ENHANCED RECOMMENDER SYSTEMS THROUGH CROSS-DOMAIN KNOWLEDGE TRANSFER

Qian Zhang

Faculty of Engineering and Information Technology

University of Technology Sydney

A thesis submitted for the Degree of

Doctor of Philosophy

June 2018

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Qian Zhang

June 2018

Acknowledgements

I would like to express my sincere gratitude to my principal supervisor A./Professor Guangquan Zhang. Without his patience and encouragement, I would not have been able to finish this PhD journey. His richness in knowledge, his responsibility, his care for his students and his enthusiasm to research affected me deeply. During my PhD, the discussions I had with Professor Zhang inspired me and taught me a lot about research methodology. I also would like to express my thankfulness to my co-supervisor Distinguished Professor Jie Lu. The discussions with her and her valuable reviews and critical comments on my paper writing improve the quality of my dissertation a lot. I appreciate the selfless care and support of both these academics. I would also like to express my thankfulness to my co-supervisor Dr Dianshuang Wu. I thank him for all those discussions on the experiments and for his corrections to my thesis.

I would like to express my thankfulness to every member of the Decision Systems & e-Service Intelligence Lab (DeSI). It was such a joyful time to spend three-and-a-half years with you. Together the collision of our ideas sparked and inspired my research. I would like to especially thank Wei Wang, Peng Hao, Ruiping Yin and Feng Liu for the discussions on recommender systems and transfer learning.

I kindly thank Ms Jemima Moore and Ms Michele Mooney for proofreading my thesis and publications. I am grateful to the School of Software in the Faculty of Engineering and Information Technology at the University of Technology Sydney for the travel support. I also thank the financial support I received from Australian Research Council (ARC) discovery project grant, which supports me to finish my study.

I would like to express my thankfulness to my family, my parents, my brother, my sister-in-law and my two nephews. Thank you for all your understanding and unconditional love and support during my three-and-a-half years abroad. I love you all.

Abstract

Recommender systems are widely used and have developed rapidly with the explosion of Web 2.0 technologies. The aim of recommender systems is to provide users with items (products or services) that match the users' preferences. Recommender systems provide users with personalized online product and service recommendations and are a ubiquitous part of today's online entertainment smorgasbord.

However, many real-world recommender systems suffer from data sparsity and user-preference drift issues which degrade the recommendation performance and lead to a poor user experience. For the user-preference drift issue, time-window and instance decaying approaches are widely applied, but one research gap is that existing methods proposed for adaptation and weighting decay are biased, since the direction of user preference drift was not appropriately addressed in their study. For the data sparsity issue, cross-domain recommender systems are used to handle data sparsity issues. These systems transfer knowledge from one domain that has adequate preference information to another domain that does not. One significant research gap in the existing methods is that they cannot ensure the knowledge extracted from the source domain is consistent with the target domain, which may impact the accuracy of the recommendations. This research addresses the aforementioned research gaps.

In this research, to solve these problems and enhance recommender systems, a user profile is enhanced with more information in various domains, including data in different time windows and different categories. For recommender systems with time labels, fuzzy set and fuzzy relation theories are adopted to model uncertain user behavior. A distance measure together with a related statistical guarantee is proposed to detect whether a user preference has drifted or not. A fuzzy user-preference drift detection-based recommendation method is proposed to model user preference and predict user ratings in temporal dynamics. For a cross-domain recommender system, to ensure knowledge consistency between the two domains, two sets of methods are developed for two different scenarios. For cross-domain recommender systems with non-overlapping entities, an adaptive knowledge transfer method for cross-domain recommender systems with consistent information transfer is proposed and applied to a telecom product recommender system and a business partner recommender system (Smart BizSeeker). Knowledge consistency is based on user and item latent groups, and domain adaptation techniques are used to map and adjust these groups in both domains to maintain consistency during the transfer learning process. For cross-domain recommender systems with partially overlapping entities, a kernel-induced knowledge transfer method is proposed. Domain adaptation is used to adjust the feature spaces of overlapping entities and diffusion kernel completion is used to obtain the non-overlapping entity correlations between two domains. It shows even with a small number of overlapping entities, knowledge transferred from the source domain to the target domain is very applicable and beneficial.

To conclude, this research addresses user-preference inconsistency which occurs in recommender systems in both different time windows and different categories.

Different contexts (e.g. time) can be treated as different domains. Thus, this research aims to improve prediction accuracy and enhance recommender systems through cross-domain knowledge transfer. Extensive experiment results show that our proposed methods can generally achieve significant improvement in accuracy compared with the existing approaches.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Due to the rapid development of information technology in e-services, a huge amount of information is sent to users when they are making decisions on any kinds of e-service. Usually, users are unwilling to explore the vast amount of information offered by companies in relation to their products and services and instead, choose a product or service in which they are truly interested. Thus, in the fiercely competitive marketplace, it is crucial for a company to help customers deal with information overload in decision-making so as to retain their loyalty (Schafer et al., 1999). Providing personalized e-services to customers is an appropriate approach to solve the information overload problem and improve the user's experience. In this way, services to different users are customized, making it easy for customers to find what they need.

Search engines (Brin and Page, 1998) and recommender systems (Resnick et al., 1994) emerged in the mid-1990s from information retrieval and filtering. After this,

recommender systems become the most commonly used technique to solve the information overload problem brought about by Web 2.0 to provide personalized e-services (Lu et al., 2015b). The recommendation process predicts users' potential interest (user's preference) in items that they haven't previously bought, according to users' historical records (explicitly as ratings on historical item and implicitly as click on a web page) thus creating a personalized list for which users can choose (Adomavicius and Tuzhilin, 2005). In the past two decades, recommender systems have been broadly adopted in different areas, both academia and industry. Application areas include movie recommendations on Netflix (Bennett et al., 2007), music recommendations on Pandora (Foote, 1997), commodity recommendations on Amazon.com (Linden et al., 2003), future event recommendation s(Minkov et al., 2010), business-to-business recommendations (Shambour and Lu, 2012) and even medical decision support recommendations (Wiesner and Pfeifer, 2010).

Recommendation techniques can be roughly divided into three clusters (Lu et al., 2015b): one is content-based, one is **C**ollaborative **F**iltering (CF)-based and the other is knowledge-based. The content-based recommender systems aim to recommend items that are similar to something a specific user was interested in previously (McFee et al., 2012; Tkalcic et al., 2013; Yin et al., 2013). CF-based techniques assume that users who have similar interests in the past will share similar tastes in the future (Baltrunas and Ricci, 2014; Cai et al., 2014; Liu et al., 2012; Wu et al., 2013). Knowledge-based techniques offer items to users based on the knowledge gained from domain experts or inferred from the available attributes (Carrer-Neto et al., 2012; Knijnenburg et al., 2012; Zhang et al., 2016). CF has unique advantages in many real-world areas, such as selecting a movie, book or music, when no additional side information about the item attributes or user profiles

is available. Over the last two decades, CF has been comprehensively exploited from basic memory-based methods (Deshpande and Karypis, 2004) to various model-based methods such as matrix factorization (Rendle, 2012), probabilistic models (Liu et al., 2013) or deep learning models (Wang et al., 2015). Matrix factorization techniques stemmed from Netflix Prize competition (Koren et al., 2009), and it is still one of the most popular algorithms due to its high accuracy. Model-based methods, especially matrix factorization techniques can be extended with other information such as social network data (Mao et al., 2017), context data (McAuley and Yang, 2016) or user reviews (Zhang et al., 2014b).

Recommender systems have achieved success in many areas, especially for Amazon (Wei et al., 2007) and Netflix (Bennett et al., 2007). It is one of the most widely used techniques to deal with the information overload problem. The demand for accurate recommendation is high since presenting inaccurate predictions reduces user satisfaction with the system (Cosley et al., 2003). There are two reasons why the performance of recommender systems may be impaired. One is that the data collection process used by recommender systems remains static (Campos et al., 2014). Traditional recommender system techniques assume that user preference is relatively static over a period time. Although this assumption is valid under most situations, with the rapid development and wide use of the World Wide Web, user-preference drifts are no longer negligible. For instance, a user's preference for a movie today may be quite different from one-year or one-month ago; a user's preference for accommodation may change when they are in employment compared to when they are students; a user will choose different materials to read at home compared to when they are on holiday. Recommendations based on historical user records may be inconsistent with the user's new request. The second reason is the

well-known data sparsity problem (Khan et al., 2014; Sarwar, 2001). The items are numerous and one user can only react to a very small part of the whole list of the items. Most recommendation techniques, especially the most widely used CF, are not able to handle the sparsity problem (Xu et al., 2017). The performance of these techniques is not stable, as new users come into the system with few ratings. It is difficult to identify user patterns of behavior when user profiles are incomplete. This scenario is common when a user starts to user the system. If the system fails to provide practical support, new users will quickly lose interest and stop consulting the system.

To solve this problem and enhance recommender systems, a user profile is enhanced with more information in various domains, including data in different time windows and different categories. Usually, a user contributed data such as ratings or tags are weighted equally in most recommender systems. However, in real-world recommender systems, a user's preference and an item's functionality and popularity change over time. The variations in the data distribution with time results in the concept drift phenomenon (Widmer and Kubat, 1996). The phenomenon of unpredicted changes in the underlying data distribution over time is referred to as concept drift, which was first formulated by (Harries and Horn, 1995). Previous research shows that recommendation results are remarkably improved when changes in a user's preference are considered (Lathia et al., 2009). Most previous methods are based on the assumption that recent data weighs more than older data (Ding and Li, 2005). Based on this, gradual forgetting and time decaying based on windowing have been applied in techniques such as collaborative filtering (Koren, 2010). Since they fail to detect the change but only consider rating changes in a time order, the proposed weighting decay can be biased. Therefore, there is a

need to develop methods and models to handle user-preference drift more delicately to improve the performance of recommender systems.

We realized that we had insufficient data in one domain, such as movies, but relatively rich data in another domain, such as books. If two domains are explicitly or implicitly related, transfer learning can be conducted on recommender systems to extract shared knowledge from a domain with relatively denser data (Long et al., 2014a). The core assumption of CF fits well with the transfer learning framework. For example, users who favor romantic novels tend to enjoy romantic movies. In this way, a newly launched recommender system can be promoted by data from another related domain (Liu et al., 2016). Although data in the target domain may not be sufficient, users tend to be more active on social media or other online websites. Systems using transfer learning techniques are known as **C**ross-**D**omain **R**ecommender **S**ystem (CDRS), which is a recommender system designed to provide recommendation for the target domain using information extracted from the source domain. The knowledge shared between two domains is extracted and transferred differently in different entity overlap scenarios. Existing cross-domain recommender systems usually assume that no entity is overlapping or that there is fully one-to-one mapping. The non-overlapping methods collectively extract knowledge shared between the source and target domain, mostly on group-level user behavior (Li et al., 2009a). For the fully overlapping methods, the original source and target rating matrixes are collectively factorized and the features of the entities are extracted (Pan and Yang, 2013). Constraints are established on corresponding entities to force them to be completely the same. The constraints act as a bridge to allow knowledge from the source domain to be transferred to the target domain. The most crucial concern in CDRS is how to extract consistent

knowledge that can be shared between two related domains. Existing methods cannot ensure the knowledge extracted from the source domain is consistent with that in the target domain. This partially overlapping scenario falls in between the fully overlapping and non-overlapping is seldom studied.

To sum up, although recommender systems have been researched for around twenty years, there are still some crucial questions to be answered including a fundamental issue, prediction accuracy. This study will handle user-preference inconsistency which occurs in recommender systems in both different time windows and different domains. Different contexts (e.g. time) can be treated as different domains (Cantador et al., 2015). Thus, this research aims to improve prediction accuracy and enhance recommender systems through cross-domain knowledge transfer.

1.2 Research Questions and Objectives

This research aims to enhance recommender system through cross-domain knowledge transfer. This research will answer research questions as follows:

QUESTION 1. How to detect user-preference drift over time and react to provide guidance for the recommendation generation process?

QUESTION 2. How to ensure consistent knowledge transfer in cross-domain recommender systems?

QUESTION 3. How to transfer knowledge in cross-domain recommender systems through partially overlapping entities?

To answer these research questions, this research aims to achieve the following objectives:

OBJECTIVE 1. To develop a fuzzy user-preference drift detection-based recommendation method to handle inconsistent user preferences over time.

This objective corresponds to research question 1. User preferences change for a range of reasons and item features and user behaviors are usually subjective, incomplete and vague. Fuzzy sets and fuzzy relations are applied to recommender systems to profile user preferences at a specific time point or during a time interval. A fuzzy user preference is modeled and a set of algorithms related to detecting user preference drift is developed based on the user preference profile to describe whether a user preference is consistent over the entire user historical records. A recommendation method which can adapt to user-preference drift is proposed.

OBJECTIVE 2. To develop an adaptive knowledge transfer method for cross-domain recommender systems by consistent information transfer with non-overlapping entities.

This objective corresponds to research question 2. Following the definition in objective 2, an adaptive knowledge transfer method for CDRS is developed where user and item latent groups are adjusted by domain adaptation techniques. In this method, knowledge transferred from the source domain is consistent with the target domain. Therefore, the effectiveness of CDRS is ensured and the performance of CDRS is improved.

OBJECTIVE 3. To develop a kernel-induced knowledge transfer method for cross-domain recommender systems with partially-overlapping entities.

This objective corresponds to research question 3. Following the definition in objective 2, a kernel-induced knowledge transfer method that can eliminate the divergence of latent features between two domains and an adaptive knowledge transfer method for CDRS are developed. In this method, knowledge transfer is

achieved through only a small part of the entities that overlap between the two domains.

1.3 Research Significance

1.3.1 Theoretical Significance

Theoretically, the research develops a set of recommendation methods and solves the three following issues in recommender systems:

- **User profile modeling.** This research solves the user profile modeling problem to take different domains (time or category) into consideration. First, analyzing user preference and related item attributes can detect whether user preference has drifted or not. Second, this research defines how to adjust user/item group representations to ensure consistent knowledge, revealing a way to properly model user/item groups. Third, this research defines how to adjust user/item latent features to enable knowledge transfer between two domains through the overlapping entities.

- **Matrix factorization.** This research provides a framework for matrix tri-factorization and bi-factorization for consistent knowledge transfer in CDRS. This complements and improves the whole methodology of matrix factorization techniques used in CDRS.

- **Data sparsity.** This research addresses the data sparsity problem which is fundamental and challenging in recommender systems. By cross-domain strategies, the data sparsity problem is alleviated. Our research shows that

time-related data, cross-domain data without overlapping entities and cross-domain data with partially overlapping entities can contribute to solving data sparsity problem.

1.3.2 Practical Significance

In practical terms, this research provides guidelines on how to improve the performance of recommendation by addressing the following practical problems:

- **Time-aware recommendation.** Analysis of the user-item rating matrix shows that user's preferences change over time. A customer's choice of products/services change for various reasons such as mood, season, or social trends. A system which neglects these changes cannot provide appropriate recommendation results to customers. Thus, a method that can detect and react to changes in a user's preferences will help recommender systems to understand in what the user is interested now. In this way, businesses will better understand market development trends and make up-to-date recommendation to users.

- **Recommendation accuracy.** Data uncertainty and scarcity is very common in real-world recommender system applications. This research provides various ways to improve the accuracy of recommendation and provides users with better decision-making support. In particular, the proposed system and the new users who have recently entered the system can benefit from this research.

- **Recommendation diversity.** There are many online systems that sell different types of goods. The historical records of users are across

different categories. This research provides a way to satisfy the demand for recommendation across domains. Since recommendations can be made in different categories of goods, the potential diversity of recommendations is improved.

1.4 Thesis Structure

The logical structure of this thesis (the chapters and the corresponding research questions) and the relationship between the chapters are shown in Figure 1.1. The main contents of each chapter are summarized as follows:

Chapter 2 presents a systematic literature review related to this research. In this chapter, first a general recommender system definition is given, followed by a classification of recommendation techniques. Then, a mathematical definition of transfer learning is presented together with recent research development of this area. Finally, CDRSs are reviewed. Related methods and techniques are classified into three groups according to the information and entity overlapping scenarios. After this comprehensive review, the current research gaps are discussed and summarized.

Chapter 3 proposes a fuzzy user-preference drift detection-based recommendation method to accommodate change in the user's taste. In this chapter, a fuzzy user-preference consistency model is built based on fuzzy set theories, and a user-preference drift detection method and statistical guarantee are proposed based on concept drift techniques to provide guidance for recommendation generation. Empirical experiments are conducted on synthetic and real-world MovieLens datasets. The results show that the proposed approach improves the performance of recommender systems.

Figure 1.1 Thesis structure

Chapter 4 develops an adaptive knowledge transfer method for cross-domain recommender systems with **C**onsistent **I**nformation **T**ransfer (CIT) with non-overlapping entities. Knowledge consistency is based on user and item latent groups, and domain adaptation techniques are used to map and adjust these groups in both domains to maintain consistency during the transfer learning process. Experiments were conducted on five real-world datasets in three categories: movies, books, and music. The results for nine cross-domain recommendation tasks show that CIT outperforms five benchmarks and increases the accuracy of recommendations in the target domain, especially with sparse data. In practice, our proposed method is applied to a telecom product recommender system and a business partner recommender system (Smart BizSeeker) to enhance personalized decision making for both businesses and individual customers.

Chapter 5 develops a kernel-induced knowledge transfer method for cross-domain recommender systems with partially overlapping entities. Domain adaptation is used to adjust the feature spaces of overlapping entities, while diffusion kernel completion is used to correlate the non-overlapping entities between the two domains. In the proposed method, knowledge is effectively transferred through the overlapping entities, alleviating data sparsity issues. Experiments conducted on four datasets, each with three sparsity ratios, show that **Ker**nel-induced **K**nowledge **T**ransfer (KerKT) outperforms six benchmarks and increases the accuracy of recommendations in the target domain. Additionally, the results indicate that transferring knowledge from the source domain to the target domain is both possible and beneficial with even small overlaps.

Chapter 6 summarizes the contributions of this research and discusses research issues for further study.

1.5 Publications Related to This Thesis

Following is a list of the refereed international journal and conference papers during my PhD research that have been published or are currently under review:

Published:

1. **Q. Zhang**, D. Wu, J. Lu, F. Liu and G. Zhang, "A cross-domain recommender system with consistent information transfer," *Decision Support System*, vol. 104, pp. 49-63, 2017. (ERA Rank A*)

2. **Q. Zhang**, D. Wu, J. Lu and G. Zhang, "Fuzzy user-interest drift detection based recommender systems," in *Proceedings of the Twenty-sixth IEEE International Conference on Fuzzy Systems*, pp. 1274-1281, IEEE, 2017. (ERA Rank A)

3. **Q. Zhang**, G. Zhang, J. Lu and D. Wu, "A framework of hybrid recommender system for personalized clinical prescription," in *Proceedings of the International Conference on Intelligent Systems and Knowledge Engineering* (*ISKE*), pp. 189-195, IEEE, 2017. (ERA Rank B)

Submitted:

1. **Q. Zhang**, J. Lu, D. Wu and G. Zhang, "A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities," *IEEE Transactions on Neural Networks and Learning Systems*, (ERA Rank A)

Chapter 2

Literature Review

This chapter presents a literature review of relevant studies in connection with this research. Cross-domain recommender systems are applications of transfer learning techniques on recommender system in order to solve the data sparsity problem, a universal and practical issue which occurs in recommender systems. In this chapter, we review the research related to the three aspects of the above issue: recommender systems, transfer learning and cross-domain recommender systems. In Section 2.1, the problem of recommender systems, its recent development and the most commonly used techniques are introduced. This is followed by a review of transfer learning methods related to recommender systems in Section 2.2. Finally, in Section 2.3, a formal definition of cross-domain recommender systems is given, with a review of different methods that are related to this research.

2.1 Recommender Systems

The explosive growth in information on the web and the rapid increase in e-services has given users a huge amount of choice, which may make decision making more complex. Recommender systems are primarily devised to assist individuals who are short on experience or knowledge deal with the vast number of choices in relation to items (Shapira et al., 2011). Recommender systems take advantage of various sources of information to predict preferences of users in relation to different items (Bobadilla et al., 2013). This area of research has been the focus of great interest for the past twenty years from both academia and industry. Research in this field is motivated by the potential profit recommender systems have generate for businesses such as Amazon (Schafer et al., 1999). Recommender systems were first applied in E-commerce to solve the information overload problem caused by Web 2.0 and were quickly expanded to the personalization of e-government, e-business, e-learning, e-tourism (Lu et al., 2015b). Nowadays, recommender systems are an indispensable part of Internet websites such as Amazon.com, YouTube, Netflix, Yahoo, Facebook, Last.fm, and Meetup.

In brief, recommender systems are designed to estimate the utility of an item and predict whether it is worth recommending. The core part of recommender systems is a function to define the utility of a specific item to a user (Adomavicius and Tuzhilin, 2005). A final recommendation list containing a set of items in a ranked order will be provided. This list is ranked according to the utility of all the items the user has not consumed. Basically, the utility of an item is presented as ratings of a user. Recommender systems is to find an item for the user to maximize the utility function. Predicting the utility of items to a particular user varies in different recommendation algorithms. Referencing the classical taxonomies of

previous research (Adomavicius and Tuzhilin, 2005; Burke, 2002; Lu et al., 2015b), recommendation techniques are categorized in four types: content-based, CF-based, knowledge-based and hybrid approaches. These four categories will be reviewed in the following subsections.

2.1.1 Content-based Recommender Systems

As the name suggests, content-based recommender systems make use of the content of an item's description to predict its utility based on the user's profile (Shardanand and Maes, 1995). Content-based recommender systems aim to recommend items that are similar to those in which a specific user was interested previously. The recommendation process of content-based recommender systems is as follows (Pazzani and Billsus, 2007):

2.1.1.1 Item representation

Different item properties are extracted from documents/descriptions. The properties are referred to as "attributes", "characteristics", "fields" or "variables" in different publications. For instance, a movie can be represented by attributes: genres, directors, writers, actors, storyline, etc. These properties can be obtained directly from structured data like a table or unstructured data like an article/news. One of the most commonly used retrieval techniques in content-based recommender systems is a keyword-based model or the **V**ector **S**pace **M**odel (VSM) with **T**erm **F**requency-**I**nverse **D**ocument **F**requency (TF-IDF) weighting (Salton et al., 1975).

2.1.1.2 User profiling

Content-based recommender systems profile a user's preferences from items in a user's consumption records. Usually, the profile comprises information on what the user has liked or disliked in the past. Hence, the profiling process is a typical binary classification problem which has been well studied in machine learning and data mining fields. Classic methods such as Naïve Bayes, nearest neighbor algorithms and decision trees are used in this step (Sebastiani, 2002).

2.1.1.3 Filtering and recommendation

Once the user's profile is established, the system compares the item's attributes with the user's profile and finds the most relevant items from which to form a recommendation list. Recommendation in content-based recommender system is a filtering and matching up process between the item representation and the user profile based on the features acquired in the first two steps. The final result is to forward the matched items and remove those items the user tends to dislike. So the relevance evaluation of recommendation is dependent on the accuracy of the item's representation and the user's profile (Herlocker et al., 2004).

Content-based recommender system has several advantages as illustrated in previous studies (Lops et al., 2011). First, content-based recommendation is based on item representation, so it is user independent. As a result, this kind of system does not suffering from the data sparsity problem, a serious problem in CF-based recommender systems which is discussed in Section 2.1.2. Second, content-based recommender systems are able to recommend new items to users so the system can solve the new item cold start problem. Lastly, content-based recommender systems can provide a clear explanation of the recommendation result. The transparency of

this kind of system is a great advantage compared to other kind of techniques in real-world applications.

On the contrary, there are several limitations in content-based recommender systems (Adomavicius and Tuzhilin, 2005; Balabanović and Shoham, 1997). To begin with, although the new item problem can be solved or alleviated using content-based recommender systems, they suffer from the new user problem as the lack of user profile information will seriously affect the accuracy of the recommendation result. Furthermore, a content-based system will always choose similar items for users, leading to overspecialization in recommendation. A user tends to become bored with similar recommendation lists because most users want to learn about new and fashionable items rather than being limited to items similar to those they have previously used. Finally, items sometimes cannot easily be represented in the specific form required by content-based recommender systems. This kind of system is more suitable for articles or news recommendation rather than images or music.

2.1.2 Collaborative Filtering-based Recommender Systems

Content-based recommender systems are independent of other users but dependent on the user's own historical records, while CF-based recommender systems infer the utility of an item according to other users' ratings (Balabanović and Shoham, 1997). The term "collaborative filtering" was first used in Tapestry (Goldberg et al., 1992), one of the first systems to implement CF algorithms. This technique has been widely researched in academia (Liu et al., 2014; Resnick et al., 1994) and was quickly applied in industry (Linden et al., 2003). Twenty years on, CF is still the most popular technique applied in recommender systems (Ricci et al., 2011).

The basic assumption underpinning the CF technique is that users who share similar interests will consume similar items. So, a system using the CF technique relies on other users who share similar preferences to the given user. A classic scenario in CF is to predict a user's ratings on unconsumed items from a user-item rating matrix. And it is related to a matrix completion problem (Hu et al., 2013b). Generally, ratings are explicitly expressed on a scale from 1-5 but sometimes binary feedback reflecting a user's action or inaction, such as bookmarking a web page, clicking on a piece of news, or viewing a product page is used (Pan et al., 2008a). Compared with content-based techniques, CF has its own advantages. First, the content information is not always available which limits the development of content-based techniques. Without this limitation, CF techniques have wider applications. Second, the CF techniques are easily implemented. They can be applied to any online e-commerce website related with rating system. Third, CF techniques contribute to serendipity of recommender systems, which is the main reason that they are applied to websites such as Netflix or Amazon. These techniques can help users find items that are hidden in the long tail but very attractive to them, thus, support their decision and improve the profits of suppliers as well. CF-based techniques are classified into two categories (Breese et al., 1998): memory-based CF and model-based CF, which is explained in detail in the following sub-sections.

2.1.2.1 Memory-based Collaborative Filtering

Memory-based CF is an early generation CF using heuristic algorithms to calculate similarity values between users or items, and can therefore be divided into two types: user-based CF and item-based CF (Deshpande and Karypis, 2004). The core algorithm used in the memory-CF technique is the nearest neighbor algorithm. The

recommendation calculates and ranks the rating of a target user on different items based on the ratings of a user's/item's neighbors. This algorithm is well accepted because of its simplicity, efficiency and ability to produce accurate result. It is implemented in the following steps (Su and Khoshgoftaar, 2009):

(1) Similarity calculation

Before calculating the predicted rating, the weights of each user/item should be calculated according to its distance to the target user/item. The aim of different weightings on users/items is to select the nearest neighbors from the whole set. This step is vital for the final accuracy of recommendation. So different measuring methods are applied in this step. In this part, correlation-based similarity will be introduced (Billsus and Pazzani, 1998; Lang, 1995).

Let the rating matrix be $\boldsymbol{X} \in \mathbb{R}^{M \times N}$, suggesting that m users and n items are in the matrix. Two items i and j can be represented by the ratings of each user in a vector. The similarity between two vectors can be measured by cosine similarity. Vector cosine similarity of items i and j is calculated by:

$$W_{(i,j)}^{cos} = \frac{\sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} r_{u,i} \times r_{u,j}}{\sqrt{\sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} r_{u,i}^2} \times \sqrt{\sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} r_{u,j}^2}} \tag{2.1}$$

where $\mathcal{U}_i$ is the user set containing users who have rated item i, and $\mathcal{U}_j$ is the user set containing users who have rated item j. $\mathcal{U}_i \cap \mathcal{U}_j$ represents users who have co-rated item i and item j together. $r_{u,i}$ and $r_{u,j}$ are the ratings of user u for items i and j respectively.

Cosine similarity is used in item-based CF but is seldom used in user-based CF because it fails to take differences in means and variances between users' rating scales into consideration. This is adjusted in **P**earson **C**orrelation **C**oefficient (PCC),

a popular measurement method used in both item-based and user-based CF. In user-based CF (Sarwar et al., 2001), the PCC similarity between two users u and v is calculated by

$$W_{u,v}^{PCC} = \frac{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} (r_{u,i} - \bar{r}_u) \times (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} (r_{u,i} - \bar{r}_u)^2} \times \sqrt{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} (r_{v,i} - \bar{r}_v)^2}} \tag{2.2}$$

where $\mathcal{I}_u$ is the item set which is rated by users u, and $\mathcal{I}_v$ is the item set which is rated by user v. $\mathcal{I}_u \cap \mathcal{I}_v$ represent items which are co-rated by user u and user v together. $\bar{r}_u$ and $\bar{r}_v$ are the average ratings of user u and user v on all the items in $\mathcal{I}_u \cap \mathcal{I}_v$ respectively.

For item-based CF (Resnick et al., 1994), the PCC similarity between two items i and j is calculated by

$$W_{i,j}^{PCC} = \frac{\sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} (r_{u,i} - \bar{r}_i) \times (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} (r_{u,i} - \bar{r}_u)^2} \times \sqrt{\sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} (r_{u,j} - \bar{r}_j)^2}} \tag{2.3}$$

where $\mathcal{U}_i$ is the user set who rated item i, and $\mathcal{U}_j$ is the user set who rated item j. $\mathcal{U}_i \cap \mathcal{U}_j$ represents users who have co-rated item i and item j together. $\bar{r}_i$ and $\bar{r}_j$ are the average ratings of item i and item j rated by all the users in $\mathcal{U}_i \cap \mathcal{U}_j$ respectively.

(2) Rating prediction

After the weighting matrix is calculated, the rating of a target use u on a specific item i i.e., $r_{u,i}$ can be predicted. This prediction is mostly dependent on the neighborhood of the target user or item. In item-based CF with a cosine similarity, the prediction is made by (Sarwar et al., 2001),

$$r_{u,i} = \frac{\sum_{j \in \mathcal{I}_u} r_{u,j} \times W_{i,j}}{\sum_{j \in \mathcal{I}_u} |W_{i,j}|} \tag{2.4}$$

where $\mathcal{I}_u$ is the item set containing all the other items which user u has rated except item i, and $W_{i,j}$ is the similarity between target item i and another item j whose rating is known.

In user-based CF with a PCC similarity, the prediction is (Resnick et al., 1994),

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v \in \mathcal{U}_i} (r_{v,i} - \bar{r}_v) \times W_{u,v}}{\sum_{v \in \mathcal{U}_i} |W_{u,v}|} \tag{2.5}$$

where $\mathcal{U}_i$ is the user set consisting of all the users who have rated item i, and $W_{u,v}$ is the similarity between target user u and another user v who has rated item i.

(3) Top-N recommendation

Either in item-based CF or user-based CF, the k most similar users/items are selected based on the weights calculated in (1). Then, the k nearest neighborhoods are used to calculate the ratings of each user on each item the target user has not purchased. The rating predictions are sorted in decreasing order and the top-N items are chosen to form the recommendation list.

Although memory-based CF is well known for its easy implementation and relatively effective application in practice, this technique still has some non-negligible drawbacks (Adomavicius and Tuzhilin, 2005). Firstly, this technique is not able to deal with the cold-start problem. When a new user/item comes in the system, there is no rating for the system to make predictions for the new user/item. Secondly, if the item is not new but is unpopular, it doesn't have many ratings from its consumers. Memory-based CF is unlikely to recommend unpopular items to users. Therefore, the coverage of recommendation is limited. Thirdly, it cannot provide real-time recommendation. The heuristic process will take a long time to give a recommendation result especially when the dimension of the user-item rating matrix is high. This problem is partly solved by a pre-calculated and pre-stored

weighting matrix in item-based CF (Deshpande and Karypis, 2004), but the system scalability still cannot meet the practical needs.

2.1.2.2 Model-based Collaborative Filtering

Model-based CF builds a model to predict a user's rating on items using machine learning or data mining methods rather than heuristic methods as discussed in the previous section. This technique was originally designed to remedy the defects in memory-based CF, but it has been widely researched to solve different problems in specific domains. In addition to the user-item rating matrix, other side information is used, such as location, tags and reviews (Shi et al., 2014). The model-based CF technique is a good choice to combine side information with the rating matrix. In this part, three models are reviewed: matrix factorization-based, graph-based and context-aware recommender systems.

(1) Matrix factorization-based recommender systems

Matrix factorization starts to be used in recommender system from **p**robabilistic **L**atent **S**emantic **A**nalysis (pLSA) (Hofmann, 2004). Matrix factorization emanated from Netflix Prize competition (Koren, 2008), and it is still one of the most popular algorithms in this field. This model is also called **S**ingular **V**alue **D**ecomposition (SVD) (Koren et al., 2009). It projects both user space and item space onto the same latent factor space so that they are comparable. Furthermore, a probabilistic explanation for matrix factorization is given in (Mnih and Salakhutdinov, 2008). Three advantages of matrix factorization contribute to its popularity. First, the dimension of the user-item rating matrix can be reduced significantly, so the scalability of the system employing matrix factorization is secured. Second, the factorization process makes a dense rating matrix, so that the sparsity problem can

be alleviated (Luo et al., 2016). Users who only have a few ratings can acquire relatively more accurate recommendation through matrix factorization, which is a significant improvement compared with memory-based methods. Third, matrix factorization is highly suitable for integrating various side information (Liu et al., 2015). This helps to profile user preferences and improves the performance of recommender systems.

McAuley and Leskovec (2013a) combine latent rating dimensions with latent review topics, thus a method integrating matrix factorization and topic models is proposed to better understand user preferences through user-generated reviews. With sentiment analysis, Diao et al. (2014) proposed an unsupervised approach to uncover user's aspects and sentiments on movie ratings. Qian et al. (2014) mined item attributes and topics together to explain user rating behavior through a probabilistic view. McAuley et al. (2015) modeled human sense of object appearance through images and contributed it to cloth recommendation (He and McAuley, 2016). Yang et al. (2017) adopted matrix factorization with user trust relationships to reflect the opinion effect of users. All this work shows that matrix factorization is a fundamental brick in the recommender system area and has the advantages of incorporating various side information. A comprehensive survey on how side information is used and how it affects recommender systems can be found in (Shi et al., 2014).

(2) Graph-based recommender systems

Formally, a graph G is a non-empty finite set V of elements called vertices together with a possibly empty set E of pairs of vertices called edges. In graph-based model, users and items are always represented by vertices and the relationships between them are represented by edges. The graph is a bipartite graph whose

vertices set V can be classified into two non-empty sets V_{user} and V_{item} corresponding to sets of users and items (He et al., 2017). The weight of an edge is the measurement of how closely the two vertices are related. The proximity of two vertices is calculated directly by ratings (Gori et al., 2007) or similarities (Fouss et al., 2007) in recommender systems.

As users and items are fitted in graph-based models, many graph theories can be applied in recommendation. One approach is to use the probabilistic graphic model. Ranking is dependent on the probability of reaching an item vertex in a random-walk. Well-known algorithms like PageRank (Page et al., 1999) are applied and many variants (Gori et al., 2007; Mao et al., 2017; Yildirim and Krishnamoorthy, 2008) have been deployed. In addition to ratings or rating similarities, other relationships like trust (Jamali and Ester, 2009), folksonomy and taxonomy (Liang et al., 2010) are also employed in graph-based models.

(3) Context-aware recommender systems

Traditional recommender systems ignore "context" in which users are located in when recommending items to them. Context information (e.g. mood, location, time, device) is quite important for recommender systems to provide accurate recommendations to users (Adomavicius and Tuzhilin, 2015). These systems are named context-aware recommender systems. For instance, it is not appropriate for a recommender system to recommend a restaurant in the central business district to a user who now is in a suburban area. There are also various examples indicating that a user's preference is influenced by context: a user's preference on weekdays may be quite different from that on weekends; a user's preference may change when they are with friends compared with when they are alone; a user will choose different materials to read when they are eating breakfast or when they are about to

go to bed. Thus, understanding a user's preference in changing contexts is crucial for designing a good recommendation method or system (Panniello et al., 2014; Wang et al., 2016).

According to previous research, time is important context information in recommendation (Hong et al., 2012). By exploiting time information in the recommendation process, the performance of recommender systems is improved (Campos et al., 2014). A well-known example is the winning model named timeSVD++ in the Netflix Prize competition (Koren, 2010). Similar to context-aware recommender systems, these systems are named time-aware recommender systems, as time information serves as one context dimension. Time-aware recommender systems have attracted a lot of research attention since time-stamp data are easy to collect and it doesn't need an extra action on the part of the user. Therefore, from a theoretical and practical perspectives, time information is relevant and meaningful when designing recommender system. Referring to (Campos et al., 2014), approaches exploiting temporal context in recommender systems are divided into two: categorical time-aware approaches and time adaptive approaches.

In categorical time-aware approaches, time information is represented as a discrete variable. Baltrunas and Amatriain (2009) designed several contextual variables such as timeOfTheDay $= \langle$morning, evening$\rangle$, timeOfTheWeek = $\langle$workday, weekend$\rangle$ or timeOfTheYear = $\langle$hot_season, cold_season$\rangle$. After representing time-related data in categories, pre-filtering and post-filtering techniques used in context-aware recommender systems are applied to deal with the defined time variables. More research on combining categorical time information with recommender system techniques can be found in (Panniello et al., 2009; Rendle,

2010). These approaches are suitable for modeling the periodic interest of users but have limitations in handling dynamically changing user preferences.

Unlike categorical time-aware approaches, time information in time adaptive methods is represented by a continuous variable. The presumption is that a user's current preference is correlated to their most recent consumption records. In this way, the time distance regarding the current time is the major factor that will affect data weighting. This research is related to concept drift techniques such as instance selection and time decaying (Tsymbal, 2004). Ding and Li (2005) proposed an exponential time decay weight on the time series data. Cao et al. (2009) defined user preference patterns to detect user-preference drift with individual rating series data. Koren (2010) introduced a time factor in their matrix factorization model to handle user and item bias together with user latent factors with a Netflix dataset. Xiang et al. (2010) proposed a session-based graph model to capture a user's long-term and short-term preferences, respectively. McAuley and Leskovec (2013b) defined user experience as a categorical variable based on time and proposed an approach to model the evolving user expertise through user contributed reviews in a continuous time series. A temporal dynamic user modeling method is proposed to adapt the context change and user preference change together for social media recommendation (Yin et al., 2015). Zhang et al. (2014a) used a transition matrix to model user preference drift patterns and integrated it into a Bayesian matrix factorization model. In some recommendation scenarios, such as music recommendation, users tend to have recurrent behavior. This is related to time-aware recommender systems, which is important in order to accurately predict the next time users will return and how to recommend the most relevant item at the right moment. To capture the recurrent temporal patterns, Du et al. (2015) proposed

a time-sensitive recommendation model connecting self-exciting point processes and low-rank models. Another method analyzes user behavior with familiar items and models their boredom and restoring interest to predict their revisit time (Kapoor et al., 2015).

To sum up, user-preference is changing over time but this change is due to various reasons. To the best of our knowledge, time-aware recommender systems seldom detect and describe user-preference drift. They focus on modeling and adapting to individual user volatile interest. However, time-aware recommendation should consider many dimensions that affect user-preference drift. These dimensions are a user's demographic information, item content information or contextual information discussed at the beginning of this section. To model user-preference drift in time series in a finer granularity, the kind of dimension which is changing with time should be exploited. With various recommendation purposes and applications, different situations should be considered and discussed.

2.1.3 Knowledge-based Recommender Systems

In knowledge-based recommender system, recommendations are based on existing knowledge or rules of user's need and item's function (Burke, 2002). Unlike the aforementioned content-based and CF-based techniques, knowledge-based recommender systems retain a knowledge base where knowledge is extracted from a user's previous records. This knowledge base contains previous problem, constrains and corresponding solutions. Knowledge in the knowledge base is referenced when the system encounters a new recommendation problem (Smyth, 2007). Case-based reasoning is a commonly used technique for knowledge-based systems. Case-based reasoning uses previous cases to solve the current problem (Aamodt and Plaza,

1994). Different from the aforementioned content-based recommender systems, finding the similarity between products requires more structured representations. In this process, a comparison of a previous case and the current one is made, along with solution adaptation. Critiquing-based recommender systems are another kind of knowledge-based recommender systems to build user profile enhanced by user interaction with the system on choosing or critiquing items (Chen and Pu, 2012). These systems are widely used in industry and hybrid with other recommendation techniques. It is similar to the latter developed method dealing user contributed reviews.

The knowledge-based recommendation technique is of great application value especially in house sales, financial services or health decision support (Felfernig et al., 2011). These services are characterized by their specific domain knowledge and unique situation in each case. One advantage of this technique is that the new item/user problem does not exist since prior knowledge is acquired and stored in the knowledge base. Another advantage is that the user can set up constraints on the recommendation result (Felfernig and Burke, 2008). On the contrary, no advantage comes without disadvantages. The cost of building and maintaining the knowledge base is usually high in system set up and management.

All the techniques introduced in Section 2.1 are summarized in Table 2.1. Two or more techniques can be combined to overcome the limitation of any individual one. There are seven different types of hybrids: 'weighted, switching, mixed, feature combination, feature augmentation, cascade and meta-level' (Burke, 2002).

Table 2.1 Comparison of different recommendation techniques

Method	Basic Description	Representative Techniques	Advantages	Drawbacks
Content-based	Recommendations are based on the content of the item which the user preferred before.	Keyword-based Vector Space Model TF-IDF Bayes Network Decision Trees	1. User independent 2. Transparency 3. New item can be recommended	1. New user problem 2. Overspecialization 3. Content-dependent
CF-based	Recommendations are based on other users who have similar tastes.	Memory-based: Neighbor-based CF Top-N CF Model-based: Latent factor model Graph-based model Context-aware model	1. No need to consider the content 2. Easy implementation 3. Serendipity	1. Cold-start problem 2. Scalability 3. Sparsity
Knowledge-based	Recommendations are based on user's needs and item's function.	Case-based reasoning Ontology-based semantic web	1. New item and user can be recommended 2. Reasonable recommendation that meet specific needs	1. Monotonous recommendation 2. Complex knowledge database management

2.2 Transfer Learning

Machine learning has attracted great attention with an assumption that training data and test data are under the same distribution so that the trained models can solve problems of prediction or classification. However, in practice, test data is usually dynamic and diverges from the training data. This results in the current model is not applicable and needing to be rebuilt which takes great effort. Sometimes, it is not possible to retrain and build a new learning-based model since the newly collected data are not enough. There is usually not enough labels accompanying the new data. This problem is extremely serious in many real-world scenarios.

Unlike traditional machine learning, transfer learning has developed as a means of transferring knowledge from a domain with relatively rich data (source domain) to a domain with scarce data (target domain). In this definition, transfer learning aims to extract knowledge from one or more source data to assist a learning task with target data defined as:

Definition 2.1 (Transfer Learning) (Pan and Yang, 2010) Given a source domain D_s and learning task T_s, a target domain D_t and learning task T_t, transfer learning aims to improve the learning of the target predictive function $f_t(\cdot)$ in D_t using the knowledge in D_s and T_s where $D_s \neq D_t$ or $T_s \neq T_t$.

In the above definition, the condition $D_s \neq D_t$ implies that either $\mathcal{X}_s \neq \mathcal{X}_t$ or $P(X_s) \neq P(X_t)$. Similarly, the condition $T_s \neq T_t$ implies that either $Y_s \neq Y_t$ or $f_s(\cdot) \neq f_t(\cdot)$. At the same time, there exists some relationship or similarity between the feature spaces of the source domain and target domain. According to the definition, transfer learning techniques can be divided into three categories (Lu et al., 2015a):

1. *Inductive transfer learning.* The target task is different from the source task (i.e., $T_s \neq T_t$). When there is labeled data available in the target domain, inductive transfer learning is similar to multi-task learning (Kang et al., 2011). On the other hand, if there is no labeled data in the target domain, it is also known as self-taught learning (Kemker and Kanan, 2017; Raina et al., 2007).

2. *Transductive transfer learning.* The source and target tasks are the same, but the source and target domains are different (i.e., $T_s = T_t, D_s \neq D_t$). Transductive transfer learning is also used interchangeably with domain adaptation (Arnold et al., 2007). For this type of transfer learning technique, the discrepancy between source and target domain can be caused by the following reasons: different feature space (i.e., $\mathcal{X}_s \neq \mathcal{X}_t$) or different marginal distribution of feature spaces (i.e., $P(X_s) \neq P(X_t)$).

3. *Unsupervised transfer learning.* The setting is similar to inductive transfer learning but the target tasks are unsupervised learning tasks. Unsupervised transfer learning is similar to semi-supervised learning (Zhu et al., 2005), except that there are no labeled data for both the source domain and the target domain.

We divide category and review transfer learning into four types as in (Pan and Yang, 2010): instance-based, feature representation-based, parameter-based and relational knowledge-based.

2.2.1 Instance-based Transfer Learning

In instance-based inductive transfer learning, it is assumed that some source domain data are worth rescuing and can be used to construct a high-quality model for

the target domain. Representative work in this area is TrAdaBoost, a general framework to use boosting to assign different weight update strategies and train models with a large amount of source domain data and a small amount of target domain data. In this way, source domain data which is different to target domain data are automatically filtered out. To reduce negative transfer, another study developed and extended TrAdaBoost to multi-source domains settings (Yao and Doretto, 2010). Later, another two models extended TrAdaBoost and applied the transfer learning model to the multi-task level with various application scenarios (Huang et al., 2017; Wang et al., 2015).

For instance-based transductive transfer learning, the transfer learning strategy is closely related to importance sampling. Methods are developed to measure the difference between $P(D_s)$ and $P(D_t)$ caused by $P(X_s)$ and $P(X_t)$. To measure the distance between $P(X_s)$ and $P(X_t)$, a re-weighting process **K**ernel **M**ean **M**atching (KMM) is developed to match the means of source domain data and the target domain data in a **R**eproducing **K**ernel **H**ilbert **S**pace (RKHS) (Huang et al., 2006). Later, the different distribution of a source domain and target domain was called *covariate shift*. Xiao and Guo (2012) developed a method mapping target data points to similar source data points through kernelized representations of instances based on a Hilbert Schmidt Independence Criterion. Except for the kernel matching methods, an importance estimation method to minimize the **K**ullback **L**eibler (KL) divergence between the two distributions were integrated with model selection automatically (Sugiyama et al., 2008). Moreover, some unsupervised domain adaptation methods were developed to re-weight the training instances to minimize the mismatch between source and target domain distributions (Chu et al., 2013; Yamada et al., 2012).

2.2.2 Feature Representation-based Transfer Learning

In inductive transfer learning, feature representation-based transfer aims to find "good" features that can best represent the source domain data. It is more like the common feature learning in multi-task learning (Argyriou et al., 2007; Long et al., 2014b). Jebara (2004) proposed a method that extract common features with maximum entropy discrimination formalism for **S**upport **V**ector **M**achine (SVM). Argyriou et al. (2007) developed a method to learn a low-dimensional representation shared across the source and target domain through regularization and provide a convex optimization formulation for multi-task feature learning. Deng et al. (2014) proposed an inductive transfer learning mechanism suitable for several machine learning models including neural networks, fuzzy systems and kernel methods with a hidden-mapping ridge regression.

As for transductive transfer learning, feature representation-based transfer contains a number of methods involving dimensionality reduction (Pan et al., 2008b), subspace matching (Shao et al., 2012) and manifold alignment (Wang and Mahadevan, 2008). One of the fundamental methods is **M**aximum **M**ean **D**iscrepency **E**mbedding (MMDE) (Pan et al., 2008b), a method to ensure effective transfer learning to minimize the distance between distributions of the source data and target data in a low-dimensional latent space. The measure criteria is **M**aximum **M**ean **D**iscrepency (MMD) to compute the distance of distributions in a RKHS (Borgwardt et al., 2006). Furthermore, **T**ransfer **C**omponent **A**nalysis (TCA) is proposed to learn common transfer components which can reduce the divergence between the source data and target data after being projected to a subspace (Pan et al., 2011). Compared with MMDE, TCA can handle a scenario where no labeled data is available in the target domain and TCA is more efficient.

There is a lot of research in the feature representation-based transfer learning method. We list some of the representative work below: Bahadori et al. (2011) proposed a framework that combines subspace learning and transductive classification with flexible assumptions on the similarities between the source domain and target domain. Bruzzone and Marconcini (2010) extend transductive classification by taking into account the unlabeled data in the target domain. Long et al. (2014a) proposed a transfer learning framework considering not only the difference in the marginal distributions but also the conditional distributions across domains. There is also some other work on transductive learning with heterogeneous feature settings. Chang et al. (2017) used cross-domain parallel data to help label propagation in the target domain. Sometimes, there is no labeled data in the target domain, and this problem is called unsupervised domain adaptation. Some methods learn intermediate representations through modeling domains on Grassmann manifold as sampling along the geodesic, thus obtaining a subspace to learn domain-invariant features for adaption (Gong et al., 2014; Gopalan et al., 2011; Zheng et al., 2012). Instead of using one of the subspace on the geodesic flow, Gong et al. (2014) used all of them and used kernel trick to give a closed-form expression. This ensures its robustness to extract representations that reduce domain discrepancy.

Unsupervised transfer learning is still a challenge in the transfer learning area. Dai et al. (2008) firstly propose a **S**elf **T**aught **C**lustering (STC) method to use amounts of source data without labels to assist the clustering task with a small volume of target data. By simultaneously clustering the source and target data together, it is able to extract common representations for transferring knowledge. Later, Jiang and Chung (2012) proposed the transfer spectral clustering method

using bipartite graph co-clustering to relate source and target domain data on the same low dimensional feature embedding. Zhu et al. (2013) put forward a self-taught dimensionality reduction approach to transfer knowledge from freely available source data to a small-sized but high-dimensional target domain data. Bases learned from source data are applied to target data and joint graph sparse coding helps with the target data reconstruction in this method.

2.2.3 Parameter-based and Relational Knowledge-based Transfer Learning

There is a large body of research on feature representation-based transfer learning methods, however there is much less research on the other two methods. Parameter-based transfer learning shares parameters or statistical priors that allow a knowledge transfer benefit from the sharing. Some of these are probabilistic models that share statistical priors (Bonilla et al., 2008; Jing et al., 2014). Through preserving the statistical property and geometric structure in two domains, Long et al. (2014a) extracted common factors and proposed a graph co-regularization method. Gao et al. (2008) combined multiple models and assigned local weights according to the structures of the two domains. Deng et al. (2016) enhanced the parameter learning of fuzzy systems with two knowledge-leverage strategies. For relational knowledge-based transfer learning, its assumption is different from the traditional one, i.e., independent and identically distributed (i.i.d.). It requires that the transfer learning strategy is able to identify the relational structure and match the source and target domain on that level. Mihalkova et al. (2007) considered transfer learning with Markov logic networks so that its performance is revised and improved in the target domain through structure mapping with the source domain. Kumaraswamy

et al. (2015) used “type-matching” between two predicates and identified potentially similar objects across domains. In this way, logic rules from the source domain help parameter learning in the target domain. Recently, this type of transfer learning was defined as “knowledge translation” in (Jiang et al., 2016).

2.3 Cross-domain Recommender Systems

Recommender systems have been in existence for more than twenty years with wide application (Lu et al., 2015b). With great success and promising future, recommender systems are developed to provide users with more accurate and various options. Sometimes, a recommendation increases from a single-domain to multi-domain. The correlation of several domains needs to be exploited. It can benefit every single domain, meanwhile possibly mining user preferences that cannot be found with single domain data. For example, an active user in a movie domain is likely to be interested in books and music related to the movie they like. Another reason to exploit several domains together is to solve the data sparsity or cold-start problem. There may be insufficient data in one domain, but relatively rich data in another domain. For example, one user may have few records in a book category in an online review and rating system, but a lot of movie ratings available. The abundance of data in another domain can assist the recommendation in a specific target domain. To sum up, the demand for rich and diverse recommendation together with the ability to alleviate the data sparsity problem drives the development of the CDRS.

In CDRS, recommendation tasks in the target domain predict the unobserved interaction of users and items. Cross-domain recommender systems improve the

accuracy of the target recommendation task using the knowledge in the source domain. Compared with the other transfer learning methods, the biggest difference is that there is no explicit feature space in CDRS. So CDRS cannot be simply classified as any kind of transfer learning method. In other words, CDRS is a practical application of transfer learning techniques in the recommender system area. From a practical view, CDRSs provide multi-domain recommendation for online shopping retailers selling various categories of goods and provide a solution for the data sparsity problem at the same time.

Two different types of CDRSs have been developed. Some methods connect two domains through other auxiliary information rather than preference data (Shi et al., 2014). On the other hand, CDRSs based on preference data can be designed in various ways according to the overlap of users and item, the form the data takes, or the tasks the system needs to handle (Cantador et al., 2015). We classify CDRS using the aforementioned different scenarios and the three types of CDRSs are reviewed in this section as follows:

1. *Cross-domain recommender systems with side information.* Except rating matrixes in the source and target domains, there exists some other side information in the source domain and/or target domain.

2. *Cross-domain recommender system with non-overlapping entities.* There are no overlapping users or items between two domains.

3. *Cross-domain recommender systems with partially or fully overlapping entities.* There are overlapping users and/or items between two domains.

2.3.1 CDRSs with Side Information

For this type of recommender system, it is assumed that some side information on entities is available, either user generated information, social information or item attributes. **C**ollective **M**atrix **F**actorization (CMF) is designed for scenarios where a user-item rating matrix and an item-attribute matrix for the same group of items are available (Singh and Gordon, 2008). It collectively factorizes these two matrixes by sharing item parameters since the items are the same. Since then, some methods have been developed which exploit social network information to help with cross-domain recommender systems. Hybrid random walk was proposed to bridge cross-domain knowledge through user social information (Jiang et al., 2015). Chen et al. (2013) integrated social information with preference data by sharing implicit cluster-level tensors from multiple domains. Tiroshi et al. (2013) reported a graph-based recommendation method for users with multiple online social networks like Facebook and LinkedIn. Yang et al. (2015) used a bipartite graph to represent the relationships between entities across heterogeneous domains and exploit hidden similarity to help recommendations in two domains.

Except for social network information, there are also lots of user generated tags along with the online systems which can be auxiliary data for CDRS. Abel et al. (2013) used a form-based user profile and tag-based profile together to investigate how the social web can be connected with recommender systems and help with cross-system user modeling. **Tag i**nformed **Co**llaborative **Fi**ltering (TagiCoFi) is a method proposed where a user-item rating matrix and user-tag matrix for the same group of users are used (Zhen et al., 2009). User similarities extracted from shared tags are used to assist the matrix factorization of the original rating matrix. On this basis, **Tag C**ross-**D**omain **C**ollaborative **F**iltering (TagCDCF) extends

TagiCoFi to two domain scenarios each containing data from these two matrixes (Hao et al., 2017). By integrating intra-domain and inter-domain correlations to matrix factorization simultaneously, TagCDCF improves the performance of the recommender system in the target domain.

2.3.2 CDRSs with Non-overlapping Entities

Methods that handle two domains with non-overlapping entities transfer knowledge from a group-level. Users and items are clustered into groups and knowledge is shared through group-level rating patterns. For example, **C**ode **B**ook **T**ransfer (CBT) clusters users and items into groups and extracts group-level knowledge as a "codebook" (Li et al., 2009a). Later, a probabilistic model named **R**ating **M**atrix **G**enerated **M**odel (RMGM) is extended from CBT, relaxing the hard group membership to soft membership (Li et al., 2009b). These two methods cannot ensure that the information on the two groups from two different domains is consistent, and the effectiveness of knowledge transfer is not guaranteed. Gao et al. (2013) extend CBT with common and domain-specific rating patterns and improves the accuracy through adjustment. Zhang et al. (2017) use a domain adaptation technique and extract consistent knowledge from the source domain. This method is superior especially when the statistics between the source domain data and the target domain data are divergent. Moreover, Zhang et al. (2016) extend RMGM with an active learning strategy with a multi-domain scenario. This enables queries across several domains considering both domain-specific and domain-independent knowledge and together benefit recommendation in each domain.

2.3.3 CDRSs with Partially or Fully Overlapping Entities

With the assumption that entities are overlapped between two domains, the source domain and target domain are bridged by constraints on the overlapping entities. Methods dealing with data where the user/item partially or fully corresponds in both domains usually collectively factorize two matrixes in each domain by sharing part of the factorization parameters. **T**ransfer **C**ollective **F**actorization (TCF) (Pan and Yang, 2013) is developed to use implicit data in the source domain to help the prediction of explicit feedback i.e., ratings in the target domain. The assumption of TCF is very strict, the being that users and items must have one-by-one mapping in two domains. Although this method can deal with heterogeneous data, the assumption limits its scope of application in practice. **C**ross-**D**omain **T**riadic **F**actorization (CDTF) models a user-item-domain tensor to integrate both explicit and implicit user feedback (Hu et al., 2013a). The users are fully-overlapping and the user factor matrix is the same, thus bridging all the domains. **C**luster-**B**ased **M**atrix **F**actorization (CBMF) tries to boost CDTF to partially-overlapping entities (Mirbakhsh and Ling, 2015). But the core of CBMF is the same as the methods with non-overlapping entities, which is to transfer knowledge at a group-level rather than using the overlapping entities as a bridge. **R**ating **O**ver **S**ite-**T**ime (ROST) (Li et al., 2015) is similar to the aforementioned two methods, but it also considers user-preference drift in different time-windows. In this situation, users/items are partially or fully overlapped.

In addition to cross-domain recommender systems discussed above, there is other work related to this research. Since the entity correspondence is not always fully available, some strategies are developed to match users or items in two domains. Using latent space matching, unknown user/item mappings are identified

in (Li and Lin, 2014). Sometimes, as the identification of the mapping is time-consuming, an active-learning framework is developed to identify the most valuable entity correspondences in the source domain (Zhao et al., 2017). Lu et al. (2013) pointed out that some source domain data are not consistent with target domain data, thus they proposed an empirical prediction error to select the knowledge to be transferred. However, the problem of inconsistent knowledge between source and target domain is not completely solved, hence further research is needed. This is also a very crucial question for both transfer learning and CDRS research areas.

Table 2.2 Summary of literature reviews of cross-domain recommender systems.

	user/item overlapping				data			task	
	one-to-one overlap	fully overlap	partially overlap	non-overlap	preference data only		other data needed	two domains	multiple domains
					heterogeneous	homogeneous			
Singh and Gordon (2008)		×					×	×	
Jiang et al. (2015)				×			×	×	
Chen et al. (2013)				×			×		×
Tiroshi et al. (2013)				×			×	×	
Yang et al. (2015)				×			×	×	
Abel et al. (2013)		×					×	×	
Hao et al. (2017)				×			×	×	
Li et al. (2009a)				×		×		×	
Gao et al. (2013)				×		×		×	
Li et al. (2009b)				×		×		×	
Zhang et al. (2016)				×		×		×	
Zhang et al. (2017)				×		×		×	
Hu et al. (2013a)		×				×		×	
Mirbakhsh and Ling (2015)			×			×		×	
Pan and Yang (2013)	×				×			×	
Li et al. (2015)		×				×		×	
Zhao et al. (2017)			×			×		×	
Lu et al. (2013)				×		×			×

Chapter 3

A Recommender System by User-preference Drift Detection

3.1 Introduction

Although recommender systems achieve great success in the past, the complex and dynamic data characteristics brought by big data (Wu et al., 2014) are not well handled in recommender systems. Traditional recommender systems assume that user preference is relatively static over a period of time, so users' history records are weighted equally. However, user preferences are changing because of the gradual evolution of their tastes, their personal experiences or the influence of popularity. This is a phenomenon commonly seen in Big Data streams and widely known as concept drift (Kifer et al., 2004). As user history records accumulated, some old user history records may be inconsistent with the user's new requests. Using all the data blindly without selectivity cannot guarantee the accuracy of prediction. For example, a user in MovieLens may register in 2005, and keeps rating movies in the

ten-year time, but his/her preferences may have changed several times during the ten years. As a result, it is not accurate to take all his/her ratings to make an up-to-date recommendation. Recommender systems failing to take it into consideration result in degradation of performance.

To address this issue, time-aware recommender systems are developed (Campos et al., 2014). Most of the methods used in time-aware recommender system tried to accommodate their models to the user-preference drift without detecting the drift. Time-window and instance decaying approaches determine weights on data instances along the time line according to the principle that old data weight less (Yin et al., 2015). Besides penalizing the old data, some methods used dynamic matrix factorization where time is considered to be one more dimension of the data (Chua et al., 2013). Since they failed to detect the change, they cannot determine the changing direction and the proposed adaptation and weighting decay would be biased. The most similar one to this study is (Cao et al., 2009), which is based on the implicit feedback, but the relation between item similarity and explicit rating was not delicately handled in their study.

In real-world recommender systems, except that user preferences keep changing, item features and user behaviors are usually subjective, incomplete and vague (Zenebe et al., 2010). First, the descriptions and documents of items are usually incomplete and ambiguous. Besides, the explicit rating cannot imply user preferences in a certain and direct way. Fuzzy set and fuzzy relation theories are a good way to deal with information uncertainty problem, which can also be adopted to recommender systems. Item features and user preferences were represented by fuzzy sets and fuzzy relations in previous research (Yera and Martinez, 2017), but

how to represent the vague and varied user preferences and make recommendations based on fuzzy user-preference relations is still a problem to be solved.

To deal with the two challenges: 1) handling the uncertainty in item features and user behaviors, 2) modeling the user-preference drift to acquire current user preferences and make recommendations in a dynamic user-preference situation with Big Data, we proposes a fuzzy user-preference drift detection based recommendation method in this chapter. To our best knowledge, this is the first time that user-preference drift is detected and reacted by modeling user preferences with item profiles and explicit ratings.

The chapter is organized as follows. Preliminaries of fuzzy logic are introduced in Section 3.2. Section 3.3 presents the fuzzy user-preference consistency model. Section 3.4 proposes user-preference drift detection method and related algorithms. The fuzzy user-preference drift detection based recommender system prototype is presented in Section 3.5. And the proposed method is tested on both synthetic data and real-world MovieLens dataset.

3.2 Preliminary of Fuzzy Logic

To handle the uncertain issues, the fuzzy set theories are used to model item profiles and to describe the user-preference consistency. Some basic definitions of fuzzy sets are reviewed from (Wang et al., 2009; Zhang and Lu, 2009).

Definition 3.1 (Fuzzy Set) (Zhang and Lu, 2009). A fuzzy set $\tilde{A}$ in a universe of discourse X is characterized by a membership function $\mu_{\tilde{A}}(x)$ which associates with each element $x \in X$ a real number in the interval $[0,1]$. The function value $\mu_{\tilde{A}}(x)$ is termed the grade of membership of x in A.

Definition 3.2 (Fuzzy Relation) (Wang et al., 2009). Let X and Y be two non-empty sets. A mapping $R : X \times Y \to [0,1]$ is called a fuzzy relation from X to Y. For $(x,y) \in X \times Y$, $R(x,y) \in [0,1]$ is referred to as the degree of relationship between x and y. Particularly, a fuzzy relation from X to X is called a fuzzy relation on X.

Definition 3.3 (λ-cut relation of R) (Wang et al., 2009). The crisp relation $R_\lambda = \{(x,y)|R(x,y) \geqslant \lambda\}(\lambda \in [0,1])$ is called the λ-cut relation of R.

Definition 3.4 (Triangular Norm) (Wang et al., 2009). A mapping $\mathcal{T}$ from $[0,1] \times [0,1] \to [0,1]$ is called a triangular norm (t-norm) if it satisfies:

(1) symmetry: $\mathcal{T}(x,y) = \mathcal{T}(y,x)$ whenever $x,y \in [0,1]$;

(2) monotonicity: $\mathcal{T}(x_1,y_1) \leqslant \mathcal{T}(x_2,y_2)$ whenever $x_1 \leqslant x_2$ and $y_1 \leqslant y_2$;

(3) associativity: $\mathcal{T}(\mathcal{T}(x,y),z) = \mathcal{T}(x,\mathcal{T}(y,z))$ whenever $x,y,z \in [0,1]$;

(4) boundary condition: $\mathcal{T}(1,x) = x$ whenever $x \in [0,1]$.

3.3 Fuzzy User-preference Consistency Model

This study aims to detect whether a user-preference drift happened in a user's history records, specific for explicit rating records. In this section, how to model the user-preference consistency is introduced.

3.3.1 Fuzzy User Preferences

In recommender systems, the item set $M = \{m_j | j = 1,2,\ldots,n_M\}$ contains all items that are provided to users. $A = \{a_1,a_2,\ldots,a_{n_A}\}$ is the common item attributes space for item set M. For each item attribute $a_k \in \mathcal{A}, 1 \leqslant k \leqslant n_A$, its value is represented

as a vector $Y_k = (y_{k1}, y_{k2}, \ldots, y_{kn_k})$, where n_k is the total number of the values for attribute a_k. Based on the item description or document, an item can be represented by its item profile, which is defined as follows:

Definition 3.5 (Item profile). Given an item $m_j \in M, 1 \leqslant j \leqslant n_M$, its item profile is defined as a vector of its values for each attribute $a_k \in \mathcal{A}$, denoted as $A_j = (Y_{1,j}, Y_{2,j}, \ldots, Y_{k,j}, \ldots, Y_{n_{A,j}})$, where $Y_{k,j}$ is the value of m_j for a_k. $Y_{k,j} = (y_{k_1,j}, y_{k_2,j}, \ldots, y_{k_l,j}, \ldots, y_{kn_{k,j}})$, where $y_{k_l,j} = 1$ if y_{kl} is a value of item m_j for attribute a_k and $y_{k_l,j} = 0$ otherwise.

For example, for a movie m_j which belongs to drama and comedy, A_j is the movie's profile, which contains attributes like genre (a_1) and cast (a_2). For attribute genre (a_1), say its value set is $Y_k =$ (drama, horror, comedy, sci-fi, action). The value set of m_j for genre (a_1) is $Y_{1,j} = (1,0,1,0,0)$. Sometimes, the item features are subjective. In that condition, item cannot be represented by an item profile described above, so the fuzzy item profile is defined.

Definition 3.6 (Fuzzy item profile). Given an item $m_j \in M, 1 \leqslant j \leqslant n_M$, fuzzy item profile is defined as a vector of its membership degree of m_j to each attribute $a_k \in \mathcal{A}$, denoted as $\tilde{A}_j = (\tilde{Y}_{1,j}, \tilde{Y}_{2,j}, \ldots, \tilde{Y}_{k,j}, \ldots, \tilde{Y}_{n_A,j})$, where $\tilde{Y}_{k,j}$ is a fuzzy sub set of Y_k. Let $\mu_{kl,j}$ be the membership degree of y_{k_l} in $\tilde{Y}_{k,j}, 1 \leqslant k \leqslant n_A, 1 \leqslant l \leqslant n_k$, $\tilde{Y}_{k,j}$ can be represented as $\tilde{Y}_{k,j} = (\mu_{k1,j}/y_{k1}, \mu_{k2,j}/y_{k2}, \ldots, \mu_{kl,j}/y_{kl}, \ldots, \mu_{kn_k,j}/y_{kn_k})$.

For example, Table 3.1 shows the representation of some rated movies by user u. For the movie m_1 and its attribute genre (a_1), say the movie genre is drama, horror, comedy and action, the membership is 1.0, 0.5, 0.1, 0 and 0.1. The fuzzy value set of m_1 for genre (a_1) is $\tilde{Y}_{1,1}$ =(1.0/drama, 0.5/horror, 0.1/comedy, 0/sci-fi, 0.1/action).

Table 3.1 Movie Representation for User u

Movie	Time	Rating	Genre				
			drama	horror	comedy	sci-fi	action
m_1	t_1	$r_1(5)$	1	0.5	0.1	0	0.1
m_2	t_2	$r_2(1)$	1	0	0.9	0.2	0
m_3	t_3	$r_3(4)$	0.2	0.7	0	0	0.6
m_4	t_4	$r_4(1)$	0.1	0	0.8	1	0
m_5	t_5	$r_5(2)$	0.6	0.1	0.6	0	0

In recommender systems, users' history records are usually explicit ratings assigned to items with timestamps. The rating is usually an integer that ranges from 1 to 5. The time-stamp is usually in a date format like "28-12-2014, 09:45:00". According to the form of users' history records, the user rated item and the rated item series are defined:

Definition 3.7 (User rated item and rated item series). A user rated item is defined to be a 3-tuple containing item, its rating and time-stamp denoted as $C(m,r,t)$, where m is the item, r is the rating and t is the time-stamp when the rating is assigned to m. Each user u has a rated item series $C_u = \langle C_1(m_1,r_1,t_1),\ldots,C_i(m_i,r_i,t_i),\ldots,C_{N_u}(m_{N_u},r_{N_u},t_{N_u})\rangle$ where N_u is the number of items user u has rated. For $\forall C_i$, C_{i+1} in C_u, it satisfies $t_i < t_{i+1}$.

Users' ratings to items are determined by their preferences at that time. User preferences cannot be obtained explicitly but can be inferred through user rated items implicitly. The user-preference is formally defined as follows:

Definition 3.8 (User-preference). User-preference is defined as the motivation that determines what item the user will consume and how he/she will like it. Concretely, the **U**ser **P**reference at time **point** (point-UP) is defined as the motivation that determines how user u rates items at time point t denoted as $x_u(t)$. The **U**ser

Preference at time **interval** (interval-UP) is defined as the motivation which determines how user u rates items from time point t_i to time point t_j denoted as $x_u(t_i,t_j) = \bigcup_{t=t_i}^{t_j} x_u(t)$. The notation $\bigcup_{t=t_i}^{t_j} x_u(t)$ denotes that interval-UP is a combination of each point-UP from time point t_i to time point t_j.

Next, how to estimate the point-UP and interval-UP is discussed. The subscript u can be omitted if there is no confusion.

Given a rated item $C_i(m_i,r_i,t_i)$, which is determined by $x(t_i)$, $x(t_i)$ can be estimated by:

$$x(t_i) = \text{argmax}\, P(m_i,r_i|x(t_i)) \tag{3.1}$$

where $P(m_i,r_i|x(t_i))$ is the probability that user u will assign r_i to m_i at time-stamp t_i given point-UP $x(t_i)$.

Corresponding to the rated item series $\langle C_1(m_1,r_1,t_1),\ldots,C_i(m_i,r_i,t_i),\ldots,C_{N_u}(m_{N_u},r_{N_u},t_{N_u})\rangle$, point-UPs in series is represented by $X_u = \langle x(t_1),\ldots,x(t_i),\ldots,x(t_{N_u})\rangle$. Based on point-UPs in series, every two adjacent point-UPs $x(t_i)$ and $x(t_{i+1})$ in X_u can be consistent or not. If they are consistent, they can be merged to form interval-UP $x(t_i,t_{i+1}) = \langle x(t_i),x(t_{i+1})\rangle$. If point-UP from $x(t_i)$ to $x(t_j)$ is consistent, interval-UP is $x(t_i,t_j) = \langle x(t_i),\ldots,x(t_j)\rangle$ corresponding to a rated item series $\langle C_i(m_i,r_i,t_i),\ldots,C_j(m_j,r_j,t_j)\rangle$.

3.3.2 Fuzzy User-preference Consistency Modeling

Based on the definitions related to item profile and user-preference, whether the user-preference is changing or not, i.e., whether the relation between point-UPs is consistent or not, should be defined. In this part, user-preference consistency is defined for preparation of user-preference drift detection.

As the latent point-UPs can be estimated by Equation (3.1), the consistency relation between two point-UPs are correlated with the items that the user rated and the ratings that the user assigned to the rated items. The similarities of the items and ratings are defined below:

Definition 3.9 (Fuzzy item similarity). Fuzzy item similarity is defined as the similarity of the two items m_i and m_j based on their fuzzy item profiles. $S(m_i, m_j)$ is the item profile similarity between item m_i and item m_j, which is calculated as follows:

$$S(m_i, m_j) = \sum_{a_k \in \mathcal{A}} S_{a_k}(m_i, m_j) \times w_{a_k} \tag{3.2}$$

where w_{a_k} is the weight of attribute a_k in attribute set $\mathcal{A}$, and $\sum_{a_k \in \mathcal{A}} w_{a_k} = 1$. The weight can be assigned by domain experts or trained from labeled data. $S_{a_k}(m_i, m_j)$ is the similarity on attribute a_k between item m_i and item m_j, which is calculated by:

$$S_{a_k}(m_i, m_j) = \frac{\sum_{l=1}^{n_k} \mu_{kl,i} \times \mu_{kl,j}}{\sqrt{\sum_{l=1}^{n_k} \mu_{kl,i}^2} \times \sqrt{\sum_{l=1}^{n_k} \mu_{kl,j}^2}} \tag{3.3}$$

where $\mu_{kl,i}$ and $\mu_{kl,j}$ represent the membership degree of y_{kl} in $\tilde{Y}_{k,i}$ and $\tilde{Y}_{k,j}$ respectively, and n_k is the cardinal number of value domain Y_k of attribute a_k.

$S(m_i, m_j)$ equals 1 when the two items are totally the same, and equals 0 the other way around. If the item attributes and values are similar, $S(m_i, m_j)$ approaches to 1. For example, in Table 3.1 the fuzzy item similarity between m_1 and m_2 is 0.7111, meaning that the fuzzy item similarity between these two movies is 0.7111.

Definition 3.10 (Rating similarity). Rating similarity is defined as the similarity measure between two rating values r_i and r_j. It is calculated as follows:

$$R(r_i, r_j) = 1 - \frac{|r_i - r_j|}{r_{max} - r_{min}} \tag{3.4}$$

where r_{max} and r_{min} are the highest rating and lowest rating that a user can assign in the recommender system.

$R(r_i, r_j)$ equals 1 when the two ratings are totally the same, and equals 0 the other way around. If the ratings are similar, $R(r_i, r_j)$ approaches to 1. For example, in Table 3.1 the rating similarity between m_1 and m_2 is 0, meaning that the two ratings assigned by user u are totally different.

As the latent point-UPs estimated by Equation (3.1), the consistency relation between two point-UPs are correlated with the two similarity measures defined above. Since the user preferences are subjective and usually cannot sharply change, therefore the relation between two point-UPs is not a crisp relation but a fuzzy relation, which is defined as fuzzy point-UP consistency relation.

Definition 3.11 (Fuzzy point-UP consistency relation). Given point-UPs in series $X_u = \langle x(t_1), \ldots, x(t_i), \ldots, x(t_N) \rangle$, a point-UP set is $I_u = \{x(t_i) | i = 1, 2, \ldots, N_u\}$, where N_u is the number of items user u has rated. Fuzzy point-UP consistency relation is a fuzzy relation on point-UP set I_u and represents the consistency relation between two point-UPs in I_u. The membership function is $P(x(t_i), x(t_j))$. λ-cut of fuzzy point-UP consistency relation is:

$$P_\lambda(x(t_i), x(t_j)) = \begin{cases} 1 & P(x(t_i), x(t_j)) > \lambda \\ 0 & P(x(t_i), x(t_j)) < \lambda \end{cases} \tag{3.5}$$

where λ is a parameter ranges in $[0, 1]$. After the λ-cut operation, point-UP consistency relation becomes a crisp relation.

Next, how to calculate point-UP consistency degree $P(x(t_i), x(t_j))$ is introduced. The point-UP consistency degree is constrained by the following two rules:

Rule 1: IF items are similar AND the ratings assigned to the two items are similar, THEN the point-UPs are consistent.

Rule 2: IF items are similar AND the ratings assigned to the two items are not similar, THEN the point-UPs are inconsistent.

To calculate the point-UP consistency degree by applying the two rules, which contain uncertain linguistic terms, we first introduce fuzzy similar relations on items and ratings to handle these linguistic terms. The fuzzy item similarity relation is defined as a fuzzy relation on item set $M = \{m_j | j = 1,2,\ldots,n_M\}$. The membership degree of the item similarity relation is calculated by the fuzzy item similarity $S(m_i,m_j)$ defined in *Definition 3.9*. The fuzzy rating similarity relation is defined as a fuzzy relation on rating set $R = \{r_j | j = 1,2,\ldots,n_R\}$. The membership degree of the rating similarity relation is calculated by the rating similarity $R(r_i,r_j)$ defined in *Definition 3.10*.

To implement one rule, one fuzzy implication relation is established. Given an antecedent and the implication relation, the consequent is inferred through fuzzy inference (Gupta and Qi, 1991). Based on that, we can obtain the degree of positive point-UP consistency and the degree of negative point-UP consistency, denoted by $P^+(x(t_i),x(t_j))$ and $P^-(x(t_i),x(t_j))$ respectively. $P^+(x(t_i),x(t_j))$ and $P^-(x(t_i),x(t_j))$ are inferred by, with a t-norm $\mathcal{T}$,

$$P^+(x_i,x_j) = \mathcal{T}(S(m_i,m_j),R(r_i,r_j)) \tag{3.6}$$

$$P^-(x_i,x_j) = \mathcal{T}(S(m_i,m_j),1-R(r_i,r_j)) \tag{3.7}$$

Table 3.2 Special value for fuzzy point-UP consistency relation.

Rating similarity \ Item similarity	Similar $R(r_i,r_j)=1$	Not similar $R(r_i,r_j)=0$
Similar $S(m_i,m_j)=1$	$P^+(x(t_i),x(t_j))=1$ $P^-(x(t_i),x(t_j))=0$	$P^+(x(t_i),x(t_j))=0$ $P^-(x(t_i),x(t_j))=1$
Not Similar $S(m_i,m_j)=0$	$P^+(x(t_i),x(t_j))=0$ $P^-(x(t_i),x(t_j))=0$	$P^+(x(t_i),x(t_j))=0$ $P^-(x(t_i),x(t_j))=0$

To better elaborate the relations defined above, the boundary values of $S(m_i,m_j)$ and $R(r_i,r_j)$ in $[0,1]^2$ and their corresponding point-UP coherence and incoherence degrees are listed in Table 3.2.

Finally, the membership function of fuzzy point-UP consistency relation $P(x(t_i),x(t_j))$ is computed by:

$$P(x(t_i),x(t_j)) = (1-\beta)P^+(x(t_i),x(t_j)) + \beta(1-P^-(x(t_i),x(t_j))) \tag{3.8}$$

where $\beta \in [0,1]$ is a weighting parameter.

For example, in Table 3.1 the fuzzy item similarity between m_1 and m_2 is 0.7111. The rating similarity between them is 1. If we use the commonly used minimum t-norm defined as: $\mathcal{T}(a,b) = \min(a,b)$, then $P^+(x(t_1),x(t_2)) = 0.7111$, $P^-(x(t_1),x(t_2)) = 0$, meaning that the degree of positive point-UP consistency is 0.7111, and the degree of negative point-UP consistency is 0. Suppose $\beta = 0.5$, then $P(x_1,x_2) = 0.8555$, the degree of point-UP consistency is 0.8555, indicating that relation between point-UPs $x(t_1)$ and $x(t_2)$ is highly consistent. Suppose $\lambda = 0.5$, then $P_{0.5}(x(t_1),x(t_2)) = 1$, indicating that point-UPs $x(t_1)$ and $x(t_2)$ implied by rated items $C_1(m_1,r_1,t_1)$ and $C_2(m_2,r_2,t_2)$ are consistent.

3.4 User-preference Drift Detection

To deal with concept drift phenomenon in recommender systems, the drift detection is adopted on the existing data in the database to see whether the user preferences remain unchanged during a period of time. In this Section, a user-preference drift detection method is proposed based on the fuzzy user-preference consistency model. First, the test statistic interval-UP density decrement for detection is defined. The statistical guarantee is then discussed. Finally, the overview detection algorithm based on the first two parts is presented.

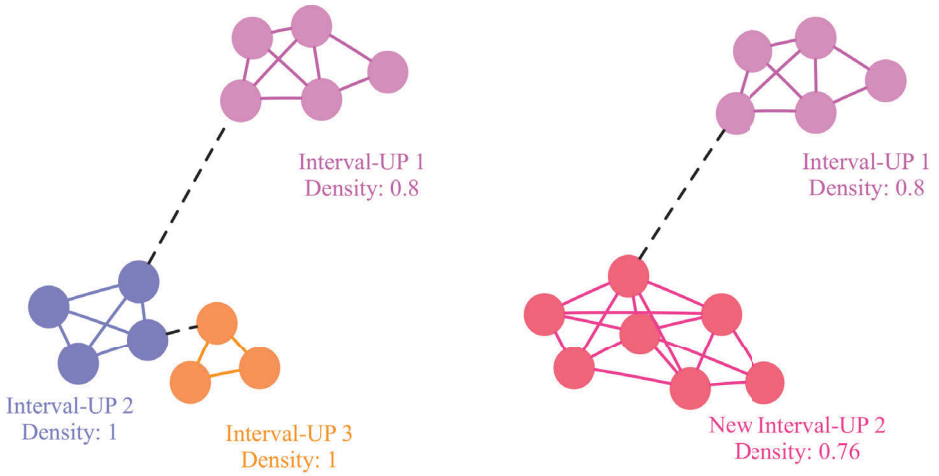
3.4.1 Interval-UP Density Decrement

Recall that in *Definition 3.11*, if two adjacent point-UPs are consistent, they can be merged to form an interval-UP. The consistency relation between two interval-UPs should also be evaluated to determine if they are consistent or not. To calculate interval-UP consistency relation, the interval-UP density is defined.

Definition 3.12 (Interval-UP density). Given an interval-UP $x(t_i,t_j) = \langle x(t_i),\ldots,x(t_j)\rangle$, the corresponding interval-UP graph is $G = (V,E)$, where $V = \{x(t_k)|(i \leqslant k \leqslant j)\}$ and $E = \{(x(t_k),x(t_l))|P_\lambda(x(t_k),x(t_l)) = 1\}$. The interval-UP density is calculated as:

$$\mathcal{D}(x(t_i,t_j)) = \frac{|E|}{max_E} \tag{3.9}$$

where $|E|$ is the number of edges contained in G and $max_E = \binom{2}{j-i+1}$ is the max number of possible edges in G. Specially, to calculate the density decrement between adjacent point-UP and interval-UP, the default density of point-UP is set to 1.



(a) Point-UP graph before merge (b) Point-UP graph after merge

Figure 3.1 An example of point-UP graph

Based on the definition above, the more consistent point-UPs are in one interval-UP, the higher interval-UP density is. An illustrative example is shown in Figure 3.1. If the point-UPs are all connected in one interval-UP, then the density of the interval-UP is 1, indicating that these point-UPs are all consistent.

If the two interval-UPs are consistent, the density of the combined interval-UP should have a high value. To determine whether two interval-UPs can form a new one, the interval-UP density decrement of two interval-UPs is defined as follows:

Definition 3.13 (Interval-UP density decrement). Given two adjacent interval-UPs $x(t_i,t_k) = \langle x(t_i),\ldots,x(t_k)\rangle$ and $x(t_{k+1},t_j) = \langle x(t_{k+1}),\ldots,x(t_j)\rangle$, let $x(t_i,t_j) = \langle x(t_i),\ldots,x(t_k),x(t_{k+1}),\ldots,x(t_j)\rangle$. The interval-UP density decrement of $x(t_i,t_k)$ and $x(t_{k+1},t_j)$ is defined as:

$$d(x(t_i,t_k),x(t_{k+1},t_j)) = \frac{\mathcal{D}(x(t_i,t_k)) + \mathcal{D}(x(t_{k+1},t_j))}{2} - \mathcal{D}(x(t_i,t_j)) \tag{3.10}$$

Interval-UP density decrement evaluates the consistency between two interval-UPs. As shown in Figure 3.1, if two interval-UPs are not consistent, the density of new interval-UP will be relatively low compared with the sum of the two interval-UPs before the merge. In this figure, interval-UP 2 can be merged with interval-UP 3 but cannot be merged with interval-UP 1. A user-preference drift happens between the two interval-UPs once the decrement is greater than ε similar to (Kifer et al., 2004). If drift happens, the two interval-UPs should not be merged. How to determine ε will be discussed in the next part.

3.4.2 Statistical Guarantee

There are two aspects of concept drift detection, one is the test statistic, the other is to provide statistical significance of the detected change. The hypothesis test is to determine whether the decrease of interval-UP density is statistically significance to say the two interval-UPs are inconsistent (i.e., a drift happens). The null hypothesis is "no drift happens (the two interval-UPs can be merged to a new one)". Interval-UP density decrement is the test statistic in the hypothesis test. Here, permutation test is conducted similar to (Lu et al., 2014) to provide statistical guarantee with a significant level α. One calculation of interval-UP density decrement is one observation denoted as $\hat{\theta}$. α is the probability of obtaining an observation as extreme as $\hat{\theta}$. Usually, the significant value is set to be 0.05, suggesting that the false positive of the test is 0.05.

Under the null hypothesis, given two interval-UP $x_1(t_1, t_p)$ containing p point-UPs and $x_2(t_{p+1}, t_{p+q})$ containing q point-UPs, they can be merged and form a new interval-UP $x_{new}(t_1, t_{p+q})$ containing $p+q$ point-UPs. One permutation is to randomly sample p point-UPs to form the interval-UP $x'_1(t_1, t_p)$ without replacement,

the remaining q point-UPs are to form the interval-UP $x'_2(t_{p+1}, t_{p+q})$. For the new clique $x'_1(t_1, t_p)$ and $x'_2(t_{p+1}, t_{p+q})$, test statistic $d(x'_1(t_1, t_p), x'_2(t_{p+1}, t_{p+q}))$ is calculated for this permutation. The permutations are repeated for Z times. The larger Z is, the more accurate the Monte Carlo simulation is. In the experiment, Z is set to 500. The threshold ε of the test statistic is calculated by:

$$\alpha \approx \hat{\alpha} = \frac{\#(\varepsilon \geqslant \hat{\theta})}{Z} \tag{3.11}$$

3.4.3 Interval-UP Density Decrement-based Drift Detection Method

Given $X_u = \langle x(t_1), \ldots, x(t_i), \ldots, x(t_N) \rangle$, to determine whether point-UP is consistent from $x(t_1)$ to $x(t_N)$, two methods are straightforward.

One is called splitting method, which takes N point-UPs as a whole and checks whether to split it at $N-1$ positions. Intuitively, we can check the $N-1$ positions and calculate the difference between the two sub-series. The other is called merging method, which takes N point-UPs as N nodes and check whether two or several of them can be merged to be interval-UP. In real-world situations, the drift of user-preference can happen more than once in the series. And the splitting method may cause error detection if user-preference in sub-series is not consistent. So the merging method is used in this study.

Algorithm 3.1 shows the process of user-preference drift detection. The original point-UP series is represented as $X_u = List(x(t_1), \ldots, x(t_i), \ldots, x(t_N))$ where point-UPs are arranged in time order. Function $\boldsymbol{d}$ is density decrement measure for two interval-UPs and function $\boldsymbol{pt}$ is permutation test for two interval-UPs. The detail of these two functions are described in Section 3.4.1 and Section 3.4.2, respectively.

Algorithm 3.1: User-preference Drift Detection

Input:
$X_u = List(x(t_1),\ldots,x(t_i),\ldots,x(t_N))$, the original point-UP series
Output:
$\mathcal{X}_u$, the final interval-UP series

1: $\mathcal{X}_u = List(C_1,\ldots,C_i,\ldots,C_N)$, $C_i = List(x(t_i))$
2: **while** $ListSize(\mathcal{X}_u) > 1$ **do**
3: **for** each $i \in ListSize(\mathcal{X}_u)$ **do**
4: $List(C_i, C_{i+1}) = \text{argmin } d(C_i, C_{i+1})$
5: $\varepsilon = pt(C_i, C_{i+1})$
6: **if** $d(C_i, C_{i+1}) < \varepsilon$ **then**
7: $\mathcal{X}_u \leftarrow (\mathcal{X}_u - C_i)$
8: $\mathcal{X}_u \leftarrow (\mathcal{X}_u - C_{i+1})$
9: $\mathcal{X}_u \leftarrow (\mathcal{X}_u \bigcup List(C_i, C_{i+1})$
10: **end if**
11: **if** $i = |\mathcal{X}_u|$ **then**
12: **break while**
13: **end if**
14: **end for**
15: **end while**
16: **return** $\mathcal{X}_u$

In the algorithm, line 1 is to initialize the original point-UP series. Lines 2-15 repetitively merge the consistent point-UPs or interval-UPs. Concretely, lines 4 and 5 are calculating the interval-UP decrement and the threshold ε for determining whether the two interval-UPs can be merged or not. The function $\boldsymbol{argmin}$ in line 4 is to find the smallest density decrement of all the paired interval-UPs. Lines 6-10 are to update the list of user-preferences. Lines 11 to 13 are to judge whether the circulation should be terminated. If all the interval-UPs are detected and any two of them cannot be merged, then the circulation is terminated even there are more than two interval-UPs in the list.

3.5 Fuzzy User-preference Drift Detection based Recommender System

3.5.1 System Overview

In this subsection, fuzzy user-preference drift detection based recommender system is developed. The proposed user-preference drift detection method is applied to prune the out-of-date preference of user and acquire up-to-date user-preference. The developed fuzzy user-preference drift detection based recommender system is to understand current user-preference and provide recommendations to meet current user-preference. Figure 3.2 shows the architecture of the recommender system that adapts to the detected user-preference drift.

For each user, based on his/her history rating records, user-preference drift detection is conducted in user-preference drift detection module. Several interval-UPs will be acquired after the detection which are implied by the rated item series. The most recent interval-UP is the current user-preference. Its corresponding sub-series of the rated item series is acquired for the recommendation. Once a user requests recommendation, the remaining sub-series acquired in the detection module will be used to provide recommendation rather than using the whole user history records. The recommendation technique can be content-based, CF-based or any hybrid method. Experiments implemented in the next section prove that the user-preference drift detection module will remarkably improve the accuracy of recommendation.

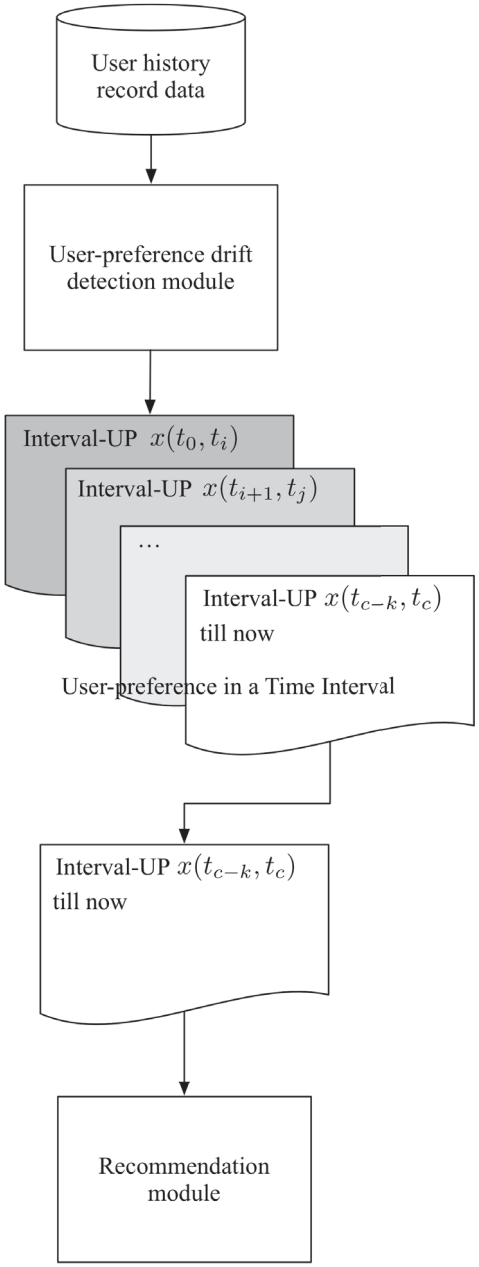


Figure 3.2 Architecture of fuzzy user-preference drift detection-based recommender system

3.5.2 Experiment

In this subsection, experiments are conducted to evaluate the performance of the user-preference drift detection based recommender system. The evaluation contains

two parts, one is about the drift detection method on synthetic data. The other is about the drift detection based recommendation on real-world data. Experiment results show that the proposed method can improve the performance of content-based and CF based recommendation.

3.5.2.1 Experiment Setup

The experiment is implemented on MovieLens 1M [1] dataset . It includes 1 million ratings of 6040 users for 3852 movies with timestamps. Each user rated at least 20 movies. There are 18 genres in this dataset. The value set for item attribute genre $Y_{genre} = \{$Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western$\}$. Each movie is associated with at least one genre. 20% of most recent ratings are taken as test set.

For the content-based and CF-based recommender systems, they rely on the predicted ratings to provide recommendation. So it is suitable to choose rating prediction-based accuracy to evaluate the prediction result. Concretely, in this study, the **M**ean **A**bsolute **E**rror (MAE) (Herlocker et al., 2004) is used to evaluate the prediction accuracy. The equation to calculate MAE is as follows:

$$MAE = \sum_{u,v,X_{uv} \in Y} \frac{|\hat{X}_{uv} - X_{uv}|}{|Y|} \tag{3.12}$$

where $\hat{X}_{uv}$ is the predicted rating of user u on item v, X_{uv} is the actual rating, Y is the test set, and $|Y|$ is the number of test ratings. The smaller the errors, the better the performance.

[1] http://grouplens.org/datasets/movielens/1m/

Table 3.3 Synthetic item similarities for user-preference drift detection.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
m_1	/	0.9749	0.9814	0.9912	0.9880	0.9943	0.9806	0.9814	0.9822	0.9910
m_2		/	0.9598	0.9874	0.9764	0.9834	0.9758	0.9790	0.9916	0.9830
m_3			/	0.9678	0.9952	0.9816	0.9800	0.9782	0.9654	0.9742
m_4				/	0.9846	0.9898	0.9901	0.9678	0.9940	0.9834
m_5					/	0.9923	0.9899	0.9768	0.9855	0.9776
m_6						/	0.9778	0.9817	0.9923	0.9840
m_7							/	0.9591	0.9827	0.9725
m_8								/	0.9654	0.9940
m_9									/	0.9730
m_{10}										/

3.5.2.2 Evaluating User-preference Drift Detection Method

Since the real-world data do not contain the label of whether user-preference drift happens in one user's data, user-preference drift detection method is evaluated on synthetic data. There are 10 items in the synthetic data and all of them are similar to each other as shown in Table 3.3. The similarity between every two items are above 0.9. There are 3 users to be evaluated whether user-preference drift happens according to their rated items as shown in Table 3.4. The rated item for each user are in chronological order, suppose from m_1 to m_{10}. For u_1, the average rating of m_1 to m_5 is 4.6, while the average rating of m_6 to m_{10} is 1.6, which is a big difference. For u_2, the average rating of m_1 to m_5 is 4.4, while the average rating of m_6 to m_{10} is 4, which is almost the same. For u_3, the average rating of m_1 to m_5 is 4.6, while the average rating of m_6 to m_{10} is 4, which is a minor difference. It is supposed that u_1 and u_3 has changed their preferences after assigning rating of m_5 to different extent, but u_2 has remained his/her preferences all the time. The drift detection result of the proposed method is the same as assumed above. The user-preference drift detection method is able to detect the change of user-preference as expected.

Table 3.4 Synthetic user ratings for user-preference drift detection.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
u_1	5	5	4	4	5	1	2	1	1	3
u_2	4	4	5	4	5	3	5	4	4	4
u_3	5	4	5	4	5	3	4	3	4	3

Table 3.5 Comparison of UDD and a traditional method on Movielens Dataset

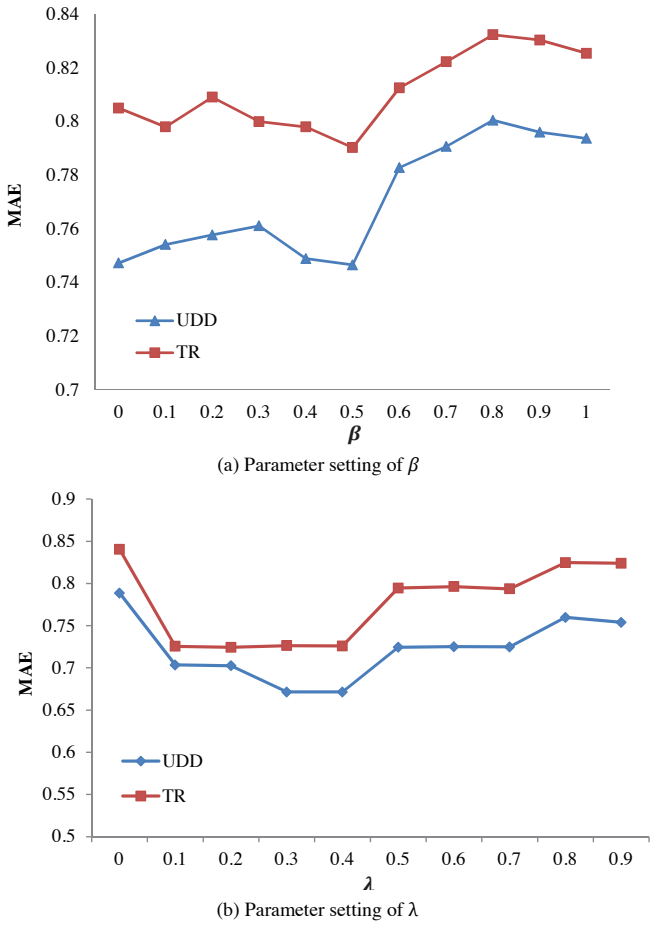
	Content-based	CF-based
TR	0.7456	0.7263
UDD	0.6948	0.6705
Improvement	6.81%	7.68%

3.5.2.3 Result and Parameter Analysis

The experiment on MovieLens dataset is to evaluate how the user-preference drift detection method will affect the performance of recommender systems. Corresponding to different recommendation techniques, item similarity in the experiment is calculated in two ways. For content-based technique, item similarity is obtained through the item profile; For CF-based technique, item similarity is calculated by item ratings.

Parameters setting in the experiment is $\beta = 0.5$, $\lambda = 0.4$. Prediction accuracy of experiment on MovieLens dataset is shown in Table 3.5. **U**ser-preference **D**rift **D**etection (UDD) is our proposed method. TR is traditional CF-based recommendation method. The UDD can improve the performance of recommendation by more than 5% of MAE either on content based or CF-based recommendation technique.

Parameter β is to control the weights of $P^+(x(t_i),x(t_j))$ and $P^-(x(t_i),x(t_j))$ in Equation (3.6). The bigger β is, the more weight is assigned to $P^-(x(t_i),x(t_j))$. To better understand how this parameter affect the performance of recommendation, β



(a) Parameter setting of β

(b) Parameter setting of λ

Figure 3.3 Parameter analysis of user-preference drift detection-based method.

is set from 0 to 1, 0.1 as a step. As shown in Figure 3.3, the best performance is achieved at $\beta = 0.5$. This result shows that $P^{+}(x(t_i),x(t_j))$ and $P^{-}(x(t_i),x(t_j))$ are both important and indispensability for calculating $P(x(t_i),x(t_j))$. Also, no matter how the parameter is assigned, the UDD is always better than traditional method. Parameter λ is to control the cut boundary of user-preference consistency relation. If $\lambda = 0$, it means that all user-preference consistent degree is 0. If $\lambda = 1$, then user-preference is consistent all time long so that the recommendation is based on the whole dataset. As shown in Figure 3.3 (b), the best performance is achieved at

$\lambda = 0.4$. Also, no matter how the parameter is assigned, the UDD is always better than traditional method.

3.6 Summary

Modeling user-preference drift is a challenge since the drift of user preferences may occur in different directions for each user in a distinct time period. Moreover, item features and user behaviors are uncertain, making the modeling even harder to be accurate and practical. In this chapter, a method to model fuzzy user-preference and detect the user-preference drift in user history records is proposed. By detecting and pruning out-of-date user-preferences, the performance of the fuzzy user-preference drift detection based recommender system is improved. The proposed method can be applied to any kind of recommender systems such as content-based, CF-based and other hybrid recommender systems. Experiments are conducted on both synthetic data and real-world MovieLens data to evaluate the drift detection method and the performance of recommender system. Both results show that the proposed method increases the recommendation accuracy as measured by MAE metric compared with recommender systems that are not sensitive to user-preference drift.

Chapter 4

A Cross-domain Recommender System with Consistent Information Transfer

4.1 Introduction

Sparsity, or the cold-start problem, remains the most challenging outstanding issue in collaborative filtering (Kim et al., 2011). If a system fails to provide practical support, new users will quickly lose interest and stop using it (Yoon et al., 2013). To solve the cold-start problem, traditional methods aim to find additional information, such as social network (Li et al., 2014), trust (Li et al., 2013) or reviews (McAuley and Yang, 2016) from within the same domain to infer user-item relationships. Unfortunately, additional information is not often available. However, where there is insufficient data in one domain, such as movies, but relatively rich data in another domain, such as books. In this way, a newly launched recommender system

in one domain is able to benefit from a mature recommender system in another domain. Such systems are known as a CDRS (Pan, 2016). Because of advantages of collaborative filtering, such as its high efficiency and its lack of content restrictions, CDRSs provide relatively high-quality recommendation together with the ability to deal with cold start problems.

CDRSs aim to use information from an alternative source domain in the target domain where sufficient preference data is unavailable. Referring to Chapter 2, the proposed method falls in the class dealing with situations where there are no intersections between the two domains (Li et al., 2009a; Lu et al., 2015a). This scenario is more widely seen in real-world applications. Sharing user ID from different data sources is almost impossible due to confidential user information.

Existing CDRS methods for preference data without intersections between two domains use shared information of users and items despite a lack of direct corresponding between domains. For example, a group of well-clustered users implies similar preference information, and a group of well-clustered items implies similar content information. From such groups, a user group to item group rating pattern, defined as group-level knowledge, can be extracted and shared as a compressed form of the original user-item rating matrix. These methods partly alleviate the sparsity problem and increase the prediction accuracy of recommender systems in target domain. However, none positively transfer knowledge to the target domain in a stable manner, which reduces the accuracy of the recommendations when there is shift between domains. Some methods are prone to failure because they use the group-level knowledge matrix directly without ensuring that the consistency of the user/item group information is maintained during transfer. Without collectively clustering or adjusting the group-level knowledge, it usually

diverges between domains. Obviously, integrating inconsistent knowledge into the target domain causes harm, rather than helping the recommender system. By ensuring the consistency of the knowledge transferred between the domains, we aim to increase the prediction accuracy of CDRSs and overcome some general problems associated with domain shift in real-world decision-making applications.

In this chapter, we investigate how to effectively transfer knowledge from the source rating matrix to help increase the prediction accuracy of the recommender system on the target rating matrix. To avoid divergence caused by domains, group-level knowledge is extracted on the basis of consistent user/item group information. That is, user/item information should be consistent in each corresponding group from source and target domain. A domain adaptation technique regulates user/item group information in both domains. Then group-level knowledge is learned to maximize the overall level of fitting in both domains. Thus, we propose a knowledge transfer method for cross-domain recommender systems by Consistent Information Transfer (CIT) with non-overlapping entities.

The remainder of the chapter is organized as follows. Section 4.2 formally defines the problem solved. In Section 4.3, we present our CIT method in three parts: an overview, the steps, and the conceptual framework of the cross-domain recommender system. Section 4.4 presents the empirical experiments on five real-world datasets spanning three categories of data. The results for nine tasks in terms of three data sparsity ratios show that our method is better than five existing non-transfer and cross-domain methods. Finally, guidelines for recommender system developers along with a discussion on the potential industry applications of the proposed method are provided in Section 4.5. Finally, the summary is provided in Section 4.6.

4.2 Problem Formulation and Motivation

In this section, a factorization view of the recommender system in one domain is given to clearly describe the problem setting. The problem under study in this chapter is then formally described. Finally, the motivation of this research is given as an example.

4.2.1 Recommendation Task based on Tri-factorization in One Domain

In a single domain, suppose there are M users and N items. The relationship between the users and the items is represented by the user-item rating matrix $\boldsymbol{X} \in \mathbb{R}^{M \times N}$ (bold letters represent a matrix). Any rating r_{ij} in $\boldsymbol{X}$ is subject to $r_{ij} \in \{1,2,3,4,5,?\}$ ("?" denotes a missing value). To construct the group-level knowledge matrix, users and items are clustered. The rating matrix $\boldsymbol{X}$ can be factorized into three matrixes (Ding et al., 2006): $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{M \times K}$ is the user-group membership matrix, $\boldsymbol{V} \in \mathbb{R}^{N \times L}$ is the item-group membership matrix, and $\boldsymbol{S} \in \mathbb{R}^{K \times L}$ is the group-level knowledge matrix. Each row of $\boldsymbol{U}$ and $\boldsymbol{V}$ contains the memberships of the user/item entity for all groups. $\boldsymbol{S}$ is the rating pattern of each user group to each item group. The recommendation task requires the prediction of user ratings for items where the rating values are not known. To calculate the missing values, the user-item rating matrix is reconstructed through $\hat{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$. Tri-factorization of $\boldsymbol{X}$ minimizes the loss function $\mathcal{L}(\boldsymbol{X}, \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T)$, which measures the error of prediction. Since $\boldsymbol{X}$ is usually sparse, the loss function

is in a weighted form as follows:

$$\mathcal{L}(\boldsymbol{X}, \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T) = \|\boldsymbol{I} \circ (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T)\|_F \tag{4.1}$$

where $\boldsymbol{I}$ is the rating indicator matrix representing whether the rating in $\boldsymbol{X}$ is observed or not, $I_{ij} \in \{0,1\}$. $I_{ij} = 1$ indicates that the rating is observed and $I_{ij} = 0$ otherwise and $\circ$ denotes the Hadamard product of matrixes.

Thus, in single-domain recommendation, $\Theta = \{\boldsymbol{U}, \boldsymbol{S}, \boldsymbol{V}\}$ are the parameters the recommender system uses to predict the ratings and provide a recommendation. The tri-factorization is:

$$\min_{\boldsymbol{U},\boldsymbol{S},\boldsymbol{V}} \mathcal{L}(\boldsymbol{X}, \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T)$$

$$\text{s. t. } \boldsymbol{U} > 0, \boldsymbol{S} > 0, \boldsymbol{V} > 0$$

4.2.2 Cross-domain Transfer Learning Recommender System

As mentioned in the Introduction, users and items are usually denoted by de-identified user and item IDs. It is often difficult to find an explicit correlation between the two domains. In this problem setting, the users/items have no correspondence across the domains and are treated as completely different users/items. We assume that explicit rating data are available for both the source and target domains. Formally, the problem is defined as:

Definition 4.1 (Cross-domain Transfer Learning Recommender System). Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, a cross-domain transfer learning recommender system aims to help recommendation tasks in the target domain predict the rating $\hat{\boldsymbol{X}}_t = \boldsymbol{U}_t \boldsymbol{S}_t \boldsymbol{V}_t^T$ using knowledge in the source rating matrix $\boldsymbol{X}_s$ and $\Theta_s = \{\boldsymbol{U}_s, \boldsymbol{S}_s, \boldsymbol{V}_s\}$, where $\mathcal{U}_s \cap \mathcal{U}_t = \emptyset$ and $\mathcal{I}_s \cap \mathcal{I}_t = \emptyset$.

$\mathcal{U}_s$ and $\mathcal{I}_s$ represent the user set and item set in the source domain, while $\mathcal{U}_t$ and $\mathcal{I}_t$ represent the user set and item set in the target domain.

4.2.3 Motivation for Developing CIT

A CDRS for movies serves as a good example for describing this problem. Consider three movie rating websites. Two sites focus on classic movies (the source domain and target domain 1); the other only contains second-rate movies (target domain 2). Figure 4.1 illustrates three scenarios.

Scenario 1: Users 1-4 in Figure 4.1 (a) and users 7-10 in Figure 4.1 (b). Although the chosen movies have different origins, all the movie subsets from the source domain and target domain 1 are quite similar. Users 1-4 in the source domain and users 7-10 in the target domain 1 also have similar movie preferences; hence, the user and item groups contain similar information in the source and target domains. In this first scenario, using the group-level knowledge directly in the target domain is effective even though there is no group-matching module.

Scenario 2: Users 5, 6 in Figure 4.1 (a) and users 11, 12 in Figure 4.1 (b). UG6 in target domain 1 has completely different information to UG3 in the source domain. Because the group-level knowledge is inconsistent (here, due to the user preference information), directly transferring that knowledge from source domain to target domain will impair the performance of the CDRS.

Scenario 3: Users 1-6 in Figure 4.1 (a) and users 13-18 in Figure 4.1 (c). As in scenario 2, UGs 7-9 have completely different group information from UGs 1-3 in the source domain, as is the case with IG 1-3 and IG 7-9. Here, both the user preference information and the item content information are inconsistent. As a result, using knowledge extracted from the source domain in target domain 2 may

produce even poorer recommendations than from a recommender system that was built solely from target domain 2.

These scenarios reflect the knowledge inconsistency problem that existing CDRSs are unable to deal with. Using knowledge from another domain without mapping and adjustment only helps to produce a more accurate prediction if there is no significant divergence between the source domain and target domain. The CIT method, described in the following section, helps to solve the problem.

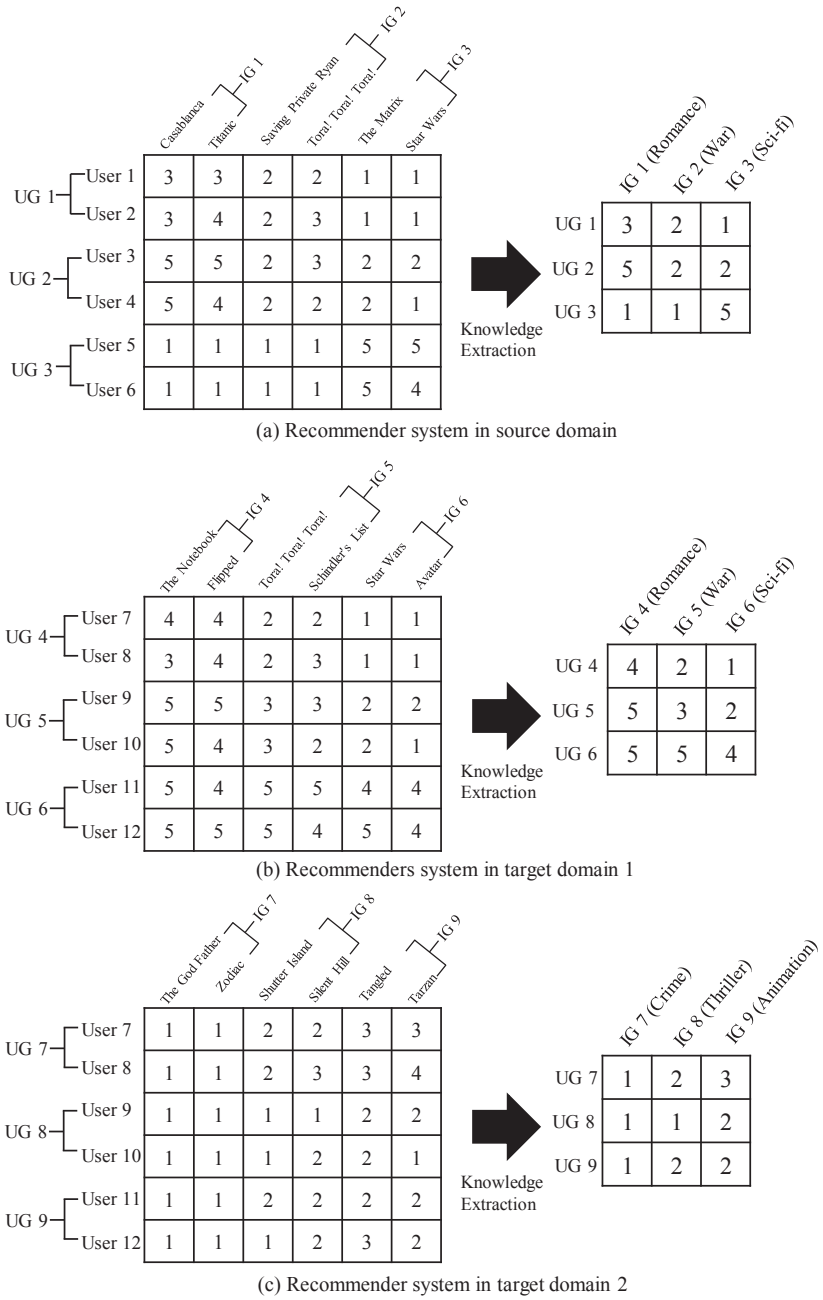


Figure 4.1 An example for CDRS

(a)-(c) Recommender systems for a source domain, target domain 1 and target domain 2. The left side shows the schematic rating matrixes; the right side shows the group-level knowledge matrixes. These represent possible groups of users and items from the left-side rating matrixes. The possible user/item group semantic meanings are annotated as UG - user group and IG - item group.

4.3 A CDRS with Consistent Information Transfer

This section introduces our CIT method beginning with an overview of the entire procedure. Each of the five steps of the method are then presented in detail followed by the system architecture to support decision-making for individuals and businesses.

4.3.1 CIT Method Overview

The proposed CIT method uses a domain adaptation technique to ensure that knowledge extracted from the source domain is consistent with the target domain and that knowledge transfer is positive. The procedure consists of five steps, as shown in Figure 4.2. 1). Users/items from the source and target domains are clustered separately into groups. 2). Domain adaptation techniques are used to generate consistent user/item latent groups in the source and target domains. 3). Consistent knowledge is extracted from the latent groups. 4). Group representations in the target domain are adjusted to retain their domain-specific characteristics. 5). A recommender system for the target domain is built. We use a specific algorithm for each step, but other clustering or domain adaptation algorithms could be substituted.

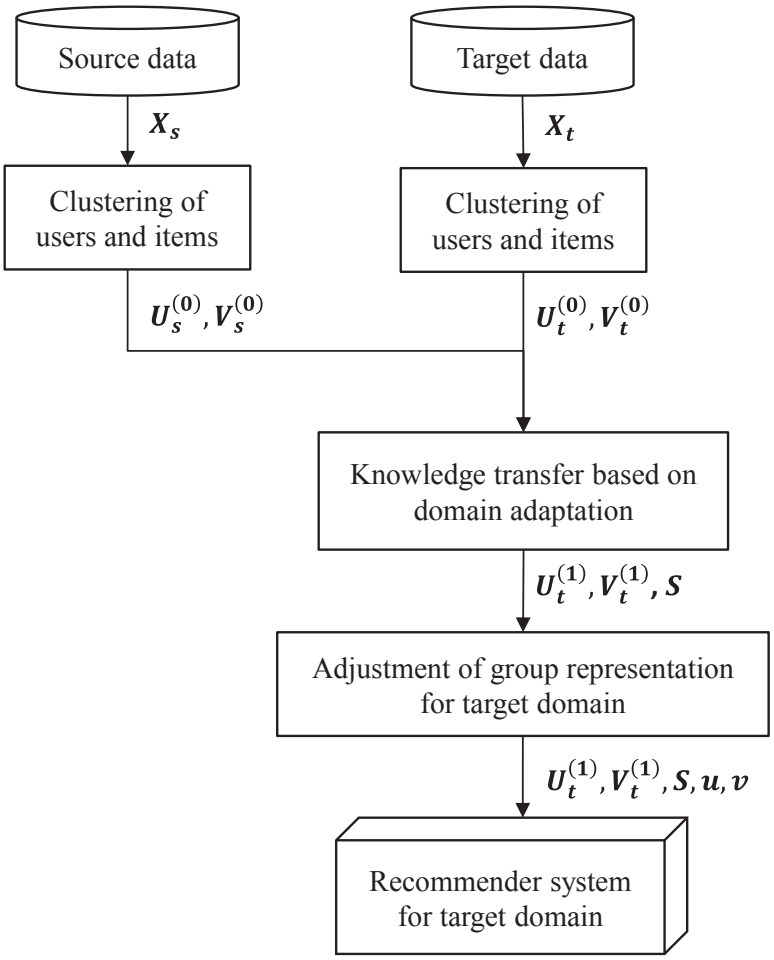


Figure 4.2 The CIT method procedure

Note: The notations in the figure correspond to the equations that follow in this section.

4.3.2 CIT Method

Our proposed CIT method consists of five steps.

4.3.2.1 Step 1: Clustering of users and items in both domains

This step clusters users and items into groups. Clustering users and items appropriately is a crucial issue. Intuitively, users may have various preferences and items may have diverse content. Therefore, it is usually more appropriate to allow both users and items to fall into multiple groups with different memberships. Thus, in this chapter, a **F**lexible **M**ixture **M**odel (FMM) is used to cluster the users and items separately (Si and Jin, 2003). The same clustering procedure is used for both the source domain and the target domain; however, for simplicity, we have only provided the description for one domain.

Suppose users are clustered into K user groups $\{Z_u^{(1)}, \ldots, Z_u^{(K)}\}$, while items are clustered into L item groups $\{Z_v^{(1)}, \ldots, Z_v^{(L)}\}$. Z_u and Z_v are two latent variables that denote the user and item groups respectively. $P(Z_u|u)$ is the conditional probability of a user belonging to a user group, denoting the group membership of the user; $P(Z_v|v)$ is the conditional probability of an item belonging to an item group, denoting its group membership. Each user group has a rating preference for each item group. r is the variable representing the preference of user groups to item groups. $P(r|Z_u, Z_v)$ is the conditional probability of r given user group Z_u and item group Z_v. The rating for a coupled user-item pair is:

$$R(u,v) = \sum_r r \sum_{Z_u, Z_v} P(r|Z_u, Z_v) P(Z_u|u) P(Z_v|v) \tag{4.2}$$

Equation (4.2) can be rewritten into matrix form:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{4.3}$$

where $\boldsymbol{U} \in \mathbb{R}^{M \times L}$ and $\boldsymbol{V} \in \mathbb{R}^{N \times L}$ are the user and item group membership matrix. U_{ij} represents the membership of user u_i for user group $Z_u^{(j)}$. U_{i*} is the ith row of matrix U representing membership of user u_i to each group. U_{*j} is the jth column of matrix $\boldsymbol{U}$ representing the membership of each user to user group $Z_u^{(j)}$. The same goes for items. $\boldsymbol{S} \in \mathbb{R}^{K \times L}$ is the group-level knowledge matrix. S_{ij} represents the preference of user group $Z_u^{(i)}$ for item group $Z_v^{(j)}$.

After clustering, the user group and item group membership matrixes $\boldsymbol{U}_s^{(0)}$, $\boldsymbol{V}_s^{(0)}$ are acquired for the source domain and $\boldsymbol{U}_t^{(0)}$, $\boldsymbol{V}_t^{(0)}$ for the target domain.

$$\boldsymbol{U}_s^{(0)} = P(Z_{u_s}|u_s),\ \boldsymbol{V}_s^{(0)} = P(Z_{v_s}|v_s) \tag{4.4}$$

$$\boldsymbol{U}_t^{(0)} = P(Z_{u_t}|u_t),\ \boldsymbol{V}_t^{(0)} = P(Z_{v_t}|v_t) \tag{4.5}$$

where $P(Z_u|u) = \frac{P(u|Z_u)P(Z_u)}{\sum_{Z_u} P(u|Z_u)P(Z_u)}$ and $P(Z_v|v) = \frac{P(v|Z_v)P(Z_v)}{\sum_{Z_v} P(v|Z_v)P(Z_v)}$. Five parameters $P(u|Z_u)$, $P(v|Z_v)$,$P(r|Z_u,Z_v)$, $P(Z_u)$ and $P(Z_v)$ are learned from the FMM (for details, see (Si and Jin, 2003)).

4.3.2.2 Step 2: Domain adaptation of the user and item groups

This step ensures information consistency between the user/item group membership matrixes of two domains. The original user group membership matrixes $\boldsymbol{U}_s^{(0)}$, $\boldsymbol{U}_t^{(0)}$and item group membership matrixes $\boldsymbol{V}_s^{(0)}$, $\boldsymbol{V}_t^{(0)}$ from the source and target domains are used as the starting point.

In one domain (say, the source domain), each column $U_{s\,*j}^{(0)}$ represents the memberships of all users in a user group j. Thus, it is reasonable to use the marginal probability distribution of column $U_{s\,*j}^{(0)}$ to represent the characteristics of the user group information from user group j. This is also applied to the other three

matrixes $\boldsymbol{V}_s^{(0)}$, $\boldsymbol{U}_t^{(0)}$, $\boldsymbol{V}_t^{(0)}$. The disparity of the marginal probability distributions of user/item group membership matrixes in both domains is used to measure the divergence of the user/item group information. If the marginal probability distributions of the memberships of the two user/item groups are the same, these two user/item groups are regarded as having the same characteristics and the same physical meanings - information in the two user/item groups is consistent. This provides a method to measure the similarity between latent user/item groups in both domains. According to the basic assumption of recommender systems, i.e., "similar users like similar items", the preferences of similar user groups to similar item groups can be shared. Therefore, if the user/item group information of two domains is consistent, this group-level knowledge can be shared by both domains. The following formal definition of consistent user/item information and consistent knowledge determines which knowledge is transferable.

Definition 4.2 (Information-consistent Tri-factorization). Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ can be factorized based on nonnegative tri-factorization:

$$\boldsymbol{X}_s = \boldsymbol{U}_s^{(0)} \boldsymbol{S}_s^{(0)} (\boldsymbol{V}_s^{(0)})^T \tag{4.6}$$

$$\boldsymbol{X}_t = \boldsymbol{U}_t^{(0)} \boldsymbol{S}_t^{(0)} (\boldsymbol{V}_t^{(0)})^T \tag{4.7}$$

If both tri-factorizations satisfy the following equations, then they are information-consistent tri-factorizations.

$$P(\boldsymbol{U}_s^{(0)}) = P(\boldsymbol{U}_t^{(0)}) \tag{4.8}$$

$$P(\boldsymbol{V}_s^{(0)}) = P(\boldsymbol{V}_t^{(0)}) \tag{4.9}$$

where $P(\boldsymbol{U}_s^{(0)})$ and $P(\boldsymbol{V}_s^{(0)})$ represent the marginal probability distributions of $\boldsymbol{U}_s^{(0)}$ and $\boldsymbol{V}_s^{(0)}$, respectively. We say that the user group information from $\boldsymbol{U}_s^{(0)}$ and $\boldsymbol{V}_s^{(0)}$ is consistent, and the item group information from $\boldsymbol{V}_s^{(0)}$ and $\boldsymbol{V}_t^{(0)}$ is consistent. That is, the user/item groups from source and target domains are consistent. $\boldsymbol{S}_s^{(0)}$ is the "consistent knowledge" of the two matrixes $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$.

According to this definition, if the marginal probability distributions of user/item groups from source and target domains are the same, the group-level knowledge matrix can be shared, so that the consistent knowledge $\boldsymbol{S}_s^{(0)}$ can be directly used for the target rating matrix (let $\boldsymbol{S}_t^{(0)}$=$\boldsymbol{S}_s^{(0)}$). If the marginal probability distributions of the user/item group membership matrixes in both domains are not the same, we need to find other tri-factorization results that satisfy the conditions in *Definition 4.2*. Looking for a solution by trying different kinds of existing matrix factorization techniques is unattainable and time-consuming. Instead, we seek the solution by aligning consistent latent user groups and item groups through domain adaptation techniques. By adjusting the marginal probability distributions of user and item groups from the source and target domains comparatively, the similarities between the latent user and item groups are maximized. Consistent knowledge can then be extracted from the source rating matrix which can be directly used to help predict ratings in the target rating matrix.

To align consistent latent user and item groups, we need to find a projection to adjust the user/item group information of both rating matrixes so that the following equations are achieved:

$$P(\boldsymbol{\Psi}_s(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)})) = P(\boldsymbol{\Psi}_t(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)})) \tag{4.10}$$

$$P(\boldsymbol{\Phi}_s(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)})) = P(\boldsymbol{\Phi}_t(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)})) \tag{4.11}$$

It is apparent that $\boldsymbol{\Psi}_s$, $\boldsymbol{\Psi}_t$, $\boldsymbol{\Phi}_s$ and $\boldsymbol{\Phi}_t$ are the keys to ensuring that the latent groups remain consistent in both domains. We need to find maps that can force different distributions to become the same after mapping. A GFK is a domain adaptation strategy for learning robust features that is flexible against mismatch across domains and can be used to find a space for data in two domains to project into, so that the data distributions of the two domains in the projected space are similar (Gong et al., 2014). After projecting a GFK, a new representation is learned that satisfies the condition in *Definition 4.2*. Thus, we use a GFK to map $\boldsymbol{U}_s^{(0)}$, $\boldsymbol{U}_t^{(0)}$,$\boldsymbol{V}_s^{(0)}$ and $\boldsymbol{V}_t^{(0)}$ to $\boldsymbol{U}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$, $\boldsymbol{V}_s^{(1)}$ and $\boldsymbol{V}_t^{(1)}$. Based on the details of GFK (See Appendix A), $\boldsymbol{\Psi}_s$, $\boldsymbol{\Psi}_t$, $\boldsymbol{\Phi}_s$ and $\boldsymbol{\Phi}_t$ can be written as follows:

$$\boldsymbol{\Psi}_s(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) = \boldsymbol{\Psi}_G(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \times f_{zs}(\boldsymbol{U}_s^{(0)}) \tag{4.12}$$

$$\boldsymbol{\Psi}_t(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) = \boldsymbol{\Psi}_G(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \times f_{zs}(\boldsymbol{U}_t^{(0)}) \tag{4.13}$$

$$\boldsymbol{\Phi}_s(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)}) = \boldsymbol{\Phi}_G(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)}) \times f_{zs}(\boldsymbol{V}_s^{(0)}) \tag{4.14}$$

$$\boldsymbol{\Phi}_t(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)}) = \boldsymbol{\Phi}_G(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)}) \times f_{zs}(\boldsymbol{V}_t^{(0)}) \tag{4.15}$$

where $\boldsymbol{\Psi}_s(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)})$ and $\boldsymbol{\Phi}_s(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)})$ are the operators of the GFK method and f_{zs} is the function of Z-score.

Then, the adapted latent user groups of the two rating matrixes can be obtained, which are expressed as:

$$\boldsymbol{U}_s^{(1)} = \boldsymbol{\Psi}_s(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \tag{4.16}$$

$$\boldsymbol{U}_t^{(1)} = \boldsymbol{\Psi}_t(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \tag{4.17}$$

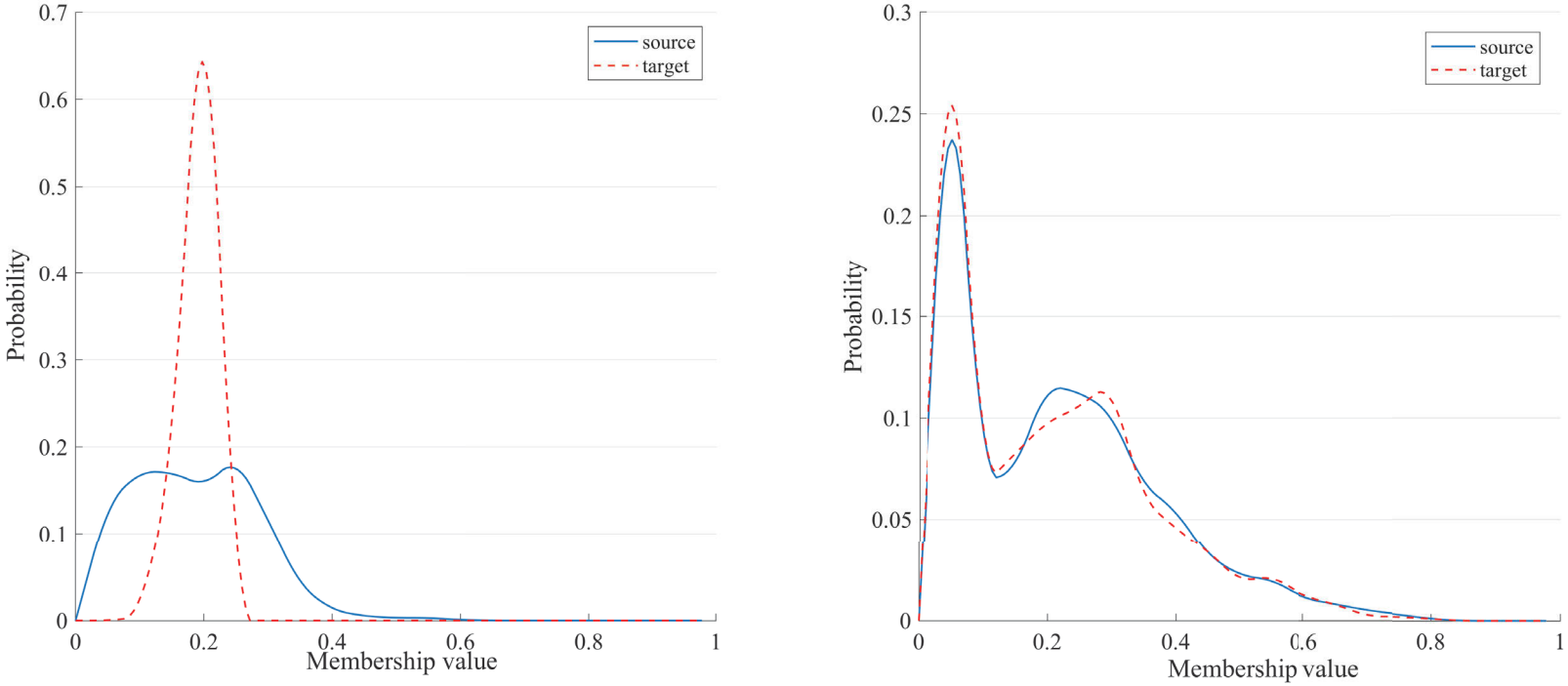
The same goes for the item groups: $\boldsymbol{V}_s^{(1)} = \boldsymbol{\Phi}_s(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)})$, $\boldsymbol{V}_s^{(1)} = \boldsymbol{\Phi}_t(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)})$. $\boldsymbol{U}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$ are user group membership matrixes unified to the same domain-invariant feature space for the source and target domains, while $\boldsymbol{V}_s^{(1)}$, $\boldsymbol{V}_t^{(1)}$ are unified item group membership matrixes.

Here, an example best illustrates the domain adaptation process of the user and item group information. Consider a source domain and a target domain that both have 1000 non-overlapping users. In each domain, the users are clustered into six user groups, with inconsistent user group information between the source and target domains. The probability distributions of the first user group (the first column of $\boldsymbol{U}_s^{(0)}$ and $\boldsymbol{U}_t^{(0)}$) is shown in Figure 4.3 (a); each is quite different. To force consistency, the information for every user group in each domain is adjusted, after which the user group information of the adapted matrixes $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$ are almost the same, as shown in Figure 4.3 (b).

4.3.2.3 Step 3: Consistent knowledge extraction

After the domain adaptation, $\boldsymbol{U}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$ are consistent, and $\boldsymbol{V}_s^{(1)}$, $\boldsymbol{V}_t^{(1)}$ are consistent. Once we have obtained consistent group representations that are meaningful across both rating matrixes, the model trained on the source rating matrix and the target rating matrix can be brought together. On this basis, the recommender systems learned from the source and target domains will share the same group-level knowledge matrix $\boldsymbol{S}$.

Consistent knowledge $\boldsymbol{S}$ is obtained by maximizing the approximation of the available data in both the source rating matrix and the target rating matrix by approximating $\boldsymbol{X}_s \approx \boldsymbol{U}_s^{(1)} \boldsymbol{S} (\boldsymbol{V}_s^{(1)})^T$ together with $\boldsymbol{X}_t \approx \boldsymbol{U}_t^{(1)} \boldsymbol{S} (\boldsymbol{V}_t^{(1)})^T$. To qualify



(a) Distribution before adjustment

(b) Distribution after adjustment

Figure 4.3 An example of user group information adjustment in two domains

Note: (a) Marginal probability distribution of the first column in $\boldsymbol{U}_s^{(0)}$ and $\boldsymbol{U}_t^{(0)}$, (b) Marginal probability distribution of the first column in $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$.

the approximation, one useful and simple measure is to use a Frobenius norm between the original rating matrix and the approximation. We have the following cost function:

$$J_s(\boldsymbol{S}) = \frac{1}{M_s N_s}\|\boldsymbol{I}_s \circ (\boldsymbol{X}_s - \boldsymbol{U}_s^{(1)}\boldsymbol{S}(\boldsymbol{V}_s^{(1)})^T)\|_F + \frac{1}{M_t N_t}\|\boldsymbol{I}_t \circ (\boldsymbol{X}_t - \boldsymbol{U}_t^{(1)}\boldsymbol{S}(\boldsymbol{V}_t^{(1)})^T)\|_F + \frac{\lambda}{2KL}\|\boldsymbol{S}\|_F \tag{4.18}$$

where $\boldsymbol{I}_s$ is a binary weighting matrix for $\boldsymbol{X}_s$, if $(I_s)_{ij} = 1$, then $(X_s)_{ij} \neq 0$ and $(I_s)_{ij} = 0$, otherwise. The same applies to $\boldsymbol{I}_t$ for $\boldsymbol{X}_t$. $\circ$ is an entry-wise product, λ is the parameter for regularization. Since the physical meaning of $\boldsymbol{S}$ is the preference that the user groups give to the item groups, it should be in range of $(0,5]$. Regularization to constrain the range of $\boldsymbol{S}$ is added to the cost function. Finally, consistent knowledge is learned through the following optimization problem:

$$\min_{\boldsymbol{S}} J_s(\boldsymbol{S})$$
$$\text{s.t. } \boldsymbol{S} > 0$$

Gradient descent is a general algorithm for optimization, which leads to the update rule: $s_{ab} \leftarrow s_{ab} + \eta_{ab}\frac{\partial J_s}{\partial \boldsymbol{S}_{ab}}$. For this problem, we need to constrain the non-negativity of $\boldsymbol{S}$. The partial derivative of the cost function has a special form, so we can use tricks to set the learning rate $\eta_{ab} = \frac{(\boldsymbol{S})_{ab}}{(A+B+\frac{\lambda \boldsymbol{S}}{2KL})_{ab}}$ to guarantee that $\boldsymbol{S}$ is nonnegative, where A$= \frac{1}{M_s N_s}(\boldsymbol{U}_s^{(1)})^T(\boldsymbol{I}_s \circ (\boldsymbol{U}_s^{(1)}\boldsymbol{S}(\boldsymbol{V}_s^{(1)})^T))\boldsymbol{V}_s^{(1)}$, $B = \frac{1}{M_t N_t}(\boldsymbol{U}_t^{(1)})^T(\boldsymbol{I}_t \circ (\boldsymbol{U}_t^{(1)}\boldsymbol{S}(\boldsymbol{V}_t^{(1)})^T))\boldsymbol{V}_t^{(1)}$. The objective function is non-increasing under the following update rule:

$$s_{ab} \leftarrow s_{ab}\frac{(\frac{1}{M_s N_s}(\boldsymbol{U}_s^{(1)})^T\boldsymbol{X}_s\boldsymbol{V}_s^{(1)} + \frac{1}{M_t N_t}(\boldsymbol{U}_t^{(1)})^T\boldsymbol{X}_t\boldsymbol{V}_t^{(1)})_{ab}}{(A+B+\frac{\lambda \boldsymbol{S}}{2KL})_{ab}} \tag{4.19}$$

The learning process is summarized in Algorithm 4.1.

Algorithm 4.1: Consistent Knowledge Extraction

Input:

$\boldsymbol{X}_s$, the source rating matrix
$\boldsymbol{X}_t$, the target rating matrix
$\boldsymbol{U}_s^{(1)}$, $\boldsymbol{V}_s^{(1)}$, user and item membership matrix of source domain
$\boldsymbol{U}_t^{(1)}$, $\boldsymbol{V}_t^{(1)}$, user and item membership matrix of target domain
($\boldsymbol{U}_s^{(1)}$, $\boldsymbol{V}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$, $\boldsymbol{V}_t^{(1)}$ are obtained from GFK algorithm)

Output:

$\boldsymbol{S}$, the consistent knowledge

Initialize $\boldsymbol{S} \in \mathbb{R}^{K\times L}$, $J(\boldsymbol{S}) \leftarrow 0$, $J(\boldsymbol{S})^{(pre)} \leftarrow 0$
while $J(\boldsymbol{S})^{(pre)} - J(\boldsymbol{S}) > \varepsilon$ **or** $J(\boldsymbol{S}) = 0$ **do**
    $J(\boldsymbol{S})^{(pre)} = J(\boldsymbol{u}, \boldsymbol{v})$
    for each element s_{ab} in $\boldsymbol{S}$ **do**
        Update s_{ab} as in Equation (4.19)
    end for
    Update $J(\boldsymbol{S})$ as in Equation (4.18)
end while
return $\boldsymbol{S}$

4.3.2.4 Step 4: Group representation regulation

The domain adaptation technique GFK is designed for unsupervised transfer learning where no label is available in the target domain. In this problem setting, some domain-specific characteristics are embedded in the small amount of available data in the target rating matrix. To reveal these idiosyncrasies of the target domain, we amend the group representations of the target rating matrix to make the model fit better to the task in target rating matrix. It is imperative that we find maps $f_u : \boldsymbol{U}_t^{(1)} \rightarrow \mathbb{R}^{M_t\times K}$ and $f_v : \boldsymbol{V}_t^{(1)} \rightarrow \mathbb{R}^{N_t\times L}$ to make $\boldsymbol{U}_t^{(1)}$ and $\boldsymbol{V}_t^{(1)}$ more suitable for the target rating matrix. At the same time, the adjustment should not impair the consistency of user groups and item groups between two domains. According to

Definition 4.2, f_u and f_v should satisfy the following equation:

$$P(\boldsymbol{S}|f_u(\boldsymbol{U}_t^{(1)}), f_v(\boldsymbol{V}_t^{(1)})) = P(\boldsymbol{S}|\boldsymbol{U}_t^{(1)}, \boldsymbol{V}_t^{(1)}) \tag{4.20}$$

Equation 4.20 ensures that the probability of each element in S will not change after mapping $\boldsymbol{U}_t^{(1)}$ and $\boldsymbol{V}_t^{(1)}$ using f_u and f_v. Here, we choose $f_u(\boldsymbol{U}_t^{(1)}) = \boldsymbol{U}_t^{(1)}\boldsymbol{u}$ and $f_v(\boldsymbol{V}_t^{(1)}) = \boldsymbol{V}_t^{(1)}\boldsymbol{v}$, where $\boldsymbol{u} \geq 0$ and $\boldsymbol{v} \geq 0$. These two maps satisfy Equation (4.20). For further details of why f_u and f_v are chosen like this, see Appendix B. Learning f_u and f_v is an optimization problem. The cost function is:

$$J_r(u,v) = \|\boldsymbol{I}_t \circ (\boldsymbol{X}_t - \boldsymbol{U}_t^{(1)}\boldsymbol{u}\boldsymbol{S}(\boldsymbol{V}_t^{(1)}\boldsymbol{v})^T)\|_F \tag{4.21}$$

The tuning factors can be learned through optimizing

$$\min_{\boldsymbol{u},\boldsymbol{v}} J_r(\boldsymbol{u},\boldsymbol{v})$$

$$\text{s.t. } u \geq 0, v \geq 0$$

Similarly, the cost function is non-increasing under the following update rules:

$$\boldsymbol{u}_{ab} \leftarrow \boldsymbol{u}_{ab}\frac{((U_t^{(1)})^T\boldsymbol{X}_t\boldsymbol{V}_t^{(1)}\boldsymbol{v}\boldsymbol{S}^T)_{ab}}{((\boldsymbol{U}_t^{(1)})^T(\boldsymbol{I}_t \circ (\boldsymbol{U}_t^{(1)}\boldsymbol{u}\boldsymbol{S}(\boldsymbol{V}_t^{(1)}\boldsymbol{v})^T))\boldsymbol{V}_t^{(1)}\boldsymbol{v}\boldsymbol{S}^T)_{ab}} \tag{4.22}$$

$$\boldsymbol{v}_{cd} \leftarrow \boldsymbol{v}_{cd}\frac{((V_t^{(1)})^T\boldsymbol{X}_t^T\boldsymbol{U}_t^{(1)}\boldsymbol{v}\boldsymbol{S}^T)_{cd}}{((\boldsymbol{V}_t^{(1)})^T(\boldsymbol{I}_t^T \circ (\boldsymbol{V}_t^{(1)}\boldsymbol{v}\boldsymbol{S}^T\boldsymbol{u}^T(\boldsymbol{U}_t^{(1)})^T))\boldsymbol{U}_t^{(1)}\boldsymbol{v}\boldsymbol{S})_{cd}} \tag{4.23}$$

Finally, the optimization problem is solved by alternatively estimating $\boldsymbol{u}$, $\boldsymbol{v}$. How $\boldsymbol{u}$, $\boldsymbol{v}$ is learned is summarized in Algorithm 4.2.

Algorithm 4.2: Group representation regulation

Input:
$\boldsymbol{X}_t$, the target rating matrix
$\boldsymbol{S}$, the consistent knowledge
$\boldsymbol{U}_t^{(1)}$, $\boldsymbol{V}_t^{(1)}$, user and item membership matrix of target domain
($\boldsymbol{U}_t^{(1)}$, $\boldsymbol{V}_t^{(1)}$ are obtained from GFK algorithm)
Output:
$\boldsymbol{u}$, user tuning factor
$\boldsymbol{v}$, item tuning factor

Initialize $\boldsymbol{u} \in \mathbb{R}^{K\times K}$, $\boldsymbol{v} \in \mathbb{R}^{L\times L}$, $J(\boldsymbol{u},\boldsymbol{v}) \leftarrow 0$, $J(\boldsymbol{u},\boldsymbol{v})^{(pre)} \leftarrow 0$
while $J(\boldsymbol{u},\boldsymbol{v})^{(pre)} - J(\boldsymbol{u},\boldsymbol{v}) > \varepsilon$ **or** $J(\boldsymbol{u},\boldsymbol{v}) = 0$ **do**
  $J(\boldsymbol{u},\boldsymbol{v})^{(pre)} = J(\boldsymbol{u},\boldsymbol{v})$
  for each element u_{ab} in $\boldsymbol{u}$ **do**
    Update u_{ab} as in Equation (4.22)
  end for
  for each element v_{cd} in $\boldsymbol{v}$ **do**
    Update v_{cd} as in Equation (4.23)
  end for
  Update $J(\boldsymbol{u},\boldsymbol{v})$ as in Equation (4.21)
end while
return $\boldsymbol{u}$, $\boldsymbol{v}$

4.3.2.5 Step 5: Recommendation in target domain

The recommendation in target domain is given by Equation (4.24).

$$\begin{cases} \hat{\boldsymbol{X}}_t = (\boldsymbol{U}_t^{(1)}\boldsymbol{u})\boldsymbol{S}(\boldsymbol{V}_t^{(1)}\boldsymbol{v})^T, \\ \boldsymbol{U}_t^{(1)} = \boldsymbol{\Psi}_G(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \times f_{zs}(\boldsymbol{U}_t^{(0)}), \\ \boldsymbol{V}_t^{(1)} = \boldsymbol{\Phi}_G(\boldsymbol{V}_s^{(0)}, \boldsymbol{V}_t^{(0)}) \times f_{zs}(\boldsymbol{V}_t^{(0)}), \end{cases} \tag{4.24}$$

where $\hat{\boldsymbol{X}}_t$ is the reconstructed user-item rating matrix for prediction, $\boldsymbol{u}$, $\boldsymbol{v}$ are user and item tuning factors for target domain, $\boldsymbol{S}$ is the consistent knowledge, $\boldsymbol{U}_s^{(0)}$, $\boldsymbol{U}_t^{(0)}$ are user group membership matrixes, and $\boldsymbol{V}_s^{(0)}$, $\boldsymbol{V}_t^{(0)}$ are item group membership matrixes for the source domain and the target domain before domain adaptation.

$\boldsymbol{U}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$ are user and item group membership matrixes for the target domain after domain adaptation. $\boldsymbol{\Psi}_G$ and $\boldsymbol{\Phi}_G$ are GFK operators to map group membership matrixes to a domain-invariant feature space, and f_{zs} is the Z-score function.

4.3.3 Architecture of CIT

In the proposed CIT method, group-level knowledge from a source domain and a target domain can be combined and augmented compared with what can be acquired independently from only the target domain. In this section, we introduce how to use the proposed CIT method when developing a recommender system to support decision making for businesses and individual customers.

A conceptual framework for a cross-domain recommender system that applies the proposed method is shown in Figure 4.4. When businesses launch a new product or service, a sufficient amount of data has not always been collected to populate the target domain. It is often easier to acquire data from another mature service - the source domain. Accordingly, a cross-domain recommendation engine can be built, based on our method, to provide better predictions of a user's preferences for items. This assists decision making for both businesses and individual customers.

For businesses, the CDRS could be used to support product development and marketing decisions. For example, businesses could predict user preferences for more accurate cross-selling or identify potential user groups to market specific products to. They could also develop product bundles based on user preference prediction. For individual customers, our proposed CDRS could be used to facilitate targeted product searches. By ranking products according to predicted preference, customers may be able to locate the most desirable products more quickly and effectively.

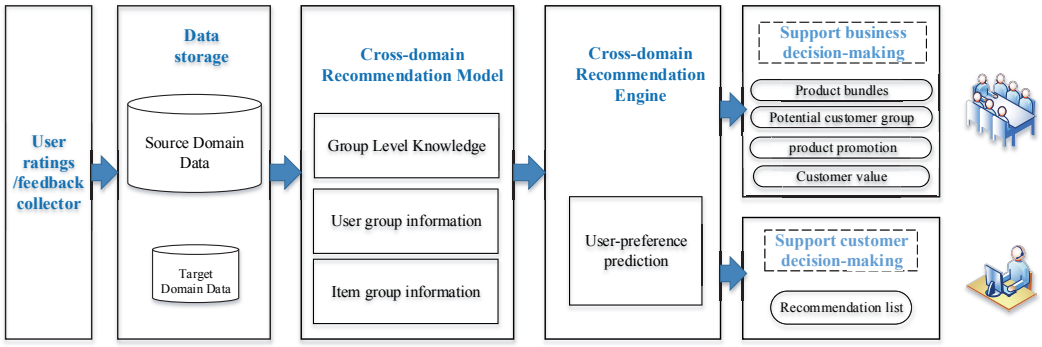


Figure 4.4 Conception framework of cross-domain recommender system using CIT.

4.4 Experiments and Analysis

Our empirical experiments are presented in this section. First, the datasets and evaluation metrics are introduced, followed by the experimental settings and the baseline methods. The results of the experiments are presented along with an analysis of the parameters.

4.4.1 Dataset and Evaluation Metrics

In testing the CIT method, it was important to choose data from different but similar domains. Previous research has considered movies, books, and music as appropriate categories for CDRS experiment tests. For a fair comparison, we have chosen the same categories and many of the same datasets for our experiments. Our tests comprise nine cross-domain recommendation tasks, including movie-to-movie and book-to-movie recommendations, common in prior research, as well as some new tasks extending to the music category that are less commonly tested. The baseline methods include three non-transfer methods and two cross-domain methods. Five

real-world datasets were used: Movielens 20M [1], Netflix [2], LibraryThing [3], Amazon Book [4] and YahooMusic [5]. Each is publicly available and has been used to test recommender systems in a variety of scenarios for recommender systems in single domain. But tests on these dataset in this novel cross-domain setting are lacking. The statistical information for these datasets is provided in Table 4.1.

Table 4.1 Statistical information on the original datasets for experiments on CIT

	movielens 20M	Netflix	library thing	Amazon	Yahoo music_1	Yahoo music_2
#user	138493	480189	7279	8026324	200000	200000
#item	26744	17770	37232	2330066	136736	136736
#rating	20000263	100480507	749401	22507155	78344627	78742463
sparsity	0.54%	1.18%	0.28%	0.0001%	0.29%	0.29%
range	0.5-5	1-5	0.5-5	1-5	1-5	1-5
average	3.5255	3.6043	3.8709	4.2958	3.1613	3.1634
std	1.0520	1.0852	0.9387	1.1115	1.5991	1.6046

In the Amazon Book dataset, we found that more than 6 million among 8 million users gave all their reviewed items the same rating. This phenomenon is very uncommon and rarely happens in real-world. As such, it was determined that these users could provide no effective contribution to the construction of a recommender system and were removed. For the Movielens20M and LibraryThing datasets, we normalized the ratings to a range of $\{1,2,3,4,5\}$. Movielens20M, LibraryThing, and YahooMusic_1 were used as the source domain, while Netflix, AmazonBook, YahooMusic_2 were used as the target domain. Across all the datasets, 2000 items that had been rated more than 10 times were randomly chosen.

[1] https://grouplens.org/datasets/movielens/20m/
[2] https://netflixprize.com/index.html
[3] https://www.librarything.com
[4] http://jmcauley.ucsd.edu/data/amazon/
[5] https://webscope.sandbox.yahoo.com./

We then filtered out the users who had given less than a total of 20 ratings. The next section describes how the users were chosen.

For the source domain data, we randomly selected 4000 users to be regular customers of the site. The sparsity ratio of source domain data was controlled at 2%. Two source domain datasets with different statistical properties were chosen to test the performance of different algorithms. For the target domain data, we randomly selected 2000 users to be regular customers of the site, and another 2000 users to be new customers. In terms of regular customers, three sparsity ratios were used to compare different algorithms in different circumstances. For new users, five observed ratings were given, and the rest of the ratings were used for evaluation. In the end, the rating matrixes for both the source and target domains were all 4000×2000 matrixes. The details of the final datasets are summarized in Table 4.2.

MAE and **R**oot **M**ean **S**quare **E**rror (RMSE) were used as the evaluation metrics. MAE is defined in Eq. (3.12) and RMSE is defined as:

$$RMSE = \sqrt{\sum_{u,v,X_{uv} \in Y} \frac{(\hat{X}_{uv} - X_{uv})^2}{|Y|}} \tag{4.25}$$

where Y is the test set, and $|Y|$ is the number of test ratings. The smaller the errors, the better the performance.

4.4.2 Experimental Settings and Baselines

Three non-transfer learning methods and two cross-domain methods were chosen as comparisons for the proposed method. The non-transfer learning methods were: PCC (Deshpande and Karypis, 2004), FMM (Si and Jin, 2003) and SVD (Koren

Table 4.2 Description of data subsets in three categories for experiments on CIT

Data_type	Data_name	Data_source	Domain	Sparsity	Average
Movie	movie_s1	Movielens20M	source	2.00%	3.66
	movie_s2	Movielens20M	source	2.00%	2.63
	movie_t1	Netflix	target	0.50%	2.68
	movie_t2	Netflix	target	1.00%	2.67
	movie_t3	Netflix	target	1.50%	2.67
Book	book_s1	LibraryThing	source	2.00%	4.02
	book_s2	LibraryThing	source	2.00%	3.72
	book_t1	Amazon	target	0.50%	3.52
	book_t2	Amazon	target	0.75%	3.53
	book_t3	Amazon	target	0.94%	3.53
Music	music_s1	YahooMusic_1	source	2.00%	4.13
	music_s2	YahooMusic_1	source	2.00%	2.73
	music_t1	YahooMusic_2	target	0.50%	2.26
	music_t2	YahooMusic_2	target	1.00%	2.26
	music_t3	YahooMusic_2	target	1.50%	2.25

et al., 2009). The cross-domain methods were: CBT (Li et al., 2009a) and RMGM (Li et al., 2009b). PCC uses user-based CF, and the number of neighborhoods was set at 50. For SVD, the latent feature number was fixed at 40, the regularization factor was set to 0.015, and the learning rate was set to 0.003. For FMM, CBT, and RMGM, the user group number and item group number were both set to 40. For the proposed method, CIT, the user group number and the item group number were both set to 40, and the regularization factor was set to 0.5. Further analysis of the parameters is provided in section 4.4.4.

For each target domain, three configurations of sparsity were settled; thus, nine cross-domain recommendation tasks each under three sparsity ratios were conducted for comparison between the baselines and the proposed method. Since the algorithms (except for PCC) need to initialize the factorized matrix randomly, we ran 20 random initializations and report the averaged results and standard deviations.

4.4.3 Results

Comparison results are given in Table 4.3, 4.4 and 4.5. The proposed method, CIT, had the lowest MAE and RMSE among all the six methods in most of the cross-domain recommendation tasks. Compared with the non-transfer learning methods, we find that our method is more effective at extracting knowledge from the source domain to apply in the target domain. This is especially significant when the statistical properties of the source rating matrix are different from those in the target rating matrix. This indicates that our method gains its benefits by keeping the user and item group information in both domains consistent. The CIT method is able to extract knowledge even when the statistical properties of the source rating matrix diverge from the target rating matrix, while CBT and RMGM may need some restricted conditions of source data.

Table 4.3 Prediction performance of CIT on a movie target domain

method	source data	MAE Sparsity			RMSE Sparsity		
		0.50%	1.00%	1.50%	0.50%	1.00%	1.50%
PCC	-	1.2609	1.2710	1.1981	1.5671	1.5789	1.4839
FMM	-	1.0164±0.0027	1.0069±0.0033	1.0029±0.0028	1.2283±0.0036	1.2143±0.0045	1.2064±0.0037
SVD	-	1.0230±0.0013	1.0227±0.0012	1.0391±0.0077	1.2372±0.0015	1.2382±0.0012	1.2544±0.0096
CBT	movie_s1	1.2868±0.0034	1.2845±0.0072	1.2836±0.0038	1.5318±0.0043	1.5290±0.0092	1.5277±0.0048
	movie_s2	1.0205±0.0007	1.0194±0.0016	1.0192±0.0010	**1.1964***±0.0003	1.1962±0.0008	1.1958±0.0004
	book_s1	1.4493±0.0075	1.4477±0.0071	1.4441±0.0066	1.7627±0.0114	1.7604±0.0107	1.7551±0.0100
	book_s2	1.3272±0.0118	1.3248±0.0071	1.3253±0.0104	1.5871±0.0159	1.5839±0.0093	1.5849±0.0134
	music_s1	1.4917±0.0189	1.4935±0.0164	1.4923±0.0146	1.8115±0.0187	1.8131±0.0155	1.8112±0.0141
	music_s2	1.0144±0.0023	1.0134±0.0019	1.0141±0.0020	1.2027±0.0027	1.2032±0.0020	1.2018±0.0030
RMGM	movie_s1	1.0347±0.0065	1.0252±0.0050	1.0214±0.0034	1.2515±0.0079	1.2402±0.0067	1.2345±0.0047
	movie_s2	1.0038±0.0022	0.9994±0.0025	0.9977±0.0028	1.2104±0.0031	1.2025±0.0033	1.1992±0.0038
	book_s1	1.0464±0.0048	1.0369±0.0046	1.0309±0.0052	1.2711±0.0060	1.2583±0.0064	1.2501±0.0069
	book_s2	1.0396±0.0033	1.0326±0.0038	1.0261±0.0043	1.2616±0.0043	1.2523±0.0048	1.2433±0.0057
	music_s1	1.0498±0.0055	1.0387±0.0056	1.0299±0.0058	1.2734±0.0078	1.2595±0.0079	1.2463±0.0076
	music_s2	1.0591±0.0061	1.0512±0.0039	1.0489±0.0046	1.2813±0.0076	1.2711±0.0052	1.2655±0.0058
CIT	movie_s1	**1.0002***±0.0025	**0.9906***±0.0027	**0.9888***±0.0025	**1.1846***±0.0027	**1.1881***±0.0034	**1.1846***±0.0027
	movie_s2	**0.9995***±0.0028	**0.9911***±0.0023	**0.9873***±0.0022	1.1987*±0.0040	**1.1887***±0.0029	**1.1828***±0.0034
	book_s1	**0.9992***±0.0022	**0.9908***±0.0019	**0.9886***±0.0023	**1.1978***±0.0034	**1.1882***±0.0026	**1.1843***±0.0028
	book_s2	**0.9993***±0.0032	**0.9907***±0.0022	**0.9889***±0.0022	**1.1985***±0.0041	**1.1886***±0.0026	**1.1853***±0.0032
	music_s1	**0.9996***±0.0033	**0.9914***±0.0022	**0.9883***±0.0018	**1.1985***±0.0045	**1.1885***±0.0029	**1.1839***±0.0025
	music_s2	**1.0004***±0.0025	**0.9931***±0.0021	**0.9892***±0.0023	**1.1997***±0.0031	**1.1886***±0.0028	**1.1848***±0.0033

Table 4.4 Prediction performance of CIT on a book target domain

method	source data	MAE Sparsity			RMSE Sparsity		
		0.50%	0.75%	0.94%	0.50%	0.75%	0.94%
PCC	-	1.2625	1.2654	1.2340	1.5737	1.5739	1.5305
FMM	-	1.0645±0.0028	1.0256±0.0022	**1.0211***±0.0029	1.3152±0.0035	1.2645±0.0033	1.2582±0.0045
SVD	-	1.0591±0.0025	1.0288±0.0021	1.1702±0.0034	1.3220±0.0028	1.2826±0.0026	1.5032±0.0047
CBT	movie_s1	1.0859±0.0009	1.0856±0.0005	1.0851±0.0006	1.3233±0.0037	1.3219±0.0023	1.3220±0.0024
	movie_s2	1.2345±0.0118	1.2334±0.0072	1.2287±0.0078	1.4271±0.0105	1.4260±0.0064	1.4215±0.0070
	book_s1	1.0896±0.0012	1.0893±0.0009	1.0895±0.0012	1.4330±0.0030	1.4310±0.0038	1.4312±0.0038
	book_s2	1.0813±0.0010	1.0814±0.0009	1.0809±0.0008	1.3569±0.0059	1.3547±0.0046	1.3525±0.0041
	music_s1	1.1128±0.0082	1.1127±0.0098	1.1105±0.0080	1.4616±0.0096	1.4618±0.0091	1.4598±0.0073
	music_s2	1.1881±0.0129	1.1906±0.0147	1.1935±0.0119	1.3895±0.0099	1.3910±0.0116	1.3930±0.0095
RMGM	movie_s1	1.0673±0.0046	1.0460±0.0051	1.0425±0.0047	1.3057±0.0066	1.2786±0.0065	1.2750±0.0070
	movie_s2	1.0594±0.0037	1.0329±0.0036	1.0277±0.0037	1.2933±0.0039	1.2614±0.0043	1.2558±0.0036
	book_s1	1.0726±0.0041	1.0440±0.0039	1.0409±0.0046	1.3339±0.0052	1.2962±0.0050	1.2914±0.0057
	book_s2	1.0649±0.0037	1.0424±0.0037	1.0376±0.0029	1.3172±0.0049	1.2883±0.0050	1.2807±0.0036
	music_s1	1.0817±0.0055	1.0588±0.0063	1.0539±0.0053	1.3432±0.0074	1.3143±0.0103	1.3082±0.0081
	music_s2	1.1028±0.0076	1.0832±0.0072	1.0713±0.0068	1.3430±0.0094	1.3196±0.0086	1.3092±0.0084
CIT	movie_s1	**1.0464***±0.0045	**1.0246**±0.0031	1.0243±0.0028	**1.2685***±0.0041	**1.2464***±0.0041	**1.2458***±0.0046
	movie_s2	**1.0456***±0.0036	**1.0249**±0.0032	1.0245±0.0024	**1.2688***±0.0040	**1.2474***±0.0035	**1.2458***±0.0022
	book_s1	**1.0465***±0.0031	1.0257±0.0028	1.0247±0.0030	**1.2705***±0.0041	**1.2468***±0.0040	**1.2458***±0.0039
	book_s2	**1.0474***±0.0045	**1.0254**±0.0026	1.0236±0.0031	**1.2707***±0.0050	**1.2476***±0.0034	**1.2448***±0.0043
	music_s1	**1.0467***±0.0040	**1.0249***±0.0024	1.0238±0.0030	**1.2711***±0.0047	**1.2465***±0.0039	**1.2442***±0.0033
	music_s2	**1.0457***±0.0030	1.0265±0.0030	1.0238±0.0032	**1.2690***±0.0030	**1.2482***±0.0036	**1.2456***±0.0028

Table 4.5 Prediction performance of CIT on a music target domain

method	source data	MAE Sparsity			RMSE Sparsity		
		0.50%	1.00%	1.50%	0.50%	1.00%	1.50%
PCC	-	1.4403	1.3617	1.3262	1.8421	1.7080	1.6489
FMM	-	1.2619±0.0023	1.2460±0.0027	1.2448±0.0028	1.5009±0.0035	1.4754±0.0057	1.4685±0.0045
SVD	-	1.2675±0.0009	1.2603±0.0009	1.2566±0.0014	1.4972±0.0011	1.4916±0.0015	1.4876±0.0015
CBT	movie_s1	1.3776±0.0030	1.3759±0.0044	1.3764±0.0029	1.6168±0.0046	1.6136±0.0069	1.6149±0.0048
	movie_s2	1.2726±0.0021	1.2734±0.0025	1.2728±0.0024	1.4663±0.0017	1.4644±0.0021	1.4634±0.0021
	book_s1	1.4666±0.0038	1.4665±0.0038	1.4656±0.0064	1.7929±0.0076	1.7926±0.0076	1.7908±0.0127
	book_s2	1.3986±0.0045	1.3973±0.0050	1.4005±0.0035	1.6598±0.0071	1.6581±0.0085	1.6634±0.0068
	music_s1	1.4971±0.0131	1.4934±0.0102	1.5012±0.0049	1.8343±0.0139	1.8310±0.0102	1.8387±0.0050
	music_s2	1.2597±0.0059	1.2598±0.0054	1.2568±0.0051	1.4604±0.0036	1.4604±0.0035	1.4603±0.0041
RMGM	movie_s1	1.2699±0.0038	1.2576±0.0048	1.2539±0.0036	1.5099±0.0062	1.4952±0.0075	1.4897±0.0056
	movie_s2	1.2482±0.0023	1.2401±0.0027	1.2406±0.0029	1.4819±0.0051	1.4698±0.0048	1.4707±0.0051
	book_s1	1.2832±0.0049	1.2690±0.0030	1.2623±0.0054	1.5374±0.0069	1.5180±0.0058	1.5094±0.0088
	book_s2	1.2757±0.0037	1.2620±0.0047	1.2575±0.0042	1.5285±0.0065	1.5096±0.0076	1.5026±0.0070
	music_s1	1.2901±0.0051	1.2767±0.0063	1.2683±0.0083	1.5497±0.0097	1.5317±0.0095	1.5199±0.0117
	music_s2	1.2881±0.0037	1.2842±0.0035	1.2799±0.0057	1.5385±0.0066	1.5328±0.0079	1.5264±0.0069
CIT	movie_s1	**1.2450***±0.0021	**1.2375***±0.0019	**1.2344***±0.0015	**1.4516***±0.0030	**1.4451***±0.0029	**1.4400***±0.0026
	movie_s2	**1.2452***±0.0019	**1.2375***±0.0018	**1.2345***±0.0020	**1.4513***±0.0026	**1.4439***±0.0030	**1.4409***±0.0038
	book_s1	**1.2449***±0.0019	**1.2385***±0.0017	**1.2350***±0.0021	**1.4511***±0.0020	**1.4448***±0.0022	**1.4411***±0.0040
	book_s2	**1.2455***±0.0015	**1.2377***±0.0016	**1.2344***±0.0017	**1.4523***±0.0026	**1.4444***±0.0029	**1.4403***±0.0023
	music_s1	**1.2449***±0.0021	**1.2379***±0.0021	**1.2349***±0.0026	**1.4511***±0.0027	**1.4445***±0.0027	**1.4404***±0.0039
	music_s2	**1.2453***±0.0023	**1.2375***±0.0018	**1.2344***±0.0021	**1.4513***±0.0025	**1.4438***±0.0021	**1.4402***±0.0031

Comparing the six methods and given the results of all nine tasks with different sparsity ratios, we can make the following observations:

1. For non-transfer learning methods, the FMM method shows superior performance compared to the memory-based method PCC and the famous matrix factorization method SVD from the Netflix competition. PCC and SVD are not very good at handling the cold-start problem. When the number of available ratings for users in target domain is limited, they fail to give good recommendations.

2. CBT is not stable and positive transfer is not guaranteed. When the statistical properties of the source rating matrix is similar to that of the target rating matrix (say movie_s2 to movie_t1/2/3), CBT is better than the non-transfer baselines. Since CBT fills the source rating matrix with the users' average ratings, the average is crucial to this method and gains more advantages on two datasets when their average ratings are close. However, many results, like movie_s1 to movie_t1/2/3, suggest that CBT grapples with negative transfer issues. Referring to the statistical properties in Table 4.2, the performance of CBT is directly related to the average of ratings. When the average rating of the source rating matrix deviates from that of the target domain, the performance of CBT is greatly impaired.

3. RMGM shows similar performance to CBT but is more stable. The rating matrixes from the source and target domains are diagonally joined in RMGM. It is necessary for the two matrixes to have similar statistical properties to extract common knowledge, but RMGM fails to note whether or not the two matrixes are similar. RMGM's results suggest that discrepancies in the

average will disturb the extraction of common knowledge, thus weakening transfer learning. We can see that positive transfer cannot be assured without a similarity guarantee of the rating matrixes for the source and target domains.

4. The proposed CIT performs better than all the other baseline methods in almost all tasks, whether or not the datasets are in the same category. CIT ensures a steady improvement compared to non-transfer learning methods. Unlike the other two cross-domain methods, CIT is also suitable for datasets with different statistical properties. The adaptation knowledge transfer in CIT ensures that the knowledge extracted from the source rating matrix is suitable for assisting recommendation in the target domain.

5. Negative transfer was always observed for CBT and RMGM when the average rating in the source domain was different from that of the target domain. This leads to a fundamental question in transfer learning: "When to transfer?" This is an area seldom studied in CDRS. Instead of determining when to transfer, our proposed method reduces the difference between the source and target domains by preserving consistent user and item group information. In the scope of this chapter, we did not see any negative transfer learning in our proposed method.

To confirm that the improvement of our CIT method over other methods was significant, we conducted a significance analysis on all pairs of experiments for each of the nine tasks in all three sparsity ratios using Friedman's test. Most of the resulting p-values were much smaller than the significance level α ($\alpha = 0.05$). Statistically significant results are marked with an asterisk $*$ in Tables 4.3, 4.4 and 4.5. Only one result was not a statistically significant improvement - the book target

Table 4.6 Comparison results of average MAE between CIT and five baselines

Task	non-transfer			cross-domain		
	PCC	FMM	SVD	CBT	RMGM	CIT
m2m	1.2433	1.0087	1.0283	1.1523	1.0137	**0.9929**
b2m	1.2433	1.0087	1.0283	1.3864	1.0354	**0.9929**
mu2m	1.2433	1.0087	1.0283	1.2532	1.0463	**0.9937**
m2b	1.2540	1.0371	1.0860	1.1589	1.0460	**1.0317**
b2b	1.2540	1.0371	1.0860	1.0853	1.0504	**1.0322**
mu2b	1.2540	1.0371	1.0860	1.1514	1.0753	**1.0319**
m2mu	1.3761	1.2509	1.2615	1.3248	1.2517	**1.2390**
b2mu	1.3761	1.2509	1.2615	1.4325	1.2683	**1.2393**
mu2mu	1.3761	1.2509	1.2615	1.3784	1.2812	**1.2392**

domain with a sparsity of 1.50% compared to the FMM non-transfer method in terms of MAE. However, CIT's performance improvement in the same scenario was significant at a data sparsity of 0.50%, suggesting that cross-domain transfer may not be required as data richness in target domain increases.

To better understand the effectiveness of transfer learning on each individual task, we calculated the average MAEs and RMSEs for each cross-domain recommendation task. The results are presented in Tables 4.6 and 4.7. The results for the nine tasks show that the proposed CIT method achieves the best performance in terms of both MAE and RMSE of the six methods.

Figure 4.5 compares the results for all the methods. Since the rating average is different between the source and target domains, the overall performance of cross-domain methods CBT and RMGM was not as good as the non-transfer learning method FMM. RMGM is relatively stable and is mostly better than SVD, while CBT fluctuates and is worse than most of the other methods. We can see that the overall performance in the music category is worse than that in the movie and book categories, indicating that the rating matrix in the music category has

Table 4.7 Comparison results of average RMSE between CIT and five baselines

Task	non-transfer			cross-domain		
	PCC	FMM	SVD	CBT	RMGM	CIT
m2m	1.5433	1.2163	1.2433	1.3628	1.2231	**1.1879**
b2m	1.5433	1.2163	1.2433	1.6724	1.2561	**1.1905**
mu2m	1.5433	1.2163	1.2433	1.5073	1.2662	**1.1907**
m2b	1.5594	1.2793	1.3693	1.3736	1.2783	**1.2538**
b2b	1.5594	1.2793	1.3693	1.3932	1.3013	**1.2544**
mu2b	1.5594	1.2793	1.3693	1.4261	1.3229	**1.2541**
m2mu	1.7330	1.4816	1.4921	1.5399	1.4862	**1.4455**
b2mu	1.7330	1.4816	1.4921	1.7263	1.5176	**1.4457**
mu2mu	1.7330	1.4816	1.4921	1.6475	1.5332	**1.4452**

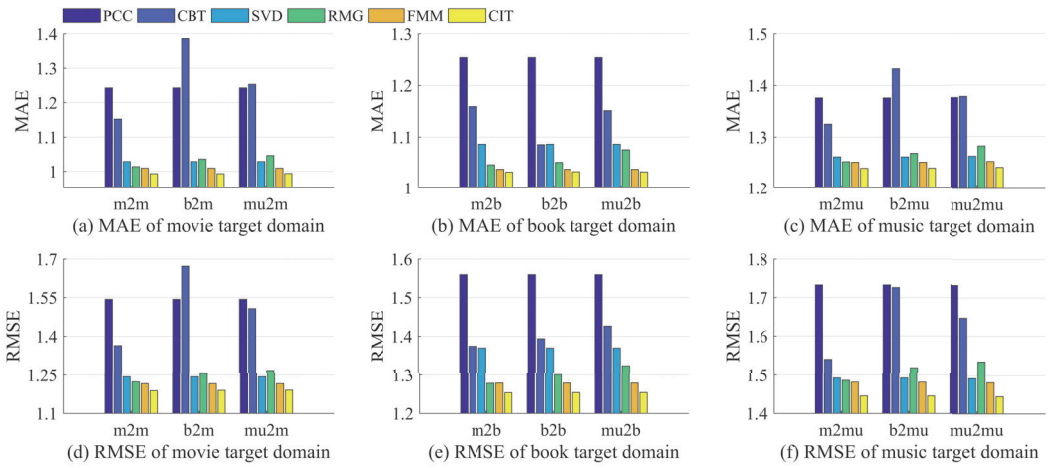


Figure 4.5 Comparison results for CIT and five other baselines.

different characteristics; however, our proposed method was still able to extract useful knowledge to help increase the prediction accuracy.

4.4.4 Parameter Analysis

In this section, we test how the parameters affect the performance of CIT. There are three parameters in the proposed CIT: K, L and λ. K is the number of user groups and L is the number of item groups. λ is the regularization factor for consistent

knowledge extraction. For simplicity, only the result for the movie to movie task has been included. Datasets with three sparsity ratios were used to test all three parameters. Both MAE and RMSE were used as evaluation metrics. As the results for MAE were similar to RMSE, only the results for RMSE have been included.

To analyze the parameter λ, K and L were fixed at 40. In Figure 4.6, we can see that RMSEs were not influenced significantly when λ was varied from 0.1 to 1.0. As for K and L, the number of user groups and the number of item groups did affect the RMSE, with a similar influence as described in previous papers: the higher the number, the higher the accuracy. In the range of 10 to 100, the influence of K and L is not significant. However, it took more time to run the algorithm when higher K and L values were chosen. This phenomenon was especially remarkable when K and L were larger than 100. To trade-off between an acceptable running speed for the algorithm and relative accuracy on RMSE, $K = 40$ and $L = 40$ were chosen for all experiments.

4.5 Discussion

Making decisions from an overwhelming volume of information is a crucial problem for both businesses and individual customers. And when a business begins operating in a new area, most existing recommender systems are not able to provide much guidance. The cross-domain recommendation method presented in this chapter is intended to help businesses and individual customers with decision-making in unchartered waters.

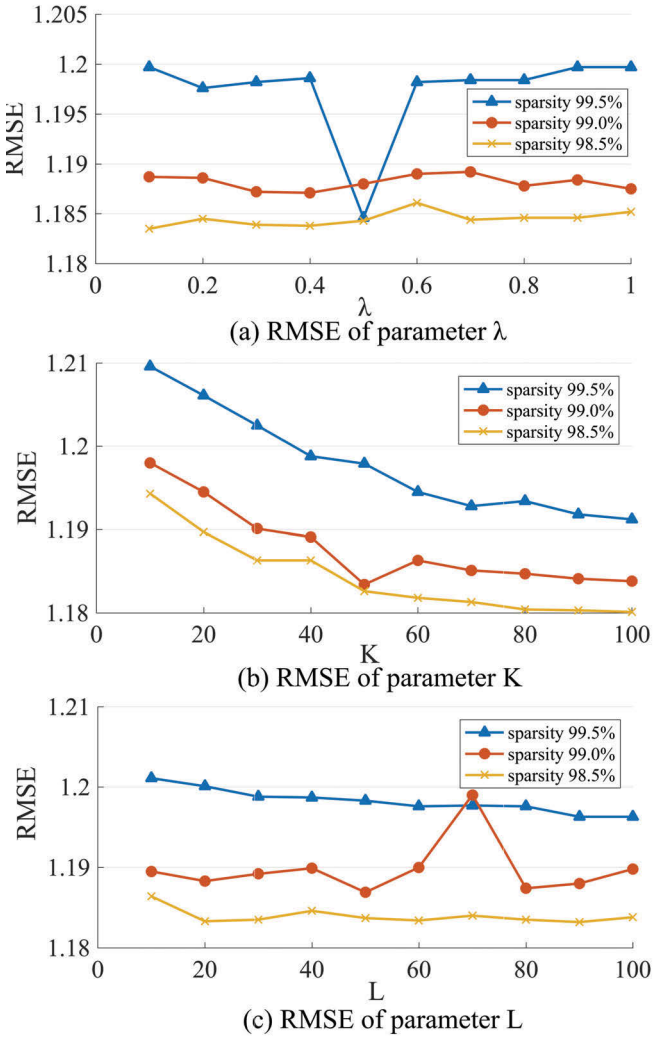


Figure 4.6 Parameter analysis of CIT.

4.5.1 Guidelines for Recommender System Developers

Recommender system developers will find the following guidelines useful:

Guideline #1: The CIT method should be used when two domains have different sparsity ratios. One domain should have a relatively sufficient amount of data; the other should be relatively sparse. There is no need to ensure user/item correspondence between the two domains.

Guideline #2: The CIT has been specially developed for two domains with divergent statistical properties (average and variance) and is appropriate for any divergence condition.

Guideline #3: If the users in target domain have no ratings at all, the CIT method is not suitable. If the sparsity ratio is more than 2.5%, developers should carefully consider whether or not to use the CIT method.

Guideline #4: The range of ratings should be normalized before using the CIT method.

4.5.2 Practical Applications

The proposed cross-domain recommendation method can be used to solve cold-start problems - a significant issue in the development and application of recommender systems. Developers can use this method to effectively transfer knowledge from a source domain with sufficient data to enhance recommendation models in a target domain. Our proposed method can be used when developing a recommender system to help businesses determine marketing strategies and to attract customers. The method can also provide end users with more effective decision-making support at the initial stage of a recommender system when very little data is available in the target domain. The improved recommendations the system provides will in turn help attract users, making the system grow more feasible and useful over time. Some examples of practical applications are provided below.

Our proposed method is used in the telecom product/service recommender system (Zhang et al., 2013). Telecommunications companies often introduce new product/service categories, such as new kinds of mobile plans. To attract customers

to their new revenue lines, it is important to generate accurate recommendations, and that requires new and specific recommendation models. However, creating an effective recommendation model with very little user and sales data can be challenging when a new product category is first introduced. Through the proposed method, sales data from a similar product category can be used as the source domain to enhance the recommendation model.

Our proposed method is also used in Smart BizSeeker, a B2B recommender system (Wu et al., 2015). Smart BizSeeker aims to recommend appropriate business partners to businesses in Australia. It also suffers from the cold-start problem, as initially there is very little rating data between businesses. However, similar B2B websites, such as Alibaba , contain a great deal of business rating data, which provides an opportunity to enhance Smart BizSeeker's recommendation model. The proposed cross-domain recommendation method effectively transfers knowledge from the rating data of other B2B websites to Smart BizSeeker to alleviate the cold-start problem.

Our method can also solve cold-start problems in G2B and G2C recommender systems (Al-Hassan et al., 2015) with a relevant source domain that contains sufficient rating data.

4.6 Summary

Developing CDRS is an efficient way to deal with the cold-start problem in recommender systems. However, using cross-domain recommendation without considering domain shift is little better than gambling (Cremonesi and Quadrana, 2014). If the knowledge extracted from the source domain just happens to fit the

target domain, the quality of recommendations may not suffer. However, if the knowledge does not, the likely result is inaccurate, poor quality recommendations. In this chapter, we proposed the CIT method to transfer consistent knowledge learned from a source domain to assist recommendations in a target domain with insufficient rating data. Unlike previous research on knowledge transfer recommender systems, our work investigates what knowledge to transfer and how to effectively transfer that knowledge from the source domain to the target domain. We put forward a tri-factorization method for a cross-domain knowledge transfer recommender system to acquire consistent knowledge. One advantage of the CIT method is that user and item groups are aligned using domain adaptation techniques to ensure consistent user/item group information in both domains. Another advantage is that the method does not require corresponding users and items across domains. Experiments were conducted on five real-world datasets spanning three categories of data and nine cross-domain recommendation tasks. The results show that the proposed CIT method achieves better performance than five other methods in both single and cross-domain settings. The CIT performs particularly well, comparatively, when there is wide deviation in the rating averages between domains.

Chapter 5

A Cross-domain Recommender System with Kernel-induced Knowledge Transfer

5.1 Introduction

To overcome data sparsity issues, CDRSs are specifically designed to provide recommendations in a target domain using information extracted from a source domain. However, the most crucial concern in CDRSs is how to extract common knowledge that can be shared between the two domains.

The methods for knowledge extraction and transfer are different depending on whether and how the entities in each domain overlap. Existing CDRSs usually assume that either none of the entities are common to both domains or they all are with full one-to-one mapping. Non-overlapping methods tend to extract shared knowledge based on collective group-level user behavior. Though many of

these methods have been designed to suit specific situations, they cannot integrate knowledge from an overlapping entity once new information becomes available. In fully-overlapping methods, the original source and target rating matrixes are collectively factorized, then the entities' features are extracted. Constraints on each entity ensure these features are exactly the same in the source and target domains so they can act as a bridge for knowledge transfer. However, practical situations rarely satisfy the "all" or "none" overlap assumption; rather, they fall somewhere in between as shown in Figure 5.1. In fully-overlapping methods, information about the overlapping entities is used to establish constraints between two domains. These constraints usually relate to the entities' features. However, even with the same user, there may be small differences in item rating patterns between two different domains, and this is called domain divergence. If the entity feature constraints are not handled delicately, knowledge transfer will suffer and reduce the prediction accuracy of the system.

Hence, cross-domain recommendation systems present the following challenges: 1) Feature inconsistency caused by data sparsity. Typically, there are no explicit features, only extracted latent features. And the observed sparse ratings do not fully represent a user's preferences, so features extracted from the same user in two different domains will be inconsistent. Thus, constructing an appropriate feature space is very challenging. 2) Feature inconsistency caused by domain heterogeneity. Extracted latent features from the overlapping entity can be aligned through domain adaptation techniques. But extracted latent features from non-overlapping entities lacks direct correlation and their features are heterogeneous. 3) Partially overlapping entities. The number of overlapping entities can be just a small part of the total number of entities in the target domain. Whether transferring knowledge through a

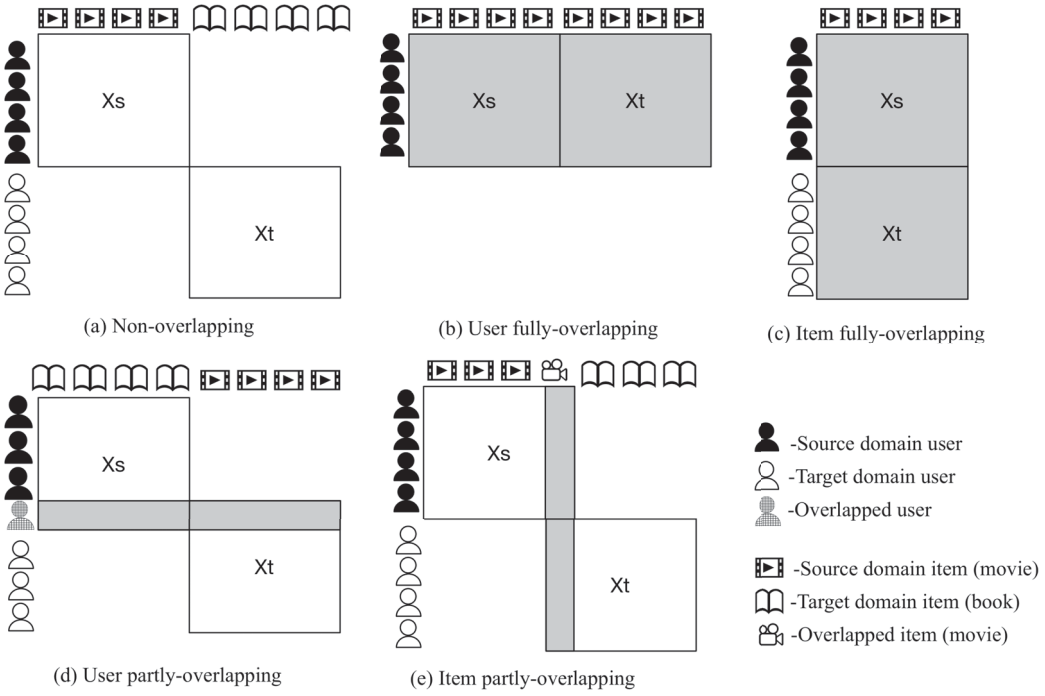


Figure 5.1 Different scenarios of overlapping entities.

small number of overlapping entities is effective, and what the shared constraints should be remain unsolved.

In this chapter, we propose a cross-domain recommender system with KerKT as a knowledge transfer method to improve recommendation performance with partially overlapping entities. We first factorize the rating matrixes separately to construct the user and item feature matrixes. To avoid divergence in the feature space caused by data sparsity, we propose a domain adaptation method to adjust the feature spaces through the overlapping entities. Then, we use a diffusion kernel to construct a full and complete entity similarity matrix, so the similarity measures can be used in heterogeneous settings. Finally, we use a more flexible constraint to jointly factorize the source and target rating matrixes.

The remainder of the chapter is organized as follows. Section 5.2 introduces the preliminaries and formally defines the problem to be solved. In Section 5.3, we present our KerKT method. Section 5.4 contains the empirical experiments. Four tasks are tested on four real-world datasets with three data sparsity ratios and three different levels of overlapping entities. The results show that our method performs better than six existing non-transfer and cross-domain methods. Finally, the summary is provided in Section 5.5.

5.2 Preliminaries and Problem Formulation

Matrix factorization is one of the most popular techniques used in recommender systems (Koren et al., 2009). In this section, a matrix factorization view of the recommender system in one domain is given to clearly describe the problem setting. The problem under study in this chapter is then formulated.

5.2.1 Recommendation Task based on Matrix Factorization in One Domain

Suppose there are M users and N items in one domain, the relationship between users and items is given as $\boldsymbol{X} \in \mathbb{R}^{M \times N}$ (bold letter represents a matrix). If a user's preferences are represented as ratings, then $\boldsymbol{X}$ is a rating matrix where $\boldsymbol{X}$ is subject to $X_{ij} \in \{1,2,3,4,5,?\}$ ("?" denotes a missing value). By minimizing its Euclidean distance to the original rating matrix $\boldsymbol{X}$ (Koren et al., 2009), $\boldsymbol{X}$ is approximated by

$$\hat{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{V}^T \tag{5.1}$$

Thus, $\boldsymbol{U} \in \mathbb{R}^{M \times K}$ is the user feature matrix and $\boldsymbol{V} \in \mathbb{R}^{N \times K}$ is the item feature matrix, which are two low-rank matrixes for users and items, respectively. The ith user and jth item are represented by the ith and jth row of the two matrixes as U_{i*} and V_{j*}. After matrix factorization, the users and items are mapped to a latent factor feature space of a lower dimensionality K.

The recommendation task is to predict the missing values in the rating matrix based on historical records of the users' preferences. Since the rating matrix $\boldsymbol{X}$ is usually extremely sparse, the low-rank approximation matrix factorization is easy to overfit. Regularization is usually used on low-rank feature matrixes to avoid this problem. In general, the optimization problem is:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \mathcal{L}(f(\boldsymbol{U},\boldsymbol{V}),\boldsymbol{X}) + \lambda \mathcal{R}(\boldsymbol{U},\boldsymbol{V}) \tag{5.2}$$

where $\mathcal{L}$ is the loss function of the predicted ratings $f(\boldsymbol{U},\boldsymbol{V})$ and the original ratings $\boldsymbol{X}$, $\mathcal{R}(\boldsymbol{U},\boldsymbol{V})$ is the regularization term, and $\lambda \geqslant 0$ is the regularization trade-off parameter. Specifically, the objective function with regularization terms to measure the loss and a Frobenius norm is (Mnih and Salakhutdinov, 2008):

$$J(\boldsymbol{U},\boldsymbol{V}) = \frac{1}{2}\|\boldsymbol{I} \circ (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T)\|_F + \frac{\lambda}{2}\|\boldsymbol{U}\|_F + \frac{\lambda}{2}\|\boldsymbol{V}\|_F \tag{5.3}$$

where $\boldsymbol{I}$ is the rating indicator matrix, $I_{ij} \in \{0,1\}$. $I_{ij} = 1$ indicates that the rating is observed, or $I_{ij} = 0$ otherwise. $\circ$ denotes the Hadamard product of the matrixes.

5.2.2 Problem Definition

The problem in this chapter is based on the assumption that ratings in the target domain are very sparse. This raises the question of how to use relatively dense

data in the source domain to assist a recommendation task in the target domain with overlapping entities. In practice, corresponding entities are not usually easy to identify. Typically, there are many unique entities between different datasets or platforms and only a few common entities. Thus, in this problem setting, the entities partially overlap. Only a small proportion of the entities in the target domain matrix $\boldsymbol{X}_t$ have observed correspondences in the source rating matrix $\boldsymbol{X}_s$. Even though the entities represent the same user and/or item, the rating a user has given or the rating an item has received can be different in each domain. The overlapping entity indicator matrix is represented by $\boldsymbol{W}^{(s,t)}$, $W_{ij}^{(s,t)} \in \{0,1\}$. $W_{ij}^{(s,t)} = 1$ indicates that the ith entity in the source domain is the same as the jth entity in the target domain, and $W_{ij}^{(s,t)} = 0$ otherwise. Without loss of generality, we require the rating rows of overlapping users to be at the top, and the corresponding users are in the same rows in both matrixes. This is achieved by permuting the rows of the original rating matrixes. Thus, the form of the entity indicator matrix $\boldsymbol{W}^{(s,t)}$ is:

$$\boldsymbol{W}^{(s,t)} = \begin{bmatrix} \boldsymbol{I}_o & 0 \\ 0 & 0 \end{bmatrix}$$

where $\boldsymbol{I}_o$ is an identity matrix of the same dimension as the number of overlapping entities. This problem is formally defined in the following.

Definition 5.1 (Cross-domain Recommender Systems with Partially Overlapping Entities). Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, a cross-domain recommender system based on partially overlapping entities is to assist with recommendation task $\hat{\boldsymbol{X}}_t = \boldsymbol{U}_t \boldsymbol{V}_t^T$ through an auxiliary source rating matrix $\boldsymbol{X}_s$ and an overlapping entity indicator matrix $\boldsymbol{W}^{(s,t)}$.

5.3 A CDRS with Kernel-induced Knowledge Transfer

This section introduces our KerKT method. The overlapping entities in each domain may be either users or items. For the purposes of this presentation, we have assumed the users overlap. Overlapping items are handled in the same way and have, therefore, been omitted from this research. The section begins with an overview of the entire method, then each of the five steps is explained in detail.

5.3.1 KerKT Method Overview

To enable knowledge sharing between the source and target domains with overlapping users, constraints on user feature matrixes are added to the collective matrix factorization of the source and target rating matrixes. Previous research assumes 'identical' factor matrixes for overlapping entities, but this assumption is too limiting to satisfy in practice. Instead, we have chosen to constrain the similarities between the entities in each domain as a bridge for knowledge transfer. However, while it is easy to measure the similarities between entities in the same domain, inter-domain entity similarities cannot be computed directly.

The overlapping entities are mapped to the same feature space through domain adaptation techniques, while the non-overlapping entities are connected by diffusion kernel completion. Thus, the similarities between all users in both domains can be measured. Further, constraining the user features using these similarities may lead to a better optimization result. The optimization problem is formalized as:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \ \mathcal{L}(f(\boldsymbol{U},\boldsymbol{V}),\boldsymbol{X}) + \lambda\mathcal{R}(\boldsymbol{U},\boldsymbol{V}) + \lambda_o\mathcal{R}_o(\boldsymbol{U}) \tag{5.4}$$

where $\mathcal{R}_o(\boldsymbol{U})$ is the regularization term for the entity similarity constraints derived from overlapping users, and $\lambda_o \geqslant 0$ is the regularization trade-off parameter.

The KerKT method consists of five steps, as shown in Figure 5.2. 1). The user features and item features are extracted separately from the source and target domains, and the two sets of user features are aligned to the same feature space through overlapping users. 2). The item features are regulated according to the original rating matrixes and the aligned user feature matrixes. 3). The user and item feature matrixes resulting from the previous two steps are used to measure the user and item similarities in one domain. 4). Kernel-induced completion is conducted to measure the inter-domain user similarities. 5) The user/item features are re-trained based on the constraints of the entity similarities, then recommendations are made. We have selected a specific algorithm to perform each step, but other suitable feature extraction or domain adaptation algorithms could be used as substitutes.

5.3.2 KerKT Method

Our proposed KerKT method is comprised of the five steps.

5.3.2.1 Step 1: Extracting and aligning user features in both domains

In this step, the source rating matrix $\boldsymbol{X}_s$ and target rating matrix $\boldsymbol{X}_t$ are separately factorized, which results in the user feature matrix $\boldsymbol{U}_s$ for the source domain and $\boldsymbol{U}_t$ for the target domain. Recall that the users in the source and target domains partially overlap. Accordingly, each user feature matrix can be divided into two parts: one containing the overlapping users; the other containing non-overlapping users. The overlapping user feature matrix for the source domain is denoted as $\boldsymbol{U}_{s,o}$

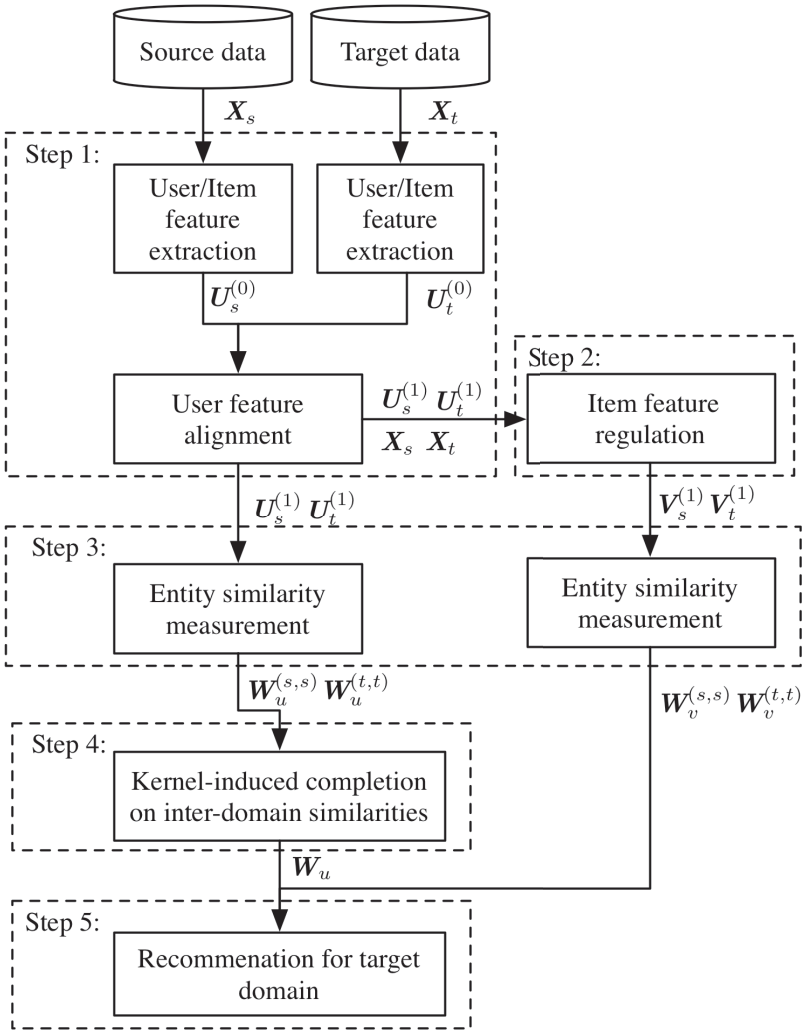


Figure 5.2 Procedure of the KerKT method.

and $\boldsymbol{U}_{s,n}$ denotes the non-overlapping matrix. The same goes for the target domain i.e., $\boldsymbol{U}_{t,o}$ and $\boldsymbol{U}_{t,n}$.

Assuming the overlapping users have similar tastes or preferences in both domains, we can use them as a bridge to transfer knowledge. However, as mentioned in the Introduction, even the same user's rating patterns may not be completely the same in two different domains. Data sparsity exacerbates this condition and may lead to two different factorized user feature vectors with different physical

meanings. Hence, setting the similarity of the overlapping user entities to 1 may lead to inaccurate similarity measurements, which would eventually negatively impact the effectiveness of the knowledge transfer in the following steps. Therefore, before using the entity correspondences as a strong condition, we need to ensure that the overlapping users in both domains are represented in the same feature space. This is referred to as "subspace alignment" in transfer learning.

The aim is to map the user feature spaces of two overlapping users into a common subspace where domain shift has been eliminated, so the overlapping users ultimately share the same feature space across both domains. In the source domain, the jth column of the overlapping user feature matrix is the representation of the jth user feature. We use a marginal probabilistic distribution of the jth column to represent the characteristics of the user features in each matrix. Thus, the goal is to minimize the differences between the marginal probabilistic distributions of the user features for the source domain and the target domain. If the marginal probability distributions of one user feature are the same in both domains, then the two user features are considered to have the same physical meaning. In this way, we can align the two user feature spaces. In our previous research, we provided a definition for information consistent tri-factorization. However, here, since the scenario and the matrix factorization model are different, this definition has been refined into a definition for consistent matrix factorization with partially overlapping users in the following.

Definition 5.2 (Consistent Matrix Factorization with Partially overlapping Users). Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix

$\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ can be factorized as follows:

$$\boldsymbol{X_s} = \begin{bmatrix} \boldsymbol{U}_{s,o} \\ \boldsymbol{U}_{s,n} \end{bmatrix} \boldsymbol{V}_s^T \tag{5.5}$$

$$\boldsymbol{X_t} = \begin{bmatrix} \boldsymbol{U}_{t,o} \\ \boldsymbol{U}_{t,n} \end{bmatrix} \boldsymbol{V}_t^T \tag{5.6}$$

where $\boldsymbol{U}_{s,o}$ and $\boldsymbol{U}_{t,o}$ are the overlapping user feature matrixes in the source domain and the target domain, and $\boldsymbol{U}_{s,n}$ and $\boldsymbol{U}_{t,n}$ are the non-overlapping user feature matrixes, respectively.

If both factorizations satisfy the following equation, then they are consistent matrix factorizations:

$$P(\boldsymbol{U}_{s,o}) = P(\boldsymbol{U}_{t,o}) \tag{5.7}$$

where $P(\boldsymbol{U}_{s,o})$ and $P(\boldsymbol{U}_{t,o})$ represent the marginal probability distribution of $\boldsymbol{U}_{s,o}$ and $\boldsymbol{U}_{t,o}$. Thus, the user feature spaces in both the source and target domains are aligned.

To solve a matrix factorization optimization problem that satisfies the above constraints is almost impossible. According to *Definition 5.2*, we can find a mapping function for those two matrixes to achieve the following equation:

$$P(\boldsymbol{\Psi_s}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)})) = P(\boldsymbol{\Psi_t}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)})) \tag{5.8}$$

A GFK is a domain adaptation strategy to find a space that two different feature spaces can be projected into, thus eliminating the divergence of two distributions. We can use this strategy to find a mapping function to align the two user feature spaces formed by overlapping users. Once the GFK operators $\boldsymbol{\Psi_s}(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)})$ are

determined, they can be used through the following mapping functions:

$$\boldsymbol{\Psi_s}(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) = \boldsymbol{U}_s^{(0)} \times \boldsymbol{\Psi_G}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)}) \tag{5.9}$$

$$\boldsymbol{\Psi_t}(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) = \boldsymbol{U}_t^{(0)} \times \boldsymbol{\Psi_G}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)}) \tag{5.10}$$

where $\boldsymbol{\Psi_G}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)})$ are the GFK operators. More details can be found in Appendix A and (Gong et al., 2014; Zhang et al., 2017) .

With the divergence eliminated through these mappings, the new representations of the overlapping users will satisfy the conditions in *Definition 5.2*. Non-overlapping users also need to be projected onto the same feature space. The mapping functions $\boldsymbol{\Psi_s}$ and $\boldsymbol{\Psi_t}$ are used for this purpose.

$$\boldsymbol{U}_s^{(1)} = \boldsymbol{\Psi_s}(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \tag{5.11}$$

$$\boldsymbol{U}_t^{(1)} = \boldsymbol{\Psi_t}(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) \tag{5.12}$$

where $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$ are the aligned user feature matrixes after mapping, and $\boldsymbol{\Psi_s}$ and $\boldsymbol{\Psi_t}$ are the mapping functions using GFK.

How the new user feature spaces are derived and how $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$ are learned is summarized in Algorithm 1.

5.3.2.2 Step 2: Item feature regulation in both domains

In matrix factorization, the user feature matrix and the item feature matrix are both low-rank matrixes that map users and items to the same k-dimensional feature space. So, once the user feature spaces are aligned, the item feature matrixes should be regularized to the new k-dimensional feature space. The new item feature

Algorithm 5.1: Consistent User Feature Extraction

Input:

$\boldsymbol{X}_s$, the source rating matrix;
$\boldsymbol{X}_t$, the target rating matrix;
$\boldsymbol{W}_u^{(s,t)}$, the overlapping user indicator matrix;

Output:

$\boldsymbol{U}_s^{(1)}$, the aligned user feature matrix in source domain;
$\boldsymbol{U}_t^{(1)}$, the aligned user feature matrix in target domain;

1: Factorize $\boldsymbol{X}_s$ and obtain user feature matrix $\begin{pmatrix} \boldsymbol{U}_{s,o}^{(0)} \\ \boldsymbol{U}_{s,n}^{(0)} \end{pmatrix}$ as in Equation (5.3)
2: Factorize $\boldsymbol{X}_t$ and obtain user feature matrix $\begin{pmatrix} \boldsymbol{U}_{t,o}^{(0)} \\ \boldsymbol{U}_{t,n}^{(0)} \end{pmatrix}$ as in Equation (5.3)
3: Obtain GFK operator $\boldsymbol{\Psi_G}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)})$ as in equation (A.2) in (Zhang et al., 2017)
4: Obtain mapping functions $\boldsymbol{\Psi_s}$ and $\boldsymbol{\Psi_t}$ as in Equation (5.9)
5: **return** $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$

matrixes are obtained by minimizing the distance between the approximations of the low-rank matrixes and the original data in the rating matrix. A Frobenius norm is used to measure the distance. The cost function of source domain matrix follows, the target domain matrix has the same form:

$$J_v(\boldsymbol{V}_s^{(1)}) = \frac{1}{2}\|\boldsymbol{I_s} \circ (\boldsymbol{X}_s - \boldsymbol{U}_s^{(1)}(\boldsymbol{V}_s^{(1)})^T)\|_F + \frac{\lambda_{V_s}}{2}\|\boldsymbol{V}_s^{(1)}\|_F \tag{5.13}$$

where λ_{V_s} is the regularization parameter. The item feature matrixes are learned by optimizing:

$$min\ J_v(\boldsymbol{V}_s^{(1)}) \tag{5.14}$$

Gradient descent is used for this optimization. The update rule is:

$$\boldsymbol{V}_s^{(1)} \leftarrow \boldsymbol{V}_s^{(1)} - \eta_{V_s}[(\boldsymbol{U}_s^{(1)}(\boldsymbol{V}_s^{(1)})^T - \boldsymbol{X}_s)\boldsymbol{U}_s^{(1)} + \lambda_{V_s}\boldsymbol{V}_s^{(1)}] \tag{5.15}$$

Algorithm 5.2: Item Feature Regularization

Input:

$\boldsymbol{X}_s$, the source rating matrix $\boldsymbol{U}_s^{(1)}$, the user feature matrix

Output:

$\boldsymbol{V}_s^{(1)}$, the regularized item feature matrix

Initialize $\boldsymbol{V}_s^{(1)} \in \mathbb{R}^{N_s \times K}$, Initialize $J_v(\boldsymbol{V}_s)$ and $J_v(\boldsymbol{V}_s)^{(pre)}$
while $J_v(\boldsymbol{V}_s)^{(pre)} - J_v(\boldsymbol{V}_s) > \varepsilon$ **do**
    $J_v(\boldsymbol{V}_s)^{(pre)} = J_v(\boldsymbol{V}_s)$
    Update $\boldsymbol{V}_s$ as in Equation (5.15)
    Update $J_v(\boldsymbol{V}_s)$ as in Equation (5.13)
end while
return $\boldsymbol{V}_s^{(1)}$

where the learning rate is η_{V_s}. $\boldsymbol{V}_t^{(1)}$ can be obtained through the same process.

This step is summarized in Algorithm 2.

5.3.2.3 Step 3: Entity similarity measures in one domain

This step calculates the user and item similarities in one domain. Since the rating matrix is very sparse, making this calculation directly from the rating matrix can lead to inaccurate results. Hence, using the PMF formulation introduced in Section 5.2.1, one rating X_{ij} is generated from a user latent feature vector U_{i*} and an item latent feature vector V_{*j}. Thus, the source rating matrix and target rating matrix can be factorized as $\boldsymbol{X}_s = \boldsymbol{U}_s \boldsymbol{V}_s^T$ and $\boldsymbol{X}_t = \boldsymbol{U}_t \boldsymbol{V}_t^T$. This is a dimensionality reduction and data compression process as users/items are mapped to a lower k-dimensional feature space (usually $k \ll M, k \ll N$). Once complete, users and items are represented as full k-dimensional feature matrixes, and the user/item similarities can be calculated from the user/item feature matrixes.

Similarity measurements are easy with user and item feature spaces in one domain since the feature spaces are homogeneous. And there are many

suitable choices for performing these calculations, such as cosine similarity, Pearson's similarity, Euclidean measurement, or the **R**adial **B**asis **F**unction (RBF) measurement. The choice depends on the situation and the characteristics of the domain. For example, cosine similarity is very popular and effective for word count and text similarity measurements due to the advantages of using angles rather than distance. Pearson's measurement tends to be more effective in memory-based collaborative filtering methods owing to its emphasis on averages. In this problem, we are measuring user similarity from a user feature matrix where the feature values are real numbers, so the following RBF measurement is the most appropriate: $W_{ij} = e^{-\frac{\|U_{i*}-U_{j*}\|^2}{\sigma^2}}$, where σ^2 is set to be the median of all the non-zero values calculated by $\|U_{i*}-U_{j*}\|^2$.

5.3.2.4 Step 4: Kernel induced completion of inter-domain user similarity

In inter-domain user similarity measurement, the user feature spaces are not the same and the user features are heterogeneous, which means their similarities cannot be calculated directly. However, given the first three steps, some user similarities between the source and target domains are now known. The overlapping entity indicator matrix $\boldsymbol{W}^{(s,t)}$ contains the observed overlapping user information. Hence, a full user similarity matrix can be constructed as:

$$\boldsymbol{W}_u = \begin{bmatrix} \boldsymbol{W}_u^{(s,s)}, & \boldsymbol{W}_u^{(s,t)} \\ \boldsymbol{W}_u^{(t,s)}, & \boldsymbol{W}_u^{(t,t)} \end{bmatrix} \tag{5.16}$$

where $\boldsymbol{W}_u^{(s,s)}$ and $\boldsymbol{W}_u^{(t,t)}$ represent the user similarities in the source and target domains, respectively, and $\boldsymbol{W}_u^{(s,t)} = (\boldsymbol{W}_u^{(t,s)})^T$ represents the inter-domain user similarities.

As in Step 2, the two feature spaces of the overlapping users are aligned to eliminate feature space divergence. Therefore, it is reasonable to set the similarity of observed overlapping users to $(W_u^{(s,t)})_{ij} = 1$ in $\boldsymbol{W}^{(s,t)}$. For now, the similarities between the overlapping users are the only known entries in the inter-domain similarity matrix. We need to complete $\boldsymbol{W}_u^{(s,t)}$ using the information from $\boldsymbol{W}_u$. Note that, here, the non-overlapping users and their features are heterogeneous. Thus, their similarities cannot be computed directly.

This matrix completion problem has a strong connection to a bipartite edge completion problem (He et al., 2017). In step 3, the user similarities are all measured within one domain. As a result, we have fully connected nodes in the graph representations of both the source and target domains, as indicated by the red nodes and blue nodes in Figure 5.3. The overlapping users are shown as purple nodes in the graph, and they act as "bridge" to couple the two graphs. To complete the user similarity matrix $\boldsymbol{W}_u$ requires filling in all the edges from the entire graph. The subscript u has been omitted to simplify the notation.

In network propagation, a random walk is a good way to reach to all the nodes. As shown in Figure 5.3, one user entity denoted as a node x in the source domain has a fully connection with all the other node in the source domain $(W^{(s,s)})_{xp}, p \in \mathcal{U}_s$, so does node y in target domain $W_{yq}^{(t,t)}, q \in \mathcal{U}_t$. If node p in source domain and node q in target domain are overlapping users, i.e., they are the same user, then $W_{pq}^{(s,t)} = 1$. The two nodes x and y are connected and their similarity can be calculated as: $W_{xy}^{(s,t)} \leftarrow W_{xp}^{(s,s)} W_{pq}^{(s,t)} W_{yq}^{(t,t)}$. By aggregating all the nodes connected to x in the source domain and y in the target domain, the edge can be completed

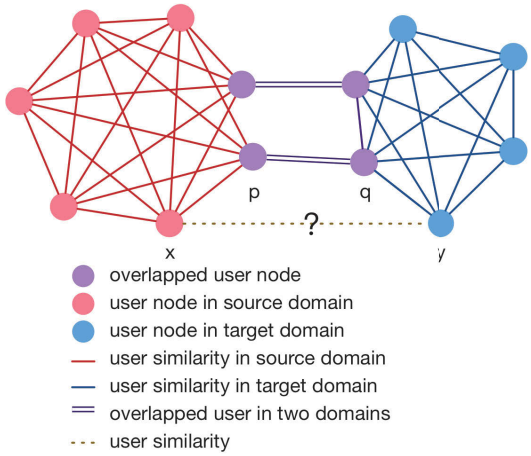


Figure 5.3 Graphical view of user relationships in source and target domains

with: $W_{xy}^{(s,t)} \leftarrow \sum_{p \in U_s} \sum_{q \in U_t} W_{xp}^{(s,s)} W_{pq}^{(s,t)} W_{yq}^{(t,t)}$. In a matrix form, this is written as:

$$W^{(s,t),(1)} = W^{(s,s)} W^{(s,t)} W^{(t,t)} \tag{5.17}$$

The above equation can be treated as a one-step random walk from both the source domain and the target domain. Generally, M steps of random walk are taken in total from the source and target sides, and all the possible steps are added together to complete the final graph:

$$W^{(s,t),(K)} = \sum_{K=0}^{M} \binom{M}{K} (W^{(s,s)})^K W^{(s,t)} (W^{(t,t)})^{M-K} \tag{5.18}$$

However, the goal in this problem is to find all the similarities between all the users in both domains. Therefore, a finite number of random walk steps may not identify all the possible relationships, but it would be more likely to associate all the indirectly connected users if K was infinite. Hence, we use the diffusion kernel

completion method (Chang et al., 2017) to complete the user similarity matrix:

$$W^{(s,t)} = e^{(\beta_s W^{(s,s)})} W^{(s,t)} e^{(\beta_t W^{(t,t)})} \tag{5.19}$$

where β_s and β_t are two positive scalars to regulate the weights of the source and the target domains.

5.3.2.5 Step 5: Collective matrix factorization with user similarity constraints

With all similarities measured and all the pairs of nodes connected, a fully connected graph can be constructed. A very common strategy for increasing computational speed is to remove edges, leave a sparse graph (Zhu et al., 2005). This approach tends to achieve better performance empirically as it emphasizes local information with high similarity while ignoring information that is likely to be false. Hence, the k nearest neighbors of each node are retained in a similar way to the original memory-based collaborative filtering strategy.

In the scenario of this chapter, only overlapping users are observed but items are non-overlapping. The users have both intra-domain and inter-domain similarities, but the items only have intra-domain similarities. Based on the assumption "similar users have similar tastes and will thus choose similar items to consume", both intra-domain and inter-domain similarities are used to constrain the proposed matrix factorization as prior knowledge. In terms of intra-domain similarities, though the data in the target domain are very sparse, they are still very valuable for measuring the similarities between users/items so as to constrain the matrix factorization. As for inter-domain similarities, users in target domain are not only correlated to users in their own domain but also in the source domain via the overlapping users. As

a consequence, users in the source domain with similar preferences to users in the target domain are transferred as knowledge to improve the performance of recommender system.

The constrains result in users who are similar tend to have similar latent factors. Specifically, the regularization form is (Zhen et al., 2009):

$$\mathcal{R}_o(\boldsymbol{U}) = tr(\boldsymbol{U}^T \boldsymbol{L} \boldsymbol{U}) \tag{5.20}$$

where $\boldsymbol{L}$ denotes a Laplacian matrix, and $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$. $\boldsymbol{W}$ is the user similarity matrix, and $\boldsymbol{D}$ is a diagonal matrix defined as $D_{ii} = \sum_j W_{ij}$. Note that although the form of regularization is quite similar, our method is different from (Zhao et al., 2017). In (Zhao et al., 2017), similarities between entities from source domain are directly used in the target domain as prior knowledge without considering the feature inconsistency between two domains. By contrast, in our method, the similarities between entities in target domain are learned by use of the domain adaption technique which ensures the feature consistency and diffusion kernel technique which makes the similarity constraints more accurate and complete. Our proposed constraints are more flexible and reasonable to satisfy in practice. We achieve the goal by minimizing the following objective function:

$$\begin{aligned} f(\boldsymbol{U_s}, \boldsymbol{V_s}, \boldsymbol{U_t}, \boldsymbol{V_t}) = & \frac{\alpha}{2} \|\boldsymbol{I}_s \circ (\boldsymbol{X}_s - \boldsymbol{U}_s \boldsymbol{V}_s^T)\|_F + \frac{1}{2} \|\boldsymbol{I}_t \circ (\boldsymbol{X}_t - \boldsymbol{U}_t \boldsymbol{V}_t^T)\|_F \\ & + \frac{\lambda_u}{2} [tr(\boldsymbol{U}_s^T \boldsymbol{L}_u^{(s,s)} \boldsymbol{U}_s) + tr(\boldsymbol{U}_s^T \boldsymbol{L}_u^{(s,t)} \boldsymbol{U}_t) \\ & + tr(\boldsymbol{U}_t^T (\boldsymbol{L}_u^{(s,t)})^T \boldsymbol{U}_s) + tr(\boldsymbol{U}_t^T \boldsymbol{L}_u^{(t,t)} \boldsymbol{U}_t)] \\ & + \frac{\lambda_v}{2} [tr(\boldsymbol{V}_s^T \boldsymbol{L}_v^{(s,s)} \boldsymbol{V}_s) + tr(\boldsymbol{V}_t^T \boldsymbol{L}_v^{(t,t)} \boldsymbol{V}_t)] \\ & + \frac{\lambda}{2} (\|\boldsymbol{U}_s\|_F + |\boldsymbol{V}_s\|_F + \|\boldsymbol{U}_t\|_F + |\boldsymbol{V}_t\|_F) \end{aligned} \tag{5.21}$$

where tr is the trace of a matrix, $\alpha \in (0,1)$ is trade-off parameter to balance the source and target domain data, and λ_u, λ_u and λ_u are the regularization parameters to control the influence of the constraints on the user similarities, item similarities and algorithm complexity. Details on how these parameters affect the proposed method are presented in Section 5.4. Using gradient descent, the objective function is minimized with the following update rules:

$$\boldsymbol{U}_s \leftarrow \boldsymbol{U}_s - \eta_{U_s}[\alpha(\boldsymbol{U}_s\boldsymbol{V}_s^T - \boldsymbol{X}_s)\boldsymbol{V}_s + \lambda_u \boldsymbol{L}_u^{(s,s)}\boldsymbol{U}_s + \frac{\lambda_u}{2}\boldsymbol{L}_u^{(s,t)}\boldsymbol{U}_t + \lambda \boldsymbol{U}_s] \tag{5.22}$$

$$\boldsymbol{V}_s \leftarrow \boldsymbol{V}_s - \eta_{V_s}[\alpha(\boldsymbol{V}_s\boldsymbol{U}_s^T - \boldsymbol{X}_s^T)\boldsymbol{U}_s + \lambda_v \boldsymbol{L}_v^{(s,s)}\boldsymbol{V}_s + \lambda \boldsymbol{U}_s] \tag{5.23}$$

$$\boldsymbol{U}_t \leftarrow \boldsymbol{U}_t - \eta_{U_t}[(\boldsymbol{U}_t\boldsymbol{V}_t^T - \boldsymbol{X}_t)\boldsymbol{V}_t + \lambda_u \boldsymbol{L}_u^{(t,t)}\boldsymbol{U}_t + \frac{\lambda_u}{2}(\boldsymbol{L}_u^{(s,t)})^T\boldsymbol{U}_s + \lambda \boldsymbol{U}_t] \tag{5.24}$$

$$\boldsymbol{V}_t \leftarrow \boldsymbol{V}_t - \eta_{V_t}[(\boldsymbol{V}_t\boldsymbol{U}_t^T - \boldsymbol{X}_t^T)\boldsymbol{U}_t + \lambda_v \boldsymbol{L}_v^{(t,t)}\boldsymbol{V}_t + \lambda \boldsymbol{U}_t] \tag{5.25}$$

By updating $\boldsymbol{U}_s$, $\boldsymbol{V}_s$, $\boldsymbol{U}_t$ and $\boldsymbol{V}_t$ iteratively, we achieve at a final optimized approximation of $\hat{\boldsymbol{X}}_t = \boldsymbol{U}_t\boldsymbol{V}_t^T$. Recommendations are given according to the rating prediction for the target domain.

5.4 Experiments and Analysis

This section presents the experimental results and related analysis. The datasets and evaluation metrics are introduced first, followed by the experimental settings and baseline methods. Then, we present the results of the empirical experiments, with a parameter analysis to conclude the section.

Table 5.1 Statistics of original datasets for experiments on KerKT

	Movielens20M	Netflix	Amazon_book	Douban_movie	Douban_book
#user	138493	480189	8026324	28718	26877
#item	26744	17770	2330066	57424	187520
#rating	20000263	100480507	22507155	2828585	1097148
sparsity	0.54%	1.18%	0.0001%	0.17%	0.02%
rating_range	0.5-5	1-5	1-5	1-5	1-5

5.4.1 Datasets and Evaluation Metrics

Our method was tested under the conditions that the source and target domains share some overlapping users and/or items. For a fair comparison, we chose movies and books as the recommendation subject - two commonly-used categories in previous research on CDRSs. Four real-world datasets were used in our experiments: Movielens[1], Netflix[2], AmazonBook[3] (He and McAuley, 2016) and Douban[4] (Zhong et al., 2014). Each of these datasets is publicly available and has been tested on single domain recommendation in a variety of situations, but rarely in cross-domain recommendation. Our experiments are a supplement to the deficiency of tests on this specific problem setting. The statistical information for these datasets is presented in Table 5.1.

From AmazonBooks, we removed all users who had given exactly the same rating for every book, as these data are not effective for constructing a recommender system (Zhang et al., 2017). Movielens20M was normalized to the range of $\{1,2,3,4,5\}$. Four cross-domain recommendation tasks were designed for experiments:

- **Task 1**: movie $\rightarrow$ movie, user-overlap, Movielens20M

[1]https://grouplens.org/datasets/movielens/20m/
[2]https://netflixprize.com/index.html
[3]http://jmcauley.ucsd.edu/data/amazon
[4]https://sites.google.com/site/erhengzhong/datasets

- **Task 2**: movie → movie, item-overlap, Netflix
- **Task 3**: book → book, item-overlap, AmazonBook
- **Task 4**: movie → book, user-overlap, Douban

In the first three tasks, we used the data from one dataset and split the entities into the source domain and the target domain to simulate entity overlaps. The fourth task was designed for Douban, a real-world rating website where users can rate movies, books, and music. We now take the Task 1 as an example to describe the data selection. Task 2 and Task 3 are similar but with overlapping items. For source domain data, we filtered out the users who had given less than a total of 20 ratings and items who had received less than 10 ratings. We randomly selected 2000 items and 2000 users, constraining the sparsity to 2% to ensure the one was relatively dense. In the 2000 users, we randomly chose 200 users to be overlapping users. We then randomly selected 1800 users that have no correspondence with the 2000 users in the source domain. Totally they compose the 2000 users for the target domain data. We also randomly chose 2000 items for the target domain data that have no intersections with items in the source domain. Three sparsity ratios on the target domain data were used to compare different algorithms in different circumstances. The details of the final datasets are summarized in Table 5.2. For evaluation metrics, MAE and RMSE were used as defined in Eq. (3.12) and (4.25).

5.4.2 Experimental Settings and Baselines

Three non-transfer learning methods were chosen for comparison: PCC (Deshpande and Karypis, 2004), FMM (Si and Jin, 2003) and **P**robabilistic **M**atrix **F**actorization (PMF) (Mnih and Salakhutdinov, 2008), along with three cross-domain recommen-

Table 5.2 Description of data subsets for four tasks on KerKT.

Task	Data_name	Data_source	Domain	Sparsity	No. of entity overlapping
Task 1	task1_s1	Movielens20M	source	2.00%	200, 100, 50
	task1_t1	Movielens20M	target	0.50%	200, 100, 50
	task1_t2	Movielens20M	target	1.00%	200, 100, 50
	task1_t3	Movielens20M	target	1.50%	200, 100, 50
Task 2	task2_s1	Netflix	source	2.00%	200, 100, 50
	task2_t1	Netflix	target	0.50%	200, 100, 50
	task2_t2	Netflix	target	1.00%	200, 100, 50
	task2_t3	Netflix	target	1.50%	200, 100, 50
Task 3	task3_s1	AmazonBook	source	2.00%	200, 100, 50
	task3_t1	AmazonBook	target	0.50%	200, 100, 50
	task3_t2	AmazonBook	target	0.63%	200, 100, 50
	task3_t3	AmazonBook	target	0.75%	200, 100, 50
Task 4	task4_s1	DoubanMovie	source	2.00%	200, 100, 50
	task4_t1	DoubanBook	target	0.50%	200, 100, 50
	task4_t2	DoubanBook	target	1.00%	200, 100, 50
	task4_t3	DoubanBook	target	1.50%	200, 100, 50

dation methods, CBT (Li et al., 2009a), RMGM (Li et al., 2009b) and PMFTL (Zhao et al., 2017). PCC is the classical memory-based collaborative filtering method. FMM is a graphical model designed to allow one user/item to be clustered into several groups simultaneously. Empirically, it has been proven to be more effective in providing recommendations to users with few historical ratings. RMGM is a cross-domain recommendation method evolving out of the single domain FMM. CBT is also a cross-domain recommendation method. Both of these methods were designed for scenarios with no overlapping users or items. PMFTL is a transfer learning method for cross-domain scenarios with entity overlap as proposed in (Zhao et al., 2017). For a fair comparison, we removed the active learning module in the originally proposed method. PMFTL was developed on the basis of PMF with partially overlapping entities. PMFTL has more relaxed constraints than TCF (Pan and Yang, 2013). TCF was designed for problems where users and items have a

one-to-one mapping. The constraints in TCF are strict. One constraint requires that the user and item feature matrixes in the source and target domains are exactly the same. Whereas, PMFTL uses similarities estimated in the source domain directly as constraints in the target domain. Since TCF cannot be used to solve the problem presented in this chapter, we did not select it for comparison.

User-based collaborative-filtering is used for PCC. The number of neighboring user is set to be 50. For FMM, CBT and RMGM, the number of user and item groups were both set to be 50. For PMF and PMFTL, we set $\lambda = \{0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 1\}$. The parameter settings for KerKT were $\alpha = 0.5$, $\lambda_u = \{0.001, 0.01\}$, $\lambda_v = \{0.001, 0.01\}$ and $\lambda = \{0.0001, 0.001, 0.01\}$. KerKT and all the baselines, except for PCC, were randomly initialized. The results of 20 random initializations were averaged; standard deviations are reported.

5.4.3 Results

The results with these four datasets are shown in Table 5.3, 5.5, 5.4 and 5.6. KerKT delivered the best performance of all the comparison methods on all four cross-domain recommendation tasks. This verifies the conclusion that using overlapping entities as a bridge for transferring knowledge is useful in CDRSs. Our analysis of the results revealed the following observations:

1. **Comparison with non-transfer learning methods.** The performance of non-transfer learning methods was relatively poor on sparse data. As the basis of CBT and RMGM, FMM was designed to predict ratings for users with little available data. Generally, FMM performed better than PMF and memory-based method PCC without transfer learning techniques. PMF is the basis of our proposed method KerKT. In all the experiment results, KerKT

significantly outperformed all the non-transfer learning recommendation techniques.

2. **Comparison with cross-domain recommendation methods for non-overlapping entities.** RMGM showed improved precision in recommendations over its basis, FMM, but sometimes the improvement was not significant (see Table 5.6). CBT did not always improve the performance of the recommender system and sometimes suffered from negative transfer, indicating that CBT is not stable when transferring knowledge (see Table 5.4). Neither of these methods uses non-overlapping entity information explicitly, but rather extract cluster-based knowledge to share between the source and target domains. KerKT outperformed both these methods in all recommendation tasks, again, proving that overlapping entities can serve as a bridge for transferring knowledge to the target domain. Methods designed for scenario of non-overlapping entities can be applied to solve the problem proposed in this chapter as a substitution. But we can see from the results that they did not show advantages since they did not use the overlap information.

3. **Comparison with cross-domain recommendation methods for partially overlapping entities.** As the methods developed for partially overlapping entities are rare and methods developed for fully overlapping entities cannot be used to solve the problem proposed in this chapter. We only choose PMFTL as a representative of cross-domain recommendation method for partially overlapping entities. PMFTL was developed on the basis of PMF with partially overlapping entities. The results of the experiments show that PMFTL was not effective in every situation since it ignores the divergence of

source domain and target domain. As a result, KerKT outperformed PMFTL in each of the four tasks with all three data sparsities.

4. **The number of overlapping entities.** We tested three different levels of overlapping entities in these experiments: 200, 100 and 50. The number of users/items in the target domain is 2000, so the overlapping entities represented just a small proportion of the total number. However, even this small proportion was still applicable to transferring knowledge from the source to the target domain. We did not observe a very obvious increase/decrease in the precision of KerKT as the number of overlapping entities increased.

Table 5.3 Overall comparison results between KerKT and six baselines on the Movielens data.

	Method	1.00%			1.50%			2.00%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	0.7877	0.7822	0.7825	0.7227	0.7201	0.7176	0.6923	0.6929	0.6928
	FMM	0.6708 (±0.0013)	0.6727 (±0.0015)	0.6704 (±0.0019)	0.6527 (±0.0013)	0.6532 (±0.0009)	0.6496 (±0.0009)	0.6458 (±0.0009)	0.6471 (±0.0007)	0.6498 (±0.0010)
	PMF	0.7007 (±0.0006)	0.7003 (±0.0008)	0.7026 (±0.0012)	0.6797 (±0.0013)	0.6853 (±0.0026)	0.6777 (±0.0010)	0.6560 (±0.0007)	0.6567 (±0.0005)	0.6601 (±0.0009)
	RMGM	0.6659 (±0.0012)	0.6698 (±0.0010)	0.6668 (±0.0024)	0.6524 (±0.0010)	0.6531 (±0.0009)	0.6500 (±0.0012)	0.6466 (±0.0008)	0.6473 (±0.0006)	0.6506 (±0.0006)
	CBT	0.7489 (±0.0030)	0.7527 (±0.0017)	0.7513 (±0.0020)	0.7499 (±0.0041)	0.7509 (±0.0037)	0.7456 (±0.0030)	0.7469 (±0.0029)	0.7478 (±0.0033)	0.7491 (±0.0029)
	PMFTL	0.7222 (±0.0023)	0.7182 (±0.0017)	0.7227 (±0.0008)	0.6931 (±0.0042)	0.6951 (±0.0050)	0.6819 (±0.0047)	0.6822 (±0.0036)	0.6764 (±0.0015)	0.6719 (±0.0037)
	KerKT	**0.6566** (±0.0029)	**0.6553** (±0.0009)	**0.6563** (±0.0017)	**0.6411** (±0.0009)	**0.6423** (±0.0005)	**0.6405** (±0.0037)	**0.6411** (±0.0007)	**0.6371** (±0.0018)	**0.6403** (±0.0005)
RMSE	PCC	1.0087	1.0028	1.0009	0.9259	0.9203	0.9172	0.8826	0.8839	0.8830
	FMM	0.8575 (±0.0014)	0.8604 (±0.0017)	0.8566 (±0.0019)	0.8339 (±0.0016)	0.8339 (±0.0010)	0.8314 (±0.0012)	0.8254 (±0.0010)	0.8294 (±0.0010)	0.8280 (±0.0011)
	PMF	0.8808 (±0.0007)	0.8812 (±0.0012)	0.8821 (±0.0007)	0.8613 (±0.0014)	0.8588 (±0.0029)	0.8574 (±0.0013)	0.8312 (±0.0008)	0.8341 (±0.0007)	0.8338 (±0.0010)
	RMGM	0.8509 (±0.0014)	0.8563 (±0.0010)	0.8511 (±0.0013)	0.8327 (±0.0014)	0.8330 (±0.0012)	0.831 (±0.0011)	0.8259 (±0.0009)	0.8291 (±0.0007)	0.8284 (±0.0010)
	CBT	0.9663 (±0.0051)	0.9703 (±0.0036)	0.9683 (±0.0047)	0.9668 (±0.0065)	0.9650 (±0.0054)	0.9611 (±0.0052)	0.9613 (±0.0050)	0.9624 (±0.0050)	0.9641 (±0.0050)
	PMFTL	0.9083 (±0.0023)	0.9050 (±0.0025)	0.9041 (±0.0031)	0.8715 (±0.0045)	0.8712 (±0.0053)	0.8574 (±0.0040)	0.8592 (±0.0041)	0.8566 (±0.0020)	0.8482 (±0.0037)
	KerKT	**0.8355** (±0.0025)	**0.8352** (±0.0007)	**0.8346** (±0.0013)	**0.8180** (±0.0016)	**0.8169** (±0.0008)	**0.8181** (±0.0064)	**0.8158** (±0.0008)	**0.8156** (±0.0038)	**0.8143** (±0.0018)

Table 5.4 Overall comparison results between KerKT and six baselines on the Netflix data.

	Method	1.00%			1.50%			2.00%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	1.0191	1.0108	1.0026	0.9099	0.9079	0.9154	0.8139	0.8188	0.8224
	FMM	0.7460 (±0.0015)	0.7452 (±0.0015)	0.7431 (±0.0016)	0.7303 (±0.0012)	0.7337 (±0.0016)	0.7349 (±0.0012)	0.7225 (±0.0010)	0.7251 (±0.0010)	0.7297 (±0.0011)
	PMF	0.8121 (±0.0017)	0.8119 (±0.0015)	0.8111 (±0.0012)	0.7681 (±0.0006)	0.7729 (±0.0005)	0.7750 (±0.0004)	0.7474 (±0.0016)	0.7514 (±0.0013)	0.7567 (±0.0011)
	RMGM	0.7432 (±0.0012)	0.7427 (±0.0010)	0.7396 (±0.0014)	0.7300 (±0.0010)	0.7338 (±0.0008)	0.7360 (±0.0015)	0.7237 (±0.0009)	0.7265 (±0.0010)	0.7319 (±0.0010)
	CBT	0.8596 (±0.0073)	0.8558 (±0.0079)	0.8600 (±0.0050)	0.8546 (±0.0051)	0.8505 (±0.0063)	0.8582 (±0.0080)	0.8497 (±0.0068)	0.8541 (±0.0077)	0.8579 (±0.0061)
	PMFTL	0.8285 (±0.0024)	0.8231 (±0.0054)	0.8243 (±0.0055)	0.7827 (±0.0037)	0.7814 (±0.0019)	0.7843 (±0.0045)	0.7649 (±0.0035)	0.7669 (±0.0042)	0.7655 (±0.0056)
	KerKT	**0.7364** (±0.0004)	**0.7375** (±0.0037)	**0.7293** (±0.0014)	**0.7183** (±0.0012)	**0.7253** (±0.0013)	**0.7250** (±0.0006)	**0.7137** (±0.0016)	**0.7172** (±0.0027)	**0.7213** (±0.0032)
RMSE	PCC	1.2968	1.2835	1.2710	1.1571	1.1567	1.1636	1.0325	1.0384	1.0447
	FMM	0.9490 (±0.0019)	0.9473 (±0.0015)	0.9417 (±0.0023)	0.9278 (±0.0010)	0.9307 (±0.0019)	0.9319 (±0.0012)	0.9172 (±0.0009)	0.9217 (±0.0011)	0.9253 (±0.0009)
	PMF	0.9978 (±0.0016)	0.9973 (±0.0013)	0.9949 (±0.0013)	0.9499 (±0.0007)	0.9541 (±0.0005)	0.9567 (±0.0005)	0.9360 (±0.0018)	0.9431 (±0.0018)	0.9461 (±0.0014)
	RMGM	0.9437 (±0.0013)	0.9432 (±0.0012)	0.9365 (±0.0014)	0.9267 (±0.0010)	0.9301 (±0.0009)	0.9322 (±0.0015)	0.9175 (±0.0008)	0.9228 (±0.0010)	0.9268 (±0.0008)
	CBT	1.0448 (±0.0027)	1.0392 (±0.0030)	1.0417 (±0.0021)	1.0363 (±0.0026)	1.0318 (±0.0030)	1.0399 (±0.0037)	1.0330 (±0.0029)	1.0367 (±0.0036)	1.0403 (±0.0030)
	PMFTL	1.0078 (±0.0021)	1.0031 (±0.0054)	1.0061 (±0.0052)	0.9675 (±0.0038)	0.9665 (±0.0016)	0.9684 (±0.0030)	0.9467 (±0.0027)	0.9509 (±0.0033)	0.9491 (±0.0042)
	KerKT	**0.9349** (±0.0004)	**0.9351** (±0.0032)	**0.9236** (±0.0012)	**0.9125** (±0.0007)	**0.9199** (±0.0016)	**0.9219** (±0.0008)	**0.9057** (±0.0008)	**0.9116** (±0.0024)	**0.9132** (±0.0028)

Table 5.5 Overall comparison results between KerKT and six baselines on the AmazonBook data.

	Method	0.75%			0.63%			0.50%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	0.7705	0.7551	1.0026	0.7704	0.7618	0.7615	0.7882	0.7718	0.9154
	FMM	0.6648 (±0.0029)	0.6630 (±0.0026)	0.7431 (±0.0019)	0.6846 (±0.0028)	0.6806 (±0.0031)	0.6822 (±0.0032)	0.7171 (±0.0034)	0.7132 (±0.0041)	0.7349 (±0.0012)
	PMF	0.6740 (±0.0011)	0.6751 (±0.0004)	0.8111 (±0.0012)	0.6934 (±0.0003)	0.6846 (±0.0010)	0.6843 (±0.0006)	0.6855 (±0.0001)	0.6962 (±0.0001)	0.7750 (±0.0004)
	RMGM	0.6581 (±0.0016)	0.6578 (±0.0021)	0.6521 (±0.0024)	0.6706 (±0.0024)	0.6671 (±0.0024)	0.6657 (±0.0026)	0.6855 (±0.0021)	0.6831 (±0.0025)	0.7316 (±0.0015)
	CBT	0.6677 (±0.0019)	0.6678 (±0.0026)	0.6717 (±0.0020)	0.6734 (±0.0022)	0.6712 (±0.0014)	0.6714 (±0.0021)	0.6753 (±0.0015)	0.6719 (±0.0030)	0.6731 (±0.0013)
	PMFTL	0.6759 (±0.0006)	0.6796 (±0.0016)	0.6815 (±0.0008)	0.6855 (±0.0003)	0.6863 (±0.0006)	0.6844 (±0.0011)	0.6886 (±0.0006)	0.6882 (±0.0004)	0.6902 (±0.0004)
	KerKT	**0.6562** (±0.0034)	**0.6511** (±0.0038)	**0.6420** (±0.0039)	**0.6529** (±0.0030)	**0.6580** (±0.0070)	**0.6502** (±0.0018)	**0.6601** (±0.0039)	**0.6611** (±0.0061)	**0.6611** (±0.0066)
RMSE	PCC	0.9203	0.9129	0.9949	0.9277	0.9337	0.9295	0.9462	0.9476	0.9567
	FMM	0.8732 (±0.0049)	0.8668 (±0.0015)	0.9417 (±0.0023)	0.8951 (±0.0034)	0.8916 (±0.0040)	0.8956 (±0.0036)	0.9402 (±0.0049)	0.9361 (±0.0051)	0.9319 (±0.0012)
	PMF	0.9203 (±0.0021)	0.9129 (±0.0012)	0.9949 (±0.0013)	0.9277 (±0.0006)	0.9337 (±0.0017)	0.9295 (±0.0009)	0.9462 (±0.0001)	0.9476 (±0.0003)	0.9567 (±0.0005)
	RMGM	0.8636 (±0.0023)	0.8576 (±0.0031)	0.8532 (±0.0032)	0.8748 (±0.0028)	0.8717 (±0.0027)	0.8734 (±0.0031)	0.8976 (±0.0028)	0.8971 (±0.0034)	0.9322 (±0.0015)
	CBT	0.9159 (±0.0031)	0.9084 (±0.0041)	0.9110 (±0.0034)	0.9143 (±0.0031)	0.9156 (±0.0024)	0.9124 (±0.0030)	0.9194 (±0.0032)	0.9716 (±0.0042)	0.9165 (±0.0020)
	PMFTL	0.9057 (±0.0016)	0.9029 (±0.0016)	0.9163 (±0.0012)	0.9292 (±0.0005)	0.9399 (±0.0012)	0.9130 (±0.0019)	0.9382 (±0.0013)	0.9442 (±0.0006)	0.9449 (±0.0003)
	KerKT	**0.8597** (±0.0049)	**0.8430** (±0.0091)	**0.8362** (±0.0054)	**0.8483** (±0.0054)	**0.8602** (±0.0137)	**0.8485** (±0.0039)	**0.8572** (±0.0067)	**0.8641** (±0.0101)	**0.8572** (±0.0104)

Table 5.6 Overall comparison results between KerKT and six baselines on the Douban data.

	Method	1.00%			1.50%			2.00%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	0.6695	0.6700	0.6700	0.6317	0.6306	0.6298	0.6076	0.6079	0.6154
	FMM	0.5950 (±0.0011)	0.6013 (±0.0015)	0.5936 (±0.0009)	0.5834 (±0.0008)	0.5857 (±0.0011)	0.5834 (±0.0006)	0.5775 (±0.0008)	0.5828 (±0.0010)	0.5787 (±0.0008)
	PMF	0.6256 (±0.0008)	0.6288 (±0.0008)	0.6248 (±0.0011)	0.5962 (±0.0008)	0.6033 (±0.0011)	0.5993 (±0.0011)	0.5848 (±0.0004)	0.5864 (±0.0005)	0.5867 (±0.0003)
	RMGM	0.5929 (±0.0008)	0.6004 (±0.0013)	0.5912 (±0.0011)	0.5835 (±0.0009)	0.5860 (±0.0009)	0.5832 (±0.0005)	0.5773 (±0.0010)	0.5835 (±0.0011)	0.5795 (±0.0006)
	CBT	0.6422 (±0.0138)	0.6627 (±0.0126)	0.6453 (±0.0145)	0.6410 (±0.0149)	0.6524 (±0.0114)	0.6324 (±0.0135)	0.6346 (±0.0096)	0.6418 (±0.0109)	0.6373 (±0.0115)
	PMFTL	0.6308 (±0.0034)	0.6353 (±0.0009)	0.6256 (±0.0011)	0.6101 (±0.0009)	0.6139 (±0.0033)	0.6077 (±0.0032)	0.6009 (±0.0019)	0.5984 (±0.0058)	0.5949 (±0.0015)
	KerKT	**0.5863** (±0.0013)	**0.5922** (±0.0020)	**0.5835** (±0.0012)	**0.5778** (±0.0013)	**0.5812** (±0.0020)	**0.5788** (±0.0019)	**0.5721** (±0.0005)	**0.5752** (±0.0016)	**0.5757** (±0.0011)
RMSE	PCC	0.8618	0.8686	0.8655	0.8052	0.8082	0.8021	0.7672	0.7724	0.7744
	FMM	0.7508 (±0.0014)	0.7585 (±0.0020)	0.7504 (±0.0011)	0.7358 (±0.0007)	0.7373 (±0.0012)	0.7337 (±0.0006)	0.7267 (±0.0007)	0.7332 (±0.0012)	0.7278 (±0.0008)
	PMF	0.7876 (±0.0011)	0.7980 (±0.0015)	0.7857 (±0.0015)	0.7500 (±0.0009)	0.7609 (±0.0014)	0.7498 (±0.0013)	0.7327 (±0.0005)	0.7394 (±0.0009)	0.7332 (±0.0003)
	RMGM	0.7461 (±0.0009)	0.7557 (±0.0016)	0.7452 (±0.0010)	0.7344 (±0.0006)	0.7371 (±0.0012)	0.7316 (±0.0007)	0.7261 (±0.0008)	0.7341 (±0.0015)	0.7273 (±0.0007)
	CBT	0.8292 (±0.0054)	0.8484 (±0.0050)	0.8312 (±0.0074)	0.8239 (±0.0071)	0.8378 (±0.0057)	0.8189 (±0.0083)	0.8159 (±0.0041)	0.8290 (±0.0041)	0.8158 (±0.0054)
	PMFTL	0.8059 (±0.0044)	0.8095 (±0.0017)	0.7896 (±0.0013)	0.7761 (±0.0017)	0.7761 (±0.0046)	0.7635 (±0.0046)	0.7605 (±0.0033)	0.7548 (±0.0085)	0.7454 (±0.0018)
	KerKT	**0.7369** (±0.0013)	**0.7429** (±0.0027)	**0.7352** (±0.0013)	**0.7268** (±0.0008)	**0.7272** (±0.0024)	**0.7246** (±0.0022)	**0.7185** (±0.0007)	**0.7258** (±0.0036)	**0.7248** (±0.0023)

To further study the overall effectiveness of our proposed KerKT method on four tasks, average MAEs and RSMEs are calculated and displayed in Table 5.7. The results show that in four cross-domain recommendation tasks, the proposed method outperforms all the other baselines. It again shows that our method has advantage in transferring knowledge from the source domain to the target domain.

Table 5.7 Comparison result of average MAE and RMSE on four tasks between KerKT and six baselines.

		Non-transfer			Cross-domain			
		PCC	FMM	PMF	RMGM	CBT	PMFTL	KerKT
MAE	Task 1	0.9134	0.7338	0.7773	0.7333	0.8541	0.7913	**0.7247**
	Task 2	0.7696	0.6872	0.6848	0.6704	0.6712	0.6840	**0.6566**
	Task 3	0.7330	0.6571	0.6798	0.6559	0.7495	0.6979	**0.6456**
	Task 4	0.6369	0.5868	0.6040	0.5864	0.6433	0.6131	**0.5803**
RMSE	Task 1	1.1608	0.9323	0.9630	0.9307	1.0370	0.9738	**0.9200**
	Task 2	0.9314	0.9005	0.9314	0.8771	0.9242	0.9267	**0.8554**
	Task 3	0.9374	0.8401	0.8579	0.8380	0.9654	0.8786	**0.8228**
	Task 4	0.8139	0.7394	0.7597	0.7375	0.8278	0.7757	**0.7292**

5.4.4 Parameter Analysis

There are three main parameters in KerKT: λ_u, λ_v and λ. Each is a trade-off parameter in the Equation (5.21). For simplicity, we have only presented the results for the Movielens dataset. This experiment was conducted with a sparsity ratio of 99.0% and 200 overlapping entities. *MAE* and *RMSE* were used as metrics. The results are presented in Figures 5.4 and 5.5.

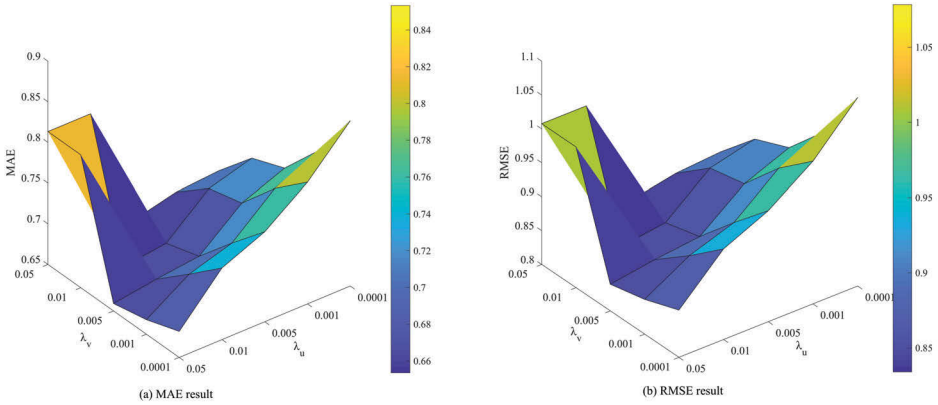


Figure 5.4 Parameter analysis of λ_u, λ_v on KerKT on Movielens dataset.

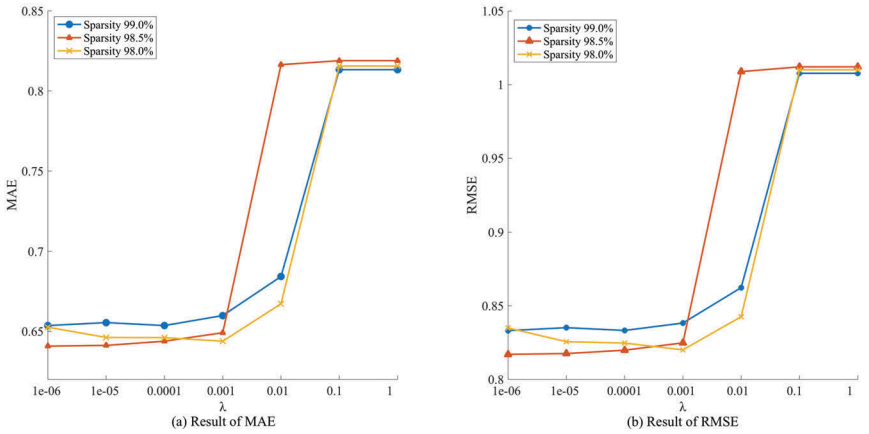


Figure 5.5 Parameter analysis of λ on KerKT on Movielens dataset.

To analyze the parameters λ_u and λ_v, we set parameter λ to 0.0001. From Figure 5.4, we can see that the *MAE* and *RMSE* change with different settings for λ_u and λ_v. These parameters reflect the influence of the user and item similarities on the matrix factorization while parameter λ restricts the complexity of the algorithm to avoid overfitting. We used a grid search to find the optimized settings for each of these parameters, λ, λ_u and λ_v, which resulted in a setting of 0.01 for all.

5.5 Summary

Today's landscape of online sales is characterized by many websites, all selling the same item, and many online shoppers with a multitude of shopping choices. Hence, in practice, it is quite common for cross-domain recommender systems to encounter overlapping entities. This chapter presents a novel cross-domain recommendation method for knowledge transfer, called KerKT, that uses overlapping entities as a bridge between the source and target domains. The method is applicable to e-commerce websites, such as Amazon, where book rating data are very dense but data in other categories are sparse. Unlike previous research, KerKT does not require that the entities be fully overlapped; it performs well in scenarios with partially overlapping entities. One advantage of this method is that it aligns the latent features of the entities extracted from the original ratings matrix. This fixes shifts in the entity feature space caused by user preference deviations between the domains. Further, the entity similarity matrix is completed through diffusion kernel completion to tackle the inconsistency caused by heterogeneous feature spaces between two domains. The similarity matrix is extended into matrix factorization with more flexible constraints to integrate the overlapping entity information. The experimental results of a comparison with six non-transfer learning and cross-domain recommendation methods show that KerKT achieved the best performance. Even with a small ratio of overlapping entities, it was still possible to transfer knowledge from the source domain to the target domain.

Chapter 6

Conclusion and Future Research

This chapter concludes the thesis and provides further research directions for this topic.

6.1 Conclusions

Recommender systems have achieved great success in the past, but the complex and dynamic data characteristics brought about by big data (Wu et al., 2014) are not well handled by recommender systems. This research focuses on solving the following three questions concerning recommender systems: 1) dynamic and uncertain user preferences; 2) inconsistent knowledge transfer across domains; and 3) the relationship between overlapping entities in recommender systems between two domains. These are still challenging problems and an investigation to provide new features to recommender systems will improve recommendation accuracy. Therefore, this research conducts a comprehensive analysis of each of the aforementioned aspects and develops a set of recommendation methods.

The main contributions of this research are as follows:

1. It develops a fuzzy user-preference drift detection-based recommendation method (to achieve Objective 1) which considers the dynamic user preferences as discussed in Chapter 3.

   A fuzzy user-preference consistency model which has the ability to deal with uncertain item features and vague user behaviors is developed. Then, a user-preference detection method is developed to describe whether the user preference is consistent in a time interval or over the entire user history. The drift detection mechanism will recognize up-to-date user preferences and will separate it from out-of-date user preferences. A recommendation method which can adapt to user-preference drift is proposed, so that user-preference drift can be sensed and the accuracy degradation caused by user-preference drift can be alleviated. By detecting and pruning out-of-date user preferences, the performance of recommender systems is improved.

2. It defines "consistent knowledge" to answer the important question of "what to transfer" in CDRSs (to achieve Objective 2) and develops an adaptive knowledge transfer method for cross-domain recommender systems with consistent information transfer (to achieve Objective 3) in Chapter 4.

   Information should be consistent for each user and item group so that group-level knowledge can be shared. In this way, the requirement as to when group-level knowledge can be transferred is addressed, which has not been considered by previous CDRSs. A formal definition of "consistent knowledge" is proposed to address the issue. A domain adaptation method that matches and adjusts user and item latent groups to maintain the consistency of group information is developed. The group-level knowledge learned on this basis represents the shared characteristics of both domains, which can help to

ensure positive transfer between the domains. An adaptive knowledge transfer method for CDRSs, called CIT is developed. This method lessens the reduction in accuracy caused by insufficient data in the target domain. It improves the performance of immature recommender systems by transferring knowledge from another related but different domain.

3. It develops a kernel-induced knowledge transfer method for cross-domain recommender systems (to achieve Objective 4) to deal with partially overlapping entity scenarios, which is the most common scenario in practice in Chapter 5.

   A domain adaptation method that aligns the feature spaces of overlapping entities is developed to match one entity's features obtained from different domains. The overlapping entities are projected to the same subspace ensuring the consistency of entity representations. At the same time, a kernel-induced completion method is used to compute entity similarities in heterogeneous situations. These two methods eliminate the divergence of features caused by data sparsity and domain heterogeneity. A new matrix factorization method is proposed with constraints that integrate intra-domain and inter-domain entity correlations obtained through overlapping entities. The two rating matrixes are collectively factorized sharing inter-domain knowledge while keeping their own domain-specific characteristics. These flexible constraints on entity correlations, relax the strict constraints on entity features compared with previous methods and ensures more useful knowledge is transferred to the target domain. An adaptive knowledge transfer method KerKT is proposed for CDRSs to deal with partially overlapping entity scenarios, which is the most common scenario in practice. The proposed

KerKT alleviates the impact on recommendation caused by data sparsity and transfers knowledge even when overlapping entities are only a small part.

6.2 Future Study

There are some limitations of this study. This study can be further advanced if these limitations are covered:

- The fuzzy user-preference drift detection method proposed in Chapter 3 needs to be applied to users who have a number of ratings in a relatively long time period. This method cannot be used with sparse data or to users who only give ratings in a short period of time
- The CIT and KerKT methods, as cross-domain recommendation methods, are applied to two domains. These two domains must be similar in some way. But how this similarity is defined or measured is not quite clear. This is a common problem happened in almost all kinds of cross-domains recommendation methods.
- The CIT method has superior advantages on divergent data between the source domain and the target domain. But its advantage may not be that obvious if the two rating matrixes from the source domain and the target domain are quite similar. In this situation, the result of CIT is similar to RMGM but slower.

This thesis identifies the following directions as future work:

- In this research, the input of the proposed method is explicit ratings. This is based on the assumption that unrated items have not been discovered by

the user yet. However, in the era of big data, there are other user-contributed data such as implicit feedback or tags assigned by users. By taking these into account, user preference can be modeled more accurately. How the explicit data and implicit data can be integrated into one system and how to extract common knowledge from the two domains with different data forms are still open and challenging issues.

- Cold-start problems are frequent in real-world applications, giving CDRSs great practical significance. However, there are still some other issues to be solved such as: the types of situations that benefit from transfer learning; the sparsity levels of the data required for the target and source domains; and how to choose the most optimal source domain to assist transfer learning. If these questions are answered, CDRS can be better applied to markets and industry.

- New customers in our experimental scenarios each have five ratings, and future work will explore 'pure' cold-start problems where new users have no ratings at all. There may a need to integrate more information, except explicit rating information such as social media or web history.

- In this research, CDRS based on bi-matrix factorization and tri-matrix factorization are exploited and the inconsistencies in these models are well handled. To address multi-dimensional data, tensor factorization is promising in developing CDRS. Information inconsistency also exists in CDRS based on tensor factorization. A possible future improvement is to develop adaptive knowledge transfer recommender systems. This will contribute to the theoretical development of matrix factorization.

- There is practical significance in studying and developing cross-domain recommender systems. Smart BizSeeker, a B2B recommender system, aims to recommend appropriate business partners to businesses in Australia (Wu et al., 2015). For future study, we are implementing our proposed method into the system.

- In this research, three methods are developed corresponding to three different situations, namely time-window, non-overlapping entity and partially overlapping entity. Our future work will focus on developing a combined framework that contains more scenarios.

Bibliography

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59.

Abel, F., Herder, E., Houben, G.-J., Henze, N., and Krause, D. (2013). Cross-system user modeling and personalization on the social web. *User Modeling and User-Adapted Interaction*, pages 1–41.

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

Adomavicius, G. and Tuzhilin, A. (2015). Context-aware recommender systems. In *Recommender systems handbook*, pages 191–226. Springer.

Al-Hassan, M., Lu, H., and Lu, J. (2015). A semantic enhanced hybrid recommendation approach: A case study of e-government tourism service recommendation system. *Decision Support Systems*, 72:97–109.

Argyriou, A., Evgeniou, T., and Pontil, M. (2007). Multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 41–48.

Arnold, A., Nallapati, R., and Cohen, W. W. (2007). A comparative study of methods for transductive transfer learning. In *The 7th IEEE International Conference on Data Mining Workshops*, pages 77–82. IEEE.

Bahadori, M. T., Liu, Y., and Zhang, D. (2011). Learning with minimum supervision: A general framework for transductive transfer learning. In *IEEE 11th International Conference on Data Mining*, pages 61–70. IEEE.

Balabanović, M. and Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72.

Baltrunas, L. and Amatriain, X. (2009). Towards time-dependant recommendation based on implicit feedback. In *Workshop on Context-aware Recommender Systems*.

Baltrunas, L. and Ricci, F. (2014). Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Modeling and User-Adapted Interaction*, 24(1-2):7–34.

Bennett, J., Lanning, S., et al. (2007). The netflix prize. In *Proceedings of KDD Cup and Workshop*, volume 2007, page 35. New York, NY, USA.

Billsus, D. and Pazzani, M. J. (1998). Learning collaborative information filters. In *The 15th International Conference on Machine Learning*, volume 98, pages 46–54.

Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, 46:109–132.

Bonilla, E. V., Chai, K. M., and Williams, C. (2008). Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems*, pages 153–160.

Borgwardt, K. M., Gretton, A., Rasch, M. J., Kriegel, H.-P., Schölkopf, B., and Smola, A. J. (2006). Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57.

Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117.

Bruzzone, L. and Marconcini, M. (2010). Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):770–787.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-adapted Interaction*, 12(4):331–370.

Cai, Y., Leung, H.-f., Li, Q., Min, H., Tang, J., and Li, J. (2014). Typicality-based collaborative filtering recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 26(3):766–779.

Campos, P. G., Díez, F., and Cantador, I. (2014). Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1-2):67–119.

Cantador, I., Fernández-Tobías, I., Berkovsky, S., and Cremonesi, P. (2015). Cross-domain recommender systems. In *Recommender Systems Handbook*, pages 919–959. Springer.

Cao, H., Chen, E., Yang, J., and Xiong, H. (2009). Enhancing recommender systems under volatile userinterest drifts. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 1257–1266. ACM.

Carrer-Neto, W., Hernández-Alcaraz, M. L., Valencia-García, R., and García-Sánchez, F. (2012). Social knowledge-based recommender system. application to the movies domain. *Expert Systems with Applications*, 39(12):10990–11000.

Chang, W.-C., Wu, Y., Liu, H., and Yang, Y. (2017). Cross-domain kernel induction for transfer learning. In *AAAI*, pages 1763–1769.

Chen, L. and Pu, P. (2012). Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction*, 22(1-2):125–150.

Chen, W., Hsu, W., and Lee, M. L. (2013). Making recommendations from multiple domains. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 892–900. ACM.

Chu, W.-S., De la Torre, F., and Cohn, J. F. (2013). Selective transfer machine for personalized facial action unit detection. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3515–3522. IEEE.

Chua, F. C. T., Oentaryo, R. J., and Lim, E.-P. (2013). Modeling temporal adoptions using dynamic matrix factorization. In *2013 IEEE 13th International Conference on Data Mining*, pages 91–100. IEEE.

Cosley, D., Lam, S. K., Albert, I., Konstan, J. A., and Riedl, J. (2003). Is seeing believing?: how recommender system interfaces affect users' opinions. In

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 585–592. ACM.

Cremonesi, P. and Quadrana, M. (2014). Cross-domain recommendations without overlapping data: myth or reality? In *Proceedings of the 8th ACM Conference on Recommender Systems*, pages 297–300. ACM.

Dai, W., Yang, Q., Xue, G.-R., and Yu, Y. (2008). Self-taught clustering. In *Proceedings of the 25th International Conference on Machine Learning*, pages 200–207.

Deng, Z., Choi, K.-S., Jiang, Y., and Wang, S. (2014). Generalized hidden-mapping ridge regression, knowledge-leveraged inductive transfer learning for neural networks, fuzzy systems and kernel methods. *IEEE Transactions on Cybernetics*, 44(12):2585–2599.

Deng, Z., Jiang, Y., Ishibuchi, H., Choi, K.-S., and Wang, S. (2016). Enhanced knowledge-leverage-based tsk fuzzy system modeling for inductive transfer learning. *ACM Transactions on Intelligent Systems and Technology*, 8(1):11.

Deshpande, M. and Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177.

Diao, Q., Qiu, M., Wu, C.-Y., Smola, A. J., Jiang, J., and Wang, C. (2014). Jointly modeling aspects, ratings and sentiments for movie recommendation. In

Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 193–202. ACM.

Ding, C., Li, T., Peng, W., and Park, H. (2006). Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 126–135. ACM.

Ding, Y. and Li, X. (2005). Time weight collaborative filtering. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 485–492. ACM.

Du, N., Wang, Y., He, N., Sun, J., and Song, L. (2015). Time-sensitive recommendation from recurrent user activities. In *Advances in Neural Information Processing Systems*, pages 3492–3500.

Felfernig, A. and Burke, R. (2008). Constraint-based recommender systems: technologies and research issues. In *Proceedings of the 10th International Conference on Electronic Commerce*, page 3. ACM.

Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M. (2011). Developing constraint-based recommenders. In *Recommender systems handbook*, pages 187–215. Springer.

Foote, J. T. (1997). Content-based retrieval of music and audio. In *Multimedia Storage and Archiving Systems II*, volume 3229, pages 138–148. International Society for Optics and Photonics.

Fouss, F., Pirotte, A., Renders, J.-M., and Saerens, M. (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369.

Gao, J., Fan, W., Jiang, J., and Han, J. (2008). Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 283–291.

Gao, S., Luo, H., Chen, D., Li, S., Gallinari, P., and Guo, J. (2013). Cross-domain recommendation via cluster-level latent factor model. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 161–176.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70.

Gong, B., Grauman, K., and Sha, F. (2014). Learning kernels for unsupervised domain adaptation with applications to visual object recognition. *International Journal of Computer Vision*, 109(1-2):3–27.

Gopalan, R., Li, R., and Chellappa, R. (2011). Domain adaptation for object recognition: An unsupervised approach. In *IEEE International Conference on Computer Vision*, pages 999–1006. IEEE.

Gori, M., Pucci, A., Roma, V., and Siena, I. (2007). Itemrank: A random-walk based scoring algorithm for recommender engines. In *IJCAI*, volume 7, pages 2766–2771.

Gupta, M. M. and Qi, J. (1991). Theory of t-norms and fuzzy inference methods. *Fuzzy Sets and Systems*, 40(3):431–450.

Hao, P., Zhang, G., Martinez, L., and Lu, J. (2017). Regularizing knowledge transfer in recommendation with tag-inferred correlation. *IEEE Transactions on Cybernetics*.

Harries, M. and Horn, K. (1995). Detecting concept drift in financial time series prediction using symbolic machine learning. In *The 8th Australian Joint Conference on Artificial Intelligence*, pages 91–98. Citeseer.

He, R. and McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the*

25th International Conference on World Wide Web, pages 507–517. International World Wide Web Conferences Steering Committee.

He, X., Gao, M., Kan, M.-Y., and Wang, D. (2017). Birank: Towards ranking on bipartite graphs. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):57–71.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53.

Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):89–115.

Hong, W., Li, L., and Li, T. (2012). Product recommendation with temporal dynamics. *Expert Systems with Applications*, 39(16):12398–12406.

Hu, L., Cao, J., Xu, G., Cao, L., Gu, Z., and Zhu, C. (2013a). Personalized recommendation via cross-domain triadic factorization. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 595–606. ACM.

Hu, Y., Zhang, D., Ye, J., Li, X., and He, X. (2013b). Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2117–2130.

Huang, J., Smola, A. J., Gretton, A., Borgwardt, K. M., and Scholkopf, B. (2006). Correcting sample selection bias by unlabeled data. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, pages 601–608.

Huang, X., Rao, Y., Xie, H., Wong, T.-L., and Wang, F. L. (2017). Cross-domain sentiment classification via topic-related tradaboost. In *AAAI*, pages 4939–4940.

Jamali, M. and Ester, M. (2009). Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 397–406. ACM.

Jebara, T. (2004). Multi-task feature and kernel selection for svms. In *Proceedings of the 21st International Conference on Machine Learning*, page 55. ACM.

Jiang, M., Cui, P., Chen, X., Wang, F., Zhu, W., and Yang, S. (2015). Social recommendation with cross-domain transferable knowledge. *IEEE Transactions on Knowledge and Data Engineering*, 27(11):3084–3097.

Jiang, S., Lowd, D., and Dou, D. (2016). A probabilistic approach to knowledge translation. In *AAAI*, pages 1716–1722.

Jiang, W. and Chung, F.-l. (2012). Transfer spectral clustering. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 789–803.

Jing, H., Liang, A.-C., Lin, S.-D., and Tsao, Y. (2014). A transfer probabilistic collective factorization model to handle sparse data in collaborative filtering. In *IEEE International Conference on Data Mining*, pages 250–259. IEEE.

Kang, Z., Grauman, K., and Sha, F. (2011). Learning with whom to share in multi-task feature learning. In *ICML*, pages 521–528.

Kapoor, K., Subbian, K., Srivastava, J., and Schrater, P. (2015). Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, pages 233–242. ACM.

Kemker, R. and Kanan, C. (2017). Self-taught feature learning for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(5):2693–2705.

Khan, F. H., Bashir, S., and Qamar, U. (2014). Tom: Twitter opinion mining framework using hybrid classification scheme. *Decision Support Systems*, 57:245–257.

Kifer, D., Ben-David, S., and Gehrke, J. (2004). Detecting change in data streams. In *Proceedings of the 30th International Conference on Very Large Data Bases*, pages 180–191. VLDB Endowment.

Kim, H.-N., El-Saddik, A., and Jo, G.-S. (2011). Collaborative error-reflected models for cold-start recommender systems. *Decision Support Systems*, 51(3):519–531.

Knijnenburg, B. P., Willemsen, M. C., Gantner, Z., Soncu, H., and Newell, C. (2012). Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction*, 22(4-5):441–504.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434. ACM.

Koren, Y. (2010). Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Kumaraswamy, R., Odom, P., Kersting, K., Leake, D., and Natarajan, S. (2015). Transfer learning via relational type matching. In *IEEE International Conference on Data Mining*, pages 811–816. IEEE.

Lang, K. (1995). Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pages 331–339. Elsevier.

Lathia, N., Hailes, S., and Capra, L. (2009). Temporal collaborative filtering with adaptive neighbourhoods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 796–797. ACM.

Li, B., Yang, Q., and Xue, X. (2009a). Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, volume 9, pages 2052–2057.

Li, B., Yang, Q., and Xue, X. (2009b). Transfer learning for collaborative filtering via a rating-matrix generative model. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 617–624. ACM.

Li, B., Zhu, X., Li, R., and Zhang, C. (2015). Rating knowledge sharing in cross-domain collaborative filtering. *IEEE Transactions on Cybernetics*, 45(5):1068–1082.

Li, C.-Y. and Lin, S.-D. (2014). Matching users and items across domains to improve the recommendation quality. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 801–810. ACM.

Li, X., Wang, M., and Liang, T.-P. (2014). A multi-theoretical kernel-based approach to social network-based recommendation. *Decision Support Systems*, 65:95–104.

Li, Y.-M., Wu, C.-T., and Lai, C.-Y. (2013). A social recommender mechanism for e-commerce: Combining similarity, trust, and relationship. *Decision Support Systems*, 55(3):740–752.

Liang, H., Xu, Y., Li, Y., and Nayak, R. (2010). Personalized recommender system based on item taxonomy and folksonomy. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 1641–1644. ACM.

Linden, G., Smith, B., and York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80.

Liu, B., Xiong, H., Papadimitriou, S., Fu, Y., and Yao, Z. (2015). A general geographical probabilistic factor model for point of interest recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 27(5):1167–1179.

Liu, F., Zhang, G., Lu, H., and Lu, J. (2017). Heterogeneous unsupervised cross-domain transfer learning. *arXiv preprint arXiv:1701.02511*.

Liu, H., Hu, Z., Mian, A., Tian, H., and Zhu, X. (2014). A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, 56:156–166.

Liu, J., Jiang, Y., Li, Z., Zhang, X., and Lu, H. (2016). Domain-sensitive recommendation with user-item subgroup analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(4):939–950.

Liu, J., Wu, C., and Liu, W. (2013). Bayesian probabilistic matrix factorization with social relations and item contents for recommendation. *Decision Support Systems*, 55(3):838–850.

Liu, Q., Chen, E., Xiong, H., Ding, C. H., and Chen, J. (2012). Enhancing collaborative filtering by user interest expansion via personalized ranking. *IEEE Transactions on Systems, Man, and Cybernetics: Cybernetics*, 42(1):218–233.

Long, M., Wang, J., Ding, G., Pan, S. J., and Philip, S. Y. (2014a). Adaptation regularization: A general framework for transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(5):1076–1089.

Long, M., Wang, J., Ding, G., Shen, D., and Yang, Q. (2014b). Transfer learning with graph co-regularization. *IEEE Transactions on Knowledge and Data Engineering*, 26(7):1805–1818.

Lops, P., De Gemmis, M., and Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., and Zhang, G. (2015a). Transfer learning using computational intelligence: a survey. *Knowledge-Based Systems*, 80:14–23.

Lu, J., Wu, D., Mao, M., Wang, W., and Zhang, G. (2015b). Recommender system application developments: a survey. *Decision Support Systems*, 74:12–32.

Lu, N., Zhang, G., and Lu, J. (2014). Concept drift detection via competence models. *Artificial Intelligence*, 209:11–28.

Lu, Z., Zhong, E., Zhao, L., Xiang, E. W., Pan, W., and Yang, Q. (2013). Selective transfer learning for cross domain recommendation. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 641–649. SIAM.

Luo, X., Zhou, M., Li, S., You, Z., Xia, Y., and Zhu, Q. (2016). A nonnegative latent factor model for large-scale sparse matrices in recommender systems

via alternating direction method. *IEEE Transactions on Neural Networks and Learning Systems*, 27(3):579–592.

Mao, M., Lu, J., Zhang, G., and Zhang, J. (2017). Multirelational social recommendations via multigraph ranking. *IEEE Transactions on Cybernetics*, 47(12):4049–4061.

McAuley, J. and Leskovec, J. (2013a). Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 165–172. ACM.

McAuley, J., Targett, C., Shi, Q., and Van Den Hengel, A. (2015). Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM.

McAuley, J. and Yang, A. (2016). Addressing complex and subjective product-related queries with customer reviews. In *Proceedings of the 25th International Conference on World Wide Web*, pages 625–635.

McAuley, J. J. and Leskovec, J. (2013b). From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 897–908. ACM.

McFee, B., Barrington, L., and Lanckriet, G. (2012). Learning content similarity for music recommendation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(8):2207–2218.

Mihalkova, L., Huynh, T., and Mooney, R. J. (2007). Mapping and revising markov logic networks for transfer learning. In *AAAI*, volume 7, pages 608–614.

Minkov, E., Charrow, B., Ledlie, J., Teller, S., and Jaakkola, T. (2010). Collaborative future event recommendation. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 819–828. ACM.

Mirbakhsh, N. and Ling, C. X. (2015). Improving top-n recommendation for cold-start users via cross-domain information. *ACM Transactions on Knowledge Discovery from Data*, 9(4):33.

Mnih, A. and Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, pages 1257–1264.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., and Yang, Q. (2008a). One-class collaborative filtering. In *The 8th IEEE International Conference on Data Mining*, pages 502–511. IEEE.

Pan, S. J., Kwok, J. T., and Yang, Q. (2008b). Transfer learning via dimensionality reduction. In *AAAI*, volume 8, pages 677–682.

Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. (2011). Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Pan, W. (2016). A survey of transfer learning for collaborative recommendation with auxiliary data. *Neurocomputing*, 177:447–453.

Pan, W. and Yang, Q. (2013). Transfer learning in heterogeneous collaborative filtering domains. *Artificial Intelligence*, 197:39–55.

Panniello, U., Tuzhilin, A., and Gorgoglione, M. (2014). Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction*, 24(1-2):35–65.

Panniello, U., Tuzhilin, A., Gorgoglione, M., Palmisano, C., and Pedone, A. (2009). Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, pages 265–268. ACM.

Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer.

Qian, X., Feng, H., Zhao, G., and Mei, T. (2014). Personalized recommendation combining user interest and social circle. *IEEE Transactions on Knowledge and Data Engineering*, 26(7):1763–1777.

Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, pages 759–766.

Rendle, S. (2010). Time-variant factorization models. In *Context-Aware Ranking with Factorization Models*, pages 137–153. Springer.

Rendle, S. (2012). Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, 3(3):57.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). 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.

Ricci, F., Rokach, L., and Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer.

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295. ACM.

Sarwar, B. M. (2001). *Sparsity, scalability, and distribution in recommender systems*. University of Minnesota.

Schafer, J. B., Konstan, J., and Riedl, J. (1999). Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic Commerce*, pages 158–166. ACM.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47.

Shambour, Q. and Lu, J. (2012). A trust-semantic fusion-based recommendation approach for e-business applications. *Decision Support System*, 54(1):768–780.

Shao, M., Castillo, C., Gu, Z., and Fu, Y. (2012). Low-rank transfer subspace learning. In *IEEE 12th International Conference on Data Mining*, pages 1104–1109. IEEE.

Shapira, B., Ricci, F., Kantor, P. B., and Rokach, L. (2011). Recommender systems handbook.

Shardanand, U. and Maes, P. (1995). 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.

Shi, Y., Larson, M., and Hanjalic, A. (2014). Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys*, 47(1):3.

Si, L. and Jin, R. (2003). Flexible mixture model for collaborative filtering. In *Proceedings of the 20th International Conference on Machine Learning*, pages 704–711.

Singh, A. P. and Gordon, G. J. (2008). Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 650–658. ACM.

Smyth, B. (2007). Case-based recommendation. In *The adaptive web*, pages 342–376. Springer.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:4.

Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., and Kawanabe, M. (2008). Direct importance estimation with model selection and its application to covariate

shift adaptation. In *Advances in Neural Information Processing Systems*, pages 1433–1440.

Tiroshi, A., Berkovsky, S., Kaafar, M. A., Chen, T., and Kuflik, T. (2013). Cross social networks interests predictions based on graph features. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 319–322.

Tkalcic, M., Odic, A., Kosir, A., and Tasic, J. (2013). Affective labeling in a content-based recommender system for images. *IEEE Transactions on Multimedia*, 15(2):391–400.

Tsymbal, A. (2004). The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*, 106(2).

Wang, C. and Mahadevan, S. (2008). Manifold alignment using procrustes analysis. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1120–1127.

Wang, H., Wang, N., and Yeung, D.-Y. (2015). Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1244. ACM.

Wang, X., Donaldson, R., Nell, C., Gorniak, P., Ester, M., and Bu, J. (2016). Recommending groups to users using user-group engagement and time-dependent matrix factorization. In *AAAI*, pages 1331–1337.

Wang, X., Ruan, D., and Kerre, E. E. (2009). *Mathematics of fuzziness—Basic issues*, volume 245. Springer.

Wei, K., Huang, J., and Fu, S. (2007). A survey of e-commerce recommender systems. In *International Conference on Service Systems and Service Management*, pages 1–5. IEEE.

Widmer, G. and Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101.

Wiesner, M. and Pfeifer, D. (2010). Adapting recommender systems to the requirements of personal health record systems. In *Proceedings of the 1st ACM International Health Informatics Symposium*, pages 410–414. ACM.

Wu, D., Zhang, G., and Lu, J. (2015). A fuzzy preference tree-based recommender system for personalized business-to-business e-services. *IEEE Transactions on Fuzzy Systems*, 23(1):29–43.

Wu, J., Chen, L., Feng, Y., Zheng, Z., Zhou, M. C., and Wu, Z. (2013). Predicting quality of service for selection by neighborhood-based collaborative filtering. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(2):428–439.

Wu, X., Zhu, X., Wu, G.-Q., and Ding, W. (2014). Data mining with big data. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):97–107.

Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., and Sun, J. (2010). Temporal recommendation on graphs via long-and short-term preference fusion. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 723–732. ACM.

Xiao, M. and Guo, Y. (2012). Semi-supervised kernel matching for domain adaptation. In *AAAI*.

Xu, J., Yao, Y., Tong, H., Tao, X., and Lu, J. (2017). Rapare: A generic strategy for cold-start rating prediction problem. *IEEE Transactions on Knowledge and Data Engineering*, 29(6):1296–1309.

Yamada, M., Sigal, L., and Raptis, M. (2012). No bias left behind: Covariate shift adaptation for discriminative 3d pose estimation. In *European Conference on Computer Vision*, pages 674–687. Springer.

Yang, B., Lei, Y., Liu, J., and Li, W. (2017). Social collaborative filtering by trust. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(8):1633–1647.

Yang, D., He, J., Qin, H., Xiao, Y., and Wang, W. (2015). A graph-based recommendation across heterogeneous domains. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 463–472.

Yao, Y. and Doretto, G. (2010). Boosting for transfer learning with multiple sources. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1855–1862. IEEE.

Yera, R. and Martinez, L. (2017). Fuzzy tools in recommender systems: a survey. *International Journal of Computational Intelligence Systems*, 10(1):776–803.

Yildirim, H. and Krishnamoorthy, M. S. (2008). A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, pages 131–138. ACM.

Yin, H., Cui, B., Chen, L., Hu, Z., and Zhou, X. (2015). Dynamic user modeling in social media systems. *ACM Transactions on Information Systems*, 33(3):10.

Yin, H., Sun, Y., Cui, B., Hu, Z., and Chen, L. (2013). Lcars: a location-content-aware recommender system. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 221–229. ACM.

Yoon, V. Y., Hostler, R. E., Guo, Z., and Guimaraes, T. (2013). Assessing the moderating effect of consumer product knowledge and online shopping experience on using recommendation agents for customer loyalty. *Decision Support Systems*, 55(4):883–893.

Zenebe, A., Zhou, L., and Norcio, A. F. (2010). User preferences discovery using fuzzy models. *Fuzzy Sets and Systems*, 161(23):3044–3063.

Zhang, C., Wang, K., Yu, H., Sun, J., and Lim, E.-P. (2014a). Latent factor transition for dynamic collaborative filtering. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 452–460. SIAM.

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. (2016). Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 353–362. ACM.

Zhang, G. and Lu, J. (2009). A linguistic intelligent user guide for method selection in multi-objective decision support systems. *Information Sciences*, 179(14):2299–2308.

Zhang, M., Tang, J., Zhang, X., and Xue, X. (2014b). Addressing cold start in recommender systems: A semi-supervised co-training algorithm. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 73–82. ACM.

Zhang, Q., Wu, D., Lu, J., Liu, F., and Zhang, G. (2017). A cross-domain recommender system with consistent information transfer. *Decision Support Systems*, 104:49–63.

Zhang, Z., Lin, H., Liu, K., Wu, D., Zhang, G., and Lu, J. (2013). A hybrid fuzzy-based personalized recommender system for telecom products/services. *Information Sciences*, 235:117–129.

Zhao, L., Pan, S. J., and Yang, Q. (2017). A unified framework of active transfer learning for cross-system recommendation. *Artificial Intelligence*, 245:38–55.

Zhen, Y., Li, W.-J., and Yeung, D.-Y. (2009). Tagicofi: tag informed collaborative filtering. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, pages 69–76. ACM.

Zheng, J., Liu, M.-Y., Chellappa, R., and Phillips, P. J. (2012). A grassmann manifold-based domain adaptation approach. In *The 21st International Conference on Pattern Recognition*, pages 2095–2099. IEEE.

Zhong, E., Fan, W., and Yang, Q. (2014). User behavior learning and transfer in composite social networks. *ACM Transactions on Knowledge Discovery from Data*, 8(1):6:1–6:32.

Zhu, X., Huang, Z., Yang, Y., Shen, H. T., Xu, C., and Luo, J. (2013). Self-taught dimensionality reduction on the high-dimensional small-sized data. *Pattern Recognition*, 46(1):215–229.

Zhu, X., Lafferty, J., and Rosenfeld, R. (2005). *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, language technologies institute, school of computer science.

Abbreviations

CBMF	**C**luster-**B**ased **M**atrix **F**actorization
CBT	**C**ode **B**ook **T**ransfer
CDRS	**C**ross-**D**omain **R**ecommender **S**ystem
CDTF	**C**ross-**D**omain **T**riadic **F**actorization
CF	**C**ollaborative **F**iltering
CIT	**C**onsistent **I**nformation **T**ransfer
CMF	**C**ollective **M**atrix **F**actorization
DCM	**D**istribution **C**onsistency **M**aps
FMM	**F**lexible **M**ixture **M**odel
GFK	**G**eodesic **F**low **K**ernel
interval-UP	**U**ser **P**reference at time **interval**
KerKT	**Ker**nel-induced **K**nowledge **T**ransfer
KL	**K**ullback **L**eibler
KMM	**K**ernel **M**ean **M**atching
MAE	**M**ean **A**bsolute **E**rror
MMD	**M**aximum **M**ean **D**iscrepency
MMDE	**M**aximum **M**ean **D**iscrepency **E**mbedding
PCA	**P**rinciple **C**omponent **A**nalysis

PCC	**P**earson **C**orrelation **C**oefficient
pLSA	**p**robabilistic **L**atent **S**emantic **A**nalysis
PMF	**P**robabilistic **M**atrix **F**actorization
point-UP	**U**ser **P**reference at time **point**
RBF	**R**adial **B**asis **F**unction
RKHS	**R**eproducing **K**ernel **H**ilbert **S**pace
RMGM	**R**ating **M**atrix **G**enerated **M**odel
RMSE	**R**oot **M**ean **S**quare **E**rror
ROST	**R**ating **O**ver **S**ite-**T**ime
STC	**S**elf **T**aught **C**lustering
SVD	**S**ingular **V**alue **D**ecomposition
SVM	**S**upport **V**ector **M**achine
TagiCoFi	**Tag i**nformed **Co**llaborative **Fi**ltering
TCA	**T**ransfer **C**omponent **A**nalysis
TCF	**T**ransfer **C**ollective **F**actorization
TF-IDF	**T**erm **F**requency-**I**nverse **D**ocument **F**requency
UDD	**U**ser-preference **D**rift **D**etection
VSM	**V**ector **S**pace **M**odel

Appendix A

GFK Operators

$\boldsymbol{U}_s^{(1)}$, $\boldsymbol{V}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$ and $\boldsymbol{V}_t^{(1)}$ are obtained through maps $\boldsymbol{\Psi}_s$, $\boldsymbol{\Psi}_t$, $\boldsymbol{\Phi}_s$ and $\boldsymbol{\Phi}_t$. According to Equations (4.12) - (4.15), the cores of these maps are GFK operators $\boldsymbol{\Psi}_G$ and $\boldsymbol{\Phi}_G$. We refer readers to (Gong et al., 2014) for the details. Here we briefly introduce how the user membership matrixes are unified to the same domain-invariant feature space. The matrixes of items are the same as the users. Let $\boldsymbol{P}_s$,$\boldsymbol{P}_t \in \mathbb{R}^{K\times d}$ denote the two sets of bases for the subspaces of the source user membership matrix $\boldsymbol{U}_s^{(0)}$ and the target user membership matrix $\boldsymbol{U}_t^{(0)}$, where K is the dimensionality of the matrixes, i.e., the number of user groups, and d is the dimension of the subspace. The subspaces can be obtained by **P**rinciple **C**omponent **A**nalysis (PCA) or other methods. $\boldsymbol{R}_s$ is the orthogonal component to $\boldsymbol{P}_s$. By performing generalized SVD,

$$\boldsymbol{P}_t^T \boldsymbol{P}_t = \boldsymbol{U}_1 \boldsymbol{\Gamma} \boldsymbol{V}^T, \boldsymbol{R}_s^T \boldsymbol{P}_t = -\boldsymbol{U}_2 \boldsymbol{\Sigma} \boldsymbol{V}^T \tag{A.1}$$

where $\boldsymbol{\Gamma}$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ are diagonal matrixes. The diagonal elements of $\boldsymbol{\Gamma}$ and $\boldsymbol{\Sigma}$ are $\cos\theta_i$ and $\sin\theta_i$, where $i = 1,\ 2, \ldots,\ d$. θ_i are the angles between subspaces $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$. To ensure the consistency of the user groups between both domains,

the GFK operator is used to map the original user group membership matrixes to a domain-invariant space:

$$\boldsymbol{\Psi}_G(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}) = \boldsymbol{L} \tag{A.2}$$

where $\boldsymbol{L}$ is $\boldsymbol{G}$'s square root, $\boldsymbol{L}^T\boldsymbol{L} = \boldsymbol{G}$,

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{P}_s\boldsymbol{U}_1 & \boldsymbol{R}_s\boldsymbol{U}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_1 & \boldsymbol{\Lambda}_2 \\ \boldsymbol{\Lambda}_2 & \boldsymbol{\Lambda}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_1^T\boldsymbol{P}_s^T \\ \boldsymbol{U}_2^T\boldsymbol{R}_s^T \end{bmatrix} \tag{A.3}$$

where $\boldsymbol{\Lambda}_1$, $\boldsymbol{\Lambda}_2$ and $\boldsymbol{\Lambda}_3$ are diagonal matrixes whose diagonal elements are $\lambda_1 = 1 + \frac{\sin 2\theta_i}{2\theta_i}$, $\lambda_2 = \frac{\cos 2\theta_i - 1}{2\theta_i}$, $\lambda_3 = 1 - \frac{\sin 2\theta_i}{2\theta_i}$.

According to equations (4.16) and (4.17), $\boldsymbol{U}_s^{(1)}$,$\boldsymbol{V}_s^{(1)}$,$\boldsymbol{U}_t^{(1)}$ and $\boldsymbol{V}_t^{(1)}$ are obtained.

Appendix B

Proof of Maps Ensuring Consistency

A definition for maps like f_u and f_v is given as follows.

Definition B.1 (Distribution Consistency Maps). Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, the tri-factorizations of $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ are information consistent and they share consistent knowledge $\boldsymbol{S}$ such that

$$\boldsymbol{X}_s = \boldsymbol{U}_s^{(1)} \boldsymbol{S} (\boldsymbol{V}_s^{(1)})^T \tag{B.1}$$

$$\boldsymbol{X}_t = \boldsymbol{U}_t^{(1)} \boldsymbol{S} (\boldsymbol{V}_t^{(1)})^T \tag{B.2}$$

where $\boldsymbol{U}_s^{(1)}$, $\boldsymbol{U}_t^{(1)}$ are user group membership matrixes unified to the same domain-invariant feature space for the source and target domains, while $\boldsymbol{V}_s^{(1)}$ and $\boldsymbol{V}_t^{(1)}$ are unified item group membership matrixes. If maps f_u and f_v satisfy equation (4.20), we call f_u and f_v **D**istribution **C**onsistency **M**aps (DCM) for the two rating matrixes.

For a demonstration of a DCM map, we refer readers to some theoretical results in (Liu et al., 2017) for reliable unsupervised knowledge transfer including a linear

monotonic map and its related theorem. Linear monotonic map is: $f(\mathcal{X}) = \mathcal{X}\boldsymbol{u}$, $\mathcal{X} \in \mathbb{R}^{m\times n}$, $\boldsymbol{u} \in \mathbb{R}^{n\times 1}$. A theorem for reliable unsupervised knowledge transfer is then given, proving that linear monotonic maps can ensure the process of unsupervised knowledge transfer is reliable. As in our situation, we give the theorem and proof as follows:

Theorem 1. Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s\times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t\times N_t}$, the tri-factorizations of $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ are information consistent and they share a consistent knowledge $\boldsymbol{S}$ as in Equations (B.1) and (B.2). When $\boldsymbol{u} \geqslant 0$ and $\boldsymbol{v} \geqslant 0$, $f_u(\boldsymbol{U}_t^{(1)}) = \boldsymbol{U}_t^{(1)}\boldsymbol{u}$ and $f_v(\boldsymbol{V}_t^{(1)}) = \boldsymbol{V}_t^{(1)}\boldsymbol{v}$, they are DCMs for two rating matrixes.

Proof. When $\boldsymbol{u} \geqslant 0$ and $\boldsymbol{v} \geqslant 0$, $f_u(\boldsymbol{U}_t^{(1)}) = \boldsymbol{U}_t^{(1)}\boldsymbol{u}$ and $f_v(\boldsymbol{V}_t^{(1)}) = \boldsymbol{V}_t^{(1)}\boldsymbol{v}$ can satisfy the following equation:

$$P(\boldsymbol{S}|\boldsymbol{U}_t^{(1)}\boldsymbol{u}, \boldsymbol{V}_t^{(1)}) = P(\boldsymbol{S}|\boldsymbol{U}_t^{(1)}\boldsymbol{I}, \boldsymbol{V}_t^{(1)})$$

Then, to fix $f_u(\boldsymbol{U}_t^{(1)})$, we use the following equation:

$$P(\boldsymbol{S}|\boldsymbol{U}_t^{(1)}\boldsymbol{u}, \boldsymbol{V}_t^{(1)}\boldsymbol{v}) = P(\boldsymbol{S}|\boldsymbol{U}_t^{(1)}\boldsymbol{u}, \boldsymbol{V}_t^{(1)}\boldsymbol{I})$$

So, we then have

$$P(\boldsymbol{S}|f_u(\boldsymbol{U}_t^{(1)}), f_v(\boldsymbol{V}_t^{(1)})) = P(\boldsymbol{S}|\boldsymbol{U}_t^{(1)}, \boldsymbol{V}_t^{(1)})$$

Based on *Definition B.1*, $f_u(\boldsymbol{U}_t^{(1)}) = \boldsymbol{U}_t^{(1)}\boldsymbol{u}$ and $f_v(\boldsymbol{V}_t^{(1)}) = \boldsymbol{V}_t^{(1)}\boldsymbol{v}$ are DCMs. □

Hence, the linear monotonic map is proven to be a DCM, which means we can let f_u and f_v have the following expressions:

$$f_u(\boldsymbol{U}_t^{(1)}) = \boldsymbol{U}_t^{(1)}\boldsymbol{u},\ \boldsymbol{u} \leqslant 0 \tag{B.3}$$

$$f_v(\boldsymbol{V}_t^{(1)}) = \boldsymbol{V}_t^{(1)}\boldsymbol{v},\ \boldsymbol{v} \leqslant 0 \tag{B.4}$$

where $\boldsymbol{u} \in \mathbb{R}^{K\times K}$ is user tuning factor and $\boldsymbol{v} \in \mathbb{R}^{L\times L}$ is item tuning factor.