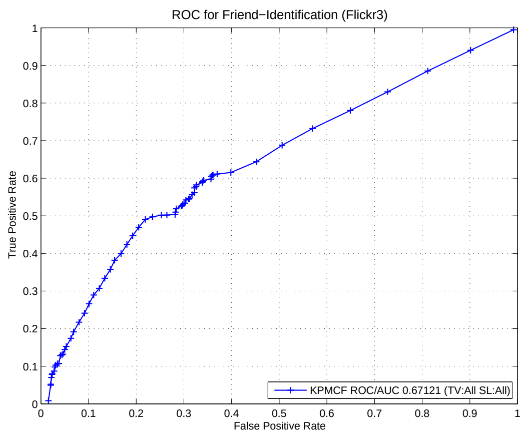


FIGURE 4.4: A sample ROC curve

Figure 4.4 show a sample *ROC* Curve, with the *AUC* value displayed in the legend label. This example demonstrates how a method can be used to classify each pair of users as friends or not, comparing with the ground truth of friendships. Intuitively, the higher the curve to the left-upper direction (or the larger the *AUC* value), the more accurate the classifier.

Precision-versus-Recall Curve - PRC

In the evaluation, we also employ metric *Precision-versus-Recall Curve (PRC)* [10], which has been introduced in section 3.4.3. For the purposes of convenience, we repeat the formula of *Precision-versus-Recall Curve* as follows (Equation 4.15):

$$\begin{aligned} &Precision = \frac{|R_a|}{|A|} \quad Recall = \frac{|R_a|}{|R|} \\ &and\ Average\ PRC:\ \overline{P}(r) = \sum_{u=1}^{N_u} \sum_{p=1}^{N_p} \frac{P_p^u(r)}{N_p} \end{aligned} \tag{4.15}$$

where A stands for the set of *retrieved items*, R for the whole *relevant items*, and R_a for the intersection of A and R (*retrieved relevant items*). Furthermore, $P_p^u(r)$ is the precision at recall level r for the p-th experiments for user u; N_p and N_u are the numbers of the experiments and the users respectively. As such, $\overline{P}(r)$ represents the *Average PRC* value at recall level r in all N_p experiments and for all N_u users.

4.6 Evaluation

To validate the effectiveness of the tie strength learned in *KPMCF*, we conducted two experiment cases on real-world datasets from *Flickr3*: *Evaluation of Friend Identification* and *Evaluation of Group Recommendation.*

In the evaluation, we extract three profile relationships (User-group, User-Location (Country), User-Label) and four interaction ones (User-Friend, Co-Tag, Co-Group and Co-Label) from the *Flickr-PASCAL* data source. We also denote the tie strength values calculated from the measuring method as *KPMCF.*

4.6.1 Evaluation of Friend Identification

In the first evaluation case of *Friend Identification*, we use the values of various relationships to classify the friendships of every user-pairs in the dataset. The friendship itself is a ground truth concluded in *User-Friend* table. The tie strength values are calculated by the *KPMCF* method. What we want to see is, whether the tie strength values can be used to classify friendship? and how the classification performance of tie strength can be compared with other relationships such as *Co-Tag*, *Co-Label* and *Co-Group*? The relationship values of the latter three relationships are taken from the corresponding matrices X^t, X^b and X^g.

we built an *ROC Curve* of each relationship by testing all the user-pairs according to corresponding relationship values, where the test value of each pair equals to one if the two users are friends, otherwise zero. Then, we measured the *Receiver Operating Characteristic / Area Under Curve (ROC/AUC)* (section 4.5.2) for each relationship.

We process the evaluation as follows. First, we use a combination of users' profile and interaction matrices to compute the *KPMCF* relationship strengths (*KPMCF*) among these users. Next, we calculate the relationship values for the other three relationships: *Co-Tag*, *Co-Label* and *Co-Group*. Then, we prepare test data with the labels of "friend" or "not friend" for all the user-pairs involved in the test data, according to the information in the $user - friend$ matrix. Finally, we perform an ROC/AUC test over the *KPMCF* and other three relationships.

The following Figure 4.5 shows the *ROC* curves of the test results, where the X- and Y-coordinates represent the False-Positive-Rate and True-Positive-Rate respectively. The corresponding AUC values are 0.67121 (*KPMCF*), 0.59459 (*Co-Tag*), 0.52984 (*Co-Label*) and 0.63864 (*Co-Group*) respectively.

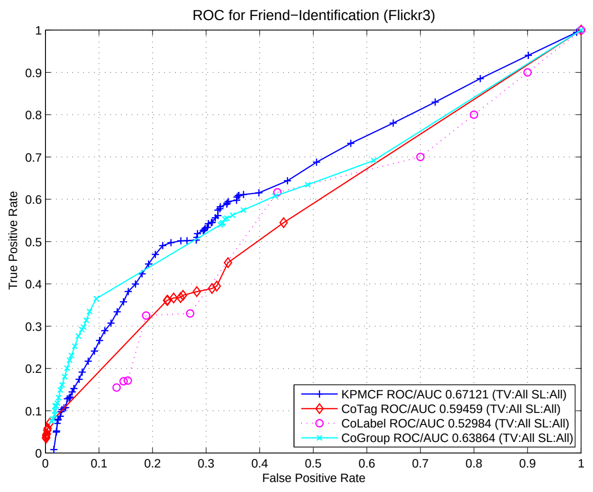


FIGURE 4.5: ROC curves

As the results demonstrate that the tie strength, represented by *KPMCF*, shows superior to all other relationships in the *Friend-Identification* application. In particular, the *KPMCF* earns the highest AUC value at 0.67121, which is about 10% higher than those of *Co-Tag* and *Co-Label*. On the other hand, the *Co-Label* relationship has the lowest AUC value at 0.52984. That means, whereas the tie strength learned from *KPMCF* is helpful for identifying friendship between users, the number of the labels commonly set by user-pairs is not a meaningful help in friendship-judgment.

4.6.2 Evaluation of Group Recommendation

The second evaluation case is *Group Recommendation.* In *Flickr* web site, users are encouraged to join different interesting groups. When a new user comes to *Flickr*, or when an existing user wants join a new group, what are the potential interesting groups (but not joined yet) for the user? In the evaluation, we use users' different relationships to make recommendations. As such, we evaluate the usefulness of each relationship for making recommendation. In this context, tie strength is considered as one of the relationships among the users.

The evaluation was conducted on four types of relationships (Tie Strength (KPMCF), Co-Tag, Co-Label and Co-Group). Firstly, we made recommendation for all the groups over the test dataset, named as G. Then, for each type of relationship, we produced a top-K recommendation list of potential interesting groups for every user.

The recommendation list and recommendation scores are determined by Equation 4.16. In the formula, a recommendation list is a sorted list of recommendation scores (rs) predicted for a particular user u against every available group g in G. For each group, this recommendation score is defined as the sum of the relationship values (normalized) between the user and other users who are joining the group. The relationship values are taken from the corresponding matrix of the evaluating relationships. In particular, for an individual user u,

$$\begin{aligned} recommendation\ list &= \underset{g \in G}{\arg\ \mathrm{sort}}\ rs(u,g) \\ where\ rs(u,g) &= \sum_{k \in User, k \neq u} rlv(u,k)\ I(k,g) \end{aligned} \tag{4.16}$$

In the Equation 4.16, $rlv(u,k)$ represents the value of a particular relationship under evaluation, and $I(k,g)$ is an indicating function, which returns one if a user k joins the group g, otherwise zero. Intuitively, if a group is joined by the other users having strong relationships with a user u, then this group will be more likely recommended to the user u.

Using all the information of the top-K recommendation lists, we then calculate the *PRC* values for the recommendation results for each relationship. In terms of *PRC*, the *relevant groups* are the joined groups of each user, and *retrieved relevant groups* are those joined groups which also appear in the top-K lists.

To compare the *PRC* performance between tie strength and other relationships, and also to investigate the impact of the tie strength, we organize four-groups of experiments, as shown in Figures 4.6, 4.7, 4.8 and 4.9. All four experiments measured the *PRC* for the four types of relationships, each experiment selecting users of different "weight levels": *High*, *Medium*, *Low* and *Average*. The three levels are generated as follows:

Normalizing all the relationship values to scale from 0 to 1.0, the users of *High* weight level are those having relationship values higher than 0.70. Similarly, the

Medium level means the relationship values are between 0.30 and 0.70, and the *Low* level indicates the relationship values are lower than 0.30. Finally, the *Average* weight level means taking all the users without selection. Figures 4.6, 4.7, 4.8 and 4.9 are the *PRC* plots for the four experiments of *Average*, *High*, *Medium* and *Low* weight levels respectively.

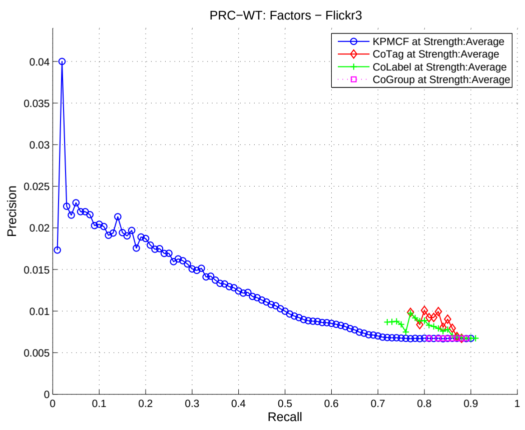


FIGURE 4.6: PRC curves - Average

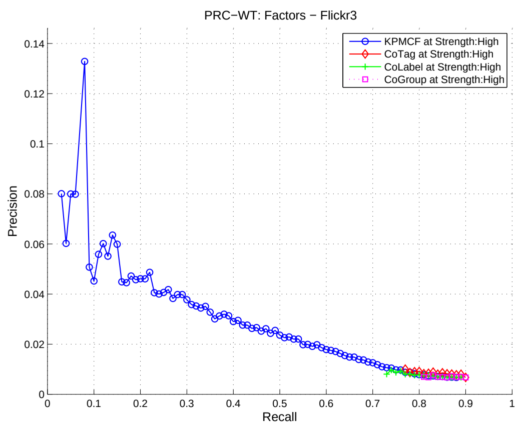


FIGURE 4.7: PRC curves - High

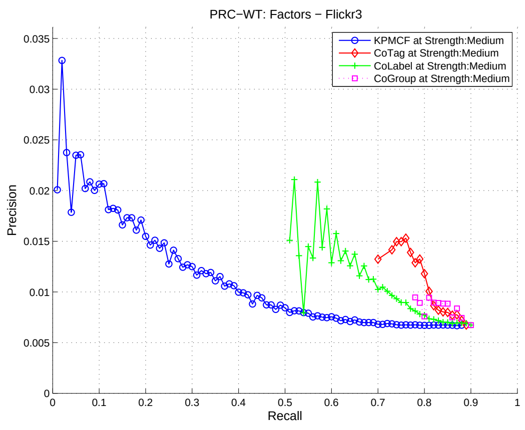


FIGURE 4.8: PRC curves - Medium

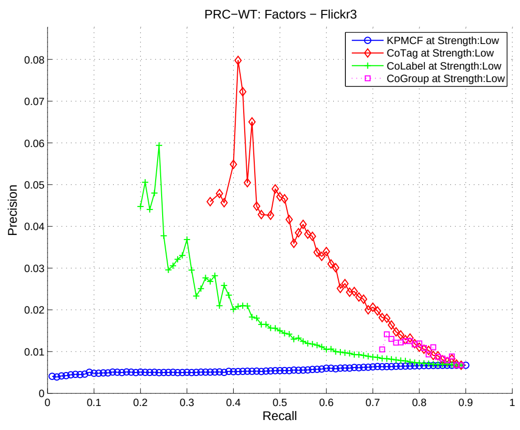


FIGURE 4.9: PRC curves - Low

As shown in the plot *Average PR* curves (Figure 4.6), the Precision-Recall performance based on Tie Strength (KPMCF) receives the best result for high-end recall levels from 0% to 70%. All other relationships deliver a *PRC* only after recall level 70%, though, some of the Co-Tag and Co-Label obtain a slightly higher *PRC* than Tie Strength after the recall level of 70%. Drawing an analogy between this experiment result and web suffering, it is as if the recommendations by tie strength come up in the first several pages, while those by other relationships come out in later pages.

It is worth noting that the Precision-Recall performance delivered by Tie Strength shows a decreasing trend when the strength level changes from high to low. In the case of *High* level tie strength (Figure 4.7), the *PRC* curve starts from the highest point 0.14 near the 10% recall level, and then steadily moves down to 0.01 at the 70% recall level. However, for the *PRC* of *Medium* level (Figure 4.8), it starts from 0.033 as its highest precision, and moves down to 0.0075 at the 70% recall level. The *PRC* for *Low* level shows a very poor performance (Figure 4.9), where the Precision remains at around 0.005 for all the recall levels. These experiment results demonstrate that the tie strength does play an important in the recommendation process: the stronger the tie strengths among the users, the higher the precision of the recommendation results.

4.7 Comparison and Summary

In this section, we compare our proposed *KPMCF* method with the studies in two closely related fields: tie strength measurement and extended matrix factorization methods.

Along with the increasing popularity of social media services, it becomes possible to capture users' both interaction and profile information. This brings greater opportunities for measuring tie strength. However, many existing studies use only interaction data to predict tie strength by using classic statistical or graph theory

[42, 61, 85]. Compare to these studies, the proposed *KPMCF* method explores a new way to measure tie strength from users' varied social relationships.

In recent years, there appear certain studies that exploit profile and/or interaction information to estimate the relationship strength for online social networks [135, 142, 149]. Of the studies, The study by Zhuang [149] utilizes similarity-based combination of kernels to capture users' interaction relationships. The combination of kernels is then used to estimate the tie strength among the users. We are impressed by the kernel learning techniques used in this study. However, although this study well seizes users' interactions using kernels where all the matrices are square ones of the size of users. This method seems not suitable for capturing users' profile information where the sizes of profile entities are basically far different from that of the users.

The study by Xiang [135] proposed a learning method of latent variable models over both of users' interaction activities and profile similarities. This method deals with two tables of the data, one is users' profile similarities, and the other is of users' interactions. To have these two tables, one needs to pre-compile all the interactions and all the profile attributes each in a single table. As the actual social relationships mostly have diverse forms, pre-compiling all information in a single table may risk the loss of important features. For instance, the number of users' education level may be only 3 or 4, whereas, that of users' ethnicity will be more than one hundred. Contrastingly, the *KPMCF* method employs kernel method to handle interaction information, but uses collective matrix factorization technique to deal with users' profile information. This flexible approach of *KPMCF* better serves the varied natures of profile characteristics and interaction behavior.

In the study by Zhao et al. [142], users' relationship strength is predicted for various activity fields based on a joint distribution of *profile strength* and *interaction strength*. These two types of "strength" are determined by the *relatedness* values of application resources - documents. Therefore, this method is limited by the application resources. Whereas, the *KPMCF* method defines tie strength by using users' latent variables, which are inferred from users' profile and interaction relationships, regardless whatever forms the application resources are. This makes *KPMCF* method more universal than others.

As matrix factorization is one of the most promising techniques of *Latent Variable Models*, using matrix factorization to infer users' tie strength is a natural choice. The general matrix factorization methods basically work on a single matrix. To exploit additional information, many studies on extended matrix factorization have drawn great interests in recent years, such as *Kernelized Matrix Factorization (KMF)* and *Collective Matrix Factorization (CMF)*. The studies of [4, 148] are two typical cases of *KMF*, although none of them is related to tie strength. The *PMA* model proposed by Agovic [4] utilizes kernels to capture the covariance for rows and columns of a target matrix, and then additively combines the kernels to generate matrix for prediction. Similarly, the *KPMF* model proposed by Zhao [148] assigns kernels to the priors of latent variables, and then infers prediction from the matrix product of latent matrices.

Although the technique of assigning kernels to priors the priors of latent variables strongly stimulates our research interest, both *PMA* and *KPMF* focused on a single rating matrix or a single kernel due to the nature of recommendation tasks. To include both of users' profile characteristics and social interactions, our *KPMCF* method makes two important extensions to matrix factorization. One extension is using multiple kernels for varied social interactions, and the other is employing collective matrix factorization to deal with multiple profiles.

Collective Matrix Factorization is an effective way to deal with the situations where one or more entities participate in multiple matrices [35, 78, 83, 119]. The framework *SoRec* proposed by Ma [83] can be used to illustrate the mechanism of *CMF*. *SoRec* introduces a social network matrix, in which a social-related term c_{ik}^{*} represents the social information following certain distribution. *SoRec* performs matrix factorization over both social network matrix and original rating matrix, where the entity "User" participates in both matrices. The major difference between *SoRec* and the proposed *KPMCF* method is that, while *SoRec* places Gaussian priors on the latent variables of users, *KPMCF* assigns the kernels of social interactions to the priors of users' latent features. Because of this assignment, *KPMCF* becomes able to measure tie strength from both of users' profile and interaction information.

Sum up this chapter, we develop a measuring method, named as *KPMCF*, to measure users' tie strengths based on various social relationships. The proposed *KPMCF* method shows three advanced features. Firstly, it uses kernel learning technique to capture users' varied interaction relationships; and secondly it performs collective matrix factorization to learn users' latent social attitudes. Lastly, it calculates users' tie strengths from the inferred latent variables instead of manifest data.

The tie strength measured by *KPMCF* can be used in various applications. In the next chapter (Chapter 5), we will develop a recommendation method that incorporates tie strength into recommendation process.

Chapter 5

Robust Recommendation with Tie Strength

In the previous chapters 3 and 4, we have developed a peer-based filtering method *CoRec* and a measuring method for tie strength *KPMCF*. In this chapter, we exploit the results derived from these two methods to build a Robust Matrix Factorization method with Tie Strength incorporated. This recommendation method is named as *TieRec*.

This chapter is organized as follows: The next section outlines the motivation of incorporating tie strength. Section 5.2 and section 5.3 jointly describe two important aspects of the *TieRec* method. The former discusses the propagation of tie strength; and the latter elaborates the robust algorithms of matrix factorization, focusing on L1 LOSS mechanism. The following sections 5.4 and section 5.5 show the comprehensive experiments and evaluations, in comparison with certain state-of-the-art *MFSR* methods. At last, section 5.6 summarizes the proposed *TieRec* method.

5.1 Matrix Factorization based Social Recommendation

When social media services become increasingly popular, incorporating users' social information has become possible and also necessary to make high quality recommendations. In this thesis, we are especially interested in matrix factorization based social recommendation (*MFSR*) methods. This is because matrix factorization has shown great promise for making highly accurate recommendations, and it also provides outstanding flexibility in working with additional information. In spite of that, to effectively incorporate social information into matrix factorization, one needs to answer three major questions: which social information is included? which techniques are used? and how to increase the recommendation qualities of both accuracy and relevance?

Since the start of recommender systems, various social information has been included in recommendation process, such as friendship and trustworthiness. In recent years, common activities on tag, label, comments, posting have also been studied in many applications. More recently, some parts of user's profile become available on the web, such as users' location, gender, education, occupation, etc. When more and more types of social information become available, an abstract and integrated form is needed to facilitate the incorporation of rich and varied social information. Tie strength should be one of the right choices. A feasibility study of incorporating tie strength in *MFSR* methods is one of the two important aspects in this chapter.

There are many *MFSR* methods proposed in recent years. Each method has its own advantages in a particular situation. When incorporating tie strength, our experiments reveal that, while certain methods (such as *SR2* [84]) show higher accuracy of the recommendations but lower relevance than other methods, some (such as *SoRec* [83]) exhibits higher relevance of the recommendations but poorer accuracy than others. From the point view of information retrieval, we believe that both accuracy and relevance are very important criteria for high quality recommender systems.

We also find that the work on robust matrix factorization helps to improve the quality of the relevance of recommendation results. There are a number of solutions to robust matrix factorization [1, 72, 133]. Of them, the *PRMF* model proposed by Wang [133] improved robustness of matrix factorization by adopting L1 LOSS technique for objective functions. Usually, Gaussian probabilistic matrix factorization employs L2 LOSS over objective function. Here, the L1 LOSS means *Least Absolute Errors* that minimizes the sum of the absolute differences between two sets of objects. Whereas, the L2 LOSS represents *Least Squares Error* which squares the differences between the elements of two sets. Owing to the nature of L2 LOSS, most proposed *MFSR* methods are still sensitive to noise, outliers and even missing entries occurring in data source. In this chapter, we develop a robust matrix factorization based on L1 LOSS technique, and meanwhile incorporate tie strength in matrix factorization.

The next two sections contribute to the algorithms of our proposed recommendation method. We firstly consolidate the relationships between active users and their recommenders (subsection 5.2.1), and formalize the propagation of tie strengths among the users (subsection 5.2.2). Then, we introduce the robust matrix factorization method based on L1 LOSS (subsection 5.3.1, and develop the algorithms of integrating tie strength (subsection 5.3.2). The proposed recommendation method is named as *TieRec*.

5.2 Consolidating Users' Tie Strength

In this section, we discuss the consolidation of tie strength with the recommenders derived in *CoRec*. We also formalize the propagation of tie strength over users' latent variables.

5.2.1 Relationships with Recommenders

As a social network based recommendation method, we want to include various types of social relations as more as possible. Besides the explicit relationships such as User-friend, we can also find and use implicit relationships, such as the relationship between a user and the peers who give him/her feedback. In the *CoRec* method, when an active user receives a set of recommendations, he/she at the same time gets a list of the peers who make recommendation for this user. We call these peers as recommenders. Now, we formalize the relationship between an active user and the recommenders.

To define the interactive relationships between an active user and all the recommenders, we deisgn a *Recommending Co-peer Graph* or *PCG*, which can be expanded from the concepts and notations introduced in Chapter 3 (3.2.1):

- For a user v in a *CPG*, a direct recommender of v is one of his/her out-bound co-peers, who recommends certain items to v. A single item i may be recommended by multiple recommenders. We define the set of these recommenders as *Direct Recommenders (DD)* for v w.r.t. item i, $DD_v(i) = \{u \mid u \in OT_i(v) \; and \; i \in I_u^f(v)\}$.
- For a user v and a recommended item i, we denote those direct and indirect recommenders who recommend item i as *Relaying Recommenders (RR)* for v w.r.t. (with regard to) i. This can be recursively defined as follows:

$$RRv(i) = DD_v(i) \cup (\cup_{u \in DD_v(i)} RR_u(i)) \tag{5.1}$$

$DD_v(i)$ represents the Direct Recommenders for v w.r.t. item i

$if \; DD_v(i) = \emptyset, \; then \; RR_v(i) = \emptyset$

- For a user v and a direct recommender $u \in DD_v(i)$, as well as a recommended item $i \in I_u^f(v)$, a *Direct Recommending Path (DRP)* between v and u w.r.t. i is defined as a set of the requesting and responding paths between v and u. That is, $DRP_v^u(i) = \{\langle v, u \rangle, \langle u, v \rangle\}$.

- Moreover, for a user v and a recommended item i, a *Combined Recommending Path (CRP)* for v w.r.t. i is defined as the combination of the union of all the *DRPs* between v and its direct recommenders $u \in DD_v(i)$ and all the *CRPs* for these recommenders with regard to the item i. This is a recursive definition, whereas if a user u does not have further direct recommenders for item i, then the *CRP* for u w.r.t. i is simply an empty set:

$$CRP_v(i) = \cup_{u \in DD_v(i)} (DRP_v^u(i) \cup CRP_u(i)) \tag{5.2}$$

$DD_v(i)$ represents the Direct Recommenders for v w.r.t. item i

$$if\ DD_v(i) = \emptyset,\ then\ CRP_v(i) = \emptyset$$

- Therefore, a *Recommending Co-peer Graph (RCG)* of an active user v_0 or RCG_{v_0} is defined by means of the concepts of *RR* and *CRP* as follows.

$$RCG_{v_0} = (v_0, V^*, E^*, I^f(v_0)) \tag{5.3}$$

$$where\ V^* = \cup_{i \in I^f(v_0)} RR_{v_0}(i),\ E^* = \cup_{i \in I^f(v_0)} CRP_{v_0}(i)$$

The graph in Figure 5.1 illustrates an *RCG* for user 63, after consolidating all the *Relaying Recommenders* for all the recommended items.

The above defined interaction graph RCG_{v_0} represents the relationship between an active user and his/her recommenders. This relationship can be used in several ways. For instance, it can be used to re-construct a special user-item matrix, based on the original one, in order to achieve higher quality of recommendations than

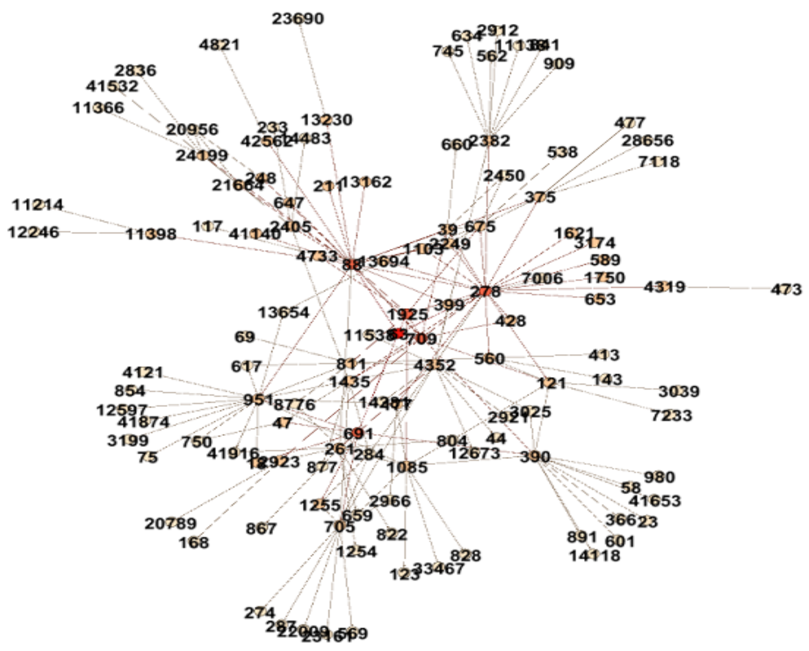


Figure 5.1: *RCG* for user 63

using the original user-item matrix. Another usage is to include this relationship in tie strength - the abstract and integrated form of social relationships among the users in a social network. Such a tie strength can be learned through the *KPMCF* method. In this thesis, we simply take the latter approach in order to focus on our *TieRec* recommendation method. For this purpose, the tie strengths between the active user v_0 and all the recommenders V^* are denoted as a 2-dimension relation $S \in \mathbb{R}^{M \times M}$ where $M = |V^*| + 1$.

5.2.2 Propagating Tie Strength

Provided that we have measured the tie strength among the users, the following question is how to bring this tie strength into matrix factorization process.

Although there are many approaches to do so, we want to take a concept of social ensemble in the proposed method. This concept closely coincides with the principle of homophily discussed in previous chapters.

As discussed so far, a contact between similar people usually occurs at a higher rate than those among dissimilar people, and moreover, co-peers are an important source of influence on people's behavior. To apply the homophily to matrix factorization based social recommendation, a user's latent features u_i can be considered as the combined effect of the latent features u_k of the user's co-peers, associating with the corresponding tie strength $s_{ik} = S(i, k)$, where $k \in RCG_i$, RCG_i is a *Recommending Co-peer Graph* of user i, and $S \in \mathbb{R}^{M \times M}$ represents the 2-dimension tie strength relation among the the active user and all the recommenders.

The association between these latent feature vectors of the users can be expressed as the following Equation 5.4:

$$u_i = \frac{\sum_{k \in RCG_i} s_{ik} u_k}{\sum_{k \in RCG_i} s_{ik}} \tag{5.4}$$

where u_i and u_k are the latent feature vectors of users i and k respectively; RCG_i is a *Recommending Co-peer Graph* of user i, and s_{ik} represents the tie strength between users i and k, where $s_{ik} = S(i, k)$, $S \in \mathbb{R}^{M \times M}$; M is the number of the users involved in the recommendation process.

By normalizing the tie strength matrix as to enforcing $\sum_{k \in RCG_i} s_{ik} = 1$, we get a simplified form of the association formula of users' latent feature vectors as follows:

$$u_i = \sum_{k \in RCG_i} s_{ik} u_k \tag{5.5}$$

Now, the inclusion of tie strengths brings two joint probabilities affecting users' latent variables. One is the general Gaussian distribution with a mean of zero, and the other is the conditional distribution given the latent feature vectors of the co-peers. these joint probabilities can be described as the following Equation 5.6:

$$\begin{aligned} & p(U|S, \sigma_U^2, \sigma_S^2) \\ & \quad \propto p(U|0, \sigma_U^2)\ p(U|S, \sigma_S^2) \\ & \quad = \prod_{i=1}^{M} \mathcal{N}(u_i|0, \sigma_U^2) \times \prod_{i=1}^{M} \mathcal{N}(u_i| \sum_{k \in RCG_i} s_{ik} u_k, \sigma_U^2) \end{aligned} \tag{5.6}$$

So far, we have prepares the tie strength, the associations between users' tie strengths and users' latent feature vectors. Now, we discuss the robust solution for matrix factorization.

5.3 Robust Matrix Factorization with Tie Strength

In this section, we firstly introduce the concept of robust matrix factorization using L1 LOSS mechanism. Then, we present the algorithms of the proposed recommendation method, which includes tie strength in the factorization process.

5.3.1 Robust Matrix Factorization

Probabilistic matrix factorization can be seen as a realization of *Latent Variable Models* or *LVM* in matrix decomposition. It is based on the assumption that the latent variables follow certain statistical distributions [12]. Based on this assumption, the study by Salakhutdinov and Mnih [110] proposed a matrix factorization model using a probabilistic approach, called *PMF*. In the design of *PMF*, it is presumed that the latent variables of the users and items follow Gaussian distribution as follows:

- for each user, the user's latent features are drawn from a Gaussian distribution,
- for each item, the item's latent natures are draw from a Gaussian distribution,
- furthermore, each rating is treated as a posterior distribution over the user's features and item's natures. It also follows Gaussian distribution.

Formally, let $R \in \mathbb{R}^{M \times N}$ be a rating matrix, representing the ratings given by *N users* for *M items*. Also, let $U \in \mathbb{R}^{M \times D}$ and $V \in \mathbb{R}^{N \times D}$ stand for the latent features of *User* and *Item* such that $R \approx UV^T$, where D is the dimensionality of the latent matrices. The above presumption can be expressed by the following formulas:

$$R = U * V' + E \; where \; E = (e_{ij}) \in \mathbb{R}^{M \times N} \tag{5.7}$$

$$p(U|\sigma_U^2) = \prod_{i=1}^{M} \mathcal{N}(u_i|0, \sigma_U^2 \boldsymbol{I}) \tag{5.8}$$

$$p(V|\sigma_V^2) = \prod_{j=1}^{N} \mathcal{N}(v_j|0, \sigma_V^2 \boldsymbol{I}) \tag{5.9}$$

$$p(R|U, V, \sigma_R^2) = \prod_{i=1}^{M} \prod_{j=1}^{N} [\mathcal{N}(r_{ij}|u_i v_j', \sigma_R^2)]^{I_{ij}} \tag{5.10}$$

where $U \in \mathbb{R}^{M \times D}$ and $V \in \mathbb{R}^{N \times D}$ represent the latent features of users and items respectively, D is the dimensionality of the latent features; $R \in \mathbb{R}^{M \times N}$ is the observed rating matrix of users over items; I_{ij} stands for an indicator function, that equals to one if user i has a rating for item j, otherwise it becomes zero.

Therefore, the log-posteriors of the latent features U and V can be described by an equation associating with Frobenius norms. This can be expressed by the following Equation 5.11:

$$log \; p(U, V|R, \sigma_R^2, \sigma_U^2, \sigma_V^2) \tag{5.11}$$

$$= \lambda_R \|W \odot (R - U * V')\|_F^2 + \lambda_U \|U\|_F^2 + \lambda_V \|V\|_F^2 + C$$

where $\lambda_R = \frac{1}{2\sigma_R^2}$, $\lambda_U = \frac{1}{2\sigma_U^2}$, $\lambda_V = \frac{1}{2\sigma_V^2}$; $\|.\|_F^2$ denotes the Frobenius norm; $W = (I_{ij}) \in \mathbb{R}^{M \times N}$ represents the availability of the observed ratings; and finally, E is an error matrix of i.i.d. zero-mean Gaussian variables with constant variance σ_R.

As such, the problem of predicting users' ratings can be solved by minimizing the Frobenius norm of the difference (*Frobenius Distance*) between the rating matrix

R and a low-rank matrix that is the product of the latent matrices U and V (Equation 5.12):

$$\operatorname*{argmin}_{U,V} \ \lambda_R \|W \odot (R - U * V')\|_F^2 + \lambda_U \|U\|_F^2 + \lambda_V \|V\|_F^2 \tag{5.12}$$

where $\odot$ means an entry-wise product between two matrices.

The above Equation 5.12 is a function of *Least Squares Error* or L2 LOSS, as its first part is a sum of the square of the differences between the elements of R and those of $U * V'$. Intuitively, because L2 LOSS squares the errors, when an error is greater than 1, one will see a much larger error result than *Least Absolute Errors* or L1 LOSS. Here, L1 LOSS is to minimize the sum of the absolute differences between two sets of objects. If this error is caused by an outlier, a model of L2 LOSS will be adjusted to minimize this single outlier at the cost of many other common and small errors. This is called a "Robust" issue. Owing to the L2 LOSS, although the *PMF* and its variants achieved great successes, they are still sensitive to noise, outliers and even missing entries appearing in target matrices.

There are a number of solutions for Robust matrix factorization [1, 72, 133]. In particular, the study by Wang [133] proposed a *PRMF* model that brings L1 LOSS in a probabilistic approach for matrix factorization. To perform matrix factorization under L1 LOSS, the elements e_{ij} of error matrix E in Equation can be set as a Laplace distribution [68] with the mean of zero and the scale parameter as *lambda*:

$$p(e_{ij}|0,\lambda) = \frac{\lambda}{2} exp(-\lambda|e_{ij}|) \tag{5.13}$$

$$where\ E = R - U * V' \tag{5.14}$$

By the Equation , the log-posteriors of U and V can be described by an L1 LOSS form as follows (Equation 5.15):

$$p(U,V|R,\lambda,\lambda_U,\lambda_V) \tag{5.15}$$

$$\propto p(R|U,V,\lambda)\ p(U|\lambda_U)\ p(V|\lambda_V)$$

$$log\ p(U,V|R,\lambda,\lambda_U,\lambda_V) \tag{5.16}$$

$$= \lambda\|R - U * V'\|_1 + \lambda_U\|U\|_F^2 + \lambda_V\|V\|_F^2 + C$$

Consequently, the problem of maximizing $p(U,V|R,\lambda,\lambda_U,\lambda_V)$ is equivalent to the minimization of the following L1 LOSS equation (Equation 5.17):

$$\underset{U,V}{\operatorname{argmin}}\ \lambda\|R - U * V'\|_1 + \lambda_U\|U\|_F^2 + \lambda_V\|V\|_F^2 \tag{5.17}$$

Once the tie strength among the users are ready, the association between users' tie strengths and latent feature vectors are established, and the robust issue has been taken into account, we are now getting in the phase of incorporating tie strength into matrix factorization.

5.3.2 Incorporating Tie Strength

In this subsection, we induce a set of algorithms that incorporate tie strength into recommendation process. In the meanwhile, we realize an inference model for the L1 LOSS based matrix factorization discussed in the previous subsection.

The aforementioned Laplace distribution makes it possible to perform L1 LOSS based matrix factorization, however, conventional gradient methods may become unsuitable for implementation because the L1 norm is basically non-smooth at zero. To address this issue, a hierarchical form of Laplace distribution is a feasible option, which was initially proposed by Lange, K. and Sinsheimer, J.S. [75]. In the hierarchical form, a Laplace distribution is suggested to be equivalently expressed as a scaled mixture of Gaussian variables (Equation 5.18):

$$\begin{aligned} L(z|\mu,\alpha^2) &= \int \mathcal{N}(z|\mu,\tau)\mathcal{E}(\tau|\alpha^2)\mathrm{d}\tau \\ where\ \mathcal{E}(t|\alpha^2) &= \frac{\alpha^2}{2}exp(-\frac{\alpha^2 t}{2}) \end{aligned} \tag{5.18}$$

Applying the above hierarchical form of Laplace distribution to the posteriors of rating distribution, the equation 5.10 can be re-formulated by the following Equation 5.19:

$$\begin{aligned} p(R|U,V,T) &= \prod_{i=1}^{M}\prod_{j=1}^{N}\mathcal{N}(r_{ij}|u_i v_j', \tau_{ij}) \\ where\ p(T|\lambda) &= \prod_{i=1}^{M}\prod_{j=1}^{N}\mathcal{E}(\tau_{ij}|(\sqrt{\frac{\lambda}{2}})^2) \end{aligned} \tag{5.19}$$

where $\tau_{ij} \in T, T \in \mathbb{R}^{M\times N}$ is named as an *embedded variable* in terms of the hierarchical form of Laplace distribution.

The above Equation 5.19 is depicted in Figure 5.2, in which the bottom part shows that a rating r_{ij} is depending on two latent feature vectors U_i and V_j, as well as the embedded latent variable $\tau_{ij} \in T$ following an exponential distribution.

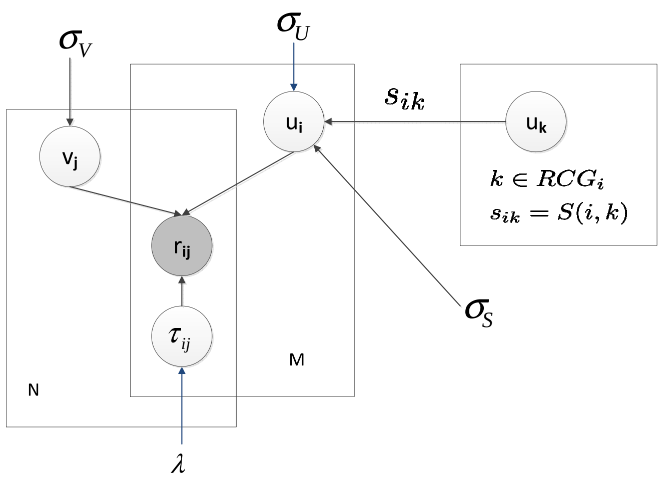


FIGURE 5.2: Graphical model of *TieRec*

The embedded latent variable τ_{ij} can be iteratively estimated in the same Expectation Maximization cycles for estimating latent feature vectors U and V. That is,

$$p(T|R,U,V,\lambda) \propto p(R|U,V,T)\ p(T|\lambda) \tag{5.20}$$

$$E[\tau_{ij}^{-1}|R,U,V] = \frac{\sqrt{\lambda}}{|r_{ij} - u_i v_j'|} \tag{5.21}$$

$$where\ \tau_{ij}^{-1} \triangleq \frac{\sqrt{\lambda}}{|r_{ij} - u_i v_j'|} \tag{5.22}$$

in fact τ_{ij}^{-1} follows an inverse Gaussian distribution.

Now, taking both of the embedded latent variable T and the strength matrix S into account, the posteriors of latent variables U and V can be described by the following Equation 5.23:

$$\begin{aligned} & p(U, V|R, T, S, \lambda, \lambda_U, \lambda_V, \lambda_S) \\ & \quad \propto p(R|U, V, T, \lambda)\ p(U|S, \lambda_U, \lambda_S)\ p(V|\lambda_V) \end{aligned} \tag{5.23}$$

The model parameters U and V can be learned by using *Conditional Expectation Maximization (CEM)* algorithm [58], in which U and V are updated alternatively, with all hyper-parameters fixed and T estimated by the above Equation 5.22.

In particular, in order to update parameter V, we in the Expectation step need to compute the expectation of the log-posterior of V w.r.t. R and T, or the lower bounding function $Q(V)$ as follows:

$$Q(V) = E_T[log\ p(V|R, T, U, \lambda, \lambda_V)] \tag{5.24}$$

Applying Bayes rule to the above $Q(V)$ Equation, we get the following Equation 5.25:

$$\begin{aligned} & p(V|R, T, U, \lambda, \lambda_V) \\ & \quad \propto\ p(R|U, V, T, \lambda)\ p(V|\lambda_V) \\ & \quad = \prod_{i=1}^{M} \prod_{j=1}^{N} \mathcal{N}(r_{ij}|u_i v_j', \tau_{ij}) \times \prod_{j=1}^{N} \mathcal{N}(v_j|0, \lambda_V \boldsymbol{I}) \end{aligned} \tag{5.25}$$

The log form of the above Equation 5.25 can be expressed as follows (Equation 5.26):

$$\begin{aligned}
&log\ p(V|R,T,U,\lambda,\lambda_V) \qquad (5.26)\\
&= log\ p(R|U,V,T,\lambda) + log\ p(V|\lambda_V)\\
&= -\frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{N}(\tau_{ij}^{-1}(r_{ij}-u_i v_j')^2) - \frac{\lambda_V}{2}\sum_{j=1}^{N} v_j v_j' + C
\end{aligned}$$

where U in this formula remains a fixed value; and C is a constant independent of other variables.

In the Maximization step, we maximize $Q(V)$ function by setting the partial derivative of the function to zero w.r.t. v_j for all the $j = 1, \cdots, N$. Let $\Omega = (\langle \tau_{ij}^{-1} \rangle)$ where $\langle \tau_{ij}^{-1} \rangle = \frac{\sqrt{\lambda}}{|r_{ij}-u_i v_j'|}$, we get the following increment formula (Equation 5.27):

$$\begin{aligned}
\Delta v_j &= \frac{\partial Q(V)}{\partial v_j} \qquad (5.27)\\
&= (U' diag(\omega_{:j})U + \lambda_V \boldsymbol{I})^{-1}\ U' diag(\omega_{:j}) r_{:j} + \lambda_V v_j
\end{aligned}$$

where $\omega_{:j}$ and $r_{:j}$ are the jth columns of Ω and R respectively.

Then, we update the parameter V by adding the increment value ΔV for all v_j:

$$v_j = v_j + \eta_v \Delta v_j,\ j = 1, \cdots, N \qquad (5.28)$$

where η_v is a parameter of learning-rate for parameter V.

The Expectation Maximization process for U is similar to that of V, except that it has an additional part of strength propagation (Equation 5.6). The lower bounding function $Q(U)$ is expressed as follows, representing the expectation of the log-posterior of U w.r.t. R and T:

$$Q(U) = E_T[log\ p(U|R, T, V, S, \lambda, \lambda_U, \lambda_S)] \tag{5.29}$$

Consequently, the posterior of U can be expanded as the following Equation 5.30:

$$\begin{aligned} & p(U|R, T, V, S, \lambda, \lambda_U, \lambda_S) \qquad (5.30) \\ & \quad \propto\ p(R|U, V, T, \lambda)\ p(U|0, \lambda_U)\ p(U|S, \lambda_S) \\ & \quad = \prod_{i=1}^{M}\prod_{j=1}^{N}\mathcal{N}(r_{ij}|u_i v_j', \tau_{ij}) \times \prod_{i=1}^{M}\mathcal{N}(u_i|0, \lambda_U) \times \prod_{i=1}^{M}\mathcal{N}(u_i|\sum_{k\in RCG_i} s_{ik}u_k, \lambda_S) \end{aligned}$$

The log form of the above Equation 5.30 is shown as follows (Equation 5.31):

$$\begin{aligned} & log\ p(U|R, T, V, S, \lambda, \lambda_U, \lambda_S) \qquad (5.31) \\ & \quad = log\ p(R|U, V, T, \lambda) + log\ p(U|0, \lambda_U) + log\ p(U|S, \lambda_S) \\ & \quad = -\frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{N}(\tau_{ij}^{-1}(r_{ij} - u_i v_j')^2) - \lambda_U \sum_{i=1}^{M} u_i u_i' \\ & \quad - \lambda_S \sum_{i=1}^{M}((u_i - \sum_{k\in RCG_i} s_{ik}u_k)^T(u_i - \sum_{k\in RCG_i} s_{ik}u_k)) + C \end{aligned}$$

where V in this formula means a fixed variable; and C is a constant independent of other variables.

Using the $\Omega = (\langle \tau_{ij}^{-1} \rangle)$ defined in Equation 5.27, the maximization of $Q(V)$ function for each u_i can be achieved by the following Equation 5.32:

$$\begin{aligned}
\Delta u_i &= \frac{\partial Q(U)}{\partial u_i} \qquad (5.32)\\
&= (V' diag(\omega_i) V + \lambda_U \boldsymbol{I})^{-1} V' diag(\omega_i) r_i + \lambda_U u_i\\
&+ \lambda_S (u_i - \sum_{k \in RCG_i} s_{ik} u_k) - \lambda_S \sum_{k \in RCG_i} s_{ik} (u_k - \sum_{l \in RCG_k} s_{kl} u_l)\\
&= (V' diag(\omega_i) V + \lambda_U \boldsymbol{I})^{-1} V' diag(\omega_i) r_i + \zeta_U u_i\\
&+ 2\ \lambda_S \sum_{k \in RCG_i} s_{ik} u_k + \lambda_S \sum_{k \in RCG_i} s_{ik} \sum_{l \in RCG_k} s_{kl} u_l
\end{aligned}$$

where $\zeta_U = -(\lambda_U + \lambda_S)$; ω_i and r_i are the ith rows of Ω and R respectively.

In the above Equation 5.32, the last item $(\lambda_S \sum_{k \in RCG_i} s_{ik} \sum_{l \in RCG_k} s_{kl} u_l)$ represents the impact of the strength propagation over the co-peers of the direct recommenders for an active user. Observed that this item hardly affect the recommendation performance of the proposed method, we took off this item by using the following simplified form (Equation 5.33) in experiments.

$$\begin{aligned}
\Delta u_i &= \frac{\partial Q(U)}{\partial u_i} \qquad (5.33)\\
&= (V' diag(\omega_{i:}) V + \lambda_U \boldsymbol{I})^{-1} V' diag(\omega_i) r_i + 2\lambda_S \sum_{k \in RCG_i} s_{ik} u_k + \zeta_U u_i
\end{aligned}$$

And finally, we update the parameter U by adding the increment value ΔU for all u_i:

$$u_i = u_i + \eta_u \Delta u_i,\ i = 1, \cdots, M \qquad (5.34)$$

where η_u is a parameter of learning-rate for parameter U.

The above Equations 5.27, 5.28, 5.33 and 5.34 are then used to estimate the latent matrices U and V, which are further multiplied for predict the "missing ratings" in the target matrix R. In the following two sections, we will show the experiments of the proposed *TieRec* method on three aspects: the performance of *TieRec* method comparing with other *MFSR* methods, the impact of Tie Strength, and the feasibility of directly exploiting tie strength in other *MFSR* methods.

5.4 Experiment Settings

In this section, we describe the experiment settings, including experimental datasets, comparing methods and evaluation metrics.

5.4.1 Data Preparation

Two datasets are used in the experiments. One is derived from *Flickr-PASCAL*, and the other from *Epinions.com.* The *Flickr-PASCAL* data is sourced from the study by McAuley and Leskovec [87], outlined in Table . The second dataset used in the experiment is *Epinions*, described in Table 5.2. This dataset is provided by the studies by Konstas [66].

TABLE 5.1: *Flickr-PASCAL* data source and relations

Flickr	Photos	Users	Groups	Tags	Locations	Label	Friends
Source	10,189	8,698	6,951	27,250	1,222	50	N/A
Relations		1,324	562	654	62	42	1324

Table 5.2: *Epinions.com* dataset

Source	user	items	ratings	items/user	friends/user	tags
Epinions.com	40,163	139,738	664,823	16.5	14.3	NA

From the *Flickr-PASCAL* data source, we extract six relationships for experiments: User-group, User-Country, User-Label, User-Friend, Co-Tag and Co-Label. For the *Epinions.com* data source, The values of User-Friend relationship is directly taken from the *Trust* table in the data source. For *Epinions.com*, we generate User-Rating and User-Friend relationships. Using these relationships, we calculate the corresponding tie strengths of the users by employing the *KPMCF* method proposed in previous chapter (Chapter 4). The derived tables and tie strengths are called *Flickr3* and *Epinions3* datasets respectively for the experiments of *TieRec*.

The recommendation task of *Flickr3* dataset is to suggest interesting groups for a new user or a user want to join a new group in the photo-sharing web service Flickr [1], where users are mostly joining several interesting groups such as groups of movies, dancing, outdoor, etc. Likewise, the recommendation task of *Epinions3* is to recommend products for review. *Epinions.com* is such a web site, on which users are encouraged to post reviews on any products for sharing with others [2]. In *Epinions.com*, users are able to assign other people as “trust” relationship in terms of their reviews.

[1]http://www.flickr.com/
[2]http://www.epinions.com/

5.4.2 Methods for Comparison

In the experiments, we employ three *MFSR* methods for evaluation: *SoRec*, *KPMF* and *SR2*. A common feature of these methods is that all methods exploit social information in the factorization process. Nevertheless, these three methods represent three different approaches of *MFSR* methods. *SoRec* takes a *Collective Matrix Factorization (CMF)* approach to factorize two target matrices: one is rating matrix and the other social network matrix. *KPMF* assign kernels as the priors to the latent variables, the kernels are learned from users' interactions. and *SR2* employs regularization techniques in factorization process, in which the social relationships is constrained in regularization terms of the objective function of matrix factorization.

SoRec

Although not explicitly declared, the *SoRec* model proposed by Ma et al. [83] employs a *Collective Matrix Factorization (CMF)* approach. The *SoRec* simultaneously factorizes two target matrices, $User - Item(R)$ and $User - User(C)$. The $User$ entity is jointly participating with both of the two target matrices. The matrix (C) is called a "social network matrix", of which each entry c^*_{ik} represents a combination of local authority and hub values on top of the trust value between user i and user k. The c^*_{ik} is defined as follows:

$$c^*_{ik} = \sqrt{\frac{d^-(k)}{d^+(i) + d^-(k)}} \times c_{ik} \tag{5.35}$$

where $d^+(i)$ represents the out-degree of user i, while $d^-(k)$ indicates the in-degree of user k; and, c_{ik} is a confidence of trust value, provided that it is explicitly stated by user i with respect to user k.

To take the "social network matrix" (c^*_{ik}) into account, the objective function of *SoRec* can be described as follows:

$$\begin{aligned} &\mathcal{L}(R, C, U, V, Z) = \frac{1}{2}\sum\sum I^R_{ij}(r_{ij} - g(u_i^T v_j))^2 \\ &+ \frac{\lambda_C}{2}\sum\sum I^C_{ik}(c^*_{ik} - g(U_i^T Z_k))^2 + \frac{\lambda_U}{2}||U||^2 + \frac{\lambda_V}{2}||V||^2 + \frac{\lambda_Z}{2}||Z||^2 \\ &where\ c^*_{ik} = \sqrt{\frac{d^-(v_k)}{d^+(v_i) + d^-(v_k)}} \times c_{ik} \end{aligned} \tag{5.36}$$

where R and C are user-rating and social network matrices respectively; U, V and Z the corresponding latent matrices of User, Item and Social-factor respectively; $d^+(v_i)$ represents the out-degree of node v_i, while $d^-(v_k)$ indicates the in-degree of node v_k.

KPMF

As kernel methods [55, 114] can be effectively measure the similarities between the nodes in graphs, the *KPMF* model of Zhou et al. [148] leverages kernel methods to incorporate the social graph in the matrix factorization. In particular, the *KPMF* model set the kernel of users' interactions as the priors of latent variables. Owing to the nature of kernel learning, *KPMF* is claimed to be especially useful when dealing with users' interaction information.

The objective function of *KPMF* can be expressed as follows:

$$E = \frac{1}{2\sigma^2}\sum_{i=1}^{N}\sum_{j=1}^{M} I(i,j)(r_{ij} - u_{i:}v_{j:}^T)^2 + \frac{1}{2}\sum_{d=1}^{D} u_{:d}^T K_U^{-1} u_{:d} + \frac{1}{2}\sum_{d=1}^{n_d} v_{:d}^T K_V^{-1} v_{:d} \tag{5.37}$$

where r_{ij} is an element of the target matrix R; $u_{i:}$ and $v_{j:}$ are row vectors of the latent matrices U and V; $u_{:d}$ and $v_{:d}$ represent column vectors of U and V; and K_U and K_V are the kernel matrices of users and items respectively. In particular, K_U is learned from the users' interactions through a Regularized Laplacian kernel [122]. As claimed in the study of [148], such a kernel K_U allows latent variables to be affected by the underlying covariances among users.

SR2

The study by Ma et al. [84] presents a different way of incorporating social information in the recommendation methods. Instead of co-factorizing two matrices (such as *SoRec*) or setting specific priors to latent variables (such as *KPMF*), *SR2* exploits social network information by embedding the information in regularization terms so as to directly constrain the objective function of factorization.

In particular, this study proposed a variety of algorithms: $SR1_{vss}$, $SR1_{pcc}$, $SR2_{vss}$ and $SR2_{pcc}$ by using different regularized schemes and distinct similarities. Of the variants, it was reported that $SR2_{pcc}$ achieved the best performance. The $SR2_{pcc}$ is used in our evaluation, simply called as *SR2*.

In the *SR2* methods, the *Friendship* among users is employed to regularize the *Objective Function.* The algorithm $SR2_{pcc}$ is an individual-based regularization with *Pearson Correlation Coefficient (PCC).* The objective function can be expressed as follows:

$$\mathcal{L}(R,U,V) = \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}(r_{ij} - u_i^T v_j)^2 \tag{5.38}$$

$$+\frac{\beta}{2}\sum_{i=1}^{M}\sum_{f\in\mathcal{F}^+(i)} sim_{if}||u_i - u_f||^2 + \lambda_1||U||^2 + \lambda_2||V||^2$$

$$where\ sim_{if} = \frac{\sum_j (r_{ij} - \overline{r}_i)(r_{fj} - \overline{r}_f)}{\sqrt{(r_{ij} - \overline{r}_i)^2}\sqrt{(r_{fj} - \overline{r}_f)^2}} \tag{5.39}$$

where r_{ij} is an element of the target matrix R; u_i and v_j are row vectors of the latent matrices U and V; I_{ij} is an indicator function; $\mathcal{F}^+(i)$ stands for the set of outlink friends of user i; sim_{if} is the general definition of *PCC*, in which $j \in T(i) \cap T(f)$; and $T(u)$ represents the items rated by the user u.

5.4.3 Evaluation Metrics

Two metrics are used in the evaluation during the experiments: *Root Mean Squared Error* or *RMSE* and *Precision-versus-Recall Curve* or *PRC.* The following two subsections outline these metrics.

Root Mean Squared Error - RMSE

Root Mean Squared Error (RMSE) is used to evaluate the accuracy performance of information retrieval algorithms, including recommendation methods. Intuitively, the *RMSE* measures how much of the differences between the original dataset and the predicted one. A special feature of *RMSE* is that it varies with the variability within the distribution of error magnitudes. In another word, the bigger the error values between the two datasets, the greater the penalization on the *RMSE* result. The following Equation 5.40 is the formula of *RMSE*:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(r_i - \widehat{r_i})^2}{N}} \tag{5.40}$$

where r_i denotes the ground-truth rating value given by a user, $\widehat{r_i}$ denotes its predicted value made by a recommendation method, and N is the total number of the ratings being predicted.

Precision-versus-Recall Curve - PRC

In the evaluation, we also employ *Precision-versus-Recall Curve (PRC)* as an important metric to measure the robustness of evaluation methods. Here *Precision* is defined as the fraction of the retrieved relevant items in the answer set, whereas *Recall* is the fraction of the retrieved relevant items of the whole set of the relevant items [10]. The formula of *Precision-versus-Recall Curve* is described as follows (Equation 5.41):

$$
\begin{aligned}
&Precision = \frac{|R_a|}{|A|} \quad Recall = \frac{|R_a|}{|R|} \\
&and\ Average\ PRC:\ \overline{P}(r) = \sum_{u=1}^{N_u}\sum_{p=1}^{N_p}\frac{P_p^u(r)}{N_p}
\end{aligned}
\tag{5.41}
$$

where A stands for the set of *retrieved items*, R for the whole *relevant items*, and R_a for the intersection of A and R (*retrieved relevant items*). Furthermore, $P_p^u(r)$ is the precision at recall level r for the p-th experiments for user u; N_p and N_u are the numbers of the experiments and the users respectively. As such, $\overline{P}(r)$ represents the *Average PRC* value at recall level r in all N_p experiments and for all N_u users.

To obtain consistent evaluation results for all the methods, we initialized all the latent matrices by an ordinary *SVD* decomposition. All the metrics were measured by the average values of the corresponding evaluation task, and each task ran for 50-100 times based on independently-generated random conditions.

In terms of the hyper-parameters, we set the dimensionality of the latent features to 10 for all the methods over all the datasets. For *TieRec* method, we set σ_V to 1 for both datasets, set σ to 1 for *Flickr3* dataset and 10 for the *Epinions3* dataset. For other methods being compared (*SoRec*, *KPMF* and *SR2*), we set the hyper-parameters to be optimized according to the corresponding studies [83, 84, 148].

5.5 Evaluation

In the experiments, we focus on the robustness of matrix factorization and impact of tie strength to recommendation methods. We conduct three evaluation cases. The first case compares the *TieRec* method with other start-of-the-art Matrix Factorization based Social Recommendation (*MFSR*) methods. The second evaluation case investigates the impact of different levels of tie strength to *TieRec* over the same datasets. The last case studies the feasibility of directly using tie strength as side information in existing *MFSR* methods.

5.5.1 Evaluation with MFSR Methods

The comparison of *TieRec* with other *MFSR* methods is shown in Figures 5.3 and 5.4 for *PRC*, as well as in Table 5.3 for *RMSE*. The *MFSR* methods being compared include *KPMF*, *SoRec* and *SR2*.

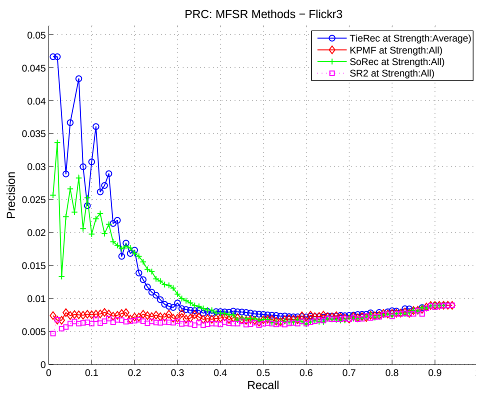


FIGURE 5.3: Comparing MFSR methods (Flickr3)

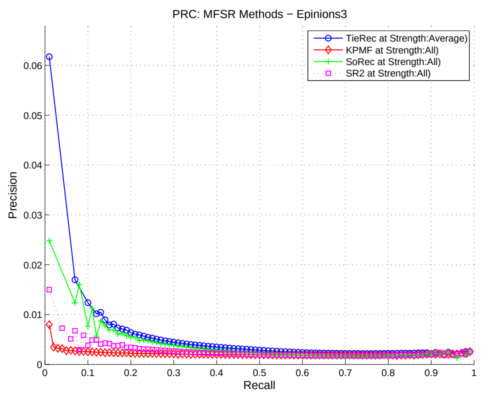


FIGURE 5.4: Comparing MFSR methods (Epinions3)

TABLE 5.3: RMSE: Comparison with MFSR methods

Dataset \ Methods	TieRec	KPMF	SoRec	SR2
Flickr3	0.098437	0.10001	0.10363	0.09958
Epinions3	0.22128	0.23203	0.25336	0.21864

As shown in Figures 5.3 and 5.4, the *PRC* measures of *TieRec* are far superior to other *MFSR* methods for both datasets *Flickr3* and *Epinions3*, especially in the first 0% to 30% recall levels in both figures. In Figures 5.3 for *Flickr3* dataset, the *Precision* values of *TieRec* are about five times higher than those of *KPMF* and *SR2*, and about 30% higher than that of *SoRec*, at all the first 20% recall levels. The *Precision* values of *TieRec* stay higher than others until 30% recall level, and then merge with the other methods after 60% recall level. Figure 5.4 shows a same pattern as that of Figure 5.3 for the *PRC* over *Epinions3* dataset, except that all the *PRC* start to merge from 40% recall level.

Comparing the *RMSE* values of all the methods in Table 5.3, we can see that, *TieRec* is leading all other methods for both datasets (0.098437 for *Flickr3* and 0.22128 for *Epinions3*), except for *SR2* method on *Epinions3* dataset (0.21864).

The above results show that the proposed *TieRec* method not only recommends more relevant items than other *MFSR* methods, but also provides higher relevance than others. The higher *RMSE* values can be considered as the contribution of the integration nature of tie strength, which helps to find other peers with high *homophily* in a social network. The superior *PRC* performance of *TieRec* demonstrates that the robust factorization mechanism embedded in *TieRec* achieves better relevance than other *MFSR* methods.

5.5.2 Impact of Tie Strengths

We investigate the impact of tie strength to the *TieRec* method at different strength levels. To this end, we run the experiments for *TieRec* method at three levels: High, Medium and Low. When applying “High” strength level to recommendation tasks, we select those users, of whom the strength values were higher than 70% of the whole range. Accordingly, the “Medium” level was between 30% and 70%, and the “Low” level was lower than 30%.

Figures 5.5 and 5.6 show that the *PRC* curves of both High and Medium strength levels outperform those of Lowstrength level for all datasets. For both datasets, the *PRC* curves of High strength level start about two or three times higher than those of Low strength level. In particular, the *PRC* curve of of High strength level remains higher than that of Low strength level until the recall level reaches 60% for *Flickr3* dataset, until 30% for *Epinions3* dataset. In the meanwhile, all the *PRC* curves of Medium strength level perform very close with those of High strength level.

The corresponding *RMSE* values are shown in Table 5.4. The *RMSE* values of High strength level are the lowest for both datasets, while those of the Low strength level perform the worst (0.09666 against 0.097909 for *Flickr3*, and 0.21084 against 0.25104 for *Epinions3*). This phenomenon coincides with the *PRC* curves in Figures 5.5 and 5.6. This observation indicates that the users with strong tie strengths will mostly contribute more relevant and more accurate recommendations.

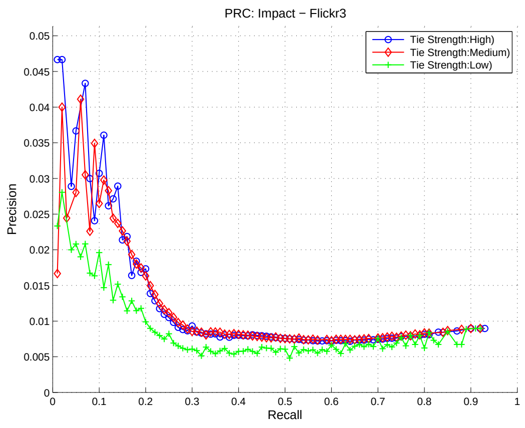


FIGURE 5.5: Impact of tie strength (Flickr3)

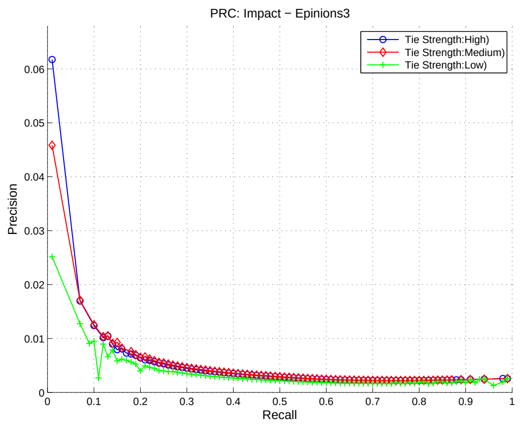


FIGURE 5.6: Impact of tie strength (Epinions3)

TABLE 5.4: RMSE: Impact of tie strength

Dataset \ Strength level	High	Medium	Low
Flickr3	0.09666	0.097723	0.097909
Epinions3	0.21084	0.25090	0.25104

5.5.3 Tie Strength as Side Information

In the experiments, we also study the feasibility of directly using tie strength as side information in existing *MFSR* methods, We conduct a series of experiments for *KPMF*, *SoRec* and *SR2* method. In the evaluation, we run each method on three conditions. The first condition is using tie strength at High strength level. The second condition is at Low strength level. The last condition is running a method using general social relationships ("Friend" in *Flickr3* and "Trust" in *Epinions3*). These three conditions are denoted as "High", "Low" and "No" respectively.

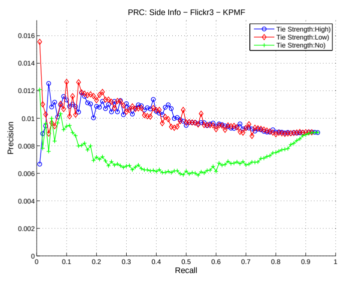


FIGURE 5.7: Flickr3-KPMF

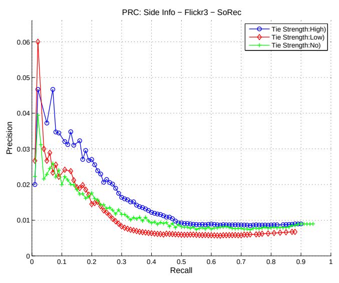


FIGURE 5.8: Flickr3-SoRec

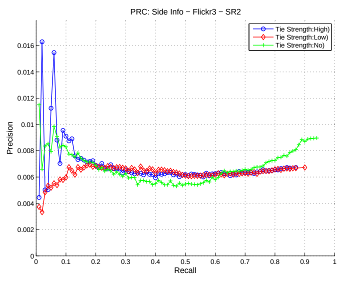


FIGURE 5.9: Flickr3-SR2

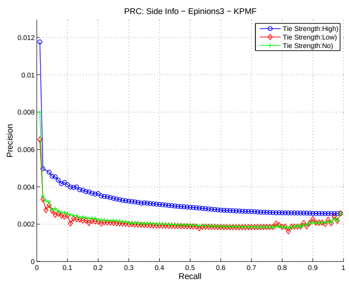


FIGURE 5.10: Epinions3-KPMF

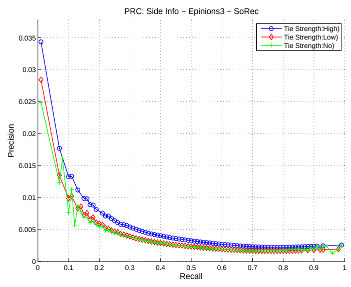


FIGURE 5.11: Epinions3-SoRec

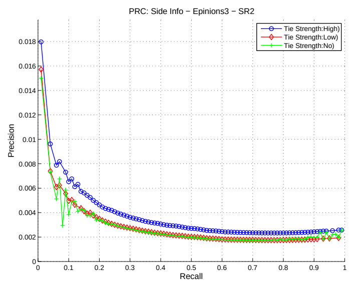


FIGURE 5.12: Epinions3-SR2

TABLE 5.5: RMSE: Tie strength as side information

Dataset \ Method	High	Low	No
Flickr3			
KPMF	0.10087	0.10184	0.10001
SoRec	0.096926	0.11452	0.10363
SR2	0.097707	0.10361	0.09958
Epinions3			
KPMF	0.22454	0.23619	0.23203
SoRec	0.24195	0.25062	0.25336
SR2	0.20958	0.2188	0.21864

Figures 5.7, 5.8 and 5.9 show the results of the evaluation for *Flickr3* dataset, while Figures 5.10, 5.11 and 5.12 for *Epinions* dataset. As shown in these figures, on the whole, the *PRC* curves using High strength level of tie strength are higher than those using primary social relationship at most recall levels, except that of *Flickr3-SR2*. However, look at the *PRC* of *Flickr3-SR2*, the *PRC* curve of "High" is about two times higher than that of 'No' until the recall level of 10%. This observation confirms that the tie strength learned from *KPMCF* method does reflect the users' overall attitudes towards other peers in a social network. These tie strengths are helpful for making recommendations.

It is interesting to note that, in most cases except for *Flickr3-KPMF*, the *PRC* curves using "Low" strength level of tie strength perform the same or even worse than those of "No". This also demonstrates that the tie strength values are very sensitive in applications. A “Low” level of tie strength reflects a weak relationship between two persons. In the applications of recommendation, such a relationship may play a downgrade role. This observation motivates us to further investigate the applicability of the tie strengths for various applications.

The *RMSE* values for all evaluation cases are shown in Table 5.5. All in all, the trends of these *RMSE* values coincide with the *PRC* curves in above figures. That is, the *RMSE* values derived from the methods using High strength level of tie strength mostly stay at the leading positions, except that of *Flickr3-KPMF*. In the meantime, most of the *RMSE* values of “Low” level are slightly worse than those of “No”.

5.6 Comparison and Summary

In this section, we sum up our proposed *TieRec* method by comparing it with other Matrix Factorization based Social Recommendation or *MFSR* methods.

A common approach taken by *MFSR* methods is to treat social connections and relationships as side information along with the user-item rating matrix [137]. From technology perspective, *MFSR* methods can be categorized into three groups: *Social Ensemble* [57, 82, 84, 102, 138], *Kernelized Matrix Factorization* [4, 148], and *Collective Matrix Factorization* [35, 78, 83, 118, 119, 139].

A common rationality held by the studies of *Social Ensemble* is that, a user’s observed ratings represent the preferences of not only the user’s own but also of his/her friends. For this reason, the study by Ma et al. [82] models the distribution of a user's observed ratings conditioned on a linear combination of the favors of all the friends of the user. Based on the same idea, the study by Jamali and Ester [57] takes trust propagation into account, and the study of Yang et al. [138] makes use of "Trust Circles". Our experiments have demonstrated that, the *SR2* proposed by Ma et al. [84] exhibits quite higher accuracy measures (*RMSE*) than other methods. However, it shows lower relevance in terms of *PRC* measures.

Collective Matrix Factorization (CMF), or Matrix Co-Factorization in certain literature, has received great attention in last few years. Especially, the studies of [35, 56, 78, 83, 119, 139] contribute to *MFSR* methods. The basic idea of *CMF* is the multiple latent features will be twistingly affected by each other when being simultaneously factorized. As shown in experiments, in comparison with other *MFSR* methods, the *SoRec* method proposed by Ma et al. [83] demonstrates outstanding relevance of *PRC* values but disappointing accuracy of *RMSE* measures.

Certain recent studies employ kernel techniques in matrix factorization to improve recommendation qualities. For example, the *PMA* model proposed by Adams et al. [2] and the *KPMF* in the study by Zhou et al. [148]. A common approach of these kernelized methods is to assign kernels to the priors of latent variables. This should be a promising approach to incorporate users' social information in recommendation process. Unfortunately, in the experiments of *KPMF*, we find that the *KPMF* model always stays at the middle positions in all the evaluation cases for either accuracy and relevance measures.

In comparison with the aforementioned *MFSR* methods, the experiments demonstrate that the proposed *TieRec* method achieves superior accuracy of *RMSE* measures to nearly all other methods in all but one evaluation cases, in the meantime it also leads far higher relevance of *PRC* than all other methods in all cases.

There are a number of solutions for robust matrix factorization [1, 72, 133]. The *TieRec* method is partially motivated by the *PRMF* model in the study of Wang et al. [133], which uses L1 LOSS to perform a probabilistic matrix factorization. While *PRMF* is a general matrix factorization on single target matrix, the *TieRec* method incorporates users' tie strengths in the factorization process. Because *PRMF* and *TieRec* work on different datasets, it is difficult to directly compare the performance of them.

It is worth noting that, although the *TieRec* method incorporates tie strength into recommendation by employing social ensemble approach, the tie strength used in the method is a static measurement. As defined in section 5.2.2 and 5.3.2, the matrix $S \in \mathbb{R}^{M \times M}$ does represent the tie strengths among the users in question. However, such a relation represents the tie strength only during a particular time frame, but it does not include the dynamic features of tie strength where the social information are evolving over time. We will improve this issue in our future work.

Recapping the discussion in this chapter, we develop a matrix factorization method for social recommendation, that incorporates users' tie strength into factorization process. In the meanwhile, the *TieRec* method employs L1 LOSS technique to increase the robustness of the method. We have conducted comprehensive experiments by using real datasets from popular social media services. The evaluation demonstrates the *TieRec* outperforms other state-of-the-art *MFSR* methods. The experiments also prove that tie strength does make significant impact to the recommendation qualities of *MFSR* methods.

Chapter 6

Conclusion and Future Work

6.1 Concluding Remarks

In this thesis, we propose a peer-based social relationship enhanced recommendation model, which especially fits in the situations where users are looking for recommendations from their peers through social circles.

Firstly, a *Regression-based* filtering method, named as *CoRec*, is designed to ensure that all the co-peers in a neighborhood participate in the recommendation processes by propagating requests and responses. This filtering method produces a twofold result. On one hand, the feedback from the peers can be considered as an initial recommendation to be directly accepted or to be further refined. On the other hand, the filtering method forms a reasonably small-sized neighborhood for each active user, of which all the co-peers are have great overlap with the user.

The information of these co-peers can also be used in other applications such as social relationship analysis.

Also, a probabilistic method is developed to measure the tie strengths among the users in a social network. The proposed method, named as *KPMCF*, employs *Collective Matrix Factorization* and *Kernel Learning* techniques to simultaneously capture users' profile characteristics and social interaction behaviors. This measuring method has three advantageous features: (i) Tie strength is measured by making use of latent features, which are learned from users' demographic and social interaction information; (ii) The learning process utilizes *Collective Matrix Factorization* to deal with the diverse forms of various relationships; (iii) Users' interaction information is captured by *Kernel Learning* techniques.

Utilizing the achievement in the above methods, we develop a robust matrix factorization method *TieRec*, that incorporates tie strength into recommendation process. The research proceeds in two directions: one is to explore the techniques of incorporating tie strength in matrix factorization process, and the other is to develop the mechanism of robust matrix factorization.

Our proposed *Peer-based Social Relationship Enhanced Recommendation Model* consists of three parts: *CoRec*, *KPMCF* and *TieRec*. These three parts can either work collaboratively to form an integrated solution, or be used independently in individual applications.

We have conducted comprehensive experiment and evaluation for all the methods of our model. Our experiments use the real datasets from popular social media services *Last.fm*, *Epinions.com* and *Flickr*. The recommendation methods being compared include two popular rating-based *CF* methods, i.e., *Pearson CF* and *Slope One*, and three state-of-the-art *MFSR* methods, i.e., *SoRec*, *SR2* and *KPMF*.

The experiments show that, the proposed model outperforms all other methods in nearly all evaluation cases in terms of widely accepted metrics: *Mean Absolute Error (MAE)*, *Root Mean Squared Error (RMSE)* and *Precision-versus-Recall Curve (PRC)*. The experiments also reveal that tie strengths do make significant contributions to recommendation qualities. Generally speaking, the stronger the tie strengths among users, the more accurate and relevant the recommendation results.

6.2 Future Work

The proposed recommendation model addresses the issues of making personalized recommendations, in the conditions where people are seeking for recommendations through social circles. The study of this thesis also raises a number of questions or issues for further work.

First, the concept of *Common-Interested-Items (CII)* helps to construct dynamic and reasonably small-sized neighborhoods. The current definition of *CII* makes use of users' ratings only. The subsequent question is: how shall we include other explicit or implicit social information in the definition of *CII*, for instance, Tags or Trustworthiness ?

Next, the proposed *KPMCF* method uses collective matrix factorization techniques to measure users' tie strength, that involves not only users' comprehensive information but also a number of parameters and hyper-parameters. To improve the quality of learning process, parameter-optimization becomes a key. A number of studies have been reported to address this issue for instance Bayesian Inference [97, 109]. Applying parameter-optimization methods to the proposed model has been listed in our to-do list.

Furthermore, as social networking has become one of the most active parts of our real life, the study on tie strength needs to challenge the dynamic social relationships where people's connections and interactions are evolving over time. In order to support the temporal features of tie strength, we will study the incorporation of the real-time evolution of tie strength into recommender systems and other various applications.

Besides the above issues, our experiments have indicated that different combinations of the profile and interaction relationships resulted in varied recommendation results. It will be a challenging task to quantitatively and qualitatively analyze the relationships between tie strength and various social factors in the context of social recommendations.

On the whole, the research of this thesis employs a variety of state-of-the-art techniques. All of them are well-established, and are continuously and actively deploying in a wide range of research fields. Following up these techniques and exploring new study topics are within our long-term research interests.

Appendix A

Evaluation System

During the research work for this thesis, we developed an evaluation system for experiment and evaluation, especially for evaluating the impacts and benefits brought by tie strength. The system is constructed in *MatLab* environment [1] with *MongoDB* database [2]. The evaluation system, named as *EvTie* was originally used for setting parameters when learning tie strength, and later on evolved into a systematic tool with comprehensive functionality such as accessing the database for retrieving raw data, learning tie strength from multiple relations, generating recommendations for selected state-of-the-art algorithms, evaluating algorithm performance, showing graphic plots for various metrics, reviewing the statistics of raw data, and so on.

Owing to its rich functions, the *EvTie* system can be used for not only internal experiment and evaluation but also for further development for public use. This

[1]http://www.mathworks.com.au/products/matlab/, last access on 17/03/2014
[2]https://www.mongodb.org/, last access on 17/03/2014

appendix provides the readers with highlights of the system, including a system overview, system modules, graphic user interfaces, and various evaluation scenarios. Interested readers can contact the author of the thesis for further details.

A.1 System Overview

A.1.1 Main Modules

Figure A.1 shows the main modules of the *EvTie* system.

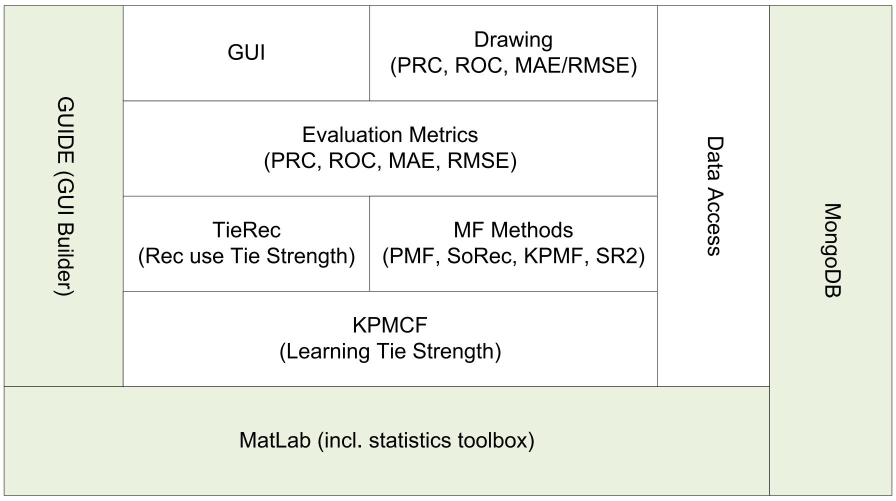

GUIDE (GUI Builder)
GUI
Drawing (PRC, ROC, MAE/RMSE)
Evaluation Metrics (PRC, ROC, MAE, RMSE)
TieRec (Rec use Tie Strength)
MF Methods (PMF, SoRec, KPMF, SR2)
KPMCF (Learning Tie Strength)
Data Access
MongoDB
MatLab (incl. statistics toolbox)

FIGURE A.1: Main Modules of evaluation system

As shown, the evaluation system is constructed in *MatLab* environment and with a *MongoDB* database, and several openly published *MatLab* and *MongoDB* related programs [3]. The main modules include:

[3]We are grateful to the corresponding authors of the following MatLab and MongoDB related programs, which are either partially or totally used in the system: (a) Mongo-MatLab-Driver, (b) prec_rec, (c) colAUC, (d) octave_pdist, (e) simpletab, (f) RoboMongo.

- *Data access* - Accessing raw data and retrieving recommendation and evaluation results;
- *KPMCF* - Learning tie strength from multiple relations;
- *TieRec* - making recommendation using Tie Strength;
- *MF Methods* - Implementation of selected social recommendation methods: PMF, SoRec, KPMF and SR2;
- *Evaluation Metrics* - Evaluating metrics, including PRC, ROC, MAE and RMSE;
- *GUI* - Interactive graphic user interfaces built up on top of GUIDE (GUI Builder);
- *Drawing* - Tools to draw PRC and ROC curves, with MAE/RMSE tables in plots.

A.1.2 Functionality

"$MATLAB^{©}$ is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MatLab, you can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in mathematical functions enable you to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or $Java^{TM}$"
(quoted from http://www.mathworks.com.au/products/matlab/).

Nearly all the codes are written by *MatLab*, with the relevant toolboxes such as *Statistics Toolbox* and *GUIDE (GUI Builder)*. The main functionality of the *EvTie* system is listed in Table A.1.

TABLE A.1: Functionality of evaluation system

MODULE	FUNCTIONALITY	FUNCTIONALITY
Learning Tie Strength		
	Settings	
		Select dataset
		Select entities
		Select parameters
	Learn tie strength	
	Recommend by TieRec	
Recommend Other Methods		
	Settings	
		Select dataset
		Select entities
		Select parameters
	Recommend by MF	
		PMF
		SoRec
		KPMF
		SR2
Evaluation		
	Settings	
		Select MF methods
		Select dataset
		Select entities
		Select parameters
	Evaluation TieRec	
	Evaluation MF	
		PMF
		SoRec
		KPMF
		SR2
	Evaluation metrics	
		PRC
		ROC
		MAE
		RMSE
Database Access		
	Save raw data	
	Retrieve raw data	
	Save recommendations	
	Retrieve recommendations	
	Save evaluation	
	Retrieve evaluation	
Drawing		
	PRC	
	ROC	
	MAE/RMSE Table	
Utilities		

A.1.3 Database

MongoDB is one of the popular NoSql database [52, 90, 130], which is "an open-source document database that provides high performance, high availability, and automatic scaling"
(quoted from https://www.mongodb.org/).

There were several reasons why we selected *MongoDB* for persistent data store in our research, and especially for the *EvTie* system. First, because *MongoDB* is a NoSql database, there is no need to define a schema before accessing data. Using this feature, we are able to easily change the data formats for research purpose; for instance, the evaluation data with gradually added metrics values.

Second, we can access *MongoDB* by directly using *MatLab* variables - especially vectors and matrices, as if we were using the cache of the *MatLab* environment. And finally, the data we were dealing with did not have complex inter-relations. Therefore, no complicated aggregation is needed. In a word, *MongoDB* brings us a simple and efficient way to handle persistent data.

In MongoDB, the database is organized according to the following rules:

- A database holds collections;
- A collection holds documents;
- A document is a set of fields;
- A field is a Key-Value pair;

- A Key is a name of string, and a Value is either a basic type value, or a document, or an array of values.

Table A.2 outlines the data store information of *EvTie* down to the level of *document*; the details of the fields are basically omitted.

Table A.2: Data store for evaluation system

DATABASE	COLLECTION	DOCUMENT	(FIELDS)
Data			
	flickr		
		prof_group	
		prof_location	
		prof_label	
		prof_tag	
		user_friend	
		co_tag	
		co_label	
		co_group	
	epinions		
		user_rating	
		usertrust	
	lastfm		
		prof_track	
		prof_tag	
		user_friend	
		co_tag	
Evaluation			
	prc_mf		
		prc_xml	
	prc_wt		
		prc_xml	
	roc		
		roc_xml	
	params		
		strength_default	
		strength_previous	
		recommend_default	
		recommend_previous	

A.2 Graphic User Interface

To develop a successful algorithm, the development tasks involve four aspects: (i) the program of the algorithm; (ii) the various combinations of the parameters; (iii) the various datasets to be tested and evaluated; and (iv) the visual presentation of the metrics measurement (some metrics such as MAE and RMSE do not need visual illustration, however, other metrics such as PRC and ROC do need them). It is the graphic user interface which integrates all these four aspects in one place, in an interactive fashion. In this evaluation system, all the graphic user interfaces are grouped into two windows: *Learning* and *Evaluation*, which will be discussed in the following two subsections.

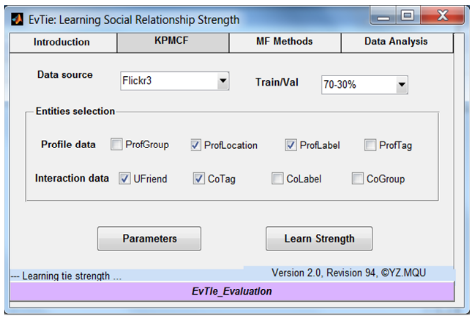
A.2.1 Learning Parameters

The first part of the GUI is to learn tie strength and to make recommendation by various methods. The Learning window can be further divided into four groups: Dashboard, KPMCF, TieRec and MF Methods.

Dashboard

There are three dashboards, each representing a type of functionality. The dashboards are presented to the user by clicking the corresponding TAB. As the names indicate, the *KPMCF* dashboard (Figure A.2) includes the functions of learning tie strength by using the *KPMCF* method, the *MF Methods* (Figure A.3) helps to make recommendations from select matrix factorization (*MF*) methods, and finally the *Data Analysis* (Figure A.4) is used to analyze the statistical characteristics of the datasets to be tested.

Figure A.5 is a pop-up view, used to adjust the parameters used in corresponding conditions. Every change in the parameters will be saved in the database in order to be used in subsequent executions.



Data source: Flickr3

Train/Val: 70-30%

Entities selection

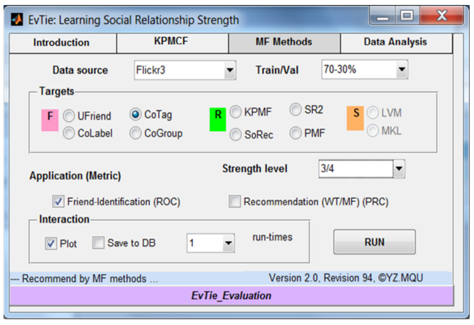
Profile data: ☐ ProfGroup ☑ ProfLocation ☑ ProfLabel ☐ ProfTag

Interaction data: ☑ UFriend ☑ CoTag ☐ CoLabel ☐ CoGroup

--- Learning tie strength ...

Version 2.0, Revision 94, ©YZ.MQU

FIGURE A.2: Learn tie strength



Data source: Flickr3

Train/Val: 70-30%

Targets

F: UFriend [✓ CoTag] CoLabel CoGroup

R: KPMF SR2 SoRec PMF

S: LVM MKL

Strength level: 3/4

Application (Metric)

☑ Friend-Identification (ROC) ☐ Recommendation (WT/MF) (PRC)

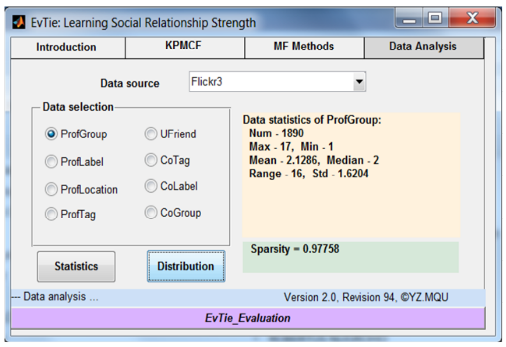
Interaction

☑ Plot ☐ Save to DB 1 run-times

--- Recommend by MF methods ...

Version 2.0, Revision 94, ©YZ.MQU

FIGURE A.3: Recommend by matrix factorization methods



Data source: Flickr3

Data selection

[✓ ProfGroup] UFriend ProfLabel CoTag ProfLocation CoLabel ProfTag CoGroup

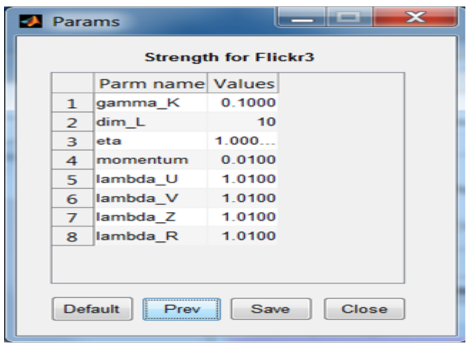
Data statistics of ProfGroup:
Num - 1890
Max - 17, Min - 1
Mean - 2.1286, Median - 2
Range - 16, Std - 1.6204

Sparsity = 0.97758

--- Data analysis ...

Version 2.0, Revision 94, ©YZ.MQU

FIGURE A.4: Data analysis



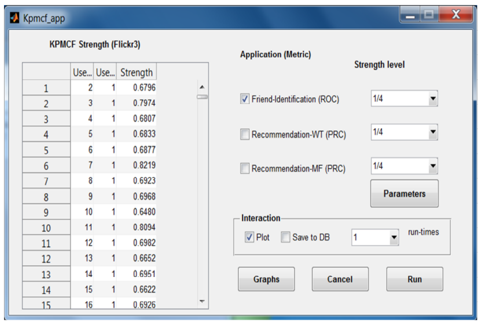
Strength for Flickr3

	Parm name	Values
1	gamma_K	0.1000
2	dim_L	10
3	eta	1.000...
4	momentum	0.0100
5	lambda_U	1.0100
6	lambda_V	1.0100
7	lambda_Z	1.0100
8	lambda_R	1.0100

FIGURE A.5: Parameters for learning method

KPMCF

From the above *KPMCF* dashboard (Figure A.2), the tie strength can be inferred according to special settings in the dashboard. The *kpmcf-app* (Figure A.6) window is a start point to apply tie strength to various applications. In the *kpmcf-app* window, the left-hand image is a list of the learned tie strength, and the right-hand image shows a number of action buttons. If the *Graph* button is selected, then a network graph of the tie strength among all the users will be shown in a pop-up window (Figure A.7). In the network graph window, the strength level can be changed by a slice bar, and different evaluation results appear. In this evaluation system, four evaluation metrics are implemented: *PRC* (section 3.4.3), *ROC* (section 4.5.2), and *MAE* and *RMSE* (section 3.4.3).


Kpmcf_app

KPMCF Strength (Flickr3)

	Use...	Use...	Strength
1	2	1	0.6796
2	3	1	0.7974
3	4	1	0.6807
4	5	1	0.6833
5	6	1	0.6877
6	7	1	0.8219
7	8	1	0.6923
8	9	1	0.6968
9	10	1	0.6480
10	11	1	0.8094
11	12	1	0.6982
12	13	1	0.6652
13	14	1	0.6951
14	15	1	0.6622
15	16	1	0.6926

Application (Metric) — Strength level

[✓] Friend-Identification (ROC) 1/4

[ ] Recommendation-WT (PRC) 1/4

[ ] Recommendation-MF (PRC) 1/4

Interaction: [✓] Plot [ ] Save to DB 1 run-times

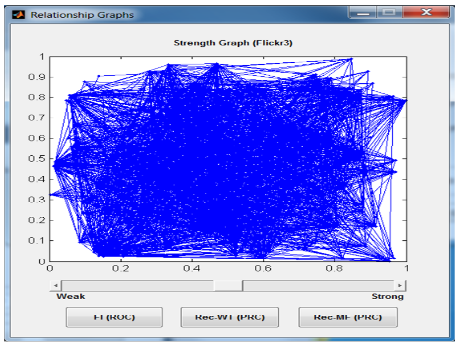


Figure A.6: KPMCF-APP Figure A.7: KPMCF-Graph

TieRec

Figures A.8 and A.9 show the performance measurement for applying multiple metrics to the recommendations made by TieRec, which incorporates the learned tie strength in the matrix factorization. In particular, Figure A.8 shows the metrics values of *PRC* and *RMSE*, and Figure A.8 shows that of *ROC*. It is the visual performance measurement which helps us to investigate and improve our algorithms.

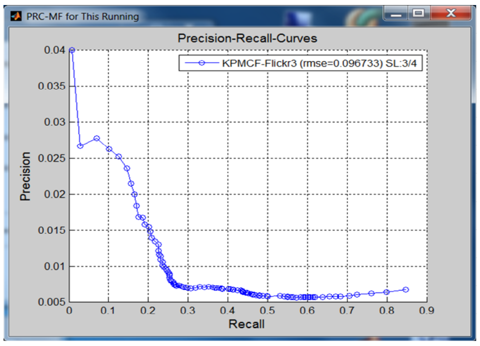


FIGURE A.8: TieRec-PRC

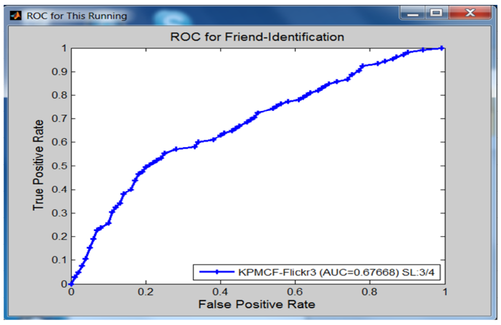


FIGURE A.9: TieRec-ROC

MF Methods

Similar to the above two figures, Figures A.10 and A.11 are the performance measurements for applying metrics to other *MF* methods. Used as two examples, Figure A.10 shows the *PRC* and *RMSE* values for the recommendations made by *SoRec*, and Figure A.11 shows *ROC* for those by *KPMF*.

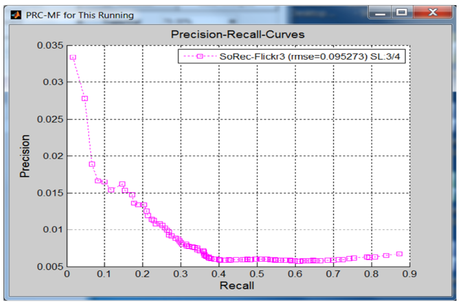


FIGURE A.10: SoRec-PRC

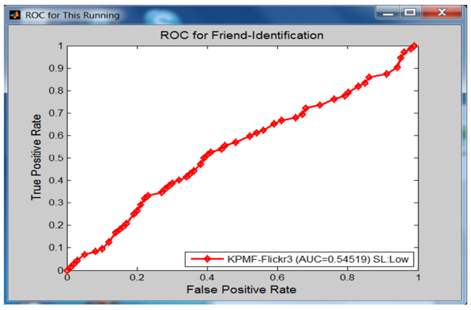


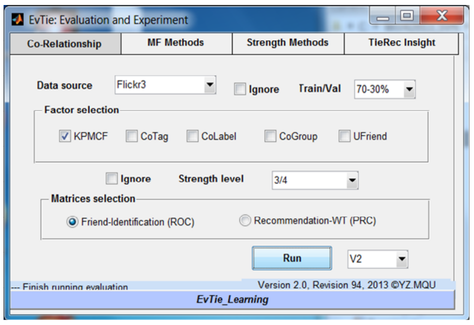
FIGURE A.11: KPMF-ROC

A.2.2 Evaluation

While the *Learning* part of the GUI basically works for a single execution of the algorithms under a specific condition, the second part of the GUI is used to evaluate the performance of the algorithms by analyzing the experiment data stored in the database. The experiment data can be accumulated execution multiples in the tens or hundreds. The evaluation can also be done in a headless fashion (refer to *headless functions* in section A.3.2).

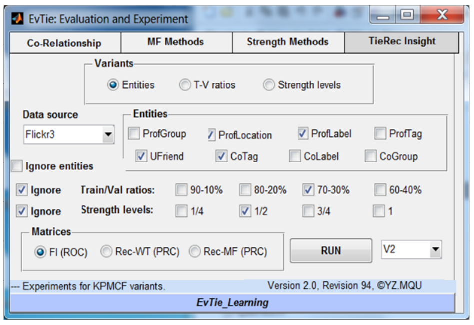
Dashboard

In this part, we provide three types of evaluations: evaluating tie strength against various collaborative relationships (Figure A.12); investigating the insight of the *TieRec* with various settings (Figure A.13); and comparing *TieRec* with certain state-of-the-art *MF* methods (*PMF*, *Sorec*, *KPMF*, *SR2*) under various conditions (Figure A.14).



Data source: Flickr3
Ignore [ ] Train/Val: 70-30%
Factor selection: [✓ KPMCF] CoTag CoLabel CoGroup UFriend
Ignore [ ] Strength level: 3/4
Matrices selection: [✓ Friend-Identification (ROC)] Recommendation-WT (PRC)
V2
--- Finish running evaluation
Version 2.0, Revision 94, 2013 ©YZ.MQU

FIGURE A.12: Evaluate Tie Strength

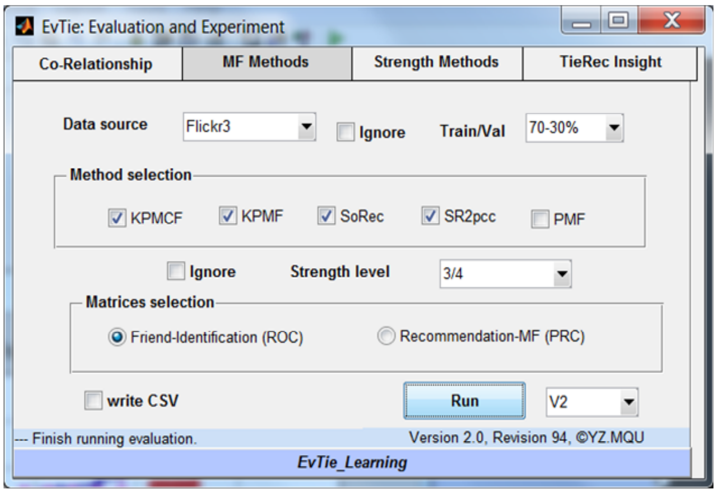


Variants: [✓ Entities] T-V ratios Strength levels
Data source: Flickr3
Ignore entities [ ]
Entities: ProfGroup [✓ ProfLocation] [✓ ProfLabel] ProfTag [✓ UFriend] [✓ CoTag] CoLabel CoGroup
[✓ Ignore] Train/Val ratios: 90-10% 80-20% [✓ 70-30%] 60-40%
[✓ Ignore] Strength levels: 1/4 [✓ 1/2] 3/4 1
Matrices: [✓ FI (ROC)] Rec-WT (PRC) Rec-MF (PRC)
V2
--- Experiments for KPMCF variants.
Version 2.0, Revision 94, ©YZ.MQU

FIGURE A.13: Insight of TieRec

Insight

Figure A.15 and Figure A.16 show the impact of tie strength on the recommendations by *TieRec* at different strength levels, for *PRC* and *ROC* measures respectively. That helps us to acquire a comprehensive insight into the *TieRec* model.



Data source: Flickr3 | Ignore [ ] | Train/Val: 70-30%

Method selection: KPMCF [✓] KPMF [✓] SoRec [✓] SR2pcc [✓] PMF [ ]

Ignore [ ] | Strength level: 3/4

Matrices selection: [✓ Friend-Identification (ROC)] Recommendation-MF (PRC)

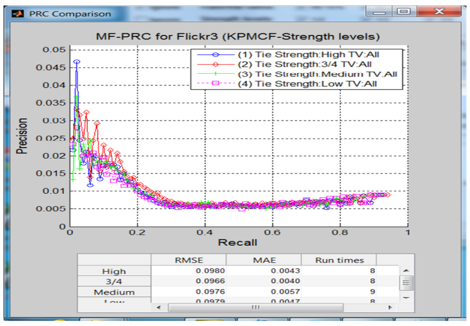
write CSV [ ] | V2

--- Finish running evaluation. | Version 2.0, Revision 94, ©YZ.MQU

EvTie_Learning

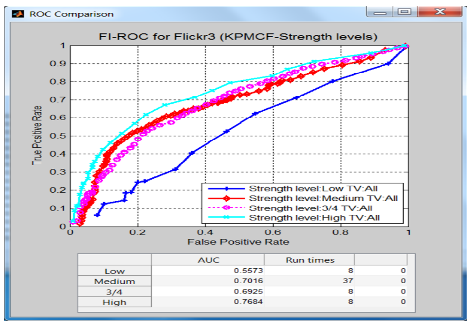
FIGURE A.14: Evaluate matrix factorization methods

As the figures show, all the *PRC* curves, *MAE* and *RMSE* values are listed in the figure window.



	RMSE	MAE	Run times
High	0.0980	0.0043	8
3/4	0.0966	0.0040	8
Medium	0.0976	0.0057	9
Low	0.0979	0.0047	8

FIGURE A.15: Insight - PRC

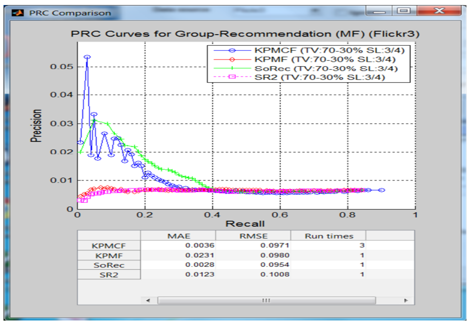


	AUC	Run times	
Low	0.5573	8	0
Medium	0.7016	37	0
3/4	0.6925	8	0
High	0.7684	8	0

FIGURE A.16: Insight - ROC

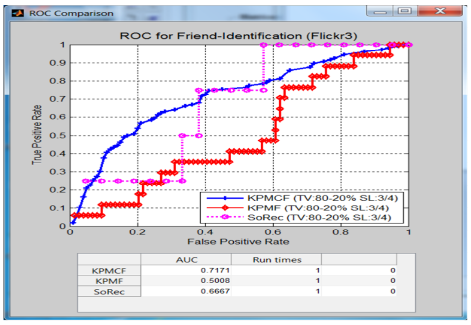
Comparison

Figures A.17 and A.18 show the comparison between *TieRec* and other *MF* methods, by applying the same datasets to these methods. Figure A.17 illustrates four *PRC* curves for corresponding methods on the *Flickr3* data set, with a training-validation-rate of 70-30% and strength level of 3/4. And, Figure A.18 shows *ROC* curves on the same *Flickr3* dataset, but with a training-validation-rate of 80-20% and strength level of 3/4.



	MAE	RMSE	Run times	
KPMCF	0.0036	0.0971	3	
KPMF	0.0231	0.0980	1	
SoRec	0.0028	0.0954	1	
SR2	0.0123	0.1008	1	

FIGURE A.17: PRC - matrix factorization Methods



	AUC	Run times	
KPMCF	0.7171	1	0
KPMF	0.5008	1	0
SoRec	0.6667	1	0

FIGURE A.18: ROC - matrix factorization Methods

A.3 Key Functions

A.3.1 Algorithm Related Functions

The core part of the evaluation system is a variety of algorithms to learn tie strength and to make recommendation by various algorithms. In particular, these algorithms include *KPMCF* (Chapter 4), *SoRec* [83], *KPMF* [148] [4], *SR2* [84], *PMF* [110] [5], and *TieRec* (Chapter 5). Table A.3 lists the major functions of these algorithms.

[4]The *KPMF* related functions are derived from the authors' coding, refer to http://www.eecs.berkeley.edu/ tinghuiz/code/kpmf.zip, last access on 10/07/2013

[5]The *PMF* related functions are derived from the authors' coding, refer to http://www.cs.toronto.edu/ rsalakhu/BPMF.html, last access on 10/07/2013

Table A.3: Tie strength and recommendation related functions

ALGORITHM	DESCRIPTION
run_kpmcf_evaluate(...)	set up dataset and parameters for learning tie strength.
make_kernel(...)	make kernels from social interactions.
kpmcf_fact(...)	perform matrix co-factorization on multiple profile matrices.
learn_strength(...)	learn tie strength from inferred latent matrix of users.
run_tierec_evaluate(...)	set up dataset and parameters for invoking gd_tierec().
gd_tierec(...)	gradient descent algorithm of TieRec.
run_pmf_evaluate(...)	set up dataset and parameters for invoking gd_pmf().
gd_pmf(...)	gradient descent algorithm of pmf.
run_kpmf_evaluate(...)	set up dataset and parameters for invoking gd_kpmf() or sgd_kpmf().
gd_kpmf(...)	Gradient descent algorithm of KPMF.
sgd_kpmf(...)	Stochastic Gradient Descent algorithm of KPMF.
run_sorec_evaluate(...)	set up dataset and parameters for invoking gd_sorec().
gd_sorec(...)	Gradient descent algorithm of SoRec.
run_sr2_evaluate(...)	set up dataset and parameters for invoking gd_sr2().
gd_sr2(...)	Gradient descent algorithm of SR2.

A.3.2 Headless Functions

While the above GUIs are developed for investigating the algorithms and parameters, the "headless" functions listed in Table A.4 are used to repeatedly and randomly run given datasets, so as to obtain statistical results of the recommendation performance. Usually, we run each case more than 50 times. One case means a particular setting for a particular dataset and method. For example, one case might be running a recommendation by using *TieRec* at a *training-validation-rate* 80-20% and with strength level *High*. The execution results will be saved to the *MongoDB* database, such that we are able to retrieve this data later on by means of statistics functions (section A.3.3).

Table A.4: Headless functions for randomly running

FUNCTION	DESCRIPTION
run_headless(iteration)	a delegation of individual headless functions.
run_headless_factors(dataset, train_val_rate)	for various collaborative relationships.
run_headless_kpmcf(dataset, train_val_rate)	for various settings of TieRec.
run_headless_methods(dataset, train_val_rate)	for various settings of MF methods.

A.3.3 Statistics Functions

There are number of statistics functions, listed in Table A.5, used to export *PRC* or *ROC* curves for evaluation and reporting purposes. These functions retrieve data from the database, and run statistical calculations on a needs basis. Usually, the data is saved when performing headless functions (section A.3.2). These statistics functions decompose the XML data and calculate the average *PRC* or *ROC* values (section 3.4.3), and finally produce corresponding *PRC* or *ROC* curves and MAE/RMSE values. All the figures from Figure 5.3 to Figure 5.6 in Chapter 5 are produced from these export functions. Most of the figures in this thesis are directly produced by these statistics functions.

TABLE A.5: Statistics functions for performance evaluation

FUNCTION	DESCRIPTION
export_prc_methods(dataset)	export PRC curves for MF methods.
export_prc_impact(dataset)	export PRC curves of various settings.
export_prc_sideinfo(method,dataset)	export PRC using tie strength in an MF method.
export_prc_factors(dataset)	export PRC curves of various relationships.
export_roc_factors(dataset)	export ROC curves of various relationships.

Bibliography

[1] H. Aanæs, R. Fisker, K. Astrom, and J.M. Carstensen. Robust factorization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(9):1215–1225, 2002.

[2] R.P. Adams, G.E. Dahl, and I. Murray. Incorporating side information in probabilistic matrix factorization with gaussian processes. *arXiv preprint arXiv:1003.4944*, 2010.

[3] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23(1):103–145, 2005.

[4] A. Agovic, A. Banerjee, and S. Chatterjee. Probabilistic matrix addition. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, 2011.

[5] K. Ali and van Stam Van W. Tivo: making show recommendations using a distributed collaborative filtering architecture. In *Proceedings of the tenth*

ACM SIGKDD international conference on Knowledge discovery and data mining, pages 394–401. ACM, 2004.

[6] G. Alonso. *Web services: concepts, architectures and applications*. Springer Verlag, 2004.

[7] J.S. Armstrong. *Principles of forecasting: a handbook for researchers and practitioners*, volume 30. Springer, 2001.

[8] K.J. Arrow. *Social choice and individual values*, volume 12. Yale university press, 2012.

[9] E-A Baatarjav, A. Amin, R. Dantu, and N.K. Gupta. Are you my friend? In *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, pages 1–5. IEEE, 2010.

[10] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[11] R. Baraglia, M. Mordacchini, P. Dazzi, and L. Ricci. A p2p recommender system based on gossip overlays (prego). In *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pages 83–90. IEEE, 2010.

[12] D.J. Bartholomew, M. Knott, and I. Moustaki. *Latent variable models and factor analysis: a unified approach*, volume 899. John Wiley & Sons, 2011.

[13] N.K. Baym and A. Ledbetter. Tunes that bind? predicting friendship strength in a music-based social network. *Information, Communication & Society*, 12(3):408–427, 2009.

[14] N.J. Belkin and W.B. Croft. Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12):29–38, 1992.

[15] R. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 43–52. IEEE, 2007.

[16] D. Ben-Shimon, A. Tsikinovsky, L. Rokach, A. Meisles, G. Shani, and L. Naamani. Recommender system from personal social networks. *Advances in Intelligent Web Mastering*, pages 47–55, 2007.

[17] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[18] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[19] A.P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[20] J.S. Breese, D. Heckerman, C. Kadie, et al. Empirical analysis of predictive algorithms for collaborative filtering. In *conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[21] S. Bundasak and K. Chinnasarn. emenu recommender system using collaborative filtering and slope one predictor. In *Computer Science and Software Engineering (JCSSE), 2013 10th International Joint Conference on*, pages 37–42. IEEE, 2013.

[22] Wray L. Buntine. Operations for learning with graphical models. *arXiv preprint cs/9412102*, 1994.

[23] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR 02, pages 238–245. ACM, 2002.

[24] P.Y. Chebotarev and E. Shamis. Characterizations of scoring methodsfor preference aggregation. *Annals of Operations Research*, 80:299–332, 1998.

[25] Jere M Cohen. Sources of peer group homogeneity. *Sociology of Education*, 1977.

[26] C. Cortes, M. Mohri, and A. Rostamizadeh. Two-stage learning kernel algorithms. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 239–246, 2010.

[27] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel target alignment. In *NIPS*, volume 2, page 4, 2001.

[28] L. Ding, D. Steil, B. Dixon, A. Parrish, and D. Brown. A relation context oriented approach to identify strong ties in social networks. *Knowledge-Based Systems*, 24(8):1187–1195, 2011.

[29] P.S. Dodds, R. Muhamad, and D.J. Watts. An experimental study of search in global social networks. *science*, 301(5634):827–829, 2003.

[30] F. Draidi, E. Pacitti, and B. Kemme. P2prec: a p2p recommendation system for large-scale data sharing. *Transactions on large-scale data-and knowledge-centered systems III*, pages 87–116, 2011.

[31] Otis Dudley Duncan, Archibald O Haller, and Alejandro Portes. Peer influences on aspirations: A reinterpretation. *American Journal of Sociology*, pages 119–137, 1968.

[32] Dd Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning about a highly connected world.* Wiley Online Library, 2012.

[33] B.S. Everitt. *An introduction to latent variable models.* Springer, 1984.

[34] L. Fahrmeir, T. Kneib, S. Lang, and B. Marx. *Regression: models, methods and applications.* Springer, 2013.

[35] Y. Fang and L. Si. Matrix co-factorization for recommendation with rich side information and implicit feedback. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 65–69. ACM, 2011.

[36] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[37] R.T. Fielding. *Architectural styles and the design of network-based software architectures.* PhD thesis, University of California, Irvine, 2000.

[38] S. Funk. Netflix update: Try this at home. 2006.

[39] M. Gao and Z. Wu. Personalized context-aware collaborative filtering based on neural network and slope one. In *Cooperative Design, Visualization, and Engineering*, pages 109–116. Springer, 2009.

[40] M. Gao, Z. Wu, and F. Jiang. Userrank for item-based collaborative filtering recommendation. *Information Processing Letters*, 111(9):440–446, 2011.

[41] Y. Gao, C. Zhang, Y. Wang, and L. Sun. A directed recommendation algorithm for user requests based on social networks. In *Embedded and Ubiquitous Computing (EUC), 2011 IFIP 9th International Conference on*, pages 457–462. IEEE, 2011.

[42] E. Gilbert and K. Karahalios. Predicting tie strength with social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 211–220. ACM, 2009.

[43] D. Goldberg, D. Nichols, B. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[44] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[45] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, 2011.

[46] K. Gottschalk, S. Graham, H. Kreger, and J. Snell. Introduction to web services architecture. *IBM systems Journal*, 41(2):170–177, 2002.

[47] M. Granovetter. The strength of weak ties. *American Journal of Sociology*, pages 1360–1380, 1973.

[48] M. Granovetter. The strength of weak ties: A network theory revisited. *Sociological Theory*, 1(1):201–233, 1983.

[49] M. Granovetter. The impact of social structure on economic outcomes. *Journal of Economic Perspectives*, 19(1):33–50, 2005.

[50] I. Guy, N. Zwerdling, D. Carmel, I. Ronen, E. Uziel, S. Yogev, and S. Ofek-Koifman. Personalized recommendation of social software items based on social relations. In *Proceedings of the third ACM conference on Recommender systems*, pages 53–60. ACM, 2009.

[51] I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel. Social media recommendation based on people and tags. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201. ACM, 2010.

[52] J. Han, E. Haihong, G. Le, and J. Du. Survey on nosql database. In *Pervasive computing and applications (ICPCA), 2011 6th international conference on*, pages 363–366. IEEE, 2011.

[53] P. Han, B. Xie, F. Yang, and R. Shen. A scalable P2P recommender system based on distributed collaborative filtering. *Expert systems with applications*, 27(2):203–210, 2004.

[54] J. Herlocker, J. A Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237. ACM, 1999.

[55] T. Hofmann, B. Schölkopf, and A.J. Smola. Kernel methods in machine learning. *The annals of statistics*, pages 1171–1220, 2008.

[56] L. Hong, A. Doumith, and B. Davison. Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 557–566. ACM, 2013.

[57] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 135–142. ACM, 2010.

[58] T. Jebara and A. Pentland. Maximum conditional likelihood via bound maximization and the cem algorithm. In *NIPS*, volume 1, page 7, 1998.

[59] T. Jiang and W. Lu. Improved slope one algorithm based on time weight. *Applied Mechanics and Materials*, 347:2365–2368, 2013.

[60] J. Jones, J. Settle, R. Bond, C. Fariss, C. Marlow, and J. Fowler. Inferring tie strength from online directed behavior. *PloS one*, 8(1):e52168, 2013.

[61] I. Kahanda and J. Neville. Using transactional information to predict link strength in online social networks. In *Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM)*, 2009.

[62] P. Kantor, L. Rokach, F. Ricci, and B. Shapira. *Recommender systems handbook*. Springer, 2011.

[63] E. Karydi and K.G. Margaritis. Parallel implementation of the slope one algorithm for collaborative filtering. In *Informatics (PCI), 2012 16th Panhellenic Conference on*, pages 174–179. IEEE, 2012.

[64] P.D. Killworth, C. McCarty, H.R. Bernard, and M. House. The accuracy of small world chains in social networks. *Social networks*, 28(1):85–96, 2006.

[65] A. Klami, G. Bouchard, and A. Tripathi. Group-sparse embeddings in collective matrix factorization. *arXiv preprint arXiv:1312.5921*, 2013.

[66] I. Konstas, V. Stathopoulos, and J.M. Jose. On social networks and collaborative recommendation. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 195–202. ACM, 2009.

[67] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[68] S. Kotz, T. Kozubowski, and K. Podgorski. *The Laplace Distribution and Generalizations: A Revisit With Applications to Communications, Exonomics, Engineering, and Finance*. Number 183. Springer, 2001.

[69] D. Krackhardt. The strength of strong ties: The importance of philos in organizations. *Networks and organizations: Structure, form, and action*, 216:239, 1992.

[70] D. Krackhardt and R.N. Stern. Informal networks and organizational crises: An experimental simulation. *Social psychology quarterly*, 51(2):123–140, 1988.

[71] S.R. Kruk and S. Decker. Semantic social collaborative filtering with foaf-realm. In *Semantic Desktop Workshop colocated with Intl. Semantic Web Conference (ISWC2005)*. Citeseer, 2005.

[72] B. Lakshminarayanan, G. Bouchard, and C. Archambeau. Robust bayesian matrix factorisation. In *International Conference on Artificial Intelligence and Statistics*, pages 425–433, 2011.

[73] Cliff Lampe, Nicole B Ellison, and Charles Steinfield. Changes in use and perception of facebook. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 721–730. ACM, 2008.

[74] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *The Journal of Machine Learning Research*, 5:27–72, 2004.

[75] K. Lange and J.S. Sinsheimer. Normal/independent distributions and their applications in robust regression. *Journal of Computational and Graphical Statistics*, 2(2):175–198, 1993.

[76] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. *Society for Industrial Mathematics*, 5:471–480, 2005.

[77] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[78] C. Lippert, S.H. Weber, Y. Huang, V. Tresp, M. Schubert, and H.P. Kriegel. Relation prediction in multi-relational domains using matrix factorization. In *Proceedings of the NIPS 2008 Workshop: Structured Input-Structured Output, Vancouver, Canada*, 2008.

[79] B. Long, X. Wu, Z.M. Zhang, and P.S. Yu. Unsupervised learning on k-partite graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 317–326. ACM, 2006.

[80] B. Long, Z.M. Zhang, X. Wu, and P.S. Yu. Spectral clustering for multi-type relational data. In *Proceedings of the 23rd international conference on Machine learning*, pages 585–592. ACM, 2006.

[81] B. Long, Z.M. Zhang, and P.S. Yu. Co-clustering by block value decomposition. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 635–640. ACM, 2005.

[82] H. Ma, I. King, and M. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 203–210. ACM, 2009.

[83] H. Ma, H. Yang, M.R. Lyu, and I. King. Sorec: Social recommendation using probabilistic matrix factorization. In *Proceeding of CIKM 2008*, pages 931–940. ACM, 2008.

[84] H. Ma, D. Zhou, C. Liu, M.R. Lyu, and I. King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296. ACM, 2011.

[85] P. Marsden and K. Campbell. Measuring tie strength. *Social Forces*, 63(2):482–501, 1984.

[86] P. Massa and P. Avesani. Trust-aware bootstrapping of recommender systems. In *ECAI 2006 Workshop on Recommender Systems*, pages 29–33. Citeseer, 2006.

[87] J. McAuley and J. Leskovec. Image labeling on a network: Using social-network metadata for image classification. In *Computer Vision–ECCV 2012*, pages 828–841. Springer, 2012.

[88] M. McPherson, L. Smith-Lovin, and J.M. Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444, 2001.

[89] L. Mekouar, Y. Iraqi, and R. Boutaba. Personalized recommendations in peer-to-peer systems. In *Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2008. WETICE'08. IEEE 17th*, pages 99–104. IEEE, 2008.

[90] P. Membrey, E. Plugge, and T. Hawkins. *The definitive guide to MongoDB: the noSQL database for cloud and desktop computing.* Apress, 2010.

[91] D. Menezes, A. Lacerda, L. Silva, A. Veloso, and N. Ziviani. Weighted slope one predictors revisited. In *Proceedings of the 22nd international conference*

on World Wide Web companion, pages 967–972. International World Wide Web Conferences Steering Committee, 2013.

[92] C.E. Metz. Basic principles of roc analysis. In *Seminars in nuclear medicine*, volume 8, pages 283–298. Elsevier, 1978.

[93] Carl D Meyer. *Matrix analysis and applied linear algebra*, volume 2. Siam, 2000.

[94] Z. Mi and C. Xu. A recommendation algorithm combining clustering method and slope one scheme. In *Bio-Inspired Computing and Applications*, pages 160–167. Springer, 2012.

[95] S. Milgram. The small-world problem. *Psychology today*, 2(1):60–67, 1967.

[96] E. Mjolsness. Labeled graph notations for graphical models. Technical report, Citeseer, 2004.

[97] S. Mohamed. *Generalised Bayesian matrix factorisation models.* PhD thesis, University of Cambridge, 2011.

[98] B. Murthi and S. Sarkar. The role of the management sciences in research on personalization. *Management Science*, 49(10):1344–1362, 2003.

[99] M.E.J. Newman. Models of the small world. *Journal of Statistical Physics*, 101(3-4):819–841, 2000.

[100] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[101] M.E.J. Newman and D.J. Watts. Renormalization group analysis of the small-world network model. *Physics Letters A*, 263(4):341–346, 1999.

[102] J. Noel, S. Sanner, K.N. Tran, P. Christen, L. Xie, E.V. Bonilla, E. Abbasnejad, and N. Della Penna. New objective functions for social collaborative filtering. In *Proceedings of the 21st international conference on World Wide Web*, pages 859–868. ACM, 2012.

[103] J-P Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, and A-L Barabási. Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences*, 104(18):7332–7336, 2007.

[104] Cara P. 99 new social media stats for 2012. *URL: http://thesocialskinny.com/99-new-social-media-stats-for-2012/ (last access on 11/03/2014)*, 2012.

[105] M. Powell and James D. *Approximation theory and methods.* Cambridge university press, 1981.

[106] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[107] E. Rich. User modeling via stereotypes. *COGNITIVE SCIENCE*, 3:329–354, 1979.

[108] G. Ruffo, R. Schifanella, and E. Ghiringhello. A decentralized recommendation system based on self-organizing partnerships. In *NETWORKING 2006. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems*, pages 618–629. Springer, 2006.

[109] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning*, pages 880–887. ACM, 2008.

[110] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. *Advances in Neural Information Processing Systems*, 20:1257–1264, 2008.

[111] G. Salton. Automatic text processing: the transformation, analysis, and retrieval of information by computer. *Addison-Wesley Series In Computer Science*, page 530, 1989.

[112] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[113] C. Schaefer, J.C. Coyne, and R.S. Lazarus. The health-related functions of social support. *Journal of behavioral medicine*, 4(4):381–406, 1981.

[114] B. Scholkopf and A. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*, volume 1. MIT press Cambridge, MA, 2002.

[115] J. Schroeder, J. Xu, and H. Chen. Crimelink explorer: Using domain knowledge to facilitate automated crime association analysis. In *Intelligence and Security Informatics*, pages 168–180. Springer, 2003.

[116] N. Seichepine, S. Essid, C. Févotte, and O. Cappé. Soft nonnegative matrix co-factorizationwith application to multimodal speaker diarization. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3537–3541. IEEE, 2013.

[117] U. Shardanand and P. Maes. Social information filtering: algorithms for automating word of mouth. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co., 1995.

[118] A.P. Singh. Efficient matrix models for relational learning. Technical report, DTIC Document, 2009.

[119] A.P. Singh and G.J. Gordon. Relational learning via collective matrix factorization. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658. ACM, 2008.

[120] A.P. Singh and G.J. Gordon. A unified view of matrix factorization models. In *Machine Learning and Knowledge Discovery in Databases*, pages 358–373. Springer, 2008.

[121] J.H. Smith. Aggregation of preferences with variable electorate. *Econometrica: Journal of the Econometric Society*, pages 1027–1041, 1973.

[122] A.J. Smola and R. Kondor. Kernels and regularization on graphs. *Learning theory and kernel machines*, pages 144–158, 2003.

[123] C. Spearman. General intelligence, objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292, 1904.

[124] Z. Sun, N. Luo, and W. Kuang. One real-time personalized recommendation systems based on slope one algorithm. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on*, volume 3, pages 1826–1830. IEEE, 2011.

[125] M. Svensén and C.M. Bishop. Pattern recognition and machine learning. 2007.

[126] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Investigation of various matrix factorization methods for large recommender systems. In *Data Mining Workshops, 2008. ICDMW'08. IEEE International Conference on*, pages 553–562. IEEE, 2008.

[127] J. Travers and S. Milgram. An experimental study of the small world problem. *Sociometry*, 32(4):425–443, 1969.

[128] A. Tveit. Peer-to-peer based recommendations for mobile commerce. In *Proceedings of the 1st international workshop on Mobile commerce*, pages 26–29. ACM, 2001.

[129] N.A. Van House. Flickr and public image-sharing: distant closeness and photo exhibition. In *Proceedings of CHI 2007*, pages 2717–2722. ACM, 2007.

[130] I.T. Varley, A. Aziz, and Aziz. No relation: The mixed blessings of non-relational databases. 2009.

[131] A.E. von Eye and C.C. Clogg. *Latent variables analysis: Applications for developmental research.* Sage Publications, Inc, 1994.

[132] J. Wang, J. Peng, and X. Cao. A distributed collaborative filtering recommendation model for p2p networks. In *Collaborative Computing: Networking, Applications and Worksharing*, pages 1–10. Springer, 2009.

[133] N. Wang, T. Yao, J. Wang, and D. Yeung. A probabilistic approach to robust matrix factorization. In *Computer Vision–ECCV 2012*, pages 126–139. Springer, 2012.

[134] I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2005.

[135] R. Xiang, J. Neville, and M. Rogati. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 981–990. ACM, 2010.

[136] B. Xie, P. Han, F. Yang, R. Shen, H. Zeng, and Z. Chen. Dcfla: A distributed collaborative-filtering neighbor-locating algorithm. *Information Sciences*, 177(6):1349–1363, 2007.

[137] X. Yang, Y. Guob, Y. Liua, and H. Steckc. A survey of collaborative filtering based social recommender systems. *Computer Communications*, (0):–, 2013.

[138] X. Yang, H. Steck, and Y. Liu. Circle-based recommendation in online social networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1267–1275. ACM, 2012.

[139] J. Yoo and S. Choi. Weighted nonnegative matrix co-tri-factorization for collaborative prediction. In *Advances in Machine Learning*, pages 396–411. Springer, 2009.

[140] Jiho Yoo, Minje Kim, Kyeongok Kang, and Seungjin Choi. Nonnegative matrix partial co-factorization for drum source separation. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 1942–1945. IEEE, 2010.

[141] H. Zhang and R. Dantu. Predicting social ties in mobile phone networks. In *Intelligence and Security Informatics (ISI), 2010 IEEE International Conference on*, pages 25–30. IEEE, 2010.

[142] X. Zhao, G. Li, J. Yuan, X. Chen, and Z. Li. Relationship strength estimation for online social networks with the study on facebook. *Neurocomputing*, 2012.

[143] Y. Zhong, L. Du, and J. Yang. Learning social relationship strength via matrix co-factorization with multiple kernels. In *Web Information Systems Engineering (WISE) 2013*, pages 15–28. Springer, 2013.

[144] Y. Zhong, J. Yang, and R. Nugroho. Incorporating tie strength in robust social recommendation. In *Big Data (BigData Congress), 2015 IEEE International Congress on*, pages 63–70. IEEE, 2015.

[145] Y. Zhong, W. Zhao, and J. Yang. Personal-hosting restful web services for social network based recommendation. *Service-Oriented Computing (SOC) 2011*, pages 661–668, 2011.

[146] Y. Zhong, W. Zhao, J. Yang, and L. Xu. Peer-based relay scheme of collaborative filtering for research literature. *COOPERATIVE INFORMATION SYSTEMS (CoopIS) 2011*, pages 321–328, 2011.

[147] Y. Zhong, X. Zheng, J. Yang, M.A. Orgun, and Y. Wang. Kpmcf: A learning model for measuring social relationship strength. In *Web Information Systems Engineering (WISE 2013*, pages 519–522. Springer, 2013.

[148] T. Zhou, H. Shan, A. Banerjee, and G. Sapiro. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *Proceedings of SDM 2012*, pages 403–414. SIAM, 2012.

[149] J. Zhuang, T. Mei, S. Hoi, X. Hua, and S. Li. Modeling social strength in social media community via kernel-based learning. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 113–122. ACM, 2011.

[150] C.N. Ziegler and G. Lausen. Paradigms for decentralized social filtering exploiting trust network structure. *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, pages 840–858, 2004.