NOVEL TECHNIQUES FOR CROSS-DOMAIN RECOMMENDATION

A thesis submitted during 2021 to the University of Hyderabad in partial fulfillment of the award of a Ph.D. degree in Computer Science

by

VEERAMACHANENI SOWMINI DEVI

SCHOOL OF COMPUTER & INFORMATION SCIENCES
UNIVERSITY OF HYDERABAD
(P.O.) CENTRAL UNIVERSITY
HYDERABAD - 500 046, INDIA

December 19, 2021

CERTIFICATE

This is to certify that the thesis entitled **"Novel Techniques for Cross-Domain Recommendation"** submitted by **Veeramachaneni Sowmini Devi** bearing **Reg. No. 15MCPC01** in partial fulfillment of the requirements for the award of **Doctor of Philosophy** in **Computer Science** is a bonafide work carried out by her under our supervision and guidance.

This thesis is free from plagiarism and has not been submitted previously in part or in full to this or any other university or institution for the award of any degree or diploma. The student has the following publications before submission of the thesis for adjudication and has produced evidence for the same.

1. "A Matrix Factorization & Clustering based Approach for Transfer Learning." *Pattern Recognition and Machine Intelligence* (2017): 77-83.
2. "A Maximum Margin Matrix Factorization based Transfer Learning Approach for Cross-Domain Recommendation." *Applied Soft Computing* 85 (2019).

Further, the student has passed the following courses towards fulfillment of coursework requirement for Ph.D.

Course Code	Name	Credits	Pass/Fail
CS 801	Data Structures and Algorithms	4	Pass
CS 802	Operating Systems and Programming	4	Pass
AI 852	Learning & Reasoning	4	Pass
AI 853	Data Mining	4	Pass

Prof. Arun K Pujari
Supervisor

Prof. Vineet Padmanabhan
Supervisor

Prof. Chakravarthy Bhagavati
Dean, School of Computer and Information Sciences

DECLARATION

I, **Veeramachaneni Sowmini Devi**, hereby declare that this thesis entitled **"Novel Techniques for Cross-Domain Recommendation"** submitted by me under the guidance and supervision of **Prof. Arun K Pujari** and **Prof. Vineet Padmanabhan** is a bonafide research work. I also declare that it has not been submitted previously in part or in full to this university or any other university or institution for the award of any degree or diploma.

Date:

Name: **Veeramachaneni Sowmini Devi**

Signature of the Student:

Reg. No. **15MCPC01**

This dissertation is dedicated to my family members and to the Almighty God.

ACKNOWLEDGEMENTS

There is a quote saying "Teaching is the profession that teaches all other professions". So, foremost, I would like to express my heartfelt gratitude to my supervisor Prof. Arun K Pujari because of whom I got motivated in my masters, to do my PhD. I am very thankful for his explanations, encouragement, suggestions, and giving his valuable guidance despite his busy schedule.

I sincerely thank my other supervisor, Prof. Vineet Padmanabhan, for his continuous support during my research, and his motivation, patience, enthusiasm, knowledge. His guidance helped in all the way through PhD and the writing of this thesis. I feel he is so student-friendly in all aspects and a good advisor and mentor for my PhD.

It is also a privilege to express my sincere regards to Mrs. Subha Lakhmi Pujari & Mrs. Mrinalini Vineet for their care, encouragement, and truthful blessings to accomplish success in my research and in my personal life too.

I would like to thank my DRC members, Prof. Hrushikesha Mohanty, Dr. Rukma Rekha, and Dr. Naveen for patiently listening to my work and giving their valuable suggestions.

I thank my fellow labmates, Venkat, Vikas, Sandeep, Ayang, Akshay, Tirupathi, and Prasad for engaging in critical discussions related to research and also support me morally whenever I need.

I am thankful to the staff of SCIS for giving me knowledge by teaching some of the subjects during my masters program, which have aided me in some way during my research.

I thank my friends Deepika, Murthy, Rama Krishna, Sneha, Ruchita, Spoorthy, and

Anji for their considerate friendship and endless support.

I also wish to thank all my childhood teachers, from whom I learned some fundamental knowledge.

I would also like to acknowledge the funding agency, Council of Scientific and Industrial Research (CSIR) Government of India, for the financial support in the form of CSIR-UGC NET-JRF/SRF.

Last but not least, I would like to express my gratitude to my family: my husband Yaswanth Gavini, for helping me in understanding some of the technical concepts and also encouraging me positively by having confidence in me; my parents Mr. Subbarao and Mrs. Kathyayani, without whom I would not be in this position, as they have always been there for me since I was a kid, agreeing to everything I needed personally and educationally; my brother Vivek, who always advises me to eat healthy food in order to gain enough strength; my in-laws Mr. Satyanarayana and Mrs. UdayaSri, brother-in-law Mr. Jayanth, co-sister Mrs. Swetha, and kids who have all been supportive in every way. They would even ask about the progress of my research regularly.

Above all, praise and gratitude to the Almighty God in whom I have the deepest faith, for his abundant blessings throughout my life and my research work, which has helped me to successfully finish the research.

Veeramachaneni Sowmini Devi

ABSTRACT

Searching the internet for items of interest have become a nightmare these days due to the sheer amount of available data. *Recommender systems* have become indispensable for many e-commerce applications to tackle this *information overload* problem. Recommender systems are of various types of which *collaborative-filtering* based recommender systems are the most popular in which the recommendation is carried out by utilizing the observed preferences of other users who have similar likings as that of the *target* user. Collaborative filtering techniques like *matrix factorization* have been demonstrated to be highly successful wherein given a partially filled User-Item rating matrix, the idea is to correctly predict the missing entries. Among different matrix factorization methods, maximum margin matrix factorization is shown to be effective in predicting the unobserved ratings when very small number of observed entries are given. While collaborative filtering techniques like matrix factorization are good at prediction, as sparsity increases (very less ratings), the accuracy of prediction expectedly falls. To address the data sparsity issue, transfer learning techniques have emerged in which the information learnt in one context is used in another. Cross-domain recommender systems or cross-domain collaborative filtering is one such method in which transfer learning is used to transfer the knowledge from dense source domain to the sparse target domain so as to improve the prediction accuracy of the target domain. In order to use transfer learning, the basic assumption is that the source and target domains are inherently related in some sense. Codebook Transfer (CBT) is one of the popular transfer learning methods of cross-domain collaborative filtering, in which the codebook which is the cluster-level rating pattern is learnt from source domain and is transferred to the target domain. Codebook basically captures the rating patterns in a

condensed form, and it is hypothesized that the condensed rating pattern is, in some sense, invariant across domains.

In this thesis, the main focus is on proposing novel codebook based cross-domain collaborative filtering techniques to address the data sparsity issue so as to improve the prediction accuracy of the sparse domain. The key challenge in designing new algorithms is to improve the prediction accuracy of target domain when both domains have no overlap of users and items. The main motivation is to construct the codebook from source domain by making use of techniques like maximum margin matrix factorisation (MMMF), clustering as well as co-clustering and thereafter to transfer the learnt codebook to the target domain. In one of the proposed methods, we generate the codebook by clustering the latent factors obtained by MMMF and then transfer the generated codebook to the target using hard membership and soft membership. In hard membership, a user or item belongs to a single cluster, whereas in soft membership a single user or item can belong to multiple clusters with some weights. The other method uses co-clustering to generate codebook, and later the codebook is processed. Applying maximum margin matrix factorization on the processed codebook gives the cluster-level latent factor vectors of the source data which are transferred to the target domain via hinge loss, to learn the target domain latent features to get the predicted target rating matrix. Another proposed method uses co-clustering technique to generate codebook from the source user-item rating matrix. The constructed codebook is transferred to the target domain by using the hingeloss function instead of squared loss in a novel way. We validate our methods by conducting experiments on different datasets and compare with some baseline methods. The results show that our methods achieve a better approximation of the sparse target matrix which in turn increases the prediction accuracy of the target domain.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ML	Machine Learning
CF	Collaborative Filtering
MF	Matrix Factorization
MMMF	Maximum Margin Matrix Factorization
PMF	Probabilistic Matrix Factorization
TL	Transfer Learning
CBT	CodeBook Transfer
RMSE	Root Mean Square Error
MAE	Mean Absolute Error

NOTATIONS

X	Source user-item rating matrix
Y	Target user-item rating matrix
m'	Number of users in source rating matrix
n'	Number of items in source rating matrix
m	Number of users in target rating matrix
n	Number of items in target rating matrix
U	User latent features
V	Item latent features
C	Codebook
ω	Observed ratings

CHAPTER 1

Introduction

In this chapter, we give a brief introduction on recommender systems wherein we outline different types of recommender systems and its usage as well as give a clear distinction between *single domain* and *cross-domain* recommender systems. A particular machine learning technique called *transfer learning* is discussed as it plays a vital role in cross-domain recommender systems which in turn is the focus area of this thesis. Having done that we layout the plan for this dissertation.

1.1 Recommender Systems and its usage

The exponential growth of the World Wide Web and the rapid rise in e-services (62) have provided consumers (users) with an overwhelming number of options, often contributing to more complicated decision-making. In (3), Chris quoted that "We are leaving the Information Age and entering the Recommendation Age." The implication of this is that, during earlier days, it was difficult to get sufficient data to make decisions and therefore individuals used to rely upon others suggestions which were given dependent on their past experiences. However, these days the case is different in the sense that the access to information is quite easy and collection of information is no longer a concern. The only problem is to make smart choices based on the available information. i.e., we are facing a situation where we can't locate our target data regardless of whether we have access to it because it is hidden somewhere in a huge pile of extraneous information. *Information overload* is the term used to describe such a scenario

and recommender systems are the technology that is being used to help users find their target information from the immense data that is available. Managing vast volumes of data effectively and efficiently is the biggest challenge of a recommender system which is why it is often said that the task of a recommender systems is that of "Information Filtering". An Information filtering system eliminates excess or undesirable data from an information source using automated or semi-automated or computerized methods before displaying it to the user. The primary objective of information filtering system is to control the information overload problem.

Recommender Systems are specifically designed to support individuals who lack skill or expertise in coping with the broad variety of choices presented (66; 2; 79). In e-commerce (10), recommender systems were first applied to deal with the information overload issue brought about by Web 2.0, and they were immediately extended to the personalization of e-business, e-government, e-tourism, and e-learning (40). Recommender Systems take advantage of many sources of knowledge to predict the preferences of users for items of interest (4). In both academia (what courses to take) and industry, this area of research has been the subject of great concern for the past twenty years, and research in this field is also driven by the potential benefit that recommendation systems can produce, for companies such as Amazon (74). These days, recommender frameworks are an irreplaceable component of Internet sites (1), for example, Yahoo, Netflix, YouTube, LinkedIn, Facebook, Amazon, and Last.fm. In short, recom-

Figure 1.1: Recommendations by Amazon

mendation systems are meant to estimate an item's effectiveness and to predict whether it is worth recommending to the user or not. *How does youtube predict what videos you may watch? How does amazon know what products you may like (Fig. 1.1)? How does Facebook give friend suggestions to you? How does LinkedIn (Fig. 1.2) know the people we might know?* that magic comes from the Recommender Systems.

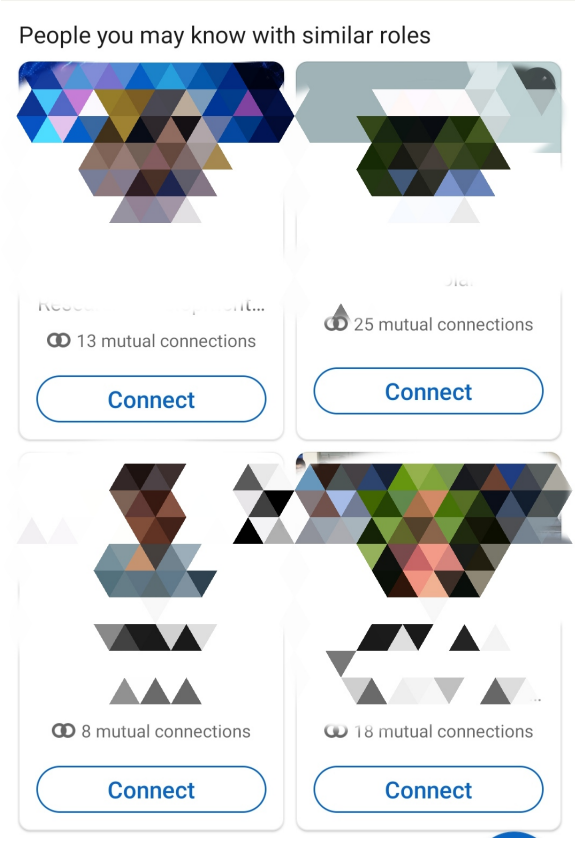


Figure 1.2: Suggestions by LinkedIn

In any Recommender Systems, there are three key elements: `user` (consumer or customer), `item` (product), and `rating` (preference) given by the user to an item. Any user can utilize the Recommender System by providing his input i.e., opinion (rating or preference) about the items he has bought or movies he has watched or books he has read,.., and thereby get the recommendations (suggestions) about the Novel items.

To produce any suggestions (recommendations), the system has to collect the data which can be of two types: *explicit* and *implicit*. For explicit data, it is often the case that a user is asked to give his/her ratings which are usually given on a particular `discrete` rating scale - say $[1-5]$, where 1 is the least rating and 5 is the highest rating. There exist `binary` preferences too, where 1 denotes that a user likes the item and 0 indicates that a user dislikes the item. In *implicit* (49; 33; 86) data collection, the user's data is

indirectly captured i.e., the user's actions are recorded on the web. The data (explicit or implicit) is advantageous in predicting the user's future preferences on the items and can guide the users (37) in a personalized way to get the interesting items.

1.2 Different Approaches in Recommender Systems

There are three different approaches (88) in recommender systems namely,

1. Content-based recommender systems (CB)
2. Collaborative Filtering (CF)
3. Hybrid recommender systems

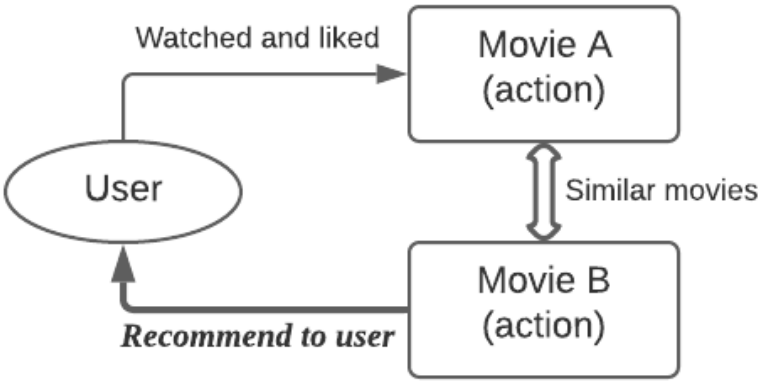


Figure 1.3: Content-based recommender systems

1. Content-based recommender systems (CB) (75; 73; 46): Recommends similar items based on a specific item the user has liked in the past. For example, to make movie recommendations, this framework uses movie (item) metadata, such as genre, description, actors, director, and so on. The notion behind these recommendation algorithms is that if a person likes a specific item, he/she likes an item that is close (similar) to it as well. It will make use of the user's past item metadata to recommend it. For instance, if we consider Figure 1.3, the user has watched Movie A which falls under the action category, and has liked the same. Then the content-based recommender system would recommend Movie B to the user, which is also an action movie as shown in the figure. Another nice example would be YouTube, where it recommends new videos that a user might possibly watch based on his/her past viewing experience.

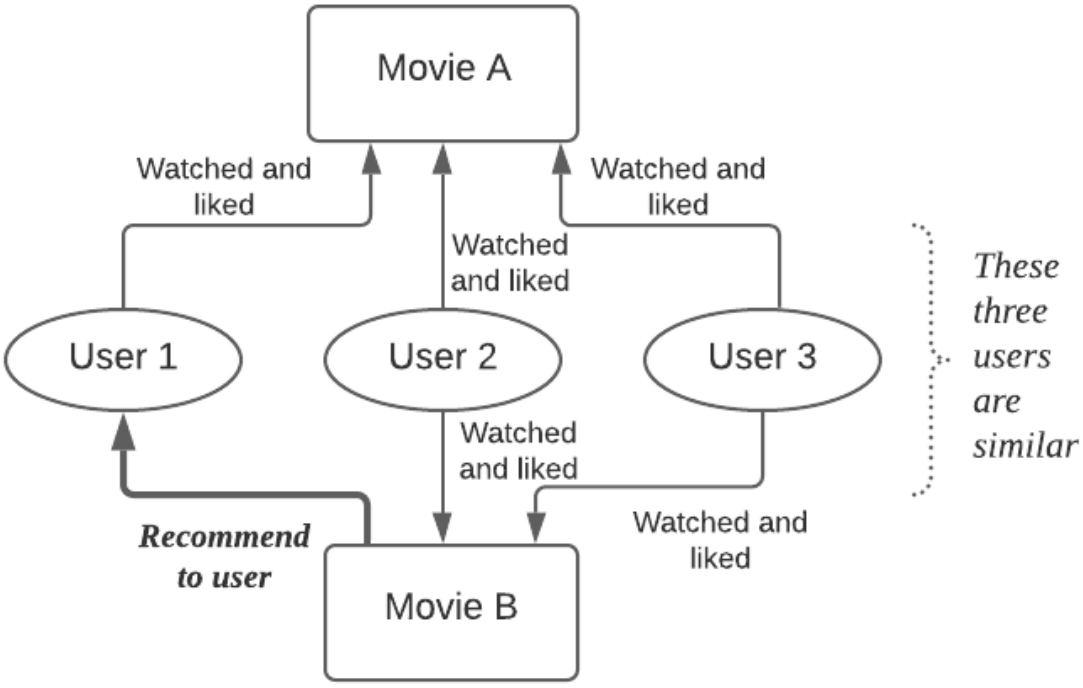


Figure 1.4: Collaborative Filtering

2. Collaborative Filtering (CF) (88; 24; 50; 83; 71): In CF, which is the most well-known technique in recommendation, the system tries to find out the users that are similar to the target user based on their ratings and recommends items the similar users have liked. It's "collaborative" because it predicts the preferences of the target user on the basis of other users' ratings. The significant assumption underlying the collaborative filtering is: users with identical preferences(tastes) in the past are probably going to have similar preferences in the future too. It is this assumption that permits to take a user's past history and extrapolate it into their future and predict items that they would enjoy (like). They will like comparative sorts of things as they preferred before. The collaborative recommendations are mainly based on user-item interactions (user-item ratings). For example, in collaborative filtering, if User 1 has preferred (liked) a Movie A which is likewise enjoyed by some other similar users, at that point Movie B will be recommended to User 1 which is loved by other similar users. If we consider Figure 1.4, User 1, User 2, and User 3 have watched Movie A and liked the same, and in addition to Movie A, User 2 and User 3 have watched Movie B too. So, in this case these three users are found to be similar, and hence Movie B will be recommended to User 1.

There are two categories (61; 84) of collaborative filtering algorithms which are *Memory based* (1; 92) and *Model based* (1; 82) algorithms. The fundamental distinction

is that memory-based algorithms use entire data to make predictions and identify top-k similar users to the target user, while model-based algorithms use the knowledge to learn/train a model that can be used to make predictions later on. This implies that the memory-based methods should necessarily have all the data in memory, whereas in model-based algorithms, once we build the model, it can make fast predictions using less amount of data than the original data.

Memory-based filtering is further divided into two categories:
User-based Filtering: These systems recommend items to a user that have been liked by similar users. Let's assume, for instance, User A and User B have a common (similar) interest in books (in the sense that they enjoy and dislike many of the same books). Now, let's say a new book was published in the market, and User A has read it and enjoyed it. Thus, it is very likely that User B would enjoy it too, and therefore, this book is recommended to User B by the system.
Item-based Filtering: These systems are highly comparable to the content based recommendation. These systems identify similar items based on the users past ratings. For instance, if User A, User B, and User C have given a rating 5 to item 1 and item 2, the system recognises these items as similar. Therefore, based on this if someone buys item 1 then item 2 is also recommended to him/her by the system.
To be concise, user-based CF is based on the idea of user similarity, and item-based CF is based on the idea of item similarity. The most popular strategy in user-based CF is to find the neighborhood of users that have similar tastes with the target user and recommend the items. In item-based CF, the reommendation of items would be made on the basis of the most similar items to the items for which the target user has given preference.

Matrix Factorization (MF) (28; 41) is one of the widely used Collaborative filtering techniques and comes under model-based filtering technique. Given a user-item rating matrix as input in which each user has rated some items, the goal is to predict how the users would rate the items they haven't rated ie., to predict the unknown ratings, such that recommendations can be made to the users. The goal of MF is to find two matrices such that the dot product of these two matrices approximates the original rating matrix. There are different types of matrix factorization methods such as regularized matrix factorization (RMF), non-negative matrix factorization (NMF), maximum margin matrix factorization (MMMF), probabilistic matrix factorization (PMF) to name a few. The

detailed discussion on MF is given in Chapter-2.

3. Hybrid recommender systems: (63; 5) The integration of both content-based and collaborative filtering methods is known as the Hybrid approach. It is possible to incorporate hybrid approaches in many ways: by separately creating and then integrating collaborative-filtering and content-based approaches; by incorporating the content-based abilities to a collaborative approach (and opposite); or by merging the approaches into one system.

The block diagram of the types of approaches is shown in Figure-1.5.

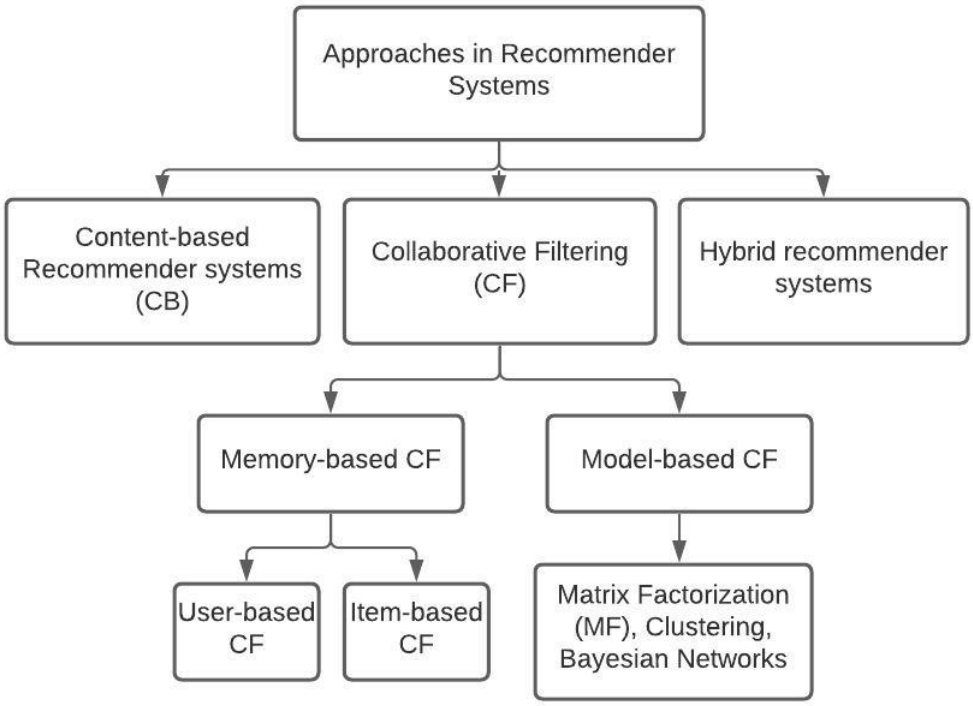


Figure 1.5: Different approaches in Recommender Systems

So far we have discussed about recommender systems (single-domain recommender systems), in which the system recommends the items of a particular domain. For instance, Netflix suggests movies, Last.fm recommends music, Douban recommends books. The major disadvantages of single-domain recommender systems are `data sparsity` issue and `cold start` problem (11).

Data-Sparsity issue: In scenarios wherein the user does not rate the items he (she) has watched/purchased/read, at that point the data (information) will be very less as such and one cannot make pertinent recommendations to the user further. *In our thesis, we*

concentrate primarily on overcoming the data-sparsity issue which is one of the major drawbacks of recommender systems.

Cold-start problem: Recommender systems are faced with the issue of cold-start in situations wherein a new user (who has not yet rated any items) wishes to utilize the recommender system. It will be an issue as the data (ratings) of that user does not exist and in such situations, one of the solutions is to take implicit ratings (browsing history,...) or perhaps ask the user to give some ratings.

1.3 Cross-Domain Recommender Systems

In order to address the limitations like *data-sparsity* of personalized single-domain recommender systems, cross-domain (Multi-disciplinary) recommender systems have come into the picture. In single-domain recommender systems, the user's past history (ratings) in the specific domain is not enough to recommend the items in that domain. In cross-domain RS (15; 39), we can use the ratings of other domains, to solve the data-sparsity issue. The key principle of Cross-Domain RS is to recommend items from one domain by using the ratings (data) from a different domain. Suppose that a user needs to be suggested a book where the ratings given by the user is very limited. We can then utilize his/her rating data from the movie domain (or some other related domain in which the active user's ratings are more) and use it for recommending books. In cross-domain RS, a technique called *Transfer Learning (TL)* (94; 99) is used in which the information (knowledge) from some auxiliary (source, dense) domain is extracted in order to use it in the sparse target domain.

1.3.1 Transfer Learning

Transfer learning (TL) is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. The main motivation behind Transfer Learning is *Domain Adaptation*, i.e., whether people can apply previously learned knowledge to solve new problems faster or with better solutions than current ones. In general, people typically have an implicit ability to transfer information (knowledge) between tasks. At the point we encounter new tasks, what we gain as knowledge while learning about one task, we use

in the same way to solve other related tasks. The more related the new task is to our previous task, the easier we can train it by cross-utilizing or transferring the previously learnt knowledge. A simple example would be, "knowledge gained while learning to recognize cars could be applied when trying to recognize trucks".

In spite of the fact that collaborative filtering based recommendation systems have become the standard nowadays, because of the data-sparsity issue (i.e., very little existing information is available), they have difficulty in making precise recommendations. In the last decade or so, cross-domain collaborative filtering strategies such as *transfer learning* (53; 54; 58; 59; 60; 96) have been proposed as a possible solution to alleviate the issue of data sparsity. In cross-domain we attempt to utilize the knowledge learned in one domain in another domain, and this can be best attained using Transfer Learning. Usually, the presumption in machine learning algorithms is that the training data and test data must be from the same domain. But this may not work in real-world situations. The primary goal of transfer learning (69; 87; 90) is to transfer the knowledge from the dense source domain to the sparse target domain. If, for example, a user has watched a lot of movies and rated them, but has very few ratings in the books domain and wants a book to be recommended, the book can be recommended utilizing his ratings from the movie domain. If we have adequate training data in one domain and want to get recommendations from another domain, then if transfer learning is used, the learning efficiency will improve to a considerable extent by reducing efforts. Transfer Learning (TL), in other words, improves learning in a new (target) task by transferring knowledge from a previously trained related (source) task.

On the other hand, common machine learning algorithms usually handle isolated tasks. Transfer learning aims to make it different by having methods to transfer the information gained or knowledge learned in one source task and use it to improve learning in a related target task. i.e., TL attempts to extract the information (knowledge) from one or more source tasks and apply the same knowledge to the target task. Traditional ML methods are such that the system must be trained on training data for any domain and then evaluated on test data of the same domain, and these algorithms vary from one domain to the next. Coming to transfer learning, one can learn an algorithm for one domain and apply it to various similar domains by using the knowledge gained from the source, such that the accuracy of the target task gets improved. For example, if we consider Figure 1.6 from recommender systems point of view, there are two different

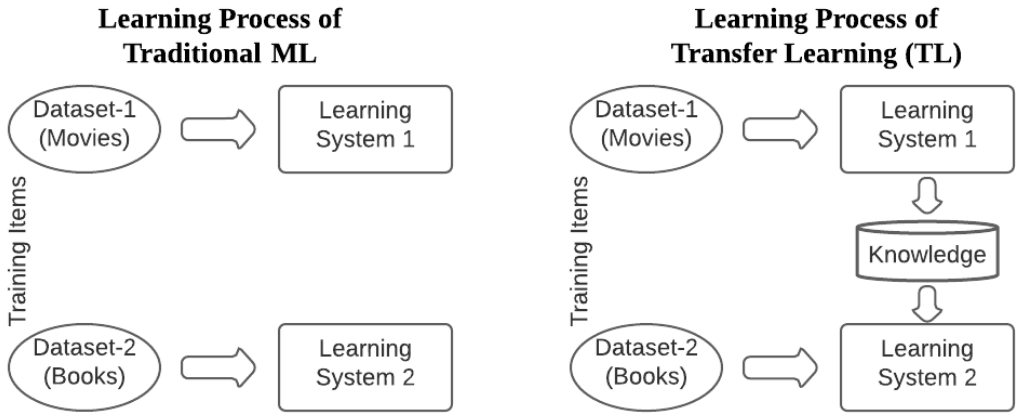


Figure 1.6: Learning process of traditional ML vs Transfer Learning

datasets namely dataset-1 (Movies) and dataset-2 (Books). Assuming that the Movie dataset is denser than that of the Book dataset, and if we consider the left side of Figure 1.6, which is the learning process of traditional ML, the learning is done in an isolated way for two datasets namely dataset-1 (Movies) and dataset-2 (Books). In this case, the recommendation performance on books data might not be good enough, as the data is sparse. Whereas in the learning process of TL (on the right side), the learning system 1 is trained on dataset-1 (Movies) which is denser, and the knowledge gained from this is extracted and utilized in the training process on dataset-2 (Books) which is sparse, so as to improve the accuracy of recommendation on books data. So, the key advantage of TL algorithms is that if there is insufficient training data in the target domain, the knowledge can be transferred from some other related domain.

In this thesis, we develop different novel techniques for cross-domain recommender systems using transfer learning approach to address the data sparsity issue and improve the prediction accuracy of the target domain.

1.4 Problem definition

Given a source user-item rating matrix X of size $m' \times n'$, where m' is the number of users (rows) in X and n' is the number of items (columns) in X, and a sparse target user-item rating matrix Y of size $m \times n$ (where the number of users is m and n the number of items) with y_{ij} known for $(i, j) \in \omega$, where ω is the set of observed ratings in target

rating matrix (Y), the goal is to find the y_{ij} for $(i, j) \notin \omega$. y_{ij} is the rating given by user i to item j in matrix Y. So, given the source data and very less target data, our goal is to predict the missing ratings of target rating matrix by utilizing the existing ratings of target matrix, and some knowledge learnt from the source rating matrix (transferring the knowledge of source). The prediction of the missing ratings should be in such a way that the error rate on the existing ratings should be minimum.

1.5 Contributions of Thesis

The major contributions of the thesis are as follows,

1. We propose a technique for cross-domain recommender systems in which we use MMMF and clustering in order to generate the knowledge (cluster-level rating pattern which is called as `codebook`) from dense source domain. We then transfer the learnt knowledge to the sparse target domain and train the same to get the membership matrices of target domain and predict the target rating matrix accurately. To the best of our knowledge, there is no research that considers MMMF while constructing codebook.
2. We extend the above method by introducing the soft membership constraint while transferring the codebook, and show that the prediction accuracy gets improved.
3. We propose a novel transfer learning method for cross-domain collaborative filtering which uses co-clustering on source domain to construct the codebook, and by processing the codebook we get the partial codebook. By applying MMMF on the partial codebook we get the cluster-level latent features of users and items. We transfer these user and item cluster-level features rather than codebook to the target domain in a novel way and learn the latent feature matrices of target domain to get the predicted matrix of target domain. As far as we know, there is no work which transfers the learnt latent features of codebook to the target domain.
4. We propose another novel method for cross-domain recommender systems in which we generate the codebook from source data by using the existing transfer learning techniques and we transfer the constructed codebook to the target domain in a novel way by using the `hinge loss` function instead of the commonly used squared loss. According to the literature, there is no research work which considers the hinge loss function while transferring the codebook of source data to target domain.

The structure of the thesis is shown in Figure 1.7.

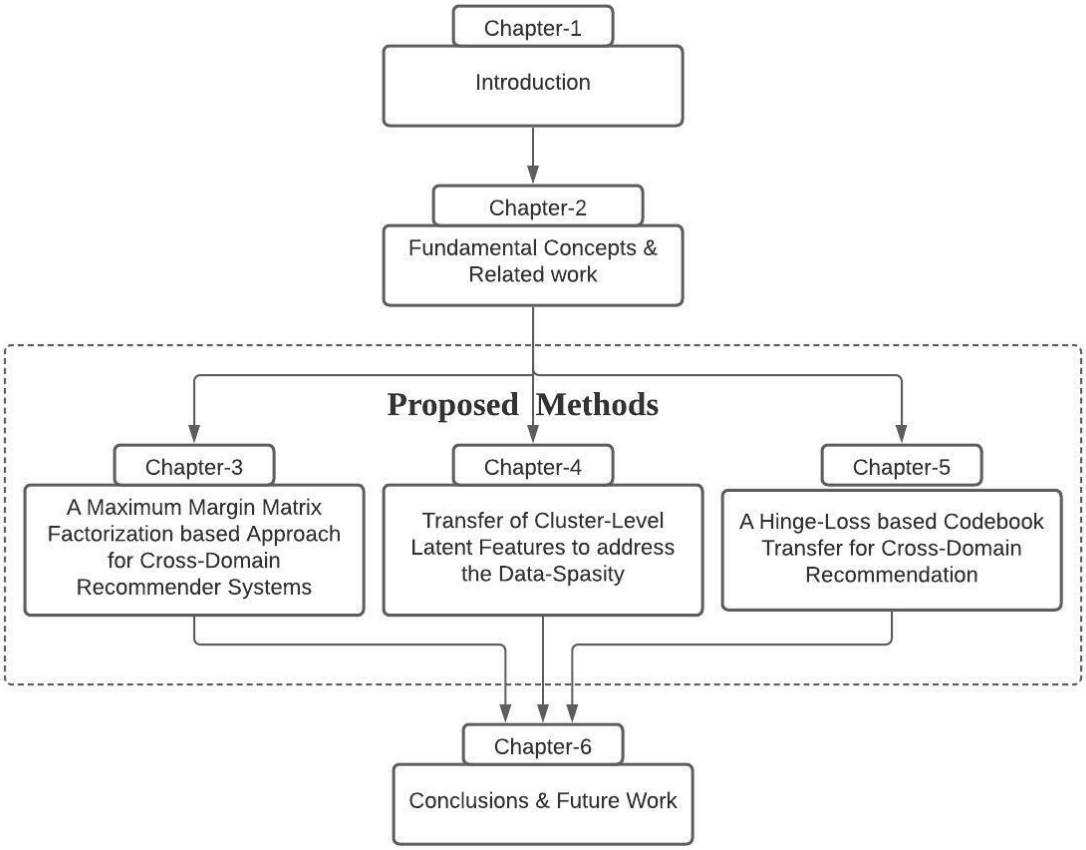


Figure 1.7: Layout of the thesis

1.6 Thesis Outline

The thesis is organized as follows.

In Chapter 2 we discuss in detail on matrix factorization technique which is one of the widely used collaborative filtering techniques. We also discuss different types of matrix factorization methods among which maximum margin matrix factorization which best suits for discrete ratings is used in our thesis. Thereafter, we discuss the transfer learning technique which is needed for cross-domain recommender systems for addressing the data sparsity problem. In the later part of the chapter, we discuss at length the related work on cross-domain recommender systems which remains the main focus area as far as this thesis is concerned.

Chapter 3 introduces a novel method of cross-domain recommendation in which we combine the maximum-margin matrix factorization technique and clustering technique to generate the codebook which is the knowledge to be transferred to the target domain.

The novel methodology of generating the knowledge from source is discussed, and transferring the learnt knowledge to target domain in different ways is also addressed in the chapter.

Chapter 4 discusses the novel technique of cross-domain collaborative filtering where the co-clustering technique is used on the rating matrix of source domain to generate the codebook. Thereafter the codebook is processed to get the partial codebook and maximum margin matrix factorization is applied on the partial codebook. We transfer the resultant latent feature matrices of the codebook to the target domain in a novel way.

In Chapter 5, we introduce another novel method for cross-domain recommender systems in which co-clustering technique is applied on the original source rating data to generate the codebook of source, and then the codebook gets transferred to the target domain in a novel way using hinge loss.

We conclude our thesis with Chapter 6, by giving future directions.

1.7 Publications of the Thesis

1. Sowmini Devi V., Vineet Padmanabhan, Arun K. Pujari. "A Matrix Factorizatino & Clustering Based Approach for Transfer Learning" Pattern Recognition and Machine Intelligence (PReMI) 2017:77-83.
2. Sowmini Devi Veeramachaneni, Arun K. Pujari, Vineet Padmanabhan, Vikas Kumar. "A Maximum Margin Matrix Factorization based Transfer Leanring Approach for Cross-Domain Recommendation." Applied Soft Computing 85 (2019).

Papers submitted / to be communicated

1. Sowmini Devi Veeramachaneni, Arun K. Pujari, Vineet Padmanabhan, Vikas Kumar. "Transfer of codebook latent factors for cross-domain recommendation with non-overlapping data", submitted to Neural Processing Letters.
2. Sowmini Devi Veeramachaneni, Arun K. Pujari, Vineet Padmanabhan, Vikas Kumar. "A Hinge-Loss based Codebook Transfer for Cross-Domain Recommendation with Nonoverlapping Data", submitted to Information Systems.
3. Sowmini Devi Veeramachaneni, Arun K. Pujari, Vineet Padmanabhan. "A Survey on Cross-Domain Recommender Systems", to be communicated.

CHAPTER 2

Fundamental Concepts & Related Work

In the previous chapter we have seen the basic introduction on recommender systems, types of recommender systems, and we have also explained the concept of cross-domain recommender systems in the backdrop of transfer learning. In this chapter, we discuss in detail about Matrix Factorization techniques which is one of the most popular and recent methods on which collaborative filtering is based and also discuss in detail on transfer learning which is essential in building cross-domain recommender systems. Having done that, we discuss at length on the major research works that have come out over the years in the area of cross-domain recommender systems trying to address to the data sparsity problem.

Recommender systems usually depend on explicit form of data (ratings given by user to items) which is placed in the form of a matrix and is often termed as user-item rating matrix wherein the rows indicate users and the columns indicate items. Collaborative filtering approaches (user-based or item-based) are simple and straightforward. One of the most successful CF techniques is Matrix Factorization (81; 65; 91; 28; 96). Matrix factorization or matrix decomposition techniques are typically more powerful as they allow one to identify the latent characteristics (latent features) underlying the user-item interactions. The goal of Matrix Factorization is to find two matrices such that the dot product of these two matrices approximates the original rating matrix. Using MF, one can find the hidden features of users and items. It characterizes both users and items by vectors of factors inferred from rating patterns. It is more applicable when

something is hidden under the data and one would like to find it. The high correlation between user and item latent features contributes to a recommendation.

2.1 Collaborative Prediction via MF

In collaborative filtering, the system tries to find out the users that are similar to the target user based on their ratings and recommends items the similar users have liked. It's "collaborative" because it predicts the preferences of the target user on the basis of other users ratings. The significant assumption underlying the collaborative filtering is: users with identical tastes (preferences) in the past are probably going to have similar preferences in the future too. It is this assumption that permits to take a user's past history and extrapolate it into their future and predict items that they would enjoy (like). They will like comparative sorts of things as they preferred before. The collaborative recommendations are mainly based on user-item interactions (user-item ratings). In collaborative prediction problem, the goal is is to predict the missing ratings using the existing ratings, and it can be viewed as a simple matrix completion problem.

Matrix Factorization (MF) (28; 91; 85; 41) techniques are a family of algorithms in collaborative filtering which extract the latent/hidden factors for users and items from a single rating matrix. By process of factorization of the rating matrix, these methods try to extract latent factors of users and items as one latent vector for each user (or, each item) that captures the characteristics of the user (or, item). The product got from multiplying the latent vector of a user with that of an item yields the rating of the user for the item. This is achieved by a low-dimensional embedding process. Formally, given a user-item rating matrix $Y \in \mathbb{R}^{m \times n}$ where m is the number of users and n is the number of items, we find two matrices, $U \in \mathbb{R}^{m \times \ell}$ and $V \in \mathbb{R}^{n \times \ell}$, where ℓ is the dimension of the embedding, such that the product is approximately equal to Y on observed entries, i.e., $U \times V^T = \hat{Y} \approx Y$. The problem can be formulated as the following optimization problem where ω denotes the set of observed entries in the rating matrix.

$$Minimize\ \mathcal{J} = \sum_{(i,j) \in \omega} \mathcal{L}(y_{ij}, u_i v_j)$$

where $\mathcal{L}(\cdot)$ is a loss function that measures the discrepancy between the observed rating

y_{ij} and its approximation $\hat{y}_{ij}$ (i.e. $u_i v_j^T$). In most of the traditional matrix factorization models, `sum-squared error` is taken as the loss function. This basic function may overfit the data, and in order to avoid overfitting, a regularization term is introduced into the optimization function.

$$Minimize\ \mathcal{J} = \sum_{(i,j)\in\omega} \mathcal{L}(y_{ij}, u_i v_j) + \mathcal{R}(U, V) \tag{2.1}$$

Here $\mathcal{R}(U, V)$ is a regularization term on user and item latent factor matrices.

2.2 Types of Matrix Factorization Techniques

There are different types of Matrix Factorization models that can be outlined as follows.

- Sparse Matrix Decomposition (LU)
- QR decomposition
- Singular Value Decomposition (SVD) or Principal Component Analysis (PCA)
- CUR matrix approximation (less accurate than SVD)
- Large Semi Definite Program (LRSDP)
- Regularized MF (RMF)
- Maximum Margin MF (MMMF)
- Probabilistic MF (PMF)
- Non-negative MF (NMF)
- Localized MF
- Divide-and-Conquer MF (parallelized technique)
- Structured MF

In our thesis, we take into consideration NMF, RMF, PMF, and MMMF as these methods have proven to be successful when applied in the domain of recommender systems. We have chosen maximum margin matrix factorization to be used in this thesis, as our primary focus is to predict the missing values given a highly sparse rating matrix. It can be seen from the literature that in the context of collaborative filtering,

Maximum Margin Matrix factorization (MMMF) (81; 65; 80; 72) is highly successful and is primarily used for prediction of discrete values $\{1, 2, \ldots, r\}$ with `hinge-loss` (or, `smooth hinge loss`). In this setting, matrix factorization approach to CF is to determine two factor matrices such that the product of these two matrices is consistent at the observed elements and the factor matrices capture the user and item characteristics. There are several approaches (64; 38; 45) to accomplish this task and the objective is to minimize a suitably defined loss function. The loss function can be minimized by different techniques such as Semi-Definite Programming(SDP), Gradient Descent(GD), Alternating Least Squares(ALS) (28), Expectation Maximization(EM) (91) to name a few. The most popular and practical approaches are variants of Gradient-search techniques where the search starts with an initial pair of factors and moves along the gradient iteratively obtaining new pairs till a minimizing point is reached. In some MF formulations, the loss function is convex (65) and it is possible to get global optimum. Most often convexity of the loss function is not guaranteed and hence the search terminates at a local minimizing point. Variants of gradient descent such as Conjugate Gradient-Search (16) or Stochastic Gradient-Search (18) are proposed earlier to get the better local minimizing point and improving the probability of reaching a global optimal.
Semidefinite programming approach is theoretically sound but is not scalable for reasonably large data. Expectation Maximization algorithm (91) for CF updates each of the factors alternately. In other words, it starts with a randomly selected initial user factor matrix and uses it to determine the item factor matrix which minimizes the error. The item factor matrix, so computed, is used in the next step to update user factor matrix. This alternating process is repeated till the termination criterion is met.

2.2.1 Non-negative Matrix Factorization (NMF)

NMF (32; 31) is also called as non-negative matrix approximation. NMF attempts to impose a restriction on individual elements of factor matrices U and V as non-negative elements. From the factor analysis view this restriction is reasonable. Suppose if we take a rating u_{i1} which says that user i likes item 1 which is an action movie, then a large u_{i1} means that user i likes the action movies very much whereas a smaller value indicates the opposite. $u_{i1} \times v_{j1}$ says the rating of user i for movie j with respect to action genre(factor). If we take an example where u_{i1} = 20 and v_{j1} =

-20, then it results in a negative score of -400, meaning that user i does not like action movies at any cost. If elements are non-negative then it becomes u_{i1} = 20 and v_{j1} = 0 resulting in 0, indicating that we can't say about user i's interest in movie j based on the "action" factor. The objective is to determine non-negative low rank matrices U and V which minimizes the following loss function.

$$\mathcal{J} = \sum_{(i,j)\epsilon\omega} (y_{ij} - u_i v_j^T)^2, such\ that\ U, V \geq 0 \tag{2.2}$$

where, ω is the set of observed (i, j) pairs.

2.2.2 Regularized Matrix Factorization (RMF)

This is the simplest formulation of all MF techniques. The objective is to determine a pair of factors such that the element-wise aggregated squared error for the observed values is minimized. In addition to the loss function, in order to avoid overfitting, a regularization constraint is added to the optimization function. The objective of RMF is to minimize $\mathcal{J}$,

$$\mathcal{J} = \sum_{(i,j)\epsilon\omega} (y_{ij} - u_i v_j^T)^2 + \lambda(||U||_F + ||V||_F) \tag{2.3}$$

where, ω is the set of observed (i, j) pairs, $\lambda > 0$ is the regularization parameter and $||.||_F$ is the Frobenius norm.
Frobenius norm of a matrix ($A_{m\times n}$) is defined as the square root of the sum of absolute squares of individual elements,
i.e., $||A||_F = \sqrt{(\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2)}$

2.2.3 Probabilistic Matrix factorization (PMF)

Probabilistic MF (PMF) is a generative model which presupposes a Gaussian distribution for the data. In this, ratings (Y) are modeled as draws from a Gaussian distribution with mean for Y_{ij} as $U_i V_j^T$. Zero-mean spherical gaussian priors are placed on U and V. i.e., Each row of U and V are drawn from a multivariate gaussian distribution with mean as 0 and precision is multiple of identity matrix I, as shown in equations

below (2.4, 2.5).

$$P(U|\sigma_U^2) = \prod_{i=1}^{m} \mathcal{N}(U_i|0, \sigma_U^2 I) \tag{2.4}$$

$$P(V|\sigma_V^2) = \prod_{j=1}^{n} \mathcal{N}(V_j|0, \sigma_V^2 I) \tag{2.5}$$

Given the user feature vectors and movie feature vectors, the distribution for the corresponding rating is given by equation (2.6),

$$P(Y|U, V, \sigma^2) = \prod_{i=1}^{m} \prod_{j=1}^{n} [\mathcal{N}(Y_{ij}|U_i V_j^T, \sigma^2)]^{I_{ij}} \tag{2.6}$$

The goal of PMF is to maximize the log-posterior of (2.6) over U and V. Maximizing the log posterior of (2.6) is equivalent to minimizing (2.7) .

$$\mathcal{J} = \frac{1}{2}(\sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij}(Y_{ij} - U_i V_j^T)^2 + \lambda_U \sum_{i}^{m} ||U||_F^2 + \lambda_V \sum_{j}^{n} ||V||_F^2) \tag{2.7}$$

where, I_{ij} is the indicator matrix whose entry is 1 if item j is rated by user i otherwise 0, $\lambda_U = \frac{\sigma^2}{\sigma_U^2}$ and $\lambda_V = \frac{\sigma^2}{\sigma_V^2}$.

2.2.4 Maximum Margin Matrix factorization (MMMF)

In the context of collaborative filtering, a variant of matrix factorization called the Maximum Margin Matrix factorization (MMMF) (81; 65; 80; 72) is shown to be successful and is primarily used for prediction of discrete values $\{1, 2, \ldots, r\}$ with hinge-loss (or, smooth hinge loss). It has a process of regularization where it constrains the norms of U and V (trace norm) instead of the dimensionality. The problem is to determine latent factor matrices $U \in \mathbb{R}^{m \times \ell}$ and $V \in \mathbb{R}^{n \times \ell}$, and $r - 1$ thresholds θ_{ia} $(1 \leq a \leq r - 1)$ for every user i so as to minimize the following objective function.

$$\mathcal{J}(U, V, \Theta) = \sum_{(i,j) \in \omega} \sum_{a=1}^{r-1} h(\mathcal{T}_{ij}^a(\theta_{ia} - u_i v_j^T)) + \frac{\lambda}{2}(||U||_F^2 + ||V||_F^2) \tag{2.8}$$

where, $\lambda > 0$ is regularization parameter,

$$\mathcal{T}_{ij}^{a} = \begin{cases} +1 & \text{if } a \geq y_{ij} \\ -1 & \text{if } a < y_{ij} \end{cases}$$

and $h(\cdot)$ is a smoothed hinge-loss function defined as,

$$h(z) = \begin{cases} 0 & \text{if } z \geq 1 \\ \frac{1}{2}(1-z)^2 & \text{if } 0 < z < 1 \\ \frac{1}{2} - z & \text{otherwise.} \end{cases}$$

It can be seen that, MMMF (Eq. 2.8) does not require $u_i v_j^T$ and y_{ij} to be closer. It expects $u_i v_j^T$ to be as small as possible if $a \geq y_{ij}$ and as large as possible if $a < y_{ij}$, when compared with θ_{ia}. It is pertinent here to discuss the geometric interpretation of MMMF. The latent factor-vector of each item (corresponding row of V) can be viewed as a point in ℓ-dimensional space and the latent factors of users (rows of U) can be viewed as decision hyperplanes in this space (29). Every pair (u_i, θ_{ir}) defines a hyperplane in ℓ-dimensional space. The objective of MMMF is to learn the embeddings of these points and hyperplanes in $\mathbb{R}^{\ell}$ such that each hyperplane (corresponding to a user) separates (or, equivalently, classifies) the items into r rating based on $(r-1)$ thresholds $\theta_{ir}(1 \leq \theta_{ir} \leq r-1)$. Each hyperplane acts as maximum-margin separator which is ensured by optimizing hinge loss function or smooth hinge loss function. The interpretation can be equivalently viewed with user-latent factors as points and item-latent factors as hyperplanes.

One can solve the optimization functions given in Equations (2.3), (2.7) and (2.8) using gradient descent method by updating U, V, Θ, using (2.9).

$$\begin{aligned} U_{t+1} &= U_t - c\,\frac{\partial \mathcal{J}}{\partial U} \\ V_{t+1} &= V_t - c\,\frac{\partial \mathcal{J}}{\partial V} \\ \Theta_{t+1} &= \Theta_t - c\,\frac{\partial \mathcal{J}}{\partial \Theta} \end{aligned} \tag{2.9}$$

where c is a trade-off parameter. $\frac{\partial \mathcal{J}}{\partial U}$, $\frac{\partial \mathcal{J}}{\partial V}$, $\frac{\partial \mathcal{J}}{\partial \Theta}$ are the partial derivatives (gradients) of $\mathcal{J}$ w.r.t U, V, Θ and are given below (Equations 2.10, 2.11, 2.12).

$$\frac{\partial \mathcal{J}}{\partial U_{i\ell}} = \lambda U_{i\ell} - \sum_{a=1}^{r-1} \sum_{j|ij\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\theta_{ia} - UV^T))V \tag{2.10}$$

$$\frac{\partial \mathcal{J}}{\partial V_{j\ell}} = \lambda V_{j\ell} - \sum_{a=1}^{r-1} \sum_{i|ij\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\theta_{ia} - UV^T))U \tag{2.11}$$

$$\frac{\partial \mathcal{J}}{\partial \Theta_{ir}} = \sum_{j|ij\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\theta_{ia} - UV^T)) \tag{2.12}$$

where,

$$h'(z) = \begin{cases} 0 & \text{if } z \geq 1 \\ z - 1 & \text{if } 0 < z < 1 \\ -1 & \text{otherwise.} \end{cases}$$

2.3 Toy example of matrix factorization

Suppose we have a set of m users (**U**) and set of n items (**I**). Let $Y = [y_{ij}]$ is $m \times n$ ($|U| \times |I|$) user/item rating matrix as shown in Table-2.1 in which there are 5 users and 4 items (i.e.,$Y_{5\times4}$). Each element $y_{ij}\ \epsilon\ \{0, 1, ...R\}$, R is total level of ratings (5 in our case). 0 represent the unknown rating. The goal is to predict the unknown ratings.

	I1	**I2**	**I3**	**I4**
U1	5	3	0	1
U2	4	0	0	1
U3	1	1	0	5
U4	1	0	0	4
U5	0	1	5	4

Table 2.1: User-Item Rating matrix (Y)

If we apply MF (with squared loss) on the example shown above (2.1), by considering the number of latent features as two (i.e., ℓ = 2), the following (Equation-2.13) are the $U_{5\times2}$, $V_{2\times4}^T$ and the predicted matrix ($\hat{Y}_{5\times4} = UV^T$) which we get initially with the random initialization of U and V. The same are shown in Figure-2.1 in which blue

color vectors are user latent feature vectors (U), and the vectors which are green in color are item latent feature vectors (V).

$$\underset{U}{\begin{pmatrix} 0.5904876 & 0.18520637 \\ 0.89651422 & 0.62040181 \\ 0.97932976 & 0.31239695 \\ 0.71248937 & 0.00531318 \\ 0.2988344 & 0.7485681 \end{pmatrix}} \underset{V^T}{\begin{pmatrix} 0.93627203 & 0.78030931 & 0.01137135 & 0.31557944 \\ 0.35834808 & 0.04508127 & 0.01842057 & 0.10453503 \end{pmatrix}} = \underset{\hat{Y}}{\begin{pmatrix} 0.6192254 & 0.78030931 & 0.01137135 & 0.31557944 \\ 0.35834808 & 0.04508127 & 0.01842057 & 0.10453503 \\ 1.0288659 & 0.7782634 & 0.0168908 & 0.3417128 \\ 0.6689878 & 0.5562016 & 0.0081998 & 0.2254024 \\ 0.5480382 & 0.2669297 & 0.0171872 & 0.1725576 \end{pmatrix}} \tag{2.13}$$

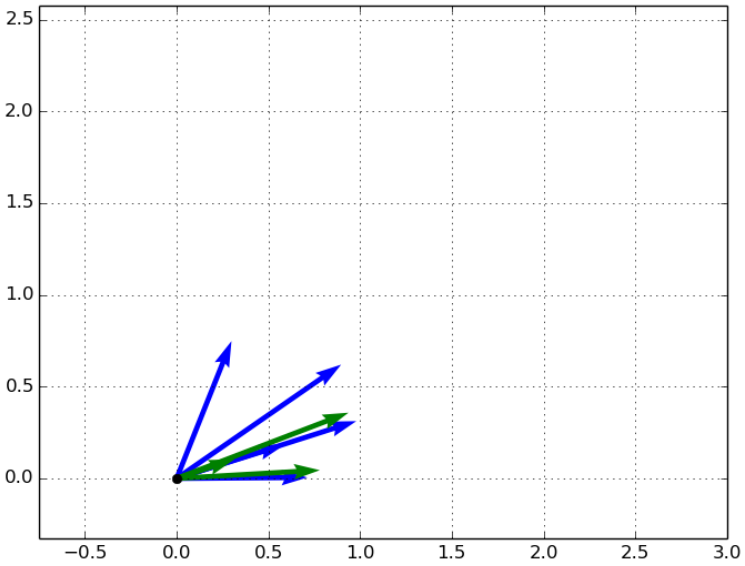


Figure 2.1: Pictorial representation of latent feature vectors (U - blue color vectors and V - green color vectors) in latent space initially

Once we calculate the predicted rating matrix (UV^T), we need to calculate the difference between the predicted values and the actual (existing) values and then we need to try to minimize the difference recursively by using Gradient Descent (GD). GD helps to find out in which direction (U and V) we should go to minimize the error and keep on going iteratively until no more error exist. So, for the given toy example, when we apply GD and go on updating U and V iteratively, to minimize the error, the updated U, V, $\hat{Y}$ which were obtained at 1500^{th} iteration are shown in Equation-2.14 and which were obtained at 3000^{th} iteration are shown in Equation-2.15. The pictorial representation of

the same is shown in Figure-2.2 and Figure-2.3.

$$\underset{U}{\begin{pmatrix} 1.77752894 & 0.29656766 \\ 1.16303033 & 0.66145795 \\ 1.30310564 & 0.67387977 \\ 1.26769425 & 0.32239559 \\ 1.38022296 & 1.97844308 \end{pmatrix}} \underset{V^T}{\begin{pmatrix} 1.77800423 & 1.21778941 & 1.15383339 & 1.6100205 \\ 0.80150418 & -0.16676241 & 1.62889942 & 0.88737676 \end{pmatrix}} = \underset{\hat{Y}}{\begin{pmatrix} 3.3982 & 2.1152 & 2.5341 & 3.1250 \\ 2.5980 & 1.3060 & 2.4194 & 2.4595 \\ 2.8570 & 1.4745 & 2.6012 & 2.6960 \\ 2.5124 & 1.4900 & 1.9879 & 2.3271 \\ 4.0398 & 1.3509 & 4.8152 & 3.9778 \end{pmatrix}} \tag{2.14}$$

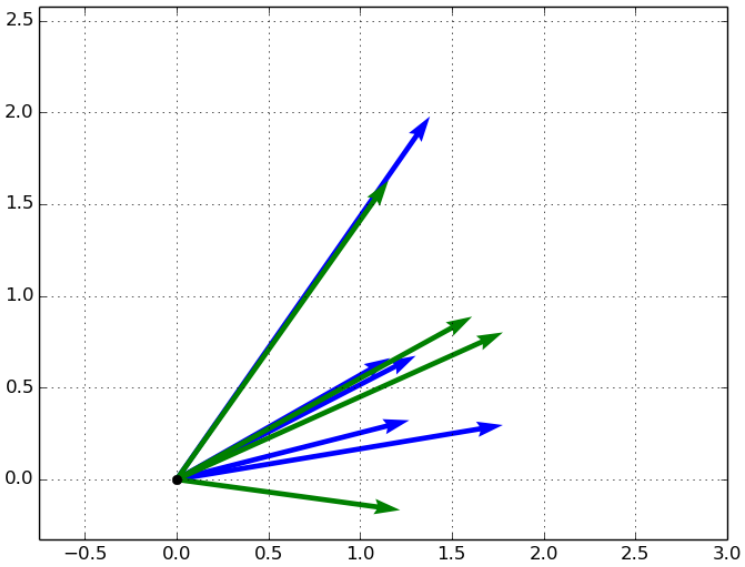


Figure 2.2: Pictorial representation of latent feature vectors (U - blue color vectors and V - green color vectors) in latent space at 1500^{th} iteration

$$\underset{U}{\begin{pmatrix} 2.11668492 & -0.46805813 \\ 1.52168501 & 0.05662858 \\ 1.01298558 & 1.6322625 \\ 0.99366696 & 1.07722617 \\ 1.17065079 & 1.84492609 \end{pmatrix}} \underset{V^T}{\begin{pmatrix} 2.08575665 & 1.38337271 & 1.24972628 & 1.08183373 \\ -0.23726548 & -0.3410333 & 1.77726989 & 1.8130988 \end{pmatrix}} = \underset{\hat{Y}}{\begin{pmatrix} 4.52594 & 3.08779 & 1.81341 & 1.44127 \\ 3.16043 & 2.08575 & 2.00233 & 1.74888 \\ 1.72556 & 0.84468 & 4.16693 & 4.05534 \\ 1.81696 & 1.00724 & 3.15633 & 3.02810 \\ 2.00396 & 0.99027 & 4.74192 & 4.61148 \end{pmatrix}} \tag{2.15}$$

$$\underset{U}{\begin{pmatrix} 2.13929538 & -0.49059839 \\ 1.73310635 & -0.28993261 \\ 1.06195536 & 2.01308987 \\ 0.92227177 & 1.57269387 \\ 1.06416843 & 1.61413303 \end{pmatrix}} \underset{V^T}{\begin{pmatrix} 2.1818626 & 1.34069235 & 1.44112731 & 0.90992741 \\ -0.64414485 & -0.23296697 & 2.07216746 & 1.96481116 \end{pmatrix}} = \underset{\hat{Y}}{\begin{pmatrix} 4.98367 & 2.98243 & 2.06639 & 0.98267 \\ 3.96816 & 2.39111 & 1.89684 & 1.00734 \\ 1.02032 & 0.95477 & 5.70187 & 4.92164 \\ 0.99923 & 0.87010 & 4.58800 & 3.92925 \\ 1.28213 & 1.05068 & 4.87836 & 4.13978 \end{pmatrix}} \tag{2.16}$$

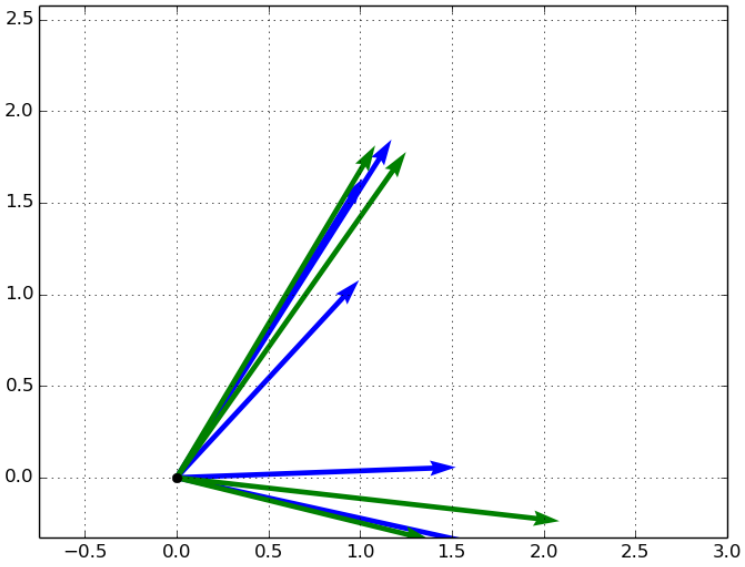


Figure 2.3: Pictorial representation of latent feature vectors (U - blue color vectors and V - green color vectors) in latent space at 3000^{th} iteration

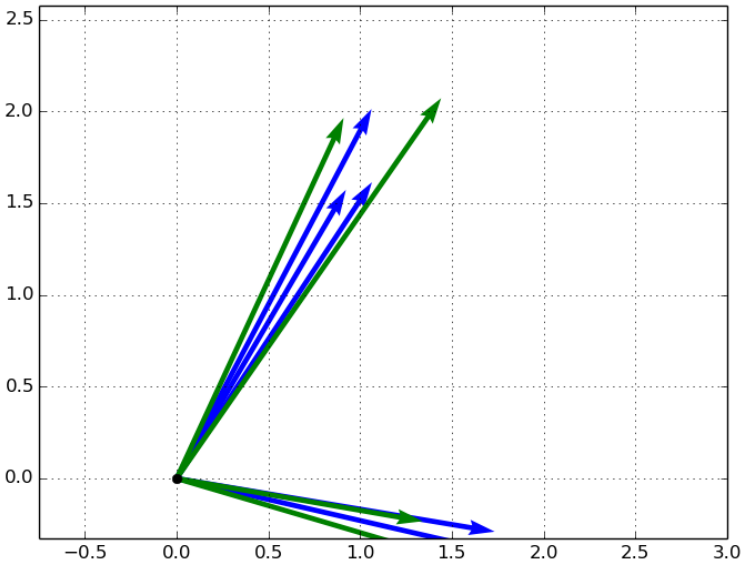


Figure 2.4: Pictorial representation of latent feature vectors (U - blue color vectors and V - green color vectors) in latent space at 4999^{th} iteration

When the latent feature vectors are updating iteratively by minimizing the error between actual and predicted values, at some point of time the error doesn't change or there will be very less variation between the current error and the previous error. At this point of time, local minima (or sometimes global minima) is obtained and the feature vectors obtained here are final vectors (U and V). For the above toy example (Figure-

2.1), the user and item latent feature vectors obtained at 4999^{th} iteration are shown in Equation-2.16, after which there is no change in the error. The corresponding vectors are depicted in latent space as shown in Figure-2.4.

2.4 Transfer Learning in recommender systems

We have seen the basics of cross-domain recommender systems (6) which can be achieved using Transfer Learning in Section-1.3 of Chapter 1. As discussed, in order to reduce the data-sparsity issue in collaborative filtering, cross-domain recommender systems (51; 95; 100) have come into picture. In recommender systems, one of the main data used to recommend the items is user-item ratings and besides these ratings (feedbacks), there exists some auxiliary data that can be used to recommend items to an active user. This auxiliary data can be used to overcome the data sparsity problem, by transferring the knowledge (transfer learning) from auxiliary domain (source) to target domain. In cross-domain recommender systems where transfer learning technique is used, before transferring the data, the following are the three main questions (52) that need to be addressed,

1. What to transfer?
2. How to transfer?
3. When to transfer?

`What to transfer?` - Before transferring the data to the target domain, we need to decide what information needs to be transferred in order to achieve more accuracy in prediction or recommendation. The auxiliary data that can be transferred to the target domain can be - user latent features, item latent features, tags, cluster-level rating pattern (codebook) etc..

`How to transfer?` - Once the data to be transferred is decided, the next question is how to transfer that data. This question can be addressed from two perspectives - "knowledge transfer algorithm styles" (collective, adaptive, integrative) and "knowledge transfer strategies" (regularization, prediction rule, constraint) which will be discussed in detail in the coming sections of the chapter.

`When to transfer?` - This question is addressed by knowing when not to transfer. Data should not be transferred if the performance is getting reduced after transferring. This is called 'negative transfer' i.e., learning in one context degrades the performance in another context.

In this thesis, we mainly address the first two questions i.e., what to transfer and how to transfer. If transferring the knowledge is diminishing the prediction accuracy of collaborative filtering, then we should not transfer the knowledge as the negative transfer occurs, and so when not to transfer is automatically addressed.

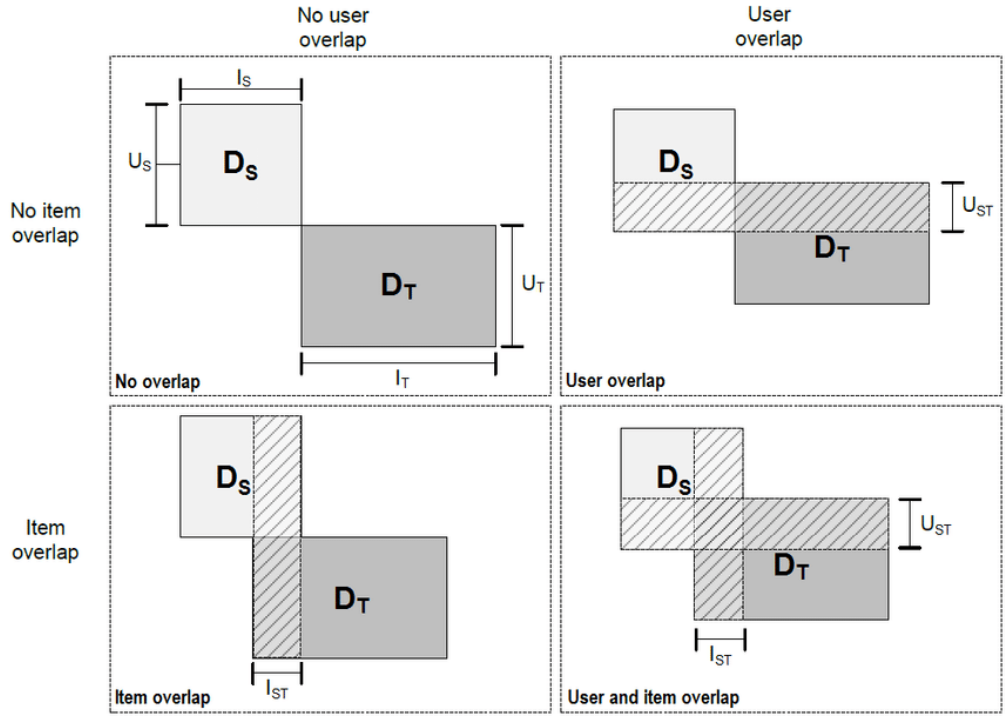


Figure 2.5: Different scenarios of user and item overlap (6)

In cross-domain recommender systems, there are different scenarios such as user overlap (user-user overlap) between source and target domains, item overlap (item-item overlap) between source and target domains, both user and item overlap (user-item overlap) between source and target domains, and no overlap of users and items (no overlap) between source and target domains. If we consider Figure 2.5 which is considered from (6), D_S is the source (auxiliary) domain, D_T is the target domain, U_S is the users from source, I_S is the set of items from source, U_T and I_T is the set of users and items from target domain, U_{ST} is the set of common users between source and target domain, I_{ST} is the set of common items between source and target domains. In the figure, there are four quadrants, in which the first quadrant shows that D_S and D_T have no user or item overlap. The scenario in the second quadrant says that there

is user overlap between D_S and D_T, but there is no item overlap. Whereas in the third quadrant there exists item overlap between D_S and D_T, but there is no user overlap. Finally, the fourth quadrant says that there exist both user and item overlap between D_S and D_T.

In all our proposed methods, we assume that there is no overlap of users and items between source and target domains, as a result, all of our proposed methods in the thesis fall under the first quadrant in which there is no overlap of users and items between source and target domains.

2.4.1 Transfer learning techniques

To address the query "how to transfer?" w.r.t. "knowledge transfer algorithm styles" we have the following three different styles,

1) Adaptive knowledge transfer
2) Collective knowledge transfer
3) Integrative knowledge transfer.

For each of these algorithm styles, three knowledge strategies and three different approaches exist which are as follows.

i) Transfer via regularization
ii) Transfer via prediction rule
iii) Transfer via constraint

a) Instance-based approach
b) Feature-based approach
c) Model-based approach

These approaches of transfer learning place a focus on which portion of knowledge is being used as a medium to promote knowledge transfer. One of the most common motivations for *instance-based* transfer learning approach is that, due to domain differences, although the source domain labeled data cannot be reused precisely, a portion of it can be used for the target domain after reweighting or resampling. In this way, the

source domain labeled instances with significant weights can be thought of as information that can be transferred between domains. The instance-based approaches make the implicit assumption that the source and target domains have many overlapping features, implying that the domains have similar or identical support.

However, in many real-world implementations, only a small portion of feature spaces from the source and target domains overlap, preventing many features from being used directly as information transfer bridges. As a consequence, some instance-based approaches for information transfer may be ineffective. In this case, *feature-based* transfer learning methods are more promising. The concept underlying feature-based methods is to find out a good feature representation for both source and target domains so that the labeled data from the source domain can be used to train a precise classifier for the target domain by projecting data onto the new representation. In this way, the learned feature representation may be treated as the domain-specific knowledge to be transferred.

Model-based transfer learning methods presume that the source and target domains share certain learning model parameters or hyperparameters. Model-based methods are motivated by the fact that a well-trained source model has acquired lots of helpful structure and that may be transferred to train a more accurate target model. As a result, the knowledge that is to be transferred is the domain-invariant structure of the model parameters.

We have studied different papers and categorized them into different algorithmic styles, as well as into different strategies and approaches discussed above. The upcoming sections gives the categorization of existing literature on cross-domain collaborative filtering using transfer learning technique to solve the data sparsity problem.

2.4.2 Generic framework of knowledge transfer

As seen, matrix factorization based techniques can be represented with a loss function, regularization term and a constraint as follows.

$$\min_{\Theta,K} \mathcal{L}(\Theta, K|Y, X) + \mathcal{R}(\Theta|K, X) + \mathcal{R}(K), \quad s.t., \Theta \in C(K, X)$$

Here $\mathcal{L}(\Theta, K|Y, X)$ is loss function, $\mathcal{R}(\Theta|K, X)$, $\mathcal{R}(K)$ are regularization terms and $\Theta \in C(K, X)$ is a constraint. Prediction rule is embedded in the loss function. X is the auxiliary data, Y is the target user-item rating matrix, K is the extracted knowledge from X, and Θ is the model parameter. We also have another matrix called as indicator matrix (I) whose values are either 1 or 0. The value is 1 if the rating exists in the corresponding entry of Y, and the entry is 0 if the rating does not exist in the target rating matrix.

2.5 Adaptive knowledge transfer

It aims to fit extracted knowledge of the auxiliary(source) domain to the target domain. It is a directed knowledge transfer approach (from auxiliary to target), and following are the two adaptive knowledge transfer strategies - (i) Transfer via constraint, $min_{\Theta}\mathcal{L}(\Theta|Y),\ s.t.\Theta \in C(K, X)$ (ii) Transfer via Regularization, $min_{\Theta}\mathcal{L}(\Theta|Y) + \mathcal{R}(\Theta|K)$.

2.5.1 Transfer via constraint

CodeBook Transfer (CBT) (35): In this, a transfer of codebook(cluster-level rating pattern) from source(auxiliary) to target is done. Initially, rows and columns of auxiliary rating matrix are co-clustered to get a cluster level rating pattern (codebook) $C \in R^{k \times k}$. Each entry in C is the mean rating of the associated co-cluster. The constructed codebook C gets transferred to the target by codebook expansion as UCV^T, and with a condition $C = \breve{C}$. Here rating pattern is shared between auxiliary and target data, and U, V are membership indicator matrices. CBT is model-based as the codebook (C-shared parameter between auxiliary and target) is discovered.

Rating Matrix Generative Model (RMGM) (36): This is an extension to CBT, in which codebook construction and expansion are done in a single step with soft membership indicator matrices. It assumes that multiple sources share a single latent pattern. In this, instead of a single auxiliary matrix, there are multiple related auxiliary rating matrices from which the relatedness can be established by finding the shared implicit cluster-level rating pattern. It is a probabilistic and model-based approach as codebook (C) is shared between different domains.

This Cluster-level rating pattern is a type of collective behavior that is more steady and transferable than individual behavior. It is especially beneficial when the explicit correspondences or overlaps between target and auxiliary data entities (users/items) are not available.

2.5.2 Transfer via regularization

Coordinate System Transfer (CST) (58): In this, latent features are getting transferred. It integrates the latent features (or coordinate systems) obtained from source domain into the target factorization (i.e., $I \odot Y \sim UCV^T$) through regularization.

$$||U - \dot{U}||_F^2 + ||V - \ddot{V}||_F^2$$

Here $U \in R^{m \times l}$, $V \in R^{n \times l}$ are latent feature matrices of target and $\dot{U} \in R^{n' \times l}$, $\ddot{V} \in R^{m' \times l}$ are user and item specific feature matrices of auxiliary data. These two regularization terms are used to constrain that U and V to be similar to $\dot{U}$ and $\ddot{V}$. Approximation is done via matrix tri-factorization (UCV^T), where V and U are orthonormal matrices. It is a feature-based approach, as the coordinate systems are getting transferred between auxiliary and target domains.

2.6 Collective knowledge transfer

In this, shared knowledge and the unshared effect of target and auxiliary data are jointly learned. This is a bi-directional knowledge transfer strategy. In this, rather than a two-step process like in adaptive transfer, the model parameter Θ and the shared knowledge K are learned concurrently. Following are the studies in collective knowledge transfer - (i) Transfer via constraint on model parameters, $min_{\Theta,K}\mathcal{L}(\Theta|Y) + \mathcal{R}(\Theta) + \mathcal{L}(K|X) + \mathcal{R}(K),\ s.t. \Theta \in C(K)$, (ii) Transfer via regularization, $min_{\Theta,K}\mathcal{L}(\Theta|Y) + \mathcal{R}(\Theta|K, X) + \mathcal{R}(K)$.

2.6.1 Transfer via constraint

Collective Matrix Factorization (CMF) (78): In this technique, user-item rating matrix $Y \in R^{m \times n}$ and one item-content matrix are collectively factorized by sharing the same item-specific latent features V. In this case, $V = \ddot{V}$, which means that item specific latent feature matrix $\ddot{V}$ is shared and acts as a bridge for knowledge transfer between two domains of data. Factorize $\tilde{Y} \sim WV^T$ and $Y \sim UV^T$, which says that item-specific latent features are shared between auxiliary and target data domains, resulting in feature-based approach.

Social Recommendation (SoRec) (42): It extends the basic matrix factorization model by together factorizing user-item rating matrix Y ($I \odot Y \sim UV^T$) and a user-user social network matrix $\dot{Y}$ ($\dot{Y} \sim \dot{U}\dot{V}^T$) with a constraint $U = \dot{U}$. In social network matrix, vertices represent users and edges represent relation between users, say, how much a user i trusts/knows a user j. It comes under feature-based approach.

TrAnsfer Learning in MUltiple Domains (TALMUD) (47): This is similar to CBT, in which a cluster-level rating pattern is shared between source(auxiliary) and target data. Instead of one auxiliary data as in CBT, TALMUD considers multiple auxiliary data and checks different combinations of users/items clusters. It generates different codebooks for each source domain and captures different levels of relatedness between the source and target domains. Knowledge is extracted from many domains and is transferred to target via some constraint saying that a certain relation exists between domains that is captured using codebooks. TALMUD measures the relatedness between different source and target domains without assuming overlapping users/items, and transfers to the target domain.

In this, there exists N different source matrices, and N different codebooks (each represented by C_n). The key problem is determining how to integrate the information(knowledge) from multiple sources and how much data (α_n) is to be transferred from each of the source domains. The relatedness coefficient(α_n) is learnt by reducing the error based on the observed target ratings. The codebook (C_n) from different source domains is transferred using sum of $\alpha_n(U_n C_n {V_n}^T)$, $n = 1, 2, ..., N$. In this the relatedness between codebooks (C_n) from different source domains is discovered as shared parameter, which results in model-based approach.

TRAnsfer collaborative filtering framework from multiple sources via ConsE nsus

Regularization (TRACER) (101): It considers the data from multiple source domains and learns the corresponding predicted matrices and transfers the knowledge. This is more related to TALMUD (47) which produces multiple outputs from various auxiliary data before assigning weights to integrate. In contrast to TALMUD, the TRACER algorithm learns and transfers the knowledge at the same time. During the lerning process, it forces the prediction outcomes for the same missing rating to converge, and while transferring the learnt knowledge it uses consensus regularization which forces all of the predicted results to be similar (say, majority value). Furthermore, the two approaches have different strategies for integrating the predicted outcomes. TALMUD gives the same weight to the predicted ratings learnt from the same source domain, and hence there will be 's' distinct weights if there exist 's' source domains. TRACER is a local majority voting approach, in which it performs majority voting on each user-item pair in the target domain. Thus, in terms of rating prediction, TRACER usually outperforms TALMUD. It comes under model-based approach.

For instance, if there are three source domains and one target domain, it learns different knowledge from source domains, and obtains predicted matrices. While transferring the knowledge, it places constraints on these matrices simultaneously, of which one could be *majority voting*. For instance, when target matrices are predicted with some ratings, we may get different predicted values from various source domains (say 4, 3, 3 from domains - 1, 2, 3 respectively). In this case, the consensus regularization forces these predicted results to be similar which in this case is the majority value - 3 (of domains 2 and 3).

Multiple INcomplete Domains Transfer Learning for Information Recommendation (MINDTL) (20): Considers the data from multiple incomplete source domains and constructs the codebooks for the domains. These learned codebooks get linearly combined and get transferred to the target domain to approximate the target rating matrix.

Transfer by Collective Factorization (TCF) (55): If the target and auxiliary data have different types of feedback data, say target is having ratings (Y), and auxiliary data is having binary feedback (X), the TCF tries to learn data dependent correlation between rows of U and columns of V^T for auxiliary and target data. Factorize $Y = UCV^T$ and $X = \tilde{U}\tilde{C}\tilde{V}^T$ collectively with constraints of sharing user-specific latent feature matrix $U = \tilde{U}$, and item-specific latent feature matrix $V = \tilde{V}$, to get U, V. Estimate C and

$\tilde{C}$ separately to capture domain-dependent information. Here, Shared latent space is constructed via matrix tri-factorization to answer what to transfer question, and in a collective way to address how to transfer. It is a feature-based approach.

interaction-rich Transfer by Collective Factorization (iTCF) (56): It extends CMF by introducing the interactions between user-specific latent features. It is more efficient when compared to TCF and more accurate when compared to CMF. It assumes the same users and same items in target and auxiliary domains. It says that for the same user, the prediction accuracy learned on target data (on numerical ratings) or auxiliary data (binary) is likely to be similar.

$$Y \sim UV^T, \quad \tilde{Y} \sim WV^T, \quad s.t. E = \tilde{E}$$

where E and $\tilde{E}$ are prediction model's errors on two data. This is a feature-based approach.

2.6.2 Transfer via regularization

Twin Bridge Transfer learning (TBT) (76): TBT considers two sets of auxiliary data (X_1, X_2), in which one set shares common set of users with target (Y), and the other shares common set of items with target. It lessens the sparsity in target data by transferring knowledge from dense auxiliary data via two pathways. i) Extracts the latent factors from auxiliary data and constructs the similarity graphs using these latent factors. ii) Transfers latent factors as well as similarity graphs to target data. As the latent factor transfer alone may transfer negative information, similarity graph transfer is also added which results in TBT.

$$||U - U_0||^2 + ||V - V_0||^2 + ||u_{i*} - u_{j*}||^2 (W_U)_{ij} + ||v_{i*} - v_{j*}||^2 (W_V)_{ij},$$

where $W_U, W_V \in 0, 1$ are weight matrices for similarity graphs, in which the distance between users/items is present.

$$W_U = \begin{cases} 1, & \text{if } u_{i*} \in N_p(u_{j*}) \text{ or } u_{j*} \in N_p(u_{i*}) \\ 0, & \text{otherwise} \end{cases}$$

Here, $N_p(u_{j*})$, $N_p(u_{i*})$ are set of p-nearest neighbors of u_{j*} and u_{i*}. Similarly W_V. As latent features and similarity graphs are getting transferred between different domains, it comes under feature-based approach.

MMMF$_{TL}$ (97): In this, a framework of different factorization techniques with transfer learning has been proposed and it is concluded that MMMF$_{TL}$ is the best choice as far as prediction accuracy is concerned. This method applies MMMF on the target domain and then iteratively selects some entities (users/items) based on some certainty measure, and finds the corresponding entities in the source domain. By utilizing the similarities between the selected entities of the source domain, it constrains the similarities of the corresponding target entities.

2.7 Integrative knowledge transfer

It integrates the actual source data into the learning task on the target data as known knowledge. Following are different integrative knowledge transfer strategies - (i) Transfer via prediction rule, $min_\Theta \mathcal{L}(\Theta|Y, A)+\mathcal{R}(\Theta)$, (ii) Transfer via regularization, $min_\Theta \mathcal{L}(\Theta|Y)+\mathcal{R}(\Theta|A)$ and (iii) Transfer via constraint, $min_\Theta \mathcal{L}(\Theta|Y) + \mathcal{R}(\Theta),\ s.t. \Theta \in C(A)$. Here instead of extracted knowledge K, raw auxiliary data A is included.

2.7.1 Transfer via prediction rule

Adding item metadata and tags (TagGSVD++) (14): This model separately captures user, item tagging information and transfers source knowledge to the target. The item metadata and the tag information are integrated into the matrix factorization process inorder to compute the rating predictions. It distinguishes between sets of tags for items and users, and factors the rating matrix into disassociate user and item components.

$$\tilde{r}_{ui} = P^T Q,$$

where, P is the user component which contains information about tags assigned by user to any item, and Q is the item component in which information about items tagged by any user exists. This is an instance-based approach.

2.7.2 Transfer via regularization

Tag Informed Collaborative Filtering (TagiCoFi) (98): In this social tagging data gets incorporated into the target numerical rating data. It tries to make two user-specific latent feature vectors as similar as possible if the two users have comparable tagging history. User-user similarity matrix is constructed from social tagging data and a regularization term is added to the basic matrix factorization method.

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\dot{S}_{ij}||U_{*i}-U_{*j}||_F^2,$$

where $\dot{S}_{ij}$ is the tag-based similarity between users i and j. The main goal is to make user-specific latent features as similar as possible, if the corresponding users have similar tagging history. This regularization term focuses on distance between a user feature vector and each of their friends' feature vectors. This is a model-based approach as it is trying to find the similarity of tags (shared parameter) between source and target domains.

Social Matrix Factorization (Social MF): Social MF investigates the influence of trust propagation and generalises the basic matrix factorization system by adding a regularization term of trustworthy friends.

$$\sum_{u=1}^{n}||U_{i*}-\sum_{u'\in T_u^+}\dot{S}_{uu'}U_{u'_*}||_F^2,$$

where, T_u^+ is the set of trustworthy friends of user u (other than u), and $\dot{S}_{uu'}$ is the similarity between users u and u' acquired via social networks. It transfers knowledge of the friends' tastes by restricting the user-specific latent features to be similar. The above regularization term defines the distance between one user's feature vector and the weighted sum of his/her friends' feature vectors.

2.7.3 Transfer via constraint

Transfer by Integrative Factorization (TIF) (59): In TIF auxiliary data is represented with uncertain ratings (score), denoted by $[a_{ui}, b_{ui}]$, and target data is represented as numerical ratings. The aim of this is to incorporate the uncertain ratings to the target

data. Integrates the auxiliary uncertain ratings as constraints into the matrix factorization as follows,

$$\hat{r} \in Ct(a_{ui}, b_{ui}),$$

which indicates that the estimated rating must fall within the range of the corresponding auxiliary uncertain rating. It comes under instance-based approach, where uncertain ratings are getting re-weighted.

Transfer by Mixed Factorization (TMF) (57): If there exist multiple types of explicit ratings, say 5-star ratings and binary (like/dislike), then two prediction tasks will be present. TMF is a mixed TL technique of feature-based and instance-based transfer, and a mixed TL algorithm of collective and integrative factorization. Here, it introduces integrative factorization method into collective factorization, to lessen the interdependency between two factorization tasks, and enable more effective knowledge transfer between **X** and **Y**.

$$\hat{r}_{ui} = U_u V_i^T + A_u V_i^T, W_u \leftrightarrow U_u$$

Here, the first term $U_u V_i^T$ is from collective factorization, and the second term $A_u V_i^T$ is from integrative factorization, and the last term $W_u \leftrightarrow U_u$ indicates the interactions between user profiles W_u and U_u. Here, $A_u = \delta_P w_p \tilde{P}_u + \delta_N w_n \tilde{N}_u$ is the fused virtual user profile for user u with boolean variables δ_P, $\delta_N \in \{0, 1\}$ and weights $w_p, w_n \geq 1$. Here $\tilde{P}_u$, are normalized user profiles constructed from user u's liked items' features, and $\tilde{N}_u$ are constructed from disliked items' latent features. This is a feature-based approach as latent features are getting transferred and also an instance-based approach as normalized user profiles are getting constructed from the liked and disliked items (i.e., re-weighting the data).

Tags as bridges between domains (TagCDCF) (77): For instance, suppose that two different domains like a movie domain and book domain have completely different users and items. However, there exists some tags which are common to both. These shared tags are used to link different domains. A constraint on tag-based similarities between user and item pairs across domains is introduced into matrix factorization. Based on tags shared between domains, construct user-user, item-item similarity matrices. Compute tag-induced cross-domain similarities using cosine similarity measure, after extracting shared tag set between domains. As we are trying to see the common tags between two domains, this is a model-based approach.

Method	What to Transfer	How to Transfer	TL approach	Transfer strategies (via)	Users/ Items/ User-item feature correspondence
CBT (35) (NMF)	Codebook	Adaptive, (tri-fact)	Model-based	Constraint	Share cluster level rating pattern
TALMUD (101) (from multiple sources)	Codebook	Collective	Model-based	Constraint (some relate dness between domains)	Share cluster-level rating pattern
CST (58) (PMF)	Latent features (principle coordinates)	Adaptive (tri-fact)	Feature-based	Regularization	Share common Users and Items, U and V are similar in both domains
iTCF (56) (PMF)	Latent features and predictability	Collective (bi-fact)	Feature-based	Constraint	Users and Items are same. It shares U, and V is same
TIF (59)	Latent features	Collective + Integrative	Feature-based + Instance-based	Constraint	Users and Items are same V-shared
TagCDCF (77) (PMF)	Correlation of tags	Integrative (integrating tag similarities , bi-fact),	Model-based	Constraint (similarities between tags)	Users and Items are not same, but share common tags
TMT (13) (NMF)	Co-occurrence matirx (of tags)	Integrative (tri-fact)	Model-based	Constraint (tags)	Users and Items are not same, but share common tags
TagiCoFi (98) (PMF)	Tags (target user similarity)	Integrative (bi-fact)	Model-based	Regularization	Constrains U to be similar based on tags
TBT (76)	Latent features + Similarity graphs	Collective	Feature-based	Regularization	Y and X_1 share users Y and X_2 share items
TagGSVD++ (14)	Implicit feedback + Item attributes + Tags	Integrative	Instance-based	Prediction	Shares same tags
CMF (PMF) (78)	Latent features	Collective (multi task, bi-fact)	Feature-based	Constraint (V=V')	Same item latent feature matrix
TCF (55) (PMF)	Latent features	Collective (tri-fact)	Feature-based	Constraint	Users and Items are same. U and V are same in both domains
MINDTL (20) (from multiple sources)	Codebook from multiple sources	Collective	Model-based	Constraint (relatedness between domains)	Share part of cluster-level rating pattern of different sources
TRACER (47) (from multiple sources)	Codebook	Collective	Model-based	Constraint (relatedness between domains) & Regularization	Share cluster-level rating pattern
MMMF$_{TL}$ (97)	Latent features	Collective (multi task, bi-fact)	Feature-based	Regularization	Shares some users and items

Table 2.2: Brief survey on different transfer learning approaches related to cross-domain collaborative filtering in alleviating the data-sparsity issue

Tag Matrix Transfer (TMT) (13): Tags are interrelated in different domains. The main assumption of TMT is that, auxiliary and target domains should share common tags even though they doesn't have same users/items. Calculate the co-occurrence distribution of all tags and construct the tag co-occurrence matrix (T) to capture the user

behavior patterns from different rating domains. Based on these, learn U, V and make prediction via $U_k T V_k$. TMT is a model-based approach, as we find the tags between source and target as shared parameters.

The categorization of all the above discussed methods is briefly shown in Table 2.2, in which the first column naming 'method' consists of names of different methods related to transfer learning for cross-domain collaborative filtering. What knowledge is to be transferred from source domain to target domain is given in the second column (What to transfer). Once the information or knowledge which is to be transferred is decided, the same knowledge is to be transferred to the target domain using different algorithmic styles which addresses the question 'How to transfer' and is given in the third column of the table. The fourth column (TL approach) gives the transfer learning approach that is followed in the particular method. For every method, there exist some strategy through which the data can be transferred and is given in the fifth column. The source and the target domains share some of the entities or latent features or rating pattern and is given in the last column.

2.8 Summary

In this chapter, we have discussed in detail about matrix factorization, and various matrix factorization models. We have also discussed in detail about transfer learning technique for recommender systems i.e., cross-domain recommender systems, and also provided the literature related to various works in cross-domain recommender systems.

CHAPTER 3

A Maximum Margin Matrix Factorization based Approach for Cross-Domain Recommender Systems

In this chapter, we discuss our proposed model for transfer learning in collaborative filtering for addressing the data sparsity issue which is one of the main drawbacks of collaborative filtering. In our proposed method, we learn a cluster level rating pattern (*codebook*) of the source domain and transfer the learnt codebook to the target domain. Transferring of codebook and finding the predicted rating matrix of target domain is done in a novel way by introducing a softness constraint into the optimization function.

In the proposed approach, maximum margin matrix factorization and clustering play a vital role while constructing the codebook. Initially, maximum margin matrix factorization (MMMF) is applied on source domain rating matrix in order to get the user and item latent feature matrices of source domain. The obtained latent feature matrices are clustered and combined to get a cluster-level rating pattern called *codebook*. Having done that, codebook transfer is used to transfer the learnt knowledge (codebook) of the source domain to the target domain and improve the prediction accuracy of the target domain which is very sparse. Transferring of codebook and finding the predicted target rating matrix is done in two different ways. In one method the target user and item membership matrices form hard membership in which a user/item can belong to a single cluster. In the second method, a softness constraint is introduced into the op-

timization function where a single user or single item belongs to multiple clusters with some weights. Various types of clustering techniques have been made use for constructing the codebook and the resultant codebook is transferred to the target domain in two different ways as discussed above.

The rest of the chapter is organized as follows: Section 3.1 gives a brief introduction about the motivation behind the use of transfer learning in the area of recommender systems. Some of the existing works related to cross-domain recommender systems are described in section 3.2. The proposed approach is given in section 3.3 and the experimental results are shown in section 3.4. Finally, we summarize our work in section 3.5.

3.1 Introduction

As discussed in Chapter 1, recommender systems usually employ techniques like Collaborative Filtering (CF) where the recommendation for a user (target user) is done by utilizing the observed preferences of other users with similar tastes as that of the target user. The other technique being Content-based (CB) recommends the items based on the items that the user has liked previously. One of the widely used CF techniques is Matrix factorization(MF) and among the different MF techniques maximum margin matrix factorization (MMMF) (81) is the most popular. However, this method can only utilize the data from a single domain and cannot take into account user-item interaction from other domains. Moreover, most CF-based recommender systems perform poorly when there are very few ratings, which is known as the data sparsity problem. To address this data sparsity, transfer learning (TL) methods have emerged. The idea behind transfer learning (53) is to extract relevant knowledge from one domain (source) and transfer to other domain (target) so as to build a predictive model across different domains. Researchers also refer this problem as Cross-Domain Collaborative Filtering. A more detailed description about Transfer Learning is discussed in Chapter 2.

In the case of recommender systems, for successful knowledge transfer, TL has to address two critical problems 1) Knowledge transfer when two domains have aligned users or items and 2) Knowledge transfer when the domains have no aligned users or items. The scenario of the first problem often doesn't exist in real-time as the

users/items in one system may not be present in the other systems. Moreover, even though if the correspondence exists, it is often expensive to map, as the users/items may have different names in different systems. The second problem is very difficult and we use a representative method to solve this issue using CBT (CodeBook Transfer) (35). CodeBook Transfer (CBT) (35) is a transfer learning approach that addresses this problem and a codebook generated from the rich information in one domain (dense) is used to predict the missing rating in another domain (sparse). Codebook essentially captures the rating patterns in a summarized form, and it is hypothesized that the summarized rating pattern is, in some sense, invariant across domains. In this chapter, We propose a novel model of CBT based collaborative filtering. As the first step, we propose a codebook construction technique based on Maximum Margin Matrix Factorization, which has natural reasons to be used in the context and has not been explored earlier. For the purpose of transfer learning using codebook, we adopt a soft membership scheme and formulate it as a non-linear optimization problem. Unlike the earlier scheme of hard membership which can be solved by a discreet optimization problem with applicable greedy method, the soft scheme requires a non-linear optimization which has higher computational complexity. However, we show that our formulation has a closed-form solution. The proposed method outperforms all major methods on codebook-based transfer learning for collaborative filtering. Our experimental analysis is exhaustive and rigorous in the sense that we provide experimental results for different alternatives. For instance, for clustering of latent factors, we use k-Means (22), DBSCAN (Density-Based Spatial Clustering of Applications with Noise) (12) and Spectral (48) clustering. k-means clustering identifies 'k' (input given by user) number of centroids and assigns each data point to the nearest cluster. It recalculates the centroid and reassigns the points to the nearest centroid, and the process continues iteratively until the cluster centroids stabilize. Spectral clustering clusters the data points by connectedness rather than using distance measure from centroids. DBSCAN clusters the data based on the density of the data points. If there are at least $minPts$ number of points within a radius of eps to the point, then it considers all those points to be part of the same cluster. We also report results of experimenting on prediction for different levels of sparsity of input matrix. More than that, we report experimental results with soft membership as well as hard membership.

3.2 Related work

As we have seen in the previous chapter, MMMF has been a very successful collaborative filtering technique in predicting ratings for unobserved entries even with a very small number of observed entries. In a sense, with a little information it can complete a sparse matrix to a satisfactory level of accuracy. But as sparsity increases, the accuracy of prediction expectedly falls. In such a situation, researchers investigate the possibility of making use of information learnt in another context. Cross-domain collaborative filtering techniques like *transfer learning* (53; 54) are proposed in the last decade as a potential solution to mitigate this problem. The main aim of transfer learning is to transfer the knowledge from the dense source domain to the sparse target domain. The problem can be stated formally as follows. Given a dense rating matrix in a source domain as $X_{m' \times n'}$, and a sparse rating matrix in target domain ($Y_{m \times n}$), the aim is to predict unknown entries of Y by making use of information available in X. One of the major questions that remains unanswered in this context is to determine whether the two domains in hand are compatible for knowledge transfer. Most of the research on transfer learning for CF makes certain assumptions on domains to apply transfer learning strategy. These assumptions include a common subset of items or users, similar attributes of users or items. The objective is to determine a sort of correspondence between two domains.

In (58) a coordinate system transfer method is proposed in which the latent features of users and items of source domain are learnt and adapted to a target domain. However, it requires that there exists either common users or items between the two domains. In (35), the authors focus on capturing the group level behavior of users on items. The main assumption is that, though the users/items are different across systems, the clusters (groups - based on age, interest etc.) of them behave similarly. So, in (35) co-clustering is applied on a separate *auxiliary* rating matrix to directly get the cluster-level rating pattern which is called as codebook. This cluster level rating pattern is then expanded to get the user and item membership matrices of the target domain in which the user/item can belong to a single cluster. Our approach differs from that of (35) in the sense that we do not use a separate dense *auxiliary* rating matrix to construct the rating pattern. In addition to that, we also extended the method as outlined in (35) by introducing the

concept of soft membership into the target user/item membership matrices.

One of the major issues involved in developing transfer learning techniques for recommendation purpose is to establish a link between the domains that are involved, so that knowledge transfer from the *source* domain to the *target* recommendation domain can take place. Domains can be linked and the transfer can happen explicitly via inter-domain similarities, common item attributes, etc. The transfer can happen also implicitly via shared user latent features or item latent features or by rating patterns which can be transferred between the domains. In (7), a framework was proposed where the items that are relevant in the source domain are chosen based on the common attributes they share with the target domain (user interested domain). In this way the inter-domain links were built via the common item attributes, however there was no overlap of users/items required between the domains.

On the other hand the transfer of knowledge by the shared latent features (of users/items) is addressed in (58) in which the latent features of users and items of source domain are learnt and adapted to a target domain by integrating the features into the factorization of target rating matrix via regularization. However, it requires either common users or items between the two domains. In (55), the latent features of source and target domains are shared in a collective way. Here, rather than learning the latent features from source and utilizing in the target, a method that simultaneously learns the latent features of both domains is proposed. The shared latent space is constructed via matrix tri-factorization. It also requires the users and items of both domains to be identical.

There are other set of methods in which rating patterns are analysed and transferred rather than latent features. Code Book Transfer (CBT) (35) is one such method in which cluster-level rating-patterns are captured. There exist latent correlations between ratings of groups of users and the groups of items, which is referred as rating pattern. In this approach, the rating matrix of source domain is analyzed to extract a codebook which is used in the target domain for prediction. Users and items, irrespective of domains, group together in such a fashion that the rating behavior of these groups remain invariant across domains. To apply the CBT concept in CF, a strategy must be devised to generate a codebook by using the rating matrix of the source domain and subsequently to devise a technique to use the codebook so generated in target domain for prediction of missing behavior. The rating matrix in source domain is assumed to be a full matrix

in (35). When the matrix is not full, the unobserved entries are filled in by a smoothing technique. Then the rows and columns of this rating matrix are grouped by a process of simultaneous clustering or co-clustering (35). Codebook matrix is generated wherein it has a row for each user group, a column for each item group, and a single rating for a pair of a user-group and an item-group. The rating for a pair of user-group and item-group is the mean rating of users and items. While using the codebook in target domain, the aim is to identify users (or, items) in the target domain with the best-fit user-group (or, item-group) of the codebook. It can be formulated as an optimization problem as follows (Eq. 3.1).

$$\min_{F_1 \in \{0,1\}^{m \times k_1}, F_2 \in \{0,1\}^{n \times k_2}} ||[Y - F_1 C F_2^T] \odot W||_F^2 \quad \text{s.t.,} F_1\mathbf{1} = 1,\ F_2\mathbf{1} = 1. \tag{3.1}$$

where $Y_{m \times n}$ is the rating matrix in the target domain, $C_{k_1 \times k_2}$ is codebook, $F_{1_{m \times k_1}}$ and $F_{2_{n \times k_2}}$ are the cluster membership matrices of users to user-group in F_1 and items to item-group in F_2, respectively. W is the indicator matrix of size $m \times n$ in which the value is 1 if the rating exists in the original rating matrix and 0 otherwise. In (35), it is assumed that a user (or an item) can belong to exactly one user-group (or, item-group). There are several extensions of CBT based transfer learning proposed in (35). In (47), the linear combination of codebooks extracted from multiple domains are used and the coefficient of linear combination is determined through an optimization process. In (20; 19), the assumption of fully dense matrix of the source domain is relaxed.

One work that comes close to ours is that of (23) in which an alternative method of generating codebook is proposed. It avoids the pre-processing (smoothing) as well as co-clustering of source matrix. From the source rating matrix, the latent factors of users U_s and items V_s are generated by a process of matrix factorization. Having done that, the latent factor-vectors are clustered separately to obtain user-group (U_c) and item-group (V_c). These mean latent vectors, U_c and V_c, of the groups are multiplied to get the codebook, C. In the original proposal (23), the codebook is used for a single domain whereas in our method we use it for cross-domain. i.e., in (23) by using the constructed codebook, the authors reconstructed the original matrix by using codebook transfer method. In our method, we have utilised the idea of constructing codebook from (23) and transfer the constructed codebook across different domains using codebook transfer method. Also, to the best of our knowledge applying MMMF in this scenario is the

novel approach. When applying MMMF there is also a threshold matrix (Θ). The usage of Θ while constructing codebook by applying MMMF is discussed in section-3.3. Once the codebook is constructed, it can be transferred to the other domain in two ways - one is to find the hard membership matrices of users/items of target domain and the other one being soft membership of users/items of target domain. Hard membership problem was addressed in (35) where a user/item can belong to exactly one cluster. But there can be scenarios in which the user/item can belong to multiple clusters with some weights which is addressed in this thesis by introducing the softness constraint into the optimization. Fig. 3.1, gives different ways of using rating information to construct codebook (C).

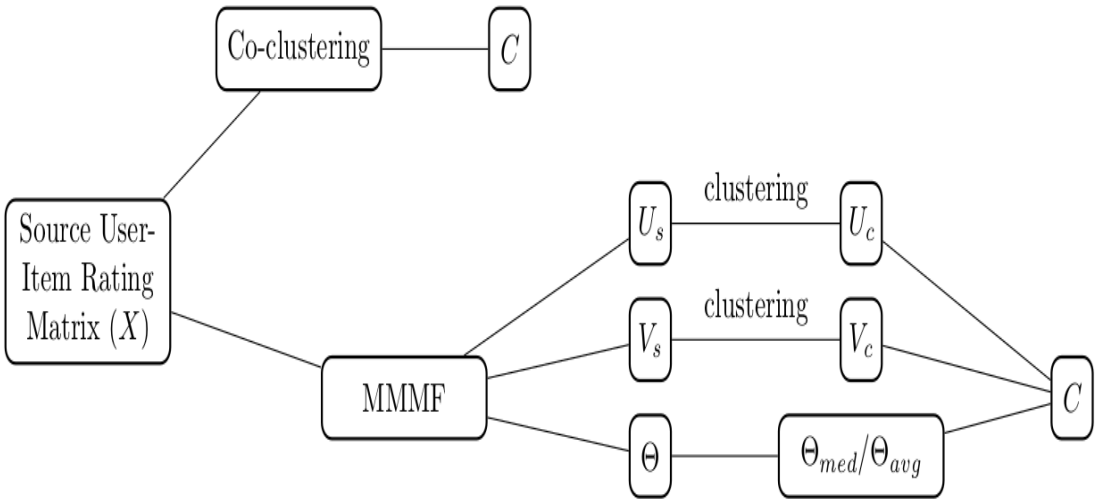


Figure 3.1: Block diagram showing construction of codebook (C) from the rating matrix

3.3 Proposed Approach

In MMMF, as stated in section 2.2.4 of Chapter 2, set of items can be viewed as set of points in a low-dimensional embedded space and set of users as decision hyperplanes in the same space. For sake of convenience, the thresholds are defined with respect to user-rating combinations. An equivalent interpretation would be to represent the set of users as points and set of items as decision hyperplanes. We take the former interpretation as proposed in the original proposal of MMMF. With this interpretation, the clustering of latent factors can be viewed as clustering of embedded points for items and grouping of embedded decision planes for users. Thus in the embedding space, a

configuration evolves describing a set of clusters, represented by the cluster-means and a set of decision hyperplanes which separate the space into r regions with the help of $(r-1)$ thresholds. This configuration is, in a sense, abstract representation of a rating pattern for the source domain. At this stage, for the purpose of transfer learning, we presume that the configuration so evolved for the source domain can also be valid for the target domain. Hence, in the next stage, it is examined whether by some embedding process the items and users of the target domain can be mapped to the embedding space conforming to the configuration. A mapping conforms to the configuration if the target-users and target-items exhibit similar rating behavior as represented by the configuration. Thus, codebook generation using clustering of latent factors obtained by MMMF-based factorization has a natural justification. The present study is motivated by this observation and this aspect has not been explored earlier.

The process of prediction using codebook, in CBT, essentially determines the membership of users and items to the user-groups and item-groups. The membership matrices F_1 or F_2 can be binary with every row having just one entry as 1 and other entries as 0. This is to ensure that an embedded point is a member of one group and no other. Hard membership problem was addressed in (35). It is pertinent to consider soft membership by assuming these matrices with entries between 0 and 1.

The step-by-step description of a generic method in this context has the following essential steps - First to factorize the rating matrix of the source domain to obtain the latent factor-vector for each user (u_s) as well as for each item (v_s). The latent factor-vectors ($U_{s_{m' \times \ell}}$, $V_{s_{n' \times \ell}}$) are grouped by any clustering algorithm with some distance metric. The representative latent factor-vector for each cluster is determined. Such cluster-level user latent factors ($U_{c_{k_1 \times \ell}}$) and item latent factors ($V_{c_{k_2 \times \ell}}$) are multiplied to obtain the codebook (C). Using the codebook so computed, optimal membership matrices are determined which minimizes the squared error.

During the clustering process, we use Cosine similarity for user latent matrix (U_s) because of considering users as hyperplanes as discussed above. Cosine similarity is generally used to measure the similarity between vectors. We use Euclidean distance for item latent matrix (V_s) as items are viewed as points in the embedded space as mentioned in the foregoing session. The mean vector for each cluster is taken as the cluster-level latent vector. In order to determine a threshold value corresponding to each

rating value for each mean user-latent factor, we have tried two different ways in our experiments; the median of the thresholds (rows of Θ), and by averaging the thresholds (rows of Θ) of those users whoever is falling into the same cluster and the resultant matrix is named as Θ_{med} or Θ_{avg} (having dimension $k_1 \times r-1$).

The construction of codebook, at this stage, consists of the following steps. We multiply the U_c and V_c^T matrices to get real-valued matrix which is then mapped to a discrete rating matrix (using Θ_c) to get the cluster level rating pattern ($C_{k_1 \times k_2}$). Mapping is done in such a way that, the entries of the matrix $U_c \times V_c^T$ and the values of Θ_c are compared. If the entry (say elements of first row ($u_{c_{1.}}$)) in $U_c \times V_c^T$ is less than $\theta_{c_{11}}$ then the rating is considered as 1. If the value is in between $\theta_{c_{11}}$ and $\theta_{c_{12}}$ we rate it as 2 and similarly if the value of the entry falls between $\theta_{c_{12}}$ and $\theta_{c_{13}}$ we rate it as 3. If the value is in the range of $\theta_{c_{13}}$ and $\theta_{c_{14}}$, we make it as 4, and finally if the value is greater than $\theta_{c_{14}}$ we rate it as 5. Here the main assumption is that there exists some implicit correspondence between the source domain user/item clusters and target domain user/item clusters. Our assumption is that the correspondence between source and target is through codebook which is a cluster level rating pattern (C).

Having talked about the construction of codebook in the source domain as mentioned above, the next step is to transfer codebook to the target domain. At this stage, it is to determine the cluster membership of each user and item in the target domain based on the constructed codebook. So, once the rating pattern is formed, we try to minimize the objective function (quadratic loss) (3.2) which expands the cluster level rating pattern so as to get the user and item membership matrices $F_{1_{m \times k_1}}$, $F_{2_{n \times k_2}}$ of the target domain. Expansion is done by duplicating the rows and columns of the codebook. If there are some users/items in the target domain which behaves like that of the i^{th} user/item cluster of the source domain, then the i^{th} row/column of the codebook gets duplicated. Thereafter the predicted matrix can be obtained using Eq. (3.3) as outlined in Algorithm 1.

$$\min_{F_1 \in \{0,1\}^{m \times k_1}, F_2 \in \{0,1\}^{n \times k_2}} ||[Y - F_1 C F_2^T] \odot W||_F^2 \quad \text{s.t.,} F_1\mathbf{1} = \mathbf{1},\, F_2\mathbf{1} = \mathbf{1}. \tag{3.2}$$

$$\tilde{Y} = W \odot Y + [1 - W] \odot [F_1 C F_2^T], \tag{3.3}$$

Algorithm 1: MMMF combined with clustering

1: **Input:** A $m' \times n'$ source rating matrix X and a $m \times n$ sparse target rating matrix Y with y_{ij} known for $(i, j) \in \omega$
2: **Output:** y_{ij} for $(i, j) \notin \omega$
3: Factorize X by MMMF to get U_s, V_s and Θ_s by solving the optimization problem (2.8) given in Chapter 2.
4: Cluster the rows of U_s with Cosine distance and rows of V_s with Euclidean metrics. Resultant matrices U_c, V_c and Θ_c represent the cluster level matrices and threshold parameters.
5: By comparing $(U_c \times V_c^T)_{ij}$ with $(\theta_c)_{ir}$ for different ratings r, generate cluster-level rating pattern (C).
6: Solve the optimization problem (3.2) to find optimal F_1, F_2.
7: Get the predicted target rating matrix $\hat{Y}$ with optimal F_1 and F_2 using (3.3).

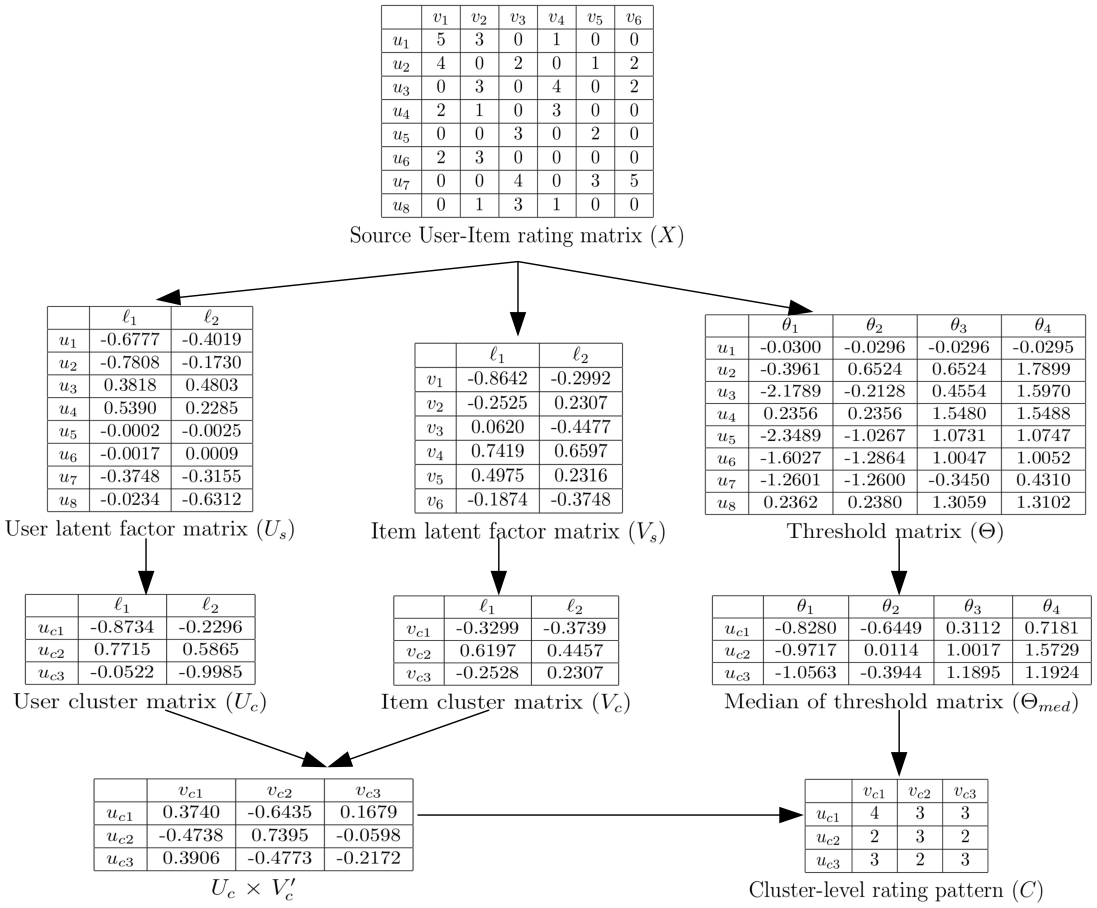


Figure 3.2: Construction of cluster-level rating pattern using source rating data

where W is the indicator matrix of size $m \times n$ in which the value is 1 if the rating exists in the original rating matrix and 0 otherwise. W ensures that the error is calculated based on observed ratings only and, $\odot$ denotes the Hadamard product (element wise product). F_1 and F_2 are binary matrices, in which the value 1 (best cluster indicator) indicates whether a user or item belongs to a particular cluster and $F_1\mathbf{1} = \mathbf{1}$, $F_2\mathbf{1} = \mathbf{1}$ ensures that each user or item belongs to only one cluster (Hard membership). For

example, if we consider [0 0 1] as one of the rows of F_1, there are three user clusters and we can say that the user belongs to the third cluster, as the entry is filled with 1 whereas the remaining entries as 0s. In a similar manner one can say about the item cluster membership by using F_2. The solution to the optimization problem (Eq. 3.2) that relates the source and the target domains is NP-hard. Smaller value of Eq. (3.2) indicates a better rating pattern between source and target while larger values indicate weak correspondence, which may result in negative transfer (68). To get the minimum local solution, Alternating Least Squares (ALS) technique is used. ALS monotonically decreases Eq. (3.2), by updating F_1 and F_2 alternatively. This has been demonstrated in (35), where F_1 and F_2 are getting updated by changing the best cluster indices in each iteration based on the minimum value of the objective function. F_1 is getting updated by fixing F_2 and then by fixing F_1, F_2 gets updated. By updating F_1 and F_2 alternatively, the value of the Eq. (3.2) gets decreased monotonically and converges to a local minimum. Once we get F_1, F_2 by solving the optimization function (3.2), we construct the predicted target matrix using Eq. (3.3) as shown in Fig. 3.3. In Fig. 3.3, Y is the original target domain rating matrix, $\tilde{Y}$ is the predicted target rating matrix in which the observed entries are retained and the missing values are predicted using $F_1 C F_2^T$. Consider Fig. 3.2, where the source rating matrix - X (presented at level-1

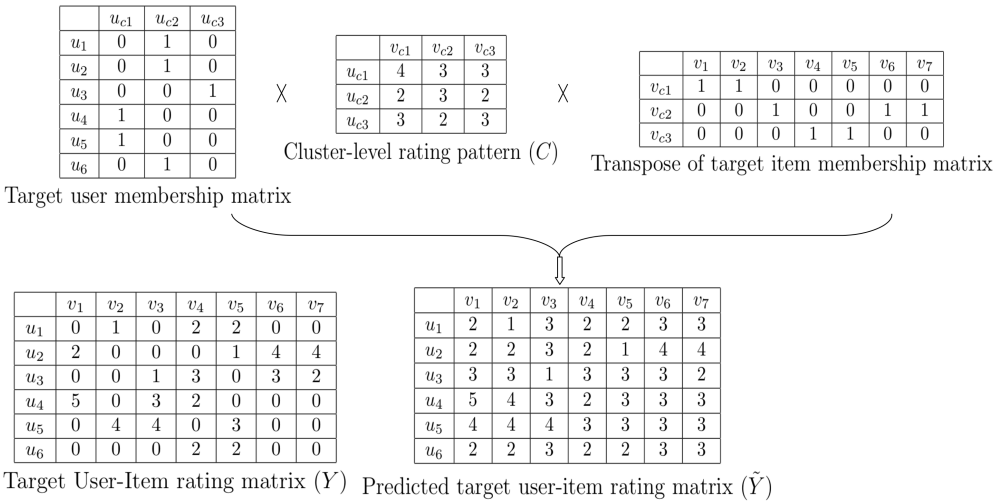


Figure 3.3: Approximation of target rating matrix using cluster-level rating pattern.

containing 8 users and 6 items) is factorized into user latent factor matrix (U_s) and item latent factor matrix (V_s) as shown in level-2. The number of latent features we consider here is 2. Besides these, a threshold matrix Θ for the users also gets learned. In our example the maximum rating (r) is 5 and so the size of Θ is 8×4. Clustering technique (with number of clusters as 3 - both k_1, k_2) is applied on U_s and V_s to get user and

item cluster matrices (U_c, V_c) which are at level-3. Also there is a Θ_{med} (Θ_c) threshold matrix which is obtained by taking the median of the thresholds of those users falling into the same cluster. In this example, users u_1, u_2, u_6, u_7 belongs to cluster-1 and u_3, u_4 belongs to cluster-2 and u_5, u_8 falls under cluster-3. So, for the first user cluster (u_{c1}), the entry in the median of threshold matrix (Θ_{med}) is the median of four rows (u_1, u_2, u_6, u_7) of Θ. For u_{c2}, it is the median of the values of u_3 and u_4 of Θ, and for u_{c3} the entry in the Θ_{med} matrix is the median of rows u_5 and u_8 of Θ matrix. Finally, at level-4 the cluster matrices are multiplied and the resultant is mapped using Θ_{med} to get cluster-level rating pattern or codebook (C) which is to be used in the target domain. In matrix C, the first entry (value 4) indicates that the users who fall into user-cluster-1 (u_{c1}) has given rating 4 to the items in item-cluster-1 (v_{c1}). Similar explanation holds for all other entries of C.

In the aforesaid scenario (Eq. 3.2), the user/item belongs to exactly one cluster which is a hard membership, whereas in real-world scenarios a user might belong to multiple clusters and for a single item there may exist many attributes. So, a user/item can belong to multiple clusters at a time with different memberships. Membership values indicate to what extent the user/item is associated with each cluster. This motivated us to introduce soft membership to the clustering. i.e., we extended the discussed method (Eq. 3.2) by introducing softness into the target membership matrices. In this case, instead of $\{0, 1\}$ values in target membership matrices we can have some real values (membership values or weights) between 0 and 1. Eq. 3.4 is the new objective function in which the soft membership is introduced.

$$\min_{F_1 \geq 0, F_2 \geq 0} ||[Y - F_1 C F_2^T] \odot W||_F^2 + \lambda_1 ||F_1||_1 + \lambda_2 ||F_2||_1 \tag{3.4}$$

Here, F_1 and F_2 are user membership and item membership matrices. Multiplicative update rules (32) (26) can be used in order to get F_1 and F_2. By considering the techniques proposed in (26) (25), one can solve Eq. (3.4) by using alternating minimization scheme in which the factor matrices are optimized one by one iteratively while fixing the other. We optimize (3.4) w.r.t F_1 and F_2 as follows. First, we solve Eq. (3.5) to get F_1 by fixing F_2.

$$\min_{F_1 \geq 0} ||[Y^T - F_2 C^T F_1^T] \odot W^T||_F^2 + \lambda_1 ||F_1||_1 \tag{3.5}$$

By applying the multiplicative update rule to (3.5) we get,

$$F_{1_{ik_1}} \leftarrow F_{1_{ik_1}} \frac{((W \odot Y)F_2C^T)_{ik_1}}{((W \odot (F_1CF_2^T))F_2C^T)_{ik_1} + \lambda_1||F_1||_1} \tag{3.6}$$

Once we get F_1, by fixing F_1 we optimize Eq. (3.7) and get F_2. Eq. (3.8) gives the multiplicative update rule for F_2.

$$\min_{F_2 \geq 0} |||[Y - F_1CF_2^T] \odot W||_F^2 + \lambda_2||F_2||_1, \tag{3.7}$$

$$F_{2_{jk_2}} \leftarrow F_{2_{jk_2}} \frac{((W^T \odot X^T)F_1C)_{jk_2}}{((W^T \odot (F_2C^TF_1^T))F_1C)_{jk_2} + \lambda_2||F_2||_1} \tag{3.8}$$

To avoid division by zero, one can add a small $\epsilon > 0$ (say, $0.1E-8$) to the denominator. Once the above equations are solved, we get the membership matrices F_1 and F_2. We are getting the matrices F_1, F_2 by transferring the cluster level rating pattern (C) from the source domain. However C is obtained by multiplying the user cluster matrix and item cluster matrix of source, which indirectly says that the user cluster centroids and item cluster centroids can also be transferred. So, once we get the membership matrices, we multiply the membership weights with the centroids of latent factor matrices (U_c and V_c), so as to get the target latent feature matrices (U_t and V_t). We also multiply

Algorithm 2: MMMF combined with soft clustering

1: **Input:** A $m' \times n'$ source rating matrix X and a $m \times n$ sparse target rating matrix Y with y_{ij} known for $(i,j) \in \omega$
2: **Output:** y_{ij} for $(i,j) \notin \omega$
3: Factorize X by MMMF to get U_s, V_s and Θ_s by solving the optimization problem (2.8) given in Chapter 2.
4: Cluster rows of U_s with cosine distance and rows of V_s with Euclidean metrics. U_c, V_c and Θ_c represent the cluster level matrices and threshold parameters.
5: By comparing $(U_c \times V_c^T)_{ij}$ with $(\Theta_c)_{ir}$ for different ratings r, get cluster level rating pattern (C).
6: Solve the optimization problem (3.4) to find optimal membership matrices F_1, F_2.
7: Multiply the weights of F_1 and F_2 with U_c and V_c to get U_t, V_t.
8: Multiply the weights of F_1 with Θ_c to get the weighted threshold matrix.
9: Product of U_t, V_t^T by mapping with weighted threshold matrix gives the target predicted matrix ($\hat{Y}$).

the threshold matrix (Θ_{med} or Θ_{avg}) with the corresponding weights of the users (F_1). The resultant target predicted matrix can be obtained by multiplying the target matrices (U_t, V_t) and mapping it to the discrete values (ratings) using the threshold matrix that we

obtained after multiplying with the weights of users. This is illustrated in Algorithm-2.

We also analyse the complexity of the proposed transfer learning method which uses MMMF to learn the codebook from the source domain, and finds the user and item membership matrices of the target domain. The time complexity of the proposed method comprises of four components - learning latent factors of source, applying clustering technique to learn the cluster level latent factors, codebook construction, learning user and item membership matrices of target domain. In the first component in which MMMF is applied, in every gradient iteration, MMMF requires $3m'n'\ell(r-1)$(30) for the update of U_s, V_s, Θ matrices. Let the total number of iterations be t_1, then the overall computational cost required by MMMF is $3t_1m'n'\ell(r-1)$. Once the latent features are learnt, the cost for performing k-means clustering (second component) to get the cluster level latent factors (U_c and V_c) of U_s and V_s are $m'\ell t_2$ and $n'\ell t_2$ respectively, where t_2 is the maximum number of iterations required to perform k-means clustering. To construct codebook, which is the third component, the computational cost required is k_1k_2l. Once the codebook is formed, we need to learn the user and item membership matrices (F_1, F_2) of target domain which is the fourth component. The computational cost required to update F_1 and F_2 is $t_3(2mnk_1+3nk_1k_2+2mn+2mk_1)$ and $t_3(2mnk_2+3mk_1k_2+2mn+2nk_2)$, where t_3 is the number of iterations. Hence the overall computational cost required by the proposed method is $(3t_1m'n'\ell(r-1)+m'\ell t_2+n'\ell t_2+t_3(2mnk_1+3nk_1k_2+2mn+2mk_1)+t_3(2mnk_2+3mk_1k_2+2mn+2nk_2))$, i.e., $\mathcal{O}(t_1m'n'\ell(r-1)+t_3mnk_1+t_3mnk_2)$. On the other hand, when MMMF is applied directly on the target data, in every gradient iteration, it requires $3mn\ell(r-1)$ for the update of U_t, V_t, Θ matrices. Let the total number of iterations be t_1, then the overall computational cost required by MMMF is $3t_1mn\ell(r-1)$, i.e., $\mathcal{O}(t_1mn\ell(r-1))$. From this we say that although the accuracy of the proposed method is high, the computational cost is slightly higher than that of MMMF. So, reducing the computation cost is part of our future work.

3.4 Experimental Analysis

The first set of datasets used in our experiments are `MovieLens-1M`[1] as *source* dataset (6040 users and 3952 movies) and `Book-Crossing`[2] as *target* dataset. From the target data (`Book-Crossing`), we consider the entities that have at least 20 ratings and discard those entities having more than 50 ratings. As a result we get the books data having 1658 users and 1362 books. In MovieLens each user has ratings within the range of 1-5, whereas in Book-Crossing the range is 1-10, and we have scaled it to 1-5 by taking the ceil value of half of the original rating. We have also experimented our methods on Synthetic datasets. We followed the method given in (30) to generate the Synthetic dataset and generated the source dataset of size 1800 $\times$ 1500 by considering 30% density. On the other side, we have generated target dataset of size 1800 $\times$ 1500 with 10% density, as the assumption is to have a sparse target data. In all the experiments, 80% of total rating data is taken for training, and the rest 20% is used for testing.

3.4.1 Evaluation Metrics

Computing prediction accuracy is one of the major criterion to evaluate rating-oriented collaborative filtering algorithms. The most commonly used metrics to measure the prediction accuracy are Root Mean Square Error (RMSE) (Eq. (5.5)) and Mean Absolute Error (MAE) (Eq. (5.6)), which depends on the difference between actual and predicted rating. We evaluate our algorithms using these metrics (RMSE and MAE), where smaller the values of these, better the performance. The values reported in the tables are the average of five runs.

$$RMSE = \sqrt{\sum_{(i,j)\epsilon\omega} \frac{(y_{ij} - \hat{y}_{ij})^2}{|\omega|}} \tag{3.9}$$

$$MAE = \sum_{(i,j)\epsilon\omega} \frac{|(y_{ij} - \hat{y}_{ij})|}{|\omega|} \tag{3.10}$$

[1] https://grouplens.org/datasets/movielens/
[2] https://grouplens.org/datasets/book-crossing/

where y_{ij} is the original rating and $\hat{y}_{ij}$ is the predicted rating, $|\omega|$ is the number of test ratings.

3.4.2 Methods for Comparison

To evaluate the performance of our methods, we consider the following approaches.

- **MMMF (81; 65):** It is one of the popular matrix factorization methods used in collaborative filtering. MMMF predicts the missing values of the rating matrix by determining the low-norm latent feature matrices of users and items. This works for single domain and we have applied this method on the target domain in our experiments.
- **MINDTL (20):** Considers the data from multiple incomplete source domains and constructs the codebooks for the domains. These learned codebooks get linearly combined and get transferred to the target domain to approximate the target rating matrix. In our experiments, we have considered only a single source domain.
- **TRACER (101):** Considers the data from multiple source domains and learns the corresponding predicted matrices and transfers the knowledge. While transferring the learnt knowledge, it uses consensus regularization which forces all of the predicted results to be similar (say, majority value). Also, this algorithm learns and transfers the knowledge at the same time. In our experiments, we have considered only a single source domain and as a result there will be no consensus regularization, but learning and transferring the knowledge happens at the same time. We thank the authors for providing the code[3] online.
- **MMMF$_{TL}$ (97):** In this paper, a framework of different factorization techniques with transfer learning has been proposed and it is concluded that MMMF$_{TL}$ is the best choice as far as prediction accuracy is concerned. This method applies MMMF on the target domain and then iteratively selects some entities (users/items) based on some certainty measure, and finds the corresponding entities in the source domain. By utilizing the similarities between the selected entities of the source domain, it constrains the similarities of the corresponding target entities.
- **CBT** (35): This method finds the codebook of the source data by performing co-clustering on users and items. Unlike in (35), we consider the whole source data for constructing the codebook. The learned codebook then gets transferred to the target domain by minimizing Eq. (3.2) so as to predict the target ratings.

For simplicity, we use the following notations for our methods,

- **KHTL** – **k**-means clustering is applied while constructing codebook on source, and **H**ard membership **T**ransfer **L**earning is used on target.

[3] https://github.com/hezi73/TRACER

- **SHTL** – **S**pectral clustering is applied in codebook construction process, and **H**ard membership **T**ransfer **L**earning is used on target.
- **DHTL** – **D**BSCAN clustering technique is used in codebook construction procedure, and **H**ard membership **T**ransfer **L**earning is used on target.
- **KSTL** – **k**-means clustering technique is applied during the codebook construction process, and **S**oft membership **T**ransfer **L**earning is used on target.
- **SSTL** – **S**pectral clustering is applied while constructing the codebook, and **S**oft membership **T**ransfer **L**earning is used on target.
- **DSTL** – **D**BSCAN clustering is used while constructing the codebook, and **S**oft membership **T**ransfer **L**earning is used on target.

Table 3.1 and Table 3.2 gives the values of RMSE and MAE of `Book-Crossing` and `Synthetic` data by using the baseline methods considered and best of our methods.

Table 3.3 and Table 3.4 gives the RMSE and MAE values on `Book-Crossing` and `Synthetic` data using our methods by utilizing Θ_{avg} to get the cluster level rating pattern. Table 3.5 and Table 3.6 gives the values of RMSE and MAE of `Book-Crossing` and `Synthetic` data when Θ_{med} is used to get the cluster level rating pattern.

Table 3.1: RMSE and MAE of baseline methods and our methods on Book-Crossing data

Method	MMMF	MMMF$_{TL}$	MINDTL	TRACER	CBT	DHTL	DSTL
RMSE	1.0017	0.9767	1.6428	1.0354	0.8682	0.9018	**0.8604**
MAE	0.6910	0.6892	1.3077	0.8421	0.6856	0.6542	**0.6182**

Table 3.2: RMSE and MAE of baseline methods and our methods on Synthetic data

Method	MMMF	MMMF$_{TL}$	MINDTL	TRACER	CBT	KHTL	KSTL
RMSE	1.4743	1.4709	1.6066	1.4834	1.4249	1.4398	**1.4162**
MAE	1.2361	1.2345	1.3213	1.2673	1.2258	1.2155	**1.2023**

Table 3.3: RMSE and MAE on Book-Crossing data when Θ_{avg} is used for mapping

Method	KHTL	KSTL	SHTL	SSTL	DHTL	DSTL
RMSE	0.9589	**0.8752**	0.9052	**0.8685**	0.9018	**0.8604**
MAE	0.6710	**0.6288**	0.6602	**0.6205**	0.6542	**0.6182**

Table 3.4: RMSE and MAE on Synthetic data when Θ_{avg} is used for mapping

Method	KHTL	KSTL	SHTL	SSTL	DHTL	DSTL
RMSE	1.4400	**1.4168**	1.4468	**1.4225**	1.4502	**1.4278**
MAE	1.2159	**1.2029**	1.2199	**1.2096**	1.2216	**1.2134**

Table 3.5: RMSE and MAE on Book-Crossing data when Θ_{med} is used for mapping

Method	KHTL	KSTL	SHTL	SSTL	DHTL	DSTL
RMSE	0.9477	**0.8746**	0.9035	**0.8678**	0.9113	**0.8610**
MAE	0.6659	**0.6282**	0.6559	**0.6199**	0.6678	**0.6187**

Table 3.6: RMSE and MAE on Synthetic data when Θ_{med} is used for mapping

Method	KHTL	KSTL	SHTL	SSTL	DHTL	DSTL
RMSE	1.4398	**1.4162**	1.4471	**1.4229**	1.4492	**1.4289**
MAE	1.2155	**1.2023**	1.2204	**1.2089**	1.2214	**1.2146**

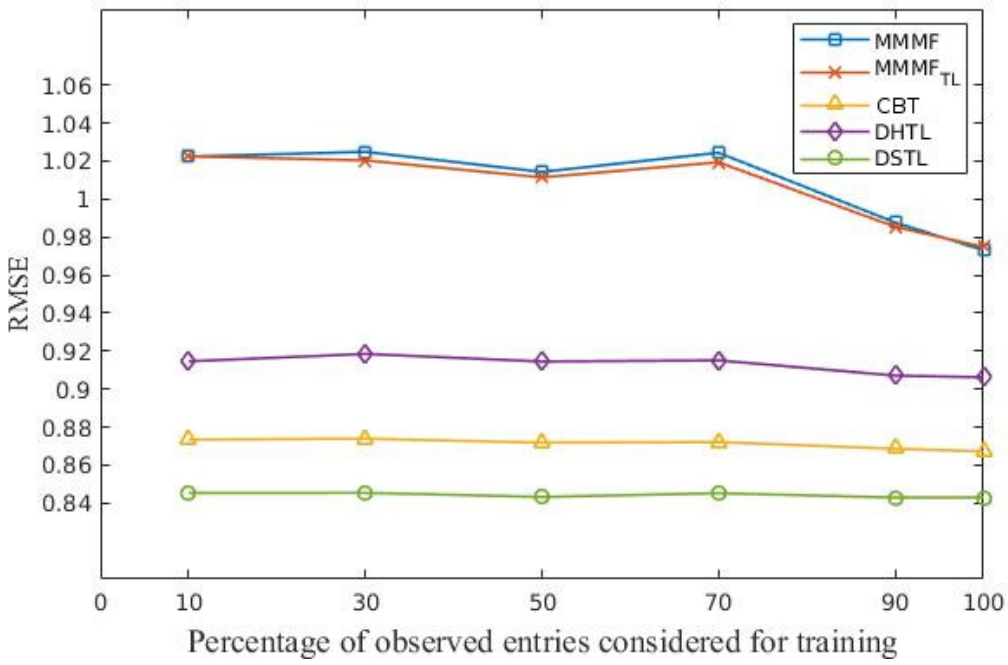


Figure 3.4: Change in RMSE on Book-Crossing data when the percentage of training data considered from training set changes

The idea behind providing the Tables 3.3, 3.4, 3.5, 3.6 is to show that in all of the cases soft membership (of users/items) transfer learning (e.g. KSTL) performs better than hard membership (of users/items) transfer learning (e.g. KHTL). Best values are given in bold in the tables. We have tuned the parameters of our methods by considering the RMSE and MAE as measures. The range of the number of clusters considered is 50 to 200, and we have fixed the number of clusters to 150 (best). In the case of DBSCAN clustering, the range of eps parameter which we have tuned is 0.001 to 0.15 and that of

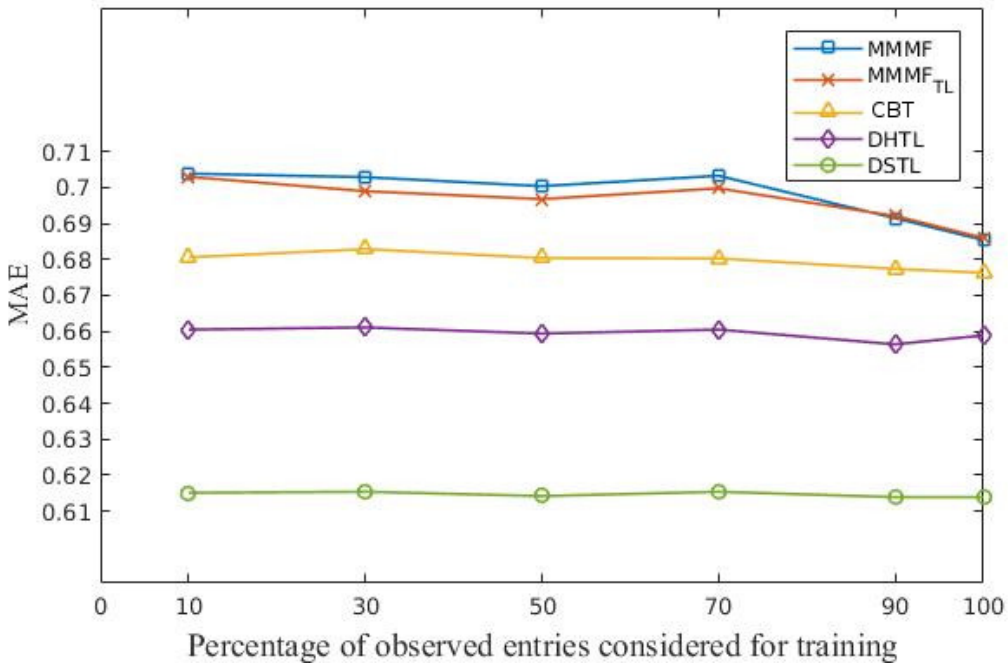


Figure 3.5: Change in MAE on Book-Crossing data when the percentage of training data considered from training set changes

$minPts$ is 3 to 8 and the results given in the table are by using the best (eps = 0.008, $minPts$ = 4 for Books data and eps = 0.005, $minPts$ = 3 for Synthetic data) of those. In all of the experiments the number of latent features (ℓ) is fixed to 100. In majority, mapping of real values of $U_c \times V_c$ using Θ_{med} is performing well when compared to that of mapping with Θ_{avg}. The reason is that there may exist some outliers in the threshold matrix and if the average is considered then the centroid may shift towards the outliers rather than towards the dense region whereas the median is less affected by the outliers and the centroid falls into the denser region. Also, in some cases where the members are connected but not compact, spectral clustering performs well when compared to that of k-means clustering. On the other hand, we have also experimented on different datasets to show the effect of sparsity level of data. We divide the target data into training (80%) and testing (20%) sets, and by fixing the testing set we considered different percentages of the training data. Fig. 3.4, 3.5 and Fig. 3.6, 3.7 depicts the effect of sparsity level (lesser the percentage of training data more the sparsity) on RMSE as well as MAE on Book-Crossing data and Synthetic data. X-axis gives the percentage of training data considered for training, and Y-axis gives the corresponding RMSE (Fig. 3.4, 3.6) or MAE (Fig. 3.5, 3.7) on testing set. Only the best comparison methods are plotted. Also, among our methods, we have plotted the best performing hard and

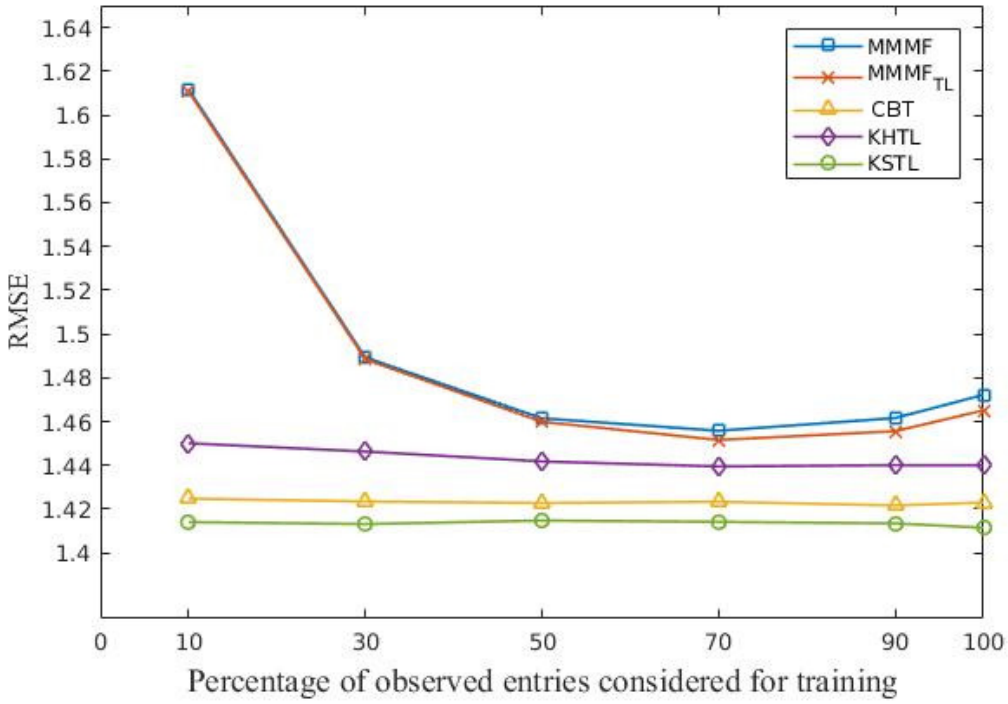


Figure 3.6: Change in RMSE on Synthetic data when the percentage of training data considered from training set changes

soft membership transfer learning methods in the case of synthetic data, namely, KHTL and KSTL. Similarly, the best performing hard and soft membership transfer learning methods in the case of Book-crossing, namely, DHTL and DSTL is also plotted. If we observe these figures, MMMF$_{TL}$ is almost performing same as that of MMMF. Also, as the sparsity level of the target rating matrix increases (i.e., less percentage of training data is considered), the performance of MMMF and MMMF$_{TL}$ decreases. On the other hand, whatever percentage of training data is considered, soft membership transfer learning method is performing better. It can also be observed that the performance of codebook based transfer learning methods is consistent with varying sparsity levels. This is due to the fact that these methods in true sense exploit the knowledge of the source domain (in the form of codebook) and effectively use the same for the prediction of the target domain. This also justifies our aim for the need of transfer learning when sufficient information is not available for prediction.

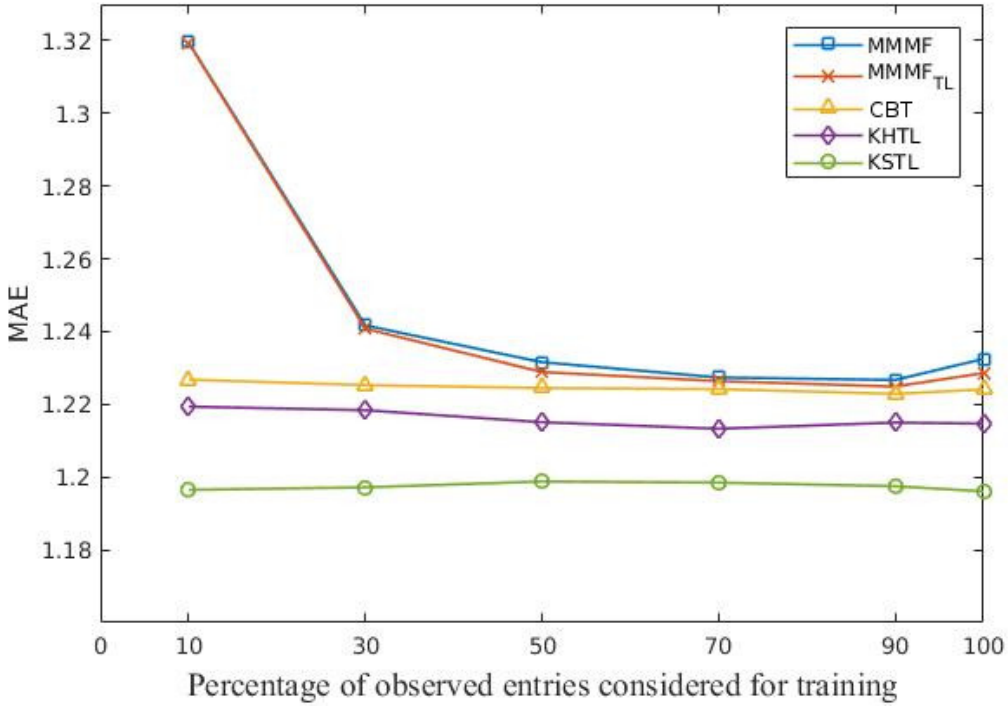


Figure 3.7: Change in MAE on Synthetic data when the percentage of training data considered from trainset changes

3.5 Summary

In this chapter a novel method for cross-domain collaborative filtering is discussed, in which MMMF is applied on the source domain data to generate the user/item latent feature matrices which are then clustered to get the user/item cluster-level latent feature matrices of the source domain. The clustered matrices are then multiplied to generate the codebook which is then transferred to the target domain. To the best of our knowledge there is no method which considers MMMF to construct the codebook from the source data. Codebook is transferred to the target domain in two different ways. In one approach, the target user and item membership matrices form hard membership in which a user/item can belong to single cluster, and in the other approach a softness constraint was introduced into the optimization function where a single user or single item belongs to multiple clusters with some weights.

CHAPTER 4

Transfer of Cluster-Level Latent Features to address the Data-Sparsity

In this chapter we discuss a novel approach (named TLFC - Transfer of Latent Features of Codebook) for transfer learning in collaborative filtering in which we transfer the latent features of codebook. In this method, the cluster-level rating pattern (codebook) of the source domain is obtained using the co-clustering technique. The obtained codebook is processed by removing some of the entries and we get the partial (processed) codebook. Applying MMMF on the processed codebook yields the latent features of codebook, that are nothing but the cluster-level latent features of source domain. These learnt cluster-level latent features of source domain are introduced into the optimization function of the target domain in a novel way via hinge loss and thereafter the prediction of target domain ratings is done.

The rest of the chapter is organized as follows: The introduction and related work are given in Section 4.1, and we discuss our proposed approach in Section 4.2. Section 4.3 discusses the experimental results. We summarize the chapter in Section 4.4.

4.1 Introduction & Related Work

Though collaborative filtering based recommender systems have become the norm these days, they have trouble in making accurate recommendations due to the data sparsity problem, i.e., very little existing information is available. To address the data

sparsity problem in recommender systems, *transfer learning* (53; 54; 96) techniques (through which cross-domain collaborative filtering is achieved) has been proposed in the literature. As discussed previously in Chapters 1, 2, in transfer learning there is a source domain which is usually considered as a *dense* domain from which knowledge is transferred to the target domain which is usually *sparse*.

One of the major issues involved in developing transfer learning techniques for recommendation purpose is to establish a link between the domains that are involved, so that knowledge can be transferred from *source* to the *target* domain. As discussed in the preceding chapters, domains can be linked and the transfer can happen explicitly via inter-domain similarities, common item attributes, etc. The transfer can happen also implicitly via shared user latent features or item latent features or by rating patterns which can be transferred between the domains. In (7), a framework was proposed where the items that are relevant in the source domain are selected based on the common attributes they have with the target domain (user interested domain). In this way, the inter-domain links were established through the common item attributes, however no overlap of users/items was required between the domains. On the other hand the transfer of knowledge through shared latent features (of users/items) is addressed in (58). The idea is to learn the hidden features present in the users and items of the source domain so that they can be integrated into the target rating matrix during the factorization process via regularisation. The success of this procedure is dependent on the presence of common users or items. In (55), the latent features of target and source are shared in a collective way. Here, rather than learning the latent features from source and utilizing them in the target, a technique is proposed wherein the latent features are simultaneously learnt from both the domains. A method called matrix tri-factorization is used to construct the shared latent space with the condition that from both the domains the users and items needs to be identical.

There are other set of methods in which *rating patterns* are analysed and transferred rather than latent features. These methods can be used in scenarios wherein users/items are not common between the domains. Rating patterns stem from the assumption that among the ratings of groups of users and groups of items a correlation could exist. One such method is codebook transfer (CBT) (35), where the main assumption is that, though the users/items are different across systems, the clusters (groups - based on age, interest etc.) of them behave similarly. It is an adaptive method which consists

of mainly two steps. One is the codebook construction and the second step is filling the target matrix by transferring the learnt codebook. As part of the initial step, the users and items that belong to the dense source domain is co-clustered to get the rating pattern at the cluster level. This rating pattern is called the *codebook* which consists of the mean rating of each of the co-clusters of users and items. Following that, the codebook is transferred to the target domain by expanding the values of the codebook. To do the same, users and items of the target domain needs to be mapped (to co-clusters) and this can be done by minimizing the quadratic loss which can be expressed as,

$$\min_{F_1 \in \{0,1\}^{m \times k_1}, F_2 \in \{0,1\}^{n \times k_2}} ||[Y - F_1 C F_2^T] \odot W||_F^2 \quad \text{s.t.,} F_1\mathbf{1} = 1,\ F_2\mathbf{1} = 1. \tag{4.1}$$

Here, $Y_{m \times n}$ is the target domain rating matrix. $C_{k_1 \times k_2}$ is codebook (rating pattern at cluster-level) which is learned from the source domain. The codebook is fixed and utilized to learn the cluster membership matrices of users ($F_{1m \times k_1}$) and items ($F_{2n \times k_2}$) of target data. A value of 1 in the indicator matrix W of size $m \times n$ shows the existence of the rating in the original rating matrix and 0 otherwise. The idea of code book transfer is to find a common latent space wherein the information obtained from C (source domain data) can be used to improve the recommendation in the target domain. i.e., here the ratings gets transferred in the condensed form (codebook).

In (36), authors have proposed a method called rating-matrix generative model which uses a probabilistic framework and fill the missing ratings of target domain by considering the rating data from multiple source rating matrices to construct the rating pattern. (47) extends the CBT by considering multiple source domains and checking different combinations of user/item clusters. It builds different codebooks for each of the source domain and extracts the relatedness between the target and each of the sources. It is based on the linear combination of different codebooks in which the learning of weights is done by the minimization of target domain prediction error. A relaxation related to the assumption of a fully dense source domain rating matrix is taken into consideration in (20; 19).

4.2 Proposed Approach

Let there be two user-item rating matrices of different domains say $X_{m' \times n'}$ (source domain matrix), $Y_{m \times n}$ (target domain matrix). Here m', m is the number of users and n', n is the number of items, and the entries of the matrices are the ratings given by the users to the items. Our goal is to predict the missing entries of the target domain more accurately using the source domain data. Figure-4.1 gives the sequential steps of the proposed method. Initially, we fill the missing entries in the source rating matrix with the mean of the ratings of that row (Step-1) and denote the filled-in rating matrix as X'. In Step-2, we apply the co-clustering on the filled-in rating matrix (X') in order to get the rating pattern at the cluster-level called as codebook ($C_{k_1 \times k_2}$). Once the codebook is obtained, we process the codebook (Step-3) by removing some of the entries of codebook and replacing by 0. Processing of codebook is done by comparing it with filled-in rating matrix as follows. Take the entry of the codebook which indicates the average of ratings given by a cluster of the users to some group of items. Compare the value with the entries of the particular users (forming a cluster) and particular items (of the cluster) of the filled-in rating matrix. If the number of entries containing the same value is more than some specific threshold percentage (th) then keep it, else we remove and replace it as zero. As there are real values in codebook and filled-in rating matrix, we don't check for the values to be exact, but instead we check for their difference to be small. The difference (error) is compared using some margin ϵ. Calculate the difference between the entries of filled-in matrix and that of codebook, and if more than some threshold percentage (th) of entries contain the margin less than or equal to $|\epsilon|$, then keep the entry as it is, else remove the entry. By following this removal of entries we get a partial codebook (C_p). Now, apply MMMF (2.8) on the processed codebook (C_p) to get the cluster-level latent feature vectors of users ($U_{c_{k_1 \times l'}}$), and items ($V_{c_{k_2 \times l'}}$) alongside a threshold matrix of users ($\Theta_{c_{k_1 \times r-1}}$), which is shown in Step-4.

Once the cluster-level latent features are obtained, we transfer the same to the target domain which is given at Step-5, by minimizing the optimization function given in Eq. 4.2. *To the best of our knowledge, there is no research which addresses transfer of cluster-level latent features and also consider hinge loss (Eq. 4.2) as the loss function while transferring the learnt knowledge of the source domain to the target domain.* Our

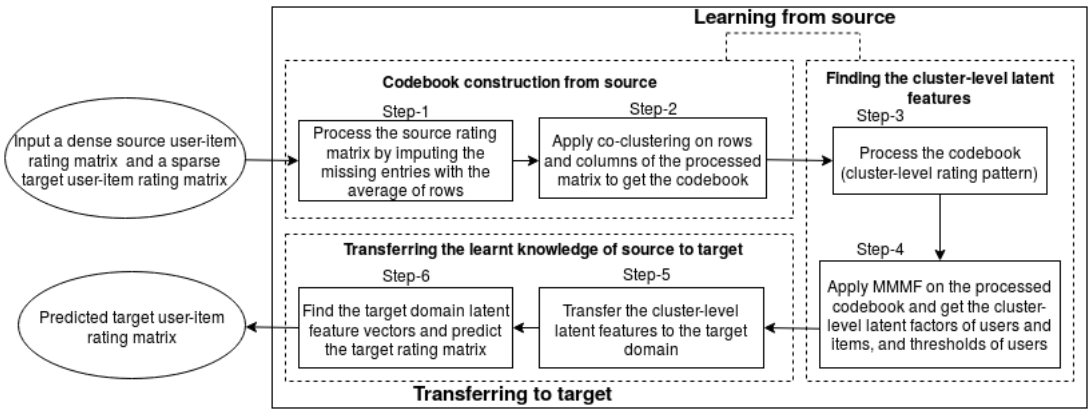


Figure 4.1: Illustration of the proposed method

assumption is that there could exist between the source and target domain some implicit correspondence through cluster-level user/item latent features (U_c, V_c).

$$\mathcal{J}(\alpha,\beta) = \sum_{(i,j)\in\omega}\sum_{a=1}^{r-1} h(\mathcal{T}_{ij}^{a}(\alpha_i\theta_{.a} - (\alpha_i U_c)(\beta_j V_c)^T)) + \lambda_1(\sum_{i=1}^{m} l_1(\sum_{k=1}^{k_1}\alpha_{ik}) + \sum_{j=1}^{n} l_1(\sum_{k=1}^{k_2}\beta_{jk})) + \lambda_2(\sum_{i=1}^{m}\sum_{k=1}^{k_1} l_2(\alpha_{ik}) + \sum_{j=1}^{n}\sum_{k=1}^{k_2} l_2(\beta_{jk})) \tag{4.2}$$

where,

$$h(z) = \begin{cases} 0 & \text{if } z \geq 1 \\ \frac{1}{2}(1-z)^2 & \text{if } 0 < z < 1 \\ \frac{1}{2} - z & \text{otherwise.} \end{cases} \tag{4.3}$$

$$\mathcal{T}_{ij}^{a} = \begin{cases} +1 & \text{if } a \geq x_{ij} \\ -1 & \text{if } a < x_{ij} \end{cases}$$

$$l_1(z) = \begin{cases} 1-z, & \text{if } z < 1 \\ z-1, & \text{if } z > 1 \\ 0, & \text{otherwise} \end{cases}$$

$$l_2(z) = \begin{cases} z, & \text{if } z < 0 \\ 0, & \text{otherwise} \end{cases}$$

$\lambda_1, \lambda_2 > 0$ are regularization parameters. $h(\cdot)$ is the smooth-hinge loss defined as given in Eq.(4.3). For a given matrix A, $A_{.a}$ is the a^{th} column of A. $l_1(z)$ ensures the row sum of α and column sum of β to be 1, whereas $l_2(z)$ ensures all elements of α and β to be positive. We have used Gradient Descent technique to optimize the Eq.(4.2), by updating the variables α and β. Initially α and β are randomly assigned, and then by calculating the gradients of Eq.(4.2) w.r.t. to α and β, we update α and β. By using the updated values of α and β, the value of the Eq.(4.2) decreases monotonically and converges to a local minimum.

The gradients of Eq.(4.2) w.r.t. variables α and β are as follows,

$$\frac{\partial \mathcal{J}}{\partial \alpha_{ik}} = \lambda_1 l_1^{'}(\sum_{k=1}^{k_1} \alpha_{ik}) + \lambda_2 l_2^{'}(\alpha_{ik}) + \sum_{a=1}^{r-1} \sum_{(i,j)\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\alpha_i \theta_{.a} - (\alpha_i U_c)(\beta_j V_c)^T))(\theta_{ka} - IU_c(\beta_j V_c)^T) \tag{4.4}$$

$$\frac{\partial \mathcal{J}}{\partial \beta_{jk}} = \lambda_1 l_1^{'}(\sum_{k=1}^{k_2} \beta_{jk}) + \lambda_2 l_2^{'}(\beta_{jk}) - \sum_{a=1}^{r-1} \sum_{(i,j)\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\alpha_i \theta_{.a} - (\alpha_i U_c)(\beta_j V_c)^T))(\alpha_i U_c)(V_{c_k})^T \tag{4.5}$$

where,

$$h'(z) = \begin{cases} 0 & \text{if } z \geq 1 \\ z - 1 & \text{if } 0 < z < 1 \\ -1 & \text{otherwise.} \end{cases}$$

$$l_1^{'}(z) = \begin{cases} -1, & \text{if } z < 1 \\ 1, & \text{if } z > 1 \\ 0, & \text{otherwise} \end{cases} \qquad l_2^{'}(z) = \begin{cases} 1, & \text{if } z < 0 \\ 0, & \text{otherwise} \end{cases}$$

I is the row vector of dimension $1 \times k_1$ containing all $1's$.

The updation of α and β variables is done as,

$$\alpha_{ik}^{t+1} = \alpha_{ik}^{t} - \eta \frac{\partial \mathcal{J}}{\partial \alpha_{ik}^{t}}$$

$$\beta_{jk}^{t+1} = \beta_{jk}^{t} - \eta \frac{\partial \mathcal{J}}{\partial \beta_{jk}^{t}}$$

Here η is the step size. Once we get the converged values of α and β, we construct the predicted target rating matrix (Step-6) using Equation (4.6), and map the resultant matrix with the threshold matrix ($\alpha.\Theta_c$) to get the target predicted rating matrix.

$$\hat{Y} = W \odot Y + [1 - W] \odot [\alpha U_c V_c^T \beta^T], \tag{4.6}$$

where $\hat{Y}$ is the predicted (approximated) target rating matrix. A value of 1 in the

Algorithm 3: Co-clustering combined with MMMF

1: **Input:** A $m' \times n'$ source rating matrix X and a $m \times n$ sparse target rating matrix Y with y_{ij} known for $(i, j) \in \omega$
2: **Output:** y_{ij} for $(i, j) \notin \omega$
3: Fill the missing entries of each row of X with the average of the rows and call it as X'.
4: Apply co-clustering on Y' to get the codebook (C).
5: Process the codebook to get the partial codebook (C_p).
6: Find U_c, V_c, Θ_c using MMMF (by minimizing Eq. 2.8) on C_p.
7: Use U_c, V_c, and find α, β of target domain by minimizing equation (4.2).
8: Using these α and β, calculate Eq. 4.6 and map with the $\alpha\Theta_c$ in order to get the discrete predicted rating matrix ($\hat{Y}$).

indicator matrix W of size $m \times n$ shows the existence of the rating in the original rating matrix and 0 otherwise. Error calculation for only the observed ratings is ensured through W and the Hadamard product (element-wise product) is denoted using $\odot$. By using the gradient descent technique as given in Eq.(4.2) a minimal solution can be obtained by updating α and β. Initially α and β are randomly assigned, and then by calculating the gradients of Eq.(4.2) w.r.t. to α and β, we update α and β. By using the updated values of α and β, the value of the Eq.(4.2) decreases monotonically until a local minima is reached. Once we get α and β by solving the optimization function (4.2), we construct the predicted target rating matrix (Step-6) using Equation (4.6), and map the resultant matrix with the threshold matrix ($\alpha.\Theta_c$) to get the target predicted rating matrix.

To be brief of Figure-4.1, it shows the flow of the proposed method, in which *learning from source* and *transferring to target* are the main steps. In the learning stage, the

cluser-level latent features from the source domain are learnt, and in the transferring stage, the learnt knowledge (cluster-level latent features) get transferred to the target domain in order to predict the missing ratings of target domain more accurately.

4.3 Experimental Analysis

`MovieLens-1M`[1] is used as the *source* dataset and `Goodbooks`[2] is used as the *target* dataset in our experiments. We have taken the first 5000 users and 3000 items from the Goodbooks data. The values of the datasets are in {0,1,2,3,4,5}. The value 0 indicates that the rating is missing, and 1 indicates the least rating, and 5 is the highest rating. Table 4.1 gives the statistics of the datasets. In our experiments, we have divided the data into training (80%) and testing (20%) sets. Root Mean Squared Error (RMSE) (Eq.4.7) and Mean Absolute Error (MAE) (Eq.4.8) are the two metrics used for evaluating our algorithms wherein smaller the values of these metrics indicate better performance.

Table 4.1: Datasets statistics

Dataset	# of Users	# of Items	% of Observed entries
MovieLens 1M	6040	3952	3.77
Goodbooks	5000	3000	1.08

4.3.1 Evaluation Metrics

From the literature it can be seen that a variety of collaborative filtering algorithms have been put forward in the last decade or so. The accuracy with which these algorithms can predict a new item/set of items vary. It is often the case that performance evaluation of these collaborative filtering algorithms is based on prediction accuracy, and Root Mean Square Error (RMSE) (Eq. (4.7)) and Mean Absolute Error (MAE) (Eq. (4.8)) are the two commonly used metrics to measure prediction accuracy. We evaluate our proposed method using RMSE and MAE. The smaller the values of RMSE

[1]https://grouplens.org/datasets/movielens/
[2]https://github.com/zygmuntz/goodbooks-10k

and MAE, better the performance of the said method.

$$RMSE = \sqrt{\sum_{(i,j)\epsilon\omega} \frac{(y_{ij} - \hat{y}_{ij})^2}{|\omega|}} \tag{4.7}$$

$$MAE = \sum_{(i,j)\epsilon\omega} \frac{|(y_{ij} - \hat{y}_{ij})|}{|\omega|} \tag{4.8}$$

where x_{ij} is the original rating, $\hat{x}_{ij}$ is the predicted rating, and $|\omega|$ is the number of test ratings.

4.3.2 The different Methods used for Comparison

Some of the baseline methods we use for evaluating the performance of our proposed method can be outlined as follows:

- **MMMF (81; 65):** Maximum Margin Matrix Factorization (MMMF) is the dominant factorization technique used in collaborative filtering. MMMF is usually applied on the input rating matrix consisting of the user-item ratings. The idea is to find the user and item latent-factor vectors which are of low rank by making use of the existing ratings. MMMF can be applied on a single domain only, and hence in our experiments, we applied it on the target domain directly.

- **MINDTL (20):** In MINDTL, codebook is constructed by taking into consideration the data from all the incomplete source domains. Here codebook for each domain is constructed.

  Following that, the constructed codebooks are linearly combined and transferred to the target domain, and the missing values of the target rating matrix gets predicted. As far as our experimental setup is concerned only a single domain is taken into consideration.

- **TRACER (101):** In TRACER, data from multiple domains are accounted for and based on this, ratings (which includes missing ratings) for all the source matrices are predicted. Thereafter the predicted knowledge is utilized by transferring it into the target domain. By making use of consensus regularisation during the knowledge transfer process, all the predicted values are forced to be similar. In a way it can be said that in TRACER at the same time learning and transferring happens. In our experiments, we have considered a single domain and therefore there is no need for consensus regularisation.

  We thank the authors for providing the code[3] online.

- **CBT (35)**: In this approach, the dense part of the source user-item rating matrix is considered, and the missing values of the rows of the dense matrix get imputed

[3]https://github.com/hezi73/TRACER

using the average of the ratings of particular row (user). The codebook is obtained from the dense user-item matrix by applying the technique of co-clustering.

In our experiments, unlike in (35), which consider only the dense part of the input data, the codebook is constructed by making use of the whole source data. Transferring of the learned codebook to the target domain is achieved by minimizing Eq. (4.1).

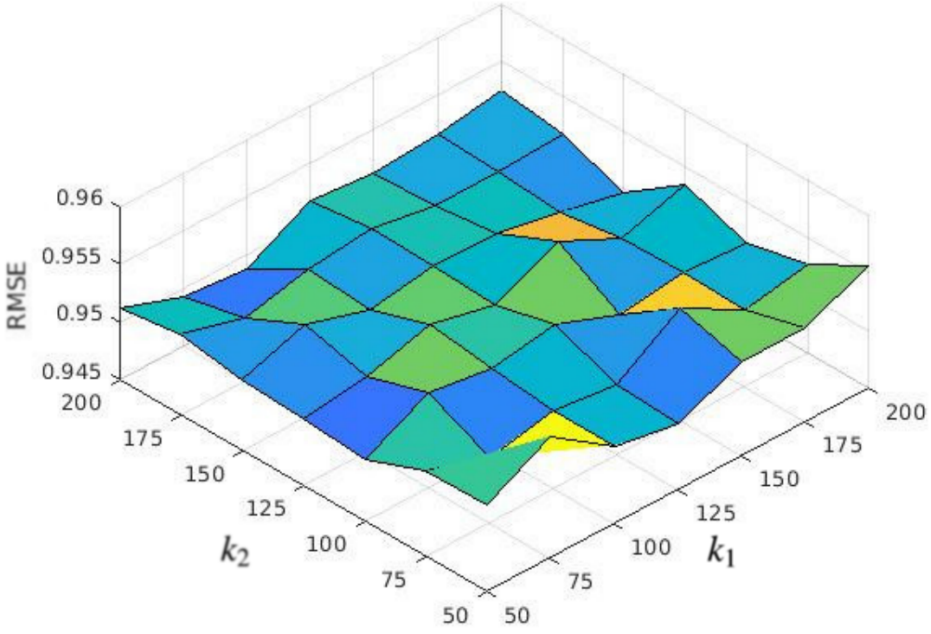


Figure 4.2: Impact of number of clusters on RMSE of Goodbooks data when MovieLens-1M is considered as source

Table 4.2: RMSE and MAE of baseline methods and TLFC method on Goodbooks data using MovieLens-1M as source

MetricMethod	MMMF	MINDTL	TRACER	CBT	TLFC
RMSE	0.9582	1.2794	0.9637	0.9641	**0.9507**
MAE	0.6501	0.9232	0.7781	0.7890	**0.6466**

We have conducted the experiments on MovieLens-1M data (source) and Goodbooks data (target), with varying number of clusters (k_1, k_2 - 25, 50, 75, 100, 125, 150, 175, 200). Fig. (4.2) gives the impact of number of clusters on RMSE and Fig. (4.3) shows the impact of number of clusters on MAE. Although there is not much change in the metric values with varying number of clusters, in our experiments we have fixed k_1 to 150, k_2 to 100, for which the best performance is achieved. By fixing the number of clusters, we have also experimented our algorithm by varying the values of threshold

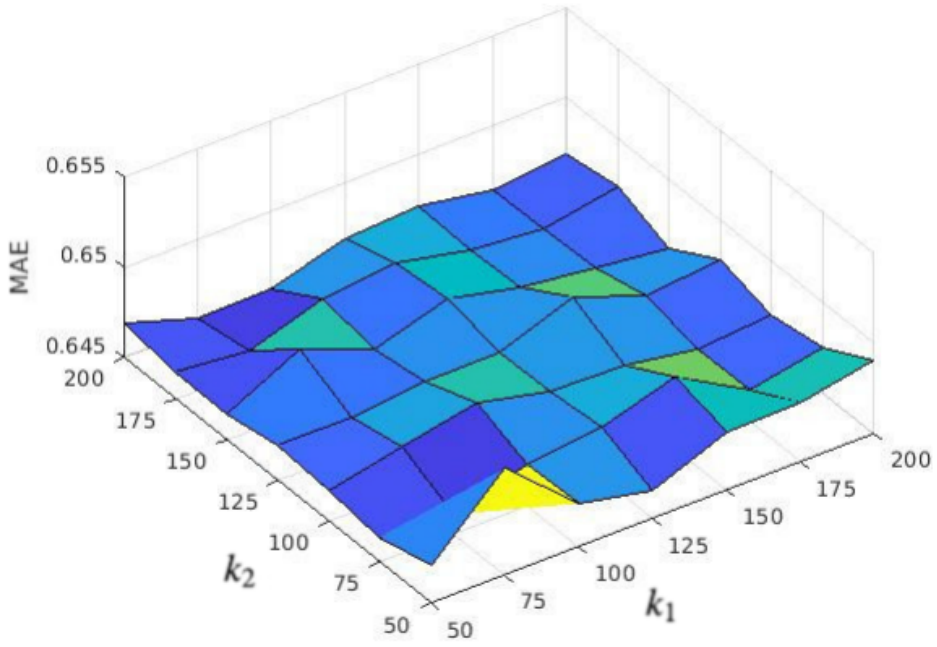


Figure 4.3: Impact of number of clusters on MAE of Goodbooks data when MovieLens-1M is considered as source

(th) and margin (ϵ). The range of th (in %) fall in {40, 50, 60, 70, 80}, and the values of ϵ considered are {0.1, 0.2, 0.3, 0.4, 0.5}. The performance of our algorithm is satisfying when the value of th is 50, and that of ϵ is 0.2. Hence, in the experiments of the proposed method, when Goodbooks data is the target and MovieLens-M is source, we have fixed the values of parameters as follows: k_1 = 150, k_2 = 100, th = 80, $\epsilon = 0.3$.

Table 4.2 shows the RMSE and MAE values on `Goodbooks` (target) data of baseline methods considered and the proposed method, by considering MovieLens-1M as source. The values reported are the average of five runs.

4.4 Summary

In this chapter, we have seen a novel model (TLFC) for cross-domain recommendation where the cluster-level latent features of the source domain are considered, and used in the target domain when the domains do not share common users or common items. As the first step, missing entries of the source rating matrix are imputed with the average of the rows to get the filled-in rating matrix. Thereafter the co-clustering

technique is used to construct the codebook of the source domain, i.e., to get the cluster-level rating pattern. After this stage, processing of codebook is done by comparing the entries of codebook with the values of the filled-in source rating matrix. The application of the Maximum margin matrix factorization technique on the processed codebook gives the cluster-level latent factor vectors of the source data. The learnt source knowledge (cluster-level latent factors) is then transferred to the target domain via hinge loss, and the learnt target domain latent features are utilized to get the predicted target rating matrix. By observing the experimental results on the benchmark data sets, we say that our model approximates the target matrix well.

CHAPTER 5

A Hinge-Loss based Codebook Transfer for Cross-Domain Recommendation

In the previous chapter we have discussed a cross-domain collaborative filtering mehod in which cluster-level latent features of codebook are transferred. In this chapter, we propose another novel transfer learning technique for cross-domain recommender systems (TCH - Transfer of codebook via Hinge loss) in which we use the co-clustering method on the original source rating matrix to generate the codebook. The obtained codebook is transferred to the target domain in a novel way using the hinge loss function instead of the squared loss function.

5.1 Introduction

Not only in recommender systems but also in many machine learning algorithms, loss functions play a significant role in empirical risk minimization and computational complexities (43; 67). So, choice of the loss function is very important. Among all the loss functions, hinge loss is more suitable for the discrete classification task. Also, among all the matrix factorization techniques, MMMF treats the collaborative prediction problem as the classification task and takes full advantage of hinge loss. As MMMF is applicable for single domain only, in this chapter, we take advantage of hinge loss and

utilise it for cross-domain recommendation problem.

In this chapter, we present a novel method of CBT based collaborative filtering. As the first step, we perform co-clustering of source rating data to construct the codebook and in the next step, the learned codebook gets transferred to the target domain in a novel way by using hinge loss which has not been tried earlier. Previous research work takes into consideration squared loss while transferring the source knowledge to the target domain in order to predict the missing ratings of the target domain. Experimental results shows that the proposed transfer learning CF method outperforms MMMF (i.e., when MMMF is applied directly on target data) and other major methods on codebook based transfer learning for collaborative filtering.

The rest of the chapter is organized as follows: Section 5.2 gives a brief description of the existing works on transfer learning in recommender systems and Section 5.3 explains the proposed approach. The experimental results are given in Section 5.4, and the summary of the work is given in section 5.5.

5.2 Related work

Matrix Factorization has been very successful and popular in predicting the missing ratings of the user-item rating matrix even when the data is too sparse. In a sense, with a little existing information it can fill-up a sparse matrix to an adequate level of accuracy. But as the sparsity increases, the accuracy level of the prediction falls. In such a situation, researchers think through the possibility of using the information learnt in another context. Here comes the concept of cross-domain recommendation. Transfer learning (53; 54) is the technique of cross-domain collaborative filtering which was proposed in the last decade as a potential solution to minify this problem. Transfer learning aims to transfer the knowledge from the dense source domain to the sparse target domain. For example, suppose that a particular user has watched many movies and have rated the same. The same user has very less ratings in another domain related to books but wants a book to be recommended, then by using his ratings from the movie domain book can be recommended. Formally, given a dense source rating matrix (source domain), and a sparse target rating matrix (target domain), the goal is to predict the missing entries of target domain by using the information available in the source

domain.

The major question that was unanswered is to determine whether both domains are suitable for knowledge transfer. Most of the transfer learning for CF methods make certain assumptions on domains to apply transfer learning strategy. These assumptions include common subset of items or users, similar attributes of items or users. The objective is to determine a sort of correspondence between two domains so that the transfer of knowledge can take place positively. Domains can be linked and the transfer can happen explicitly via inter-domain similarities, common item attributes, etc. The transfer can happen also implicitly via shared user latent features or item latent features or by rating patterns which can be transferred between the domains.

In (7), a framework was proposed in which the relevant items in the source domain are selected based on the common attributes they have with the target domain (user interested domain). The inter domain links were established through the common item attributes, however there is no overlap of users/items required between the domains. On the other hand the transfer of knowledge by the shared latent features (of users/items) is addressed in (58) in which the latent features of users and items of source domain are learnt and adapted to a target domain by integrating the features into the factorization of target rating matrix via regularization. However, it requires either common users or items between the two domains. In (55), the latent features of source and target are shared in a collective way. Here, rather than learning the latent features from source and utilizing in the target, a method which simultaneously learns the latent features of both domains is proposed. The shared latent space is constructed via matrix tri-factorization. It also requires the users and items of both domains to be identical. These methods are based on the premise that the rating behavior does not change with change of domain for the same user (or, same item) or different users (or items) with similar attributes. In (96; 97), authors presume the correspondence of users/items across different domains in order to maximize the knowledge transfer. The methods initially apply Matrix Factorization on the target rating data and learn the latent features of target and thereafter select some users/items from the target domain based on some criterion and try to find the corresponding users/items in the source domain. However as the second step, the methods enforce the selected users'/items' latent factors of target data to be same as that of the source domain, via regularization.

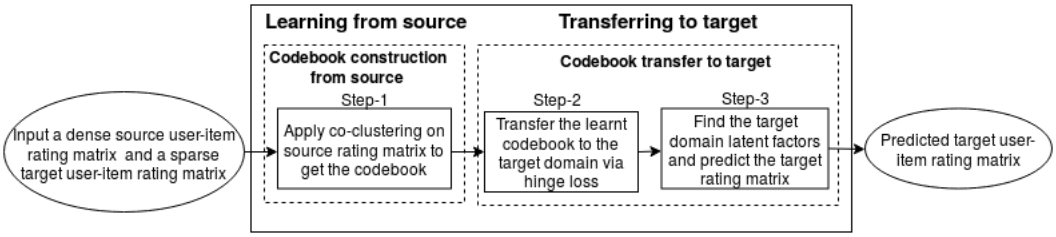


Figure 5.1: Illustration of the proposed method

If the target and source data have different types of feedback data, say target is having discrete ratings, and source data is having binary feedback, then the methods in (56; 57; 55) are applicable. In (56) it is assumed that the same users and same items exists in target and auxiliary domains. In (76) two sets of source data are taken into consideration, in which one set shares common set of users with target data, and the other shares common set of items with target data. It extracts the latent factors from source and constructs the similarity graphs from these latent factors and transfers both latent factors and similarity graphs to target data.

In Code Book Transfer (CBT) (35) CF approach, cluster-level rating-patterns are captured. The latent correlations between ratings of groups of users and groups of items exist, which is referred as cluster-level rating pattern (codebook). In this method, the source domain rating matrix is analyzed to extract a codebook which is used for prediction in the target domain. Regardless of the domains, users and items cluster together in such a way that the rating behavior of these clusters remain invariant across domains. Here, it is assumed that the source domain rating matrix is a full matrix. When the rating matrix is not full, the missing entries of the user get filled-in by an average rating of particular user. Thereafter the rows (users) and columns (items) of the filled-in rating matrix are clustered using co-clustering (35). Codebook is generated having a row for each user group and a column for each item group and a rating for a pair of user-group and item-group. The rating for a pair of user-group and item-group is the average rating of users and items in that particular group. While using the codebook in the target domain, the idea is to identify users/items in the target domain with the best-fit user-group/item-group of the transferred codebook. It can be formulated as the following optimization problem (5.1).

$$\min_{F_1 \in \{0,1\}^{m \times k_1}, F_2 \in \{0,1\}^{n \times k_2}} ||[Y - F_1 C {F_2}^T] \odot W||_F^2 \quad \text{s.t.,} F_1\mathbf{1} = 1,\ F_2\mathbf{1} = 1. \tag{5.1}$$

where $Y_{m\times n}$ is the target domain rating matrix, $C_{k_1\times k_2}$ is the codebook, $F_{1m\times k_1}$ and $F_{2n\times k_2}$ are the cluster membership matrices of users and items respectively. W is the indicator matrix of size $m \times n$ in which the value is 1 if the rating exists in the original rating matrix and 0 otherwise.

In (47), the linear combination of codebooks obtained from multiple domains are used and the coefficient of linear combination is learned through an optimization technique. In (20; 19), the assumption of fully dense rating matrix of source domain is relaxed.

The authors in (23) has come up with another method to generate codebook. The proposed method avoids pre-processing as well as co-clustering of rating matrix. In this method, the latent factors of users and items of source domain are generated by matrix factorization. The obtained latent factor-vectors are grouped separately to obtain user latent factor group and item latent factor group. The mean latent vectors of the groups are multiplied to generate a codebook.

In (35), the authors focus on extracting the group level behavior of users on items by assuming that, though the users/items are different across systems, the groups (groups - based on age, interest etc) of them behave similarly. All the missing ratings in the source rating matrix are filled in a priori by a preprocessing step and then co-clustering is applied on a filled-in source rating matrix to directly get the cluster-level rating pattern (codebook). The cluster-level rating pattern is then transferred to another domain (target). In the target domain, the user and item membership to clusters encode in the codebook so that a user (or, an item) is member of one user-cluster (or, item-cluster) represented in the codebook. In our approach, we do not use a separate preprocessing stage in the source domain and while transferring the learnt codebook of source to the target domain we use hinge loss instead of squared loss.

5.3 Proposed Method

Formally, given a dense source user-item rating matrix $X \in \mathbb{R}^{m'\times n'}$ and a sparse target user-item rating matrix $Y \in \mathbb{R}^{m\times n}$ where m', m are the number of users in source and target domain, and n', n are the number of items in source and target data, the goal is to predict the missing entries (as may users don't give ratings to many items)

in target domain using the source domain data. Prediction of missing entries should take place in such a way that the existing ratings must be approximated with less error rate.

The illustration of the proposed method is shown in Figure 5.1. Initially (Step-1), the users (rows) and items (columns) of the source rating matrix need to be simultaneously clustered (co-clustering) to construct the codebook. We need the cluster indicators of users and items, and in order to get the cluster indicators we can choose any of the co-clustering algorithms. We employed the similar formulation that was used in Orthogonal nonnegative matrix tri-factorization technique (ONMTF) (9) to get the cluster indicators. In addition, we have added the regularization term which ensures that the row sum of user cluster matrix to be one, and similarly with item cluster matrix. The source rating matrix X can be tri-factorized as follows.

$$\min_{P,Q,S} ||[X - PSQ^T] \odot W||_F^2 + \alpha||P\mathbf{1} - 1||_F^2 + \beta||Q\mathbf{1} - 1||_F^2 \tag{5.2}$$

where W is an indicator matrix of size $m' \times n'$ in which the entry is 1 if the rating exists in X and 0 otherwise. $||.||_F$ represents the Frobenius norm. The dimension of P is $m' \times k_1$, S is $k_1 \times k_2$, and Q is $n' \times k_2$. $||P\mathbf{1} - 1||_F$ ensures the row sum of P to be one, similarly with Q. Maximum value of each row of P and Q becomes the cluster indicator for the user/item in that row. These P and Q are not in a recognizable form in terms of user/item membership matrices. We use binary values to represent P and Q by setting the maximum valued entry in each row to be 1 and the others to be 0. As a result, these binary user/item cluster indicators form (membership) matrices, denoted by P_s and Q_s, for the source rating matrix. From these matrices, we can construct the codebook C as follows.

$$C = [P_s^T X Q_s] \oslash [P_s^T 1 1^T Q_s] \tag{5.3}$$

$\oslash$ indicates element-wise division. Averaging of all the ratings in each of the user/item co-cluster takes place in Eq. 5.3

Once the codebook from the source domain is constructed, transfer (Step-2) it to the target domain (Y) by substituting it in Eq. 5.4 and solve the objective function ($\mathcal{J}$) in order to get U and V of target data. We use Hinge loss to learn U and V instead of

squared loss. As the data of ours is discrete, in addition to U and V, $r-1$ thresholds θ_{ia} $(1 \leq a \leq r-1)$ for every user i has to be learned to classify the prediction value into r discrete values. Here, r is the maximum rating (say 5).

$$\mathcal{J}(U,V,\Theta) = \sum_{(i,j)\in\omega} \sum_{a=1}^{r-1} h(\mathcal{T}_{ij}^{a}(\theta_{ia} - U_{i\cdot}BV_{j\cdot}^{T})) + \frac{\lambda}{2}(||U||_F^2 + ||V||_F^2) \tag{5.4}$$

where,

$$\mathcal{T}_{ij}^{a} = \begin{cases} +1 & \text{if } a \geq y_{ij} \\ -1 & \text{if } a < y_{ij} \end{cases}$$

$h(.)$ is a smoothed hinge-loss function defined as,

$$h(z) = \begin{cases} 0 & \text{if } z \geq 1 \\ \frac{1}{2}(1-z)^2 & \text{if } 0 < z < 1 \\ \frac{1}{2} - z & \text{otherwise.} \end{cases}$$

ω is the set of observed entries, $\lambda > 0$ is regularization parameter. For any given matrix A, $A_{i\cdot}$ is the i^{th} row of A, and $A_{\cdot j}$ is the j^{th} column of A.
Gradient based approach can be used to solve the optimization function (Eq. 5.4) and the following are the gradients.

$$\frac{\partial \mathcal{J}}{\partial U_{ik_1}} = \lambda U_{ik_1} - \sum_{a=1}^{r-1} \sum_{j|ij\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\theta_{ia} - U_{i\cdot}BV_{j\cdot}^{T}))V_{j\cdot}B_{k_1\cdot}^{T}$$

$$\frac{\partial \mathcal{J}}{\partial V_{jk_2}} = \lambda V_{jk_2} - \sum_{a=1}^{r-1} \sum_{i|ij\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\theta_{ia} - U_{i\cdot}BV_{j\cdot}^{T}))U_{i\cdot}B_{\cdot k_2}$$

$$\frac{\partial \mathcal{J}}{\partial \Theta_{ia}} = \sum_{j|ij\in\omega} \mathcal{T}_{ij}^{a}.h'(\mathcal{T}_{ij}^{a}(\theta_{ia} - U_{i\cdot}BV_{j\cdot}^{T}))$$

where,

$$h'(z) = \begin{cases} 0 & \text{if } z \geq 1 \\ z-1 & \text{if } 0 < z < 1 \\ -1 & \text{otherwise.} \end{cases}$$

In gradient descent algorithms, we start with random U, V, Θ and iteratively update them using the following update rules.

$$U_{ik_1}^{t+1} = U_{ik_1}^{t} - c\frac{\partial \mathcal{J}}{\partial U_{ik_1}^{t}}$$

$$V_{jk_2}^{t+1} = V_{jk_2}^{t} - c\frac{\partial \mathcal{J}}{\partial V_{jk_2}^{t}}$$

$$\Theta_{ia}^{t+1} = \Theta_{ia}^{t} - c\frac{\partial \mathcal{J}}{\partial \Theta_{ia}^{t}}$$

c is the trade-off parameter.

Once U, V of target data are learnt, the product of U, C, and V^T (i.e., UCV^T) is mapped with the threshold matrix (Θ) and becomes the approximation (prediction$\in \{1, 2, ..., r\}$ say r = 5) of the target rating matrix (Step-3). This is illustrated in Algorithm 4.

Algorithm 4: Codebook transfer via hinge loss

1: **Input:** A $m' \times n'$ source rating matrix X and a $m \times n$ sparse target rating matrix Y with y_{ij} known for $(i, j) \in \omega$
2: **Output:** $\hat{Y}$ (y_{ij} for $(i, j) \notin \omega$)
3: Tri-factorize X using Eq. 5.2 inorder to get P, Q and S.
4: Calculate codebook C using Eq. 5.3.
5: Transfer the codebook to the target data by substituting in Eq. 5.4.
6: Solve the optimization (Eq. 5.4) using gradient descent technique inorder to get the target U, V and Θ.
7: Product of U, C, V^T (i.e., UCV^T) by mapping with the threshold matrix gives $\hat{Y}$.

Complexity Analysis: We analyze the computational complexity of the proposed method. The time complexity mainly comprises of two components: construction of codebook from source domain (Eq. 5.2), and transferring the same to the target domain (Eq. 5.4). The optimization problem given in Eq. 5.2 and Eq. 5.4 requires major computations for matrix multiplication. For the simplicity of representation, we assume that the computation cost for multiplication of two matrices, say, a $m' \times k$ matrix and a $k \times n'$ matrix, is $O(m'n'k)$. The computation cost required for updation of P, Q and S are $O(m'n'k_1)$, $O(m'n'k_2)$ and $O(m'^2k_1)$, respectively (21). Hence, the overall computation required for the construction of codebook is $O(t_1(m'n'k_1 + m'n'k_2 + m'^2k_1))$, where t_1 is the maximum number of iterations. In the target domain, the major computation is involved with the calculation of gradient w.r.t. each element of matrices U, V, and Θ. Overall

time complexity required for codebook transfer is $O(t_2(mnk_1+mnk_2+mnk_1))$, where t_2 is the maximum number of iterations. Hence, the overall computation cost of the proposed mentod is, $O(t_1(m'n'k_1 + m'n'k_2 + m'^2k_1) + t_2(2mnk_1 + mnk_2))$.

5.4 Experimental Results and Analysis

To evaluate the performance of our method we have experimented the method with different datasets. The datasets used are `MovieLens 100K`[1], `MovieLens 1M`[1], `Goodbooks`[2], `Douban Music`[3], `Douban Book`[3]. From the `Goodbooks` dataset we have considered the first 5000 users and 2000 items. From the Douban Music and Douban Book data, we have considered the first 2000 users and 2000 items. The entries of all the datasets fall in {0,1,2,3,4,5}, where 0 indicates the missing value, 1 indicates that the item is leastly liked and 5 indicates that the item is heavily liked. The statistics of the datasets are given in Table 5.1. In all the experiments, we have divided the observed data into 80% and 20%, in which 80% is used for training, and 20% is used for testing.

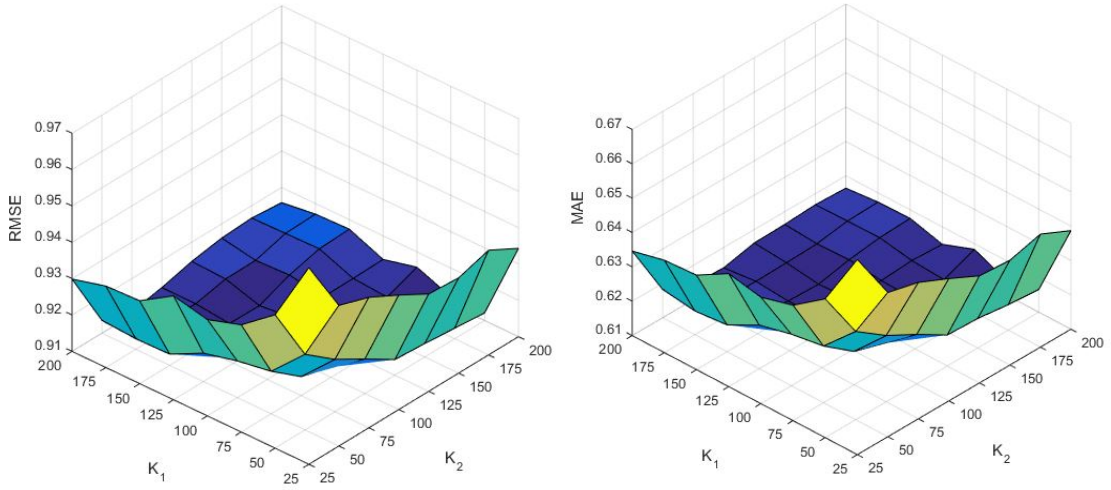
Table 5.1: Datasets statistics

Dataset	**# of Users**	**# of Items**	**% of Observed entries**
MovieLens 100K	945	1682	6.29
MovieLens 1M	6040	3952	3.77
Goodbooks (Subset)	5000	3000	1.08
Douban Music (Subset)	2000	2000	11.26
Douban Book (Subset)	2000	2000	5.51

5.4.1 Evaluation Metrics

The computation of prediction accuracy is the major criterion to evaluate the performance of rating-oriented collaborative filtering algorithms. To measure the predic-

[1]https://grouplens.org/datasets/movielens/
[2]https://github.com/zygmuntz/goodbooks-10k
[3]https://github.com/hezi73/TRACER/blob/master/douban.rar



RMSE MAE

Figure 5.2: Impact of number of clusters (k_1, k_2) on MovieLens 1M data when MovieLens 100K is considered as source

tion accuracy, Root Mean Square Error (RMSE) (Eq. (5.5)) and Mean Absolute Error (MAE) (Eq. (5.6)) are the commonly used metrics which depend on the difference between the predicted rating and the actual rating. So, we evaluate our method using these metrics (RMSE and MAE). The smaller the values of RMSE and MAE, better the performance of the method. The values reported in the tables are the average of five runs.

$$RMSE = \sqrt{\sum_{(i,j)\epsilon\omega} \frac{(y_{ij} - \hat{y}_{ij})^2}{|\omega|}} \tag{5.5}$$

$$MAE = \sum_{(i,j)\epsilon\omega} \frac{|(y_{ij} - \hat{y}_{ij})|}{|\omega|} \tag{5.6}$$

where y_{ij} is the original rating and $\hat{y}_{ij}$ is the predicted rating, $|\omega|$ is the number of test ratings.

5.4.2 Comparison Methods

To evaluate the performance of the proposed method, we consider the following baseline methods.

- **MMMF (81; 65):** It is one of the popular matrix factorization techniques used in collaborative filtering. MMMF predicts the missing entries of the given user-item rating matrix by finding the low-rank latent feature matrices of items and users. As MMMF can be applied on a single domain only, we applied it on the target domain directly in our experiments.

- **MINDTL (20):** In this approach, the data from multiple source domains is considered and the codebooks for all the domains get constructed. In the next step, the learned codebooks get linearly combined and transferred to the target domain in order to predict the missing values of the target rating matrix. In our experiments, we have considered a single source domain only.
- **TRACER (101):** This method considers the data from multiple source domains and predicts all the source rating matrices and exploit the predicted knowledge by transferring to the target domain. While transferring the knowledge, the method uses the consensus regularization which forces all the predicted values to be similar (say, majority value). In this method, learning and transferring happens at the same time. We have considered only a single source domain in our experiments and so there will be no consensus regularization, but knowledge learning and transferring happens at the same time. We thank the authors for providing the code[4] online.
- **CBT**: Finds the codebook of source data by applying co-clustering on the fully dense user-item rating matrix as given in (35). In our experiments we consider the whole source data to construct the codebook and the learned codebook gets transferred to the target domain by minimizing Eq. (5.1).

Table 5.2: RMSE and MAE of baseline methods and proposed method on MovieLens 1M data, Goodbooks data, Douban Book data

Dataset	Method	RMSE	MAE
MovieLens 1M	MMMF	0.9361	0.6402
	MINDTL	0.9948	0.7965
	TRACER	0.9800	0.8039
	CBT	0.9676	0.7746
	TCH	**0.9123**	**0.6134**
Goodbooks	MMMF	0.9604	0.6524
	MINDTL	1.2818	0.9224
	TRACER	0.9617	0.7772
	CBT	0.9634	0.7886
	TCH	**0.9470**	**0.6381**
Douban Book	MMMF	0.7976	0.5414
	MINDTL	1.1970	0.9054
	TRACER	0.8047	0.6600
	CBT	0.7828	0.6130
	TCH	**0.7785**	**0.5022**

We have conducted the experiments with varying number of clusters (k_1, k_2 - 25, 50, 75, 100, 125, 150, 175, 200) and Fig. 5.2 shows the impact of number of clusters on MovieLens 1M data when MovieLens 100K data is considered as source. Fig. 5.2a gives the impact of number of clusters on RMSE and Fig. 5.2b shows the impact of

[4]https://github.com/hezi73/TRACER

Table 5.3: RMSE and MAE of baseline methods and proposed method on MovieLens data by considering same dataset as source and target

Dataset	Method	RMSE	MAE
MovieLens 100K	MMMF	0.9828	0.6808
	MINDTL	1.9026	1.6538
	TRACER	1.0027	0.8213
	CBT	1.0264	0.8239
	TCH	**0.9653**	**0.6600**
MovieLens 1M	MMMF	0.9349	0.6389
	MINDTL	1.8545	1.5984
	TRACER	0.9892	0.8145
	CBT	1.0729	0.8725
	TCH	**0.9138**	**0.6175**

number of clusters on MAE. By observing the figures we can say that the best performance for this scenario (source - MovieLens 100K, target - MovieLens 1M) is achieved when the number of clusters are in between 100 and 150 and hence we have fixed both k_1, k_2 to 125. In the similar way, when MovieLens 1M is used as source and Goodbooks is the target, we set the number of clusters to 150. Similarly, when Douban music data is used as source and Douban book is the target, we set the number of clusters to 100.

Table 5.2 gives the values of RMSE and MAE of `MovieLens 1M` dataset, `Goodbooks` dataset, `Douban Book` dataset. For MovieLens 1M data, MovieLens 100K is considered as source and for Goodbooks data, MovieLens 1M is considered as source, and for Douban Book data, Douban music is considered as source data. The first column gives different datasets considered, whereas second column gives the methods considered for comparison along with the proposed method (TCH). The third and the fourth columns gives the RMSE and MAE values of the considered methods on datasets. By observing the table, we can say that the TCH method is performing well on any of the datasets considered.

We have also carried out experiments using our method by considering the same datasets as source and target data. It is nothing but reconstructing the same matrix by using the extracted knowledge. Table 5.3 gives the results on `MovieLens 100k` data when the same is used as source, and also on `MovieLens 1M` data when itself is used as source. By observing the metric values (RMSE, MAE) in the table, we can claim that TCH is outperforming all the comparision methods on the datasets considered.

5.5 Summary

In this chapter, we have discussed a novel codebook based transfer learning method called TCH for cross-domain recommendation. We have used a codebook construction technique, in which ONMTF is used on the source user-item rating matrix to construct the codebook. The constructed codebook is then transferred to the target domain in a novel way using the hinge loss. By observing the experimental results we say that TCH approximates the sparse target matrix well by utilizing the knowledge extracted from the source domain.

CHAPTER 6

Conclusions & Future Work

In this thesis, we have addressed one of the main issues in collaborative filtering based recommender systems, namely, data sparsity problem. To address data sparsity in cross-domain collaborative filtering recommender systems we proposed novel methodologies based on transfer learning. Whenever the data in the target domain is very less, using the technique of transfer learning, we made use of the information in source domain to transfer knowledge from source to target and thereby improved the accuracy of the prediction in the target domain.

In Chapter-1, we have discussed the basic concepts related to recommender systems and their usage, types of recommender systems, the motivation behind transfer learning which is used in cross-domain recommender systems. In Chapter-2, we had a detailed discussion on matrix factorization which is the most successful collaborative filtering technique, and also discussed the concept of transfer learning which is utilized in cross-domain recommender systems. We have also provided the literature survey on the papers related to transfer learning in collaborative filtering, which is cross-domain collaborative filtering. In Chapter-3, we have discussed our proposed technique of cross-domain recommender systems, which is based on codebook transfer. We have proposed a codebook construction technique, in which Maximum Margin Matrix Factorization is applied on the user-item rating matrix of source domain to obtain the latent-factor vectors of users and items, followed by clustering. The cluster-level latent factors of users and items are multiplied to get the codebook. By using the computed codebook, the optimal membership matrices of users and items of target domain are obtained. We have also introduced the soft membership of users and items while transferring the codebook

learned from source to the target domain. We have also shown that the codebook based transfer learning methods are performing in a consistent manner with varying sparsity levels in target domain. In the proposed method, codebook is constructed by multiplying the cluster-level user and item latent factors. In this sense, the codebook can be seen as the compressed representation of cluster-level latent factors of users and items.

In Chapter-4, we have discussed a novel technique related to transfer learning in collaborative filtering. This method, consider the cluster-level latent features of the source domain and utilize in the target when the domains do not share common users or common items. Initially, the missing entries of source domain rating matrix are imputed with the average of the rows to get the filled-in rating matrix. By making use of the co-clustering technique on the source domain, codebook is generated. Thereafter, the codebook is processed by comparing the entries of codebook with the values of the filled-in source rating matrix. Then maximum margin matrix factorization technique is applied on the processed codebook which gives the cluster-level latent factor vectors of the source data. The learnt source knowledge (cluster-level latent factors) is then transferred to the target domain via hinge loss, to learn the target domain latent features which are then used to get the predicted target rating matrix.

In Chapter-5, a novel method based on codebook based transfer learning for cross-domain recommendation is discussed. In this method, co-clustering technique is applied on the original source rating matrix to construct the codebook of source user-item rating matrix. The constructed codebook is transferred to the target domain by utilising the hinge loss function in a novel way. By observing the results it is proved that the sparse target matrix is predicted well by the proposed method.

In this thesis, the only goal we focused is to alleviate the data sparsity issue. In the future, it is worthwhile to consider the cold start issue too. The ratings are the only input data we have considered, and in the future one can consider *social tags* also as input types. Also, in real-world scenarios, user behaviour patterns change over time. As a result, precise estimation of such evolution, as well as optimization of the recommendation technique on top of it, is necessary. i.e., our recommendations should be based on users' constantly evolving feedback data. To be precise, our recommendation system should be capable of handling dynamic data too. One can also extend the works by considering multiple source domains and multiple target domains. Transfer learning

applications of other types rather than rating prediction in recommender systems could be another direction that could be pursued. Another area where one can focus is on transfer learning for group recommender systems.

REFERENCES

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

[2] Charu C. Aggarwal. *Recommender Systems: The Textbook*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[3] Chris Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion, 2006.

[4] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutierrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109 – 132, 2013.

[5] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12, 11 2002.

[6] Iván Cantador, Ignacio Fernández-Tobías, Shlomo Berkovsky, and Paolo Cremonesi. *Cross-Domain Recommender Systems*, pages 919–959. Springer US, Boston, MA, 2015.

[7] Ronald Chung, David Sundaram, and Ananth Srinivasan. Integrated personal recommender systems. In *Proceedings of the Ninth International Conference on Electronic Commerce*, ICEC '07, pages 65–74, New York, NY, USA, 2007. ACM.

[8] Paolo Cremonesi and Massimo Quadrana. Cross-domain recommendations without overlapping data: Myth or reality? In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 297–300, New York, NY, USA, 2014. ACM.

[9] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 126–135, New York, NY, USA, 2006. ACM.

[10] Robert Driskill and John Riedl. Recommender systems for e-commerce: Challenges and opportunities. In *Proceedings of the AAAI-99 Workshop on AI for Electronic Commerce, USA*, 1998.

[11] Manuel Enrich, Matthias Braunhofer, and Francesco Ricci. Cold-start management with cross-domain collaborative filtering and tags. In *International Conference on Electronic Commerce and Web Technologies*, pages 101–112. Springer, 2013.

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

[13] Zhou Fang, Sheng Gao, Bo Li, Juncen Li, and Jianxin Liao. Cross-domain recommendation via tag matrix transfer. In *IEEE International Conference on Data Mining Workshop, ICDMW 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pages 1235–1240. IEEE Computer Society, 2015.

[14] I Fernández-Tobías and Iván Cantador. Exploiting social tags in matrix factorization models for cross-domain collaborative filtering. *CEUR Workshop Proceedings*, 1245:34–40, 01 2014.

[15] Ignacio Fernández-Tobías, Iván Cantador, Marius Kaminskas, and Francesco Ricci. A generic semantic-based framework for cross-domain recommendation. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 25–32, 2011.

[16] R. Fletcher and C.M. Reeves. Function minimization by conjugate gradients. *The computer*, 7(2):149–154, 1964.

[17] Sheng Gao, Hao Luo, Da Chen, Shantao Li, Patrick Gallinari, and Jun Guo. Cross-domain recommendation via cluster-level latent factor model. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 161–176. Springer, 2013.

[18] Rainer Gemulla, Erik Nijkamp, Peter J. Haas, and Yannis Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *KDD*, pages 69–77, 2011.

[19] Ming He, Jiuling Zhang, Peng Yang, and Kaisheng Yao. Robust transfer learning for cross-domain collaborative filtering using multiple rating patterns approximation. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 225–233, New York, NY, USA, 2018. ACM.

[20] Ming He, Jiuling Zhang, and Jiang Zhang. Mindtl: Multiple incomplete domains transfer learning for information recommendation. *China Communications*, 14:218–236, 11 2017.

[21] Liang Hu, Yongheng Xing, Yanlei Gong, Kuo Zhao, and Feng Wang. Nonnegative matrix tri-factorization with user similarity for clustering in point-of-interest. *Neurocomputing*, 363:58–65, 2019.

[22] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[23] Ke Ji, Runyuan Sun, Xiang Li, and Wenhao Shu. Improving matrix approximation for recommendation via a clustering-based reconstructive method. *Neurocomput.*, 173(P3):912–920, January 2016.

[24] Xin Jin, Yanzan Zhou, and Bamshad Mobasher. A maximum entropy web recommendation system: combining collaborative and content features. In *KDD-2005*, pages 612–617, New York, NY, USA, 2005. ACM.

[25] Seung-Jun Kim, TaeHyun Hwang, and Georgios B. Giannakis. Sparse robust matrix tri-factorization with application to cancer genomics. In *3rd International Workshop on Cognitive Information Processing, CIP 2012, Baiona, Spain, May 28-30, 2012*, pages 1–6, 2012.

[26] Y. Kim and S. Choi. Weighted nonnegative matrix factorization. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1541–1544, April 2009.

[27] Yong-Deok Kim and Seungjin Choi. Weighted nonnegative matrix factorization. *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1541–1544, 2009.

[28] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[29] Vikas Kumar, Arun K Pujari, Sandeep Sahu, Venkateswara Kagita, and Vineet Padmanabhan. Collaborative filtering using multiple binary maximum margin matrix factorizations. *Information Sciences*, 380:1–11, 11 2016.

[30] Vikas Kumar, Arun K. Pujari, Sandeep Kumar Sahu, Venkateswara Rao Kagita, and Vineet Padmanabhan. Proximal maximum margin matrix factorization for collaborative filtering. *Pattern Recognition Letters*, 86:62 – 67, 2017.

[31] Daniel Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–91, 11 1999.

[32] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001.

[33] Seok Kee Lee, Yoon Ho Cho, and Soung Hie Kim. Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations. *Information Sciences*, 180(11):2142–2155, 2010.

[34] Bin Li. Cross-domain collaborative filtering: A brief survey. In *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*, pages 1085–1086. IEEE, 2011.

[35] Bin Li, Qiang Yang, and Xiangyang Xue. Can movies and books collaborate?: Cross-domain collaborative filtering for sparsity reduction. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, IJCAI'09, pages 2052–2057, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[36] Bin Li, Qiang Yang, and Xiangyang Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 617–624, New York, NY, USA, 2009. ACM.

[37] Greg Linden, Brent Smith, and Jeremy York. Industry report: Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Distributed Systems Online*, 4(1), 2003.

[38] Juntao Liu, Zhijun Yao, Yi Xiong, Wenyu Liu, and Caihua Wu. Learning conditional preference network from noisy samples using hypothesis testing. *Knowl.-Based Syst.*, 40:7–16, 2013.

[39] Antonis Loizou. *How to recommend music to film buffs: enabling the provision of recommendations from multiple domains*. PhD thesis, University of Southampton, 2009.

[40] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: A survey. *Decision Support Systems*, 74:12–32, 2015.

[41] Xin Luo, Yunni Xia, and Qingsheng Zhu. Incremental collaborative filtering recommender based on regularized matrix factorization. *Knowledge-Based Systems*, 27:271–280, 2012.

[42] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940. ACM, 2008.

[43] Hamed Masnadi-Shirazi and Nuno Vasconcelos. On the design of loss functions for classification: Theory, robustness to outliers, and savageboost. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, pages 1049–1056, USA, 2008. Curran Associates Inc.

[44] Prem Melville, Raymond J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pages 187–192, Edmonton, Alberta, 2002.

[45] Deyu Meng, Zongben Xu, Lei Zhang, and Ji Zhao. A cyclic weighted median method for l1 low-rank matrix factorization with missing entries. In *AAAI*, 2013.

[46] R. Mooney and L. Roy. Content–based book recommending using learning for text categorization. *In Proceedings of the 5th ACM Conference on Digital Libraries*, pages 195–204, June 2-7 2000.

[47] Orly Moreno, Bracha Shapira, Lior Rokach, and Guy Shani. Talmud: Transfer learning for multiple domains. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 425–434, New York, NY, USA, 2012. ACM.

[48] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2002.

[49] Edward Rolando Núúńez-Valdéz, Juan Manuel Cueva Lovelle, Oscar Sanjuán Martínez, Vicente García-Díaz, Patricia Ordoñez de Pablos, and Carlos Enrique Montenegro Marín. Implicit feedback techniques on recommender systems applied to electronic books. *Computers in Human Behavior*, 28(4):1186–1193, 2012.

[50] Rong Pan and Martin Scholz. Mind the gaps: weighting the unknown in large-scale one-class collaborative filtering. In *KDD '09'*, pages 667–676, New York, NY, USA, 2009. ACM.

[51] Sinno Jialin Pan, James T Kwok, Qiang Yang, et al. Transfer learning via dimensionality reduction. In *AAAI*, volume 8, pages 677–682, 2008.

[52] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[53] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, October 2010.

[54] Weike Pan. A survey of transfer learning for collaborative recommendation with auxiliary data. *Neurocomputing*, 177:447–453, 2016.

[55] Weike Pan, Nathan N. Liu, Evan W. Xiang, and Qiang Yang. Transfer learning to predict missing ratings via heterogeneous user feedbacks. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 2318–2323. AAAI Press, 2011.

[56] Weike Pan and Zhong Ming. Interaction-rich transfer learning for collaborative filtering with heterogeneous user feedback. *IEEE Intelligent Systems*, 29:48–54, 11 2014.

[57] Weike Pan, Shanchuan Xia, Zhuode Liu, Xiaogang Peng, and Zhong Ming. Mixed factorization for collaborative recommendation with heterogeneous explicit feedbacks. *Information Sciences*, 332:84 – 93, 2016.

[58] Weike Pan, Evan W. Xiang, Nathan N. Liu, and Qiang Yang. Transfer learning in collaborative filtering for sparsity reduction. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, pages 230–235. AAAI Press, 2010.

[59] Weike Pan, Evan W. Xing, and Qiang Yang. Transfer learning in collaborative filtering with uncertain ratings. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 662–668. AAAI Press, 2010.

[60] Weike Pan and Qiang Yang. Transfer learning in heterogeneous collaborative filtering domains. *Artificial intelligence*, 197:39–55, 2013.

[61] Aditya G. Parameswaran, Georgia Koutrika, Benjamin Bercovitz, and Hector Garcia-Molina. Recsplorer: recommendation algorithms based on precedence mining. In *SIGMOD Conference*, pages 87–98, 2010.

[62] Deuk Hee Park, Hyea Kyeong Kim, Il Young Choi, and Jae Kyeong Kim. A literature review and classification of recommender systems research. *Expert Systems with Applications*, 39(11):10059–10072, 2012.

[63] C. Porcel, A. Tejeda-Lorente, M.A. Martínez, and E. Herrera-Viedma. A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. *Information Sciences*, 184(1):1–19, 2012.

[64] Ian Porteous, Arthur U. Asuncion, and Max Welling. Bayesian matrix factorization with side information and dirichlet process mixtures. In *AAAI*, 2010.

[65] Jasson D. M. Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 713–719, New York, NY, USA, 2005. ACM.

[66] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer-Verlag, Berlin, Heidelberg, 1st edition, 2010.

[67] Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Are loss functions all the same? *Neural Comput.*, 16(5):1063–1076, May 2004.

[68] Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. To transfer or not to transfer. In *NIPS 2005 Workshop on Transfer Learning*, volume 898, 2005.

[69] Sebastian Ruder. Transfer learning - machine learning's next frontier. `http://ruder.io/transfer-learning/`, 2017.

[70] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pages 1257–1264, USA, 2007. Curran Associates Inc.

[71] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[72] K. H. Salman, Arun K. Pujari, Vikas Kumar, and Sowmini Devi Veeramachaneni. Combining swarm with gradient search for maximum margin matrix factorization. In *Proceedings of the 14th Pacific Rim International Conference on Trends in Artificial Intelligence*, PRICAI'16, pages 167–179, Switzerland, 2016. Springer.

[73] J. Salter and N. Antonopoulos. The cinemascreen recommender agent: A film recommender combining collaborative and content-based filtering. *IEEE Intelligent Systems*, 21(1):35–41, 2006.

[74] J. Ben Schafer, Joseph Konstan, and John Riedl. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, pages 158–166, New York, NY, USA, 1999. Association for Computing Machinery.

[75] Shunichi Seko, Takashi Yagi, Manabu Motegi, and Shin yo Muto. Group recommendation using feature space representing behavioral tendency and power balance among members. In *RecSys*, pages 101–108, 2011.

[76] Jiangfeng Shi, Mingsheng Long, Qiang Liu, Guiguang Ding, and Jianmin Wang. Twin bridge transfer learning for sparse collaborative filtering. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, pages 496–507, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[77] Yue Shi, Martha Larson, and Alan Hanjalic. Tags as bridges between domains: Improving recommendation with tag-induced cross-domain collaborative filtering. In Joseph A. Konstan, Ricardo Conejo, José L. Marzo, and Nuria Oliver, editors, *User Modeling, Adaption and Personalization*, pages 305–316, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[78] Ajit P. Singh and Geoffrey J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, page 650–658, New York, NY, USA, 2008. Association for Computing Machinery.

[79] Pradeep Kumar Singh, Pijush Kanti Dutta Pramanik, Avick Kumar Dey, and Prasenjit Choudhury. Recommender systems: an overview, research trends, and future directions. *International Journal of Business and Systems Research*, 15(1):14–52, 2021.

[80] V Sowmini., Venkateswara Rao Kagita, Arun K. Pujari, and Vineet Padmanabhan. Collaborative filtering by pso-based mmmf. In *SMC*, pages 569–574. IEEE, 2014.

[81] Nathan Srebro, Jason D. M. Rennie, and Tommi S. Jaakola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, pages 1329–1336. MIT Press, 2005.

[82] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009, January 2009.

[83] Xiaoyuan Su, Taghi M. Khoshgoftaar, Xingquan Zhu, and Russell Greiner. Imputation-boosted collaborative filtering using machine learning classifiers. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pages 949–950. ACM, 2008.

[84] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. Providing justifications in recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 38(6):1262–1272, 2008.

[85] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Matrix factorization and neighbor based algorithms for the netflix prize problem. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 267–274, New York, NY, USA, 2008. Association for Computing Machinery.

[86] Jian Tang, Jun Yan, Lei Ji, Ming Zhang, Shaodan Guo, Ning Liu, Xianfang Wang, and Zheng Chen. Collaborative users' brand preference mining across multiple domains from implicit feedbacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, 2011.

[87] Amit Tiroshi and Tsvi Kuflik. Domain ranking for cross domain collaborative filtering. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 328–333. Springer, 2012.

[88] E. Vozalis and K. G. Margaritis. Analysis of recommender systems' algorithms. In *The 6th Hellenic European Conference on Computer Mathematics & its Applications (HERCMA), Athens, Greece*, 2003.

[89] Zheng Wang, Yangqiu Song, and Changshui Zhang. Transferred dimensionality reduction. In *ECML/PKDD (2)*, pages 550–565, 2008.

[90] Pinata Winoto and Tiffany Tang. If you like the devil wears prada the book, will you also enjoy the devil wears prada the movie? a study of cross-domain recommendations. *New Generation Computing*, 26(3):209–225, 2008.

[91] M. Wu. Collaborative filtering via ensembles of matrix factorizations. In *KDD Cup and Workshop 2007*, pages 43–47. Max-Planck-Gesellschaft, August 2007.

[92] Xiaohua Sun, Fansheng Kong, and Song Ye. A comparison of several algorithms for collaborative filtering in startup stage. In *Proceedings. 2005 IEEE Networking, Sensing and Control, 2005.*, pages 25–28, 2005.

[93] Minjie Xu, Jun Zhu, and Bo Zhang. Nonparametric max-margin matrix factorization for collaborative prediction. In *Advances in Neural Information Processing Systems*, pages 64–72, 2012.

[94] Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno Jialin Pan. *Transfer Learning*. Cambridge University Press, 2020.

[95] Yu Zhang, Bin Cao, and Dit-Yan Yeung. Multi-domain collaborative filtering. *arXiv preprint arXiv:1203.3535*, 2012.

[96] Lili Zhao, Sinno Jialin Pan, Evan Wei Xiang, Erheng Zhong, Zhongqi Lu, and Qiang Yang. Active transfer learning for cross-system recommendation. In *AAAI*. Citeseer, 2013.

[97] Lili Zhao, Sinno Jialin Pan, and Qiang Yang. A unified framework of active transfer learning for cross-system recommendation. *Artificial Intelligence*, 245:38 – 55, 2017.

[98] Yi Zhen, Wu-Jun Li, and Dit-Yan Yeung. Tagicofi: Tag informed collaborative filtering. In *Proceedings of the Third ACM Conference on Recommender Systems*, RecSys '09, pages 69–76, New York, NY, USA, 2009. ACM.

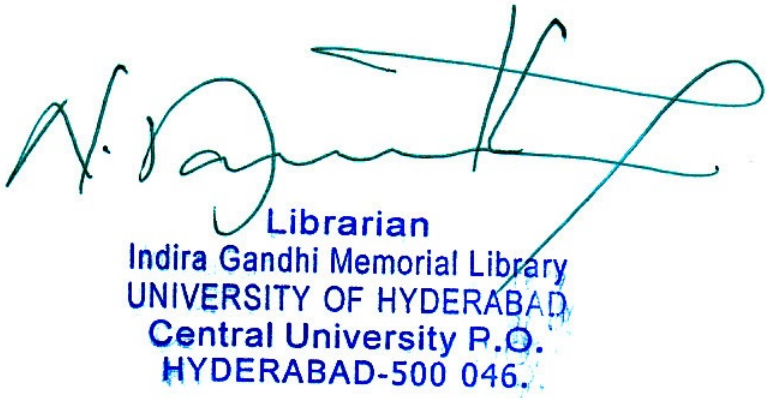
[99] Yancong Zhou, Xuemei Zhang, Yan Wang, and Bo Zhang. Transfer learning and its application research. *Journal of Physics: Conference Series*, 1920(1):012058, may 2021.

[100] Fuzhen Zhuang, Ping Luo, Hui Xiong, Yuhong Xiong, Qing He, and Zhongzhi Shi. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE Transactions on Knowledge and Data Engineering*, 22(12):1664–1678, 2009.

[101] Fuzhen Zhuang, Jing Zheng, Jingwu Chen, Xiangliang Zhang, Chuan Shi, and Qing He. Transfer collaborative filtering from multiple sources via consensus regularization. *Neural Networks*, 108:287 – 295, 2018.

Novel Techniques for Cross-Domain Recommendation

by Veeramachaneni Sowmini Devi



Submission date: 04-Jan-2022 11:35AM (UTC+0530)
Submission ID: 1737324553
File name: Sowmini-thesis-20-12-2021.pdf (3.61M)
Word count: 27371
Character count: 134415

Novel Techniques for Cross-Domain Recommendation

ORIGINALITY REPORT

37%	5%	37%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES Overall Similarity Index - 37 - (24+2+1) = 10% 04/1/2022

1	Sowmini Devi Veeramachaneni, Arun K Pujari, Vineet Padmanabhan, Vikas Kumar. "A Maximum Margin Matrix Factorization based Transfer Learning Approach for Cross-Domain Recommendation", Applied Soft Computing, 2019 Publication This Publication is by the Student. 04/1/2022	24%
2	"Pattern Recognition and Machine Intelligence", Springer Science and Business Media LLC, 2017 Publication Publication is by the Student. 04/1/2022	2%
3	V., Sowmini Devi, Venkateswara Rao Kagita, Arun K. Pujari, and Vineet Padmanabhan. "Collaborative filtering by PSO-based MMMF", 2014 IEEE International Conference on Systems Man and Cybernetics (SMC), 2014. Publication Student Publication. 04/1/2022	1%
4	Weike Pan. "A survey of transfer learning for collaborative recommendation with auxiliary data", Neurocomputing, 2016 Publication	1%

5	hdl.handle.net Internet Source	<1 %
6	csse.szu.edu.cn Internet Source	<1 %
7	Recommender Systems Handbook, 2015. Publication	<1 %
8	"PRICAI 2016: Trends in Artificial Intelligence", Springer Nature, 2016 Publication	<1 %
9	Submitted to Aristotle University of Thessaloniki Student Paper	<1 %
10	"Advances in Knowledge Discovery and Data Mining", Springer Science and Business Media LLC, 2013 Publication	<1 %
11	"Knowledge Science, Engineering and Management", Springer Science and Business Media LLC, 2017 Publication	<1 %
12	Submitted to University of North Texas Student Paper	<1 %
13	Fuzhen Zhuang, Jing Zheng, Jingwu Chen, Xiangliang Zhang, Chuan Shi, Qing He. "Transfer collaborative filtering from multiple	<1 %

sources via consensus regularization", Neural Networks, 2018
Publication

14	Cheng-Te Li, Chia-Tai Hsu, Man-Kwan Shan. "A Cross-Domain Recommendation Mechanism for Cold-Start Users Based on Partial Least Squares Regression", ACM Transactions on Intelligent Systems and Technology, 2018 Publication	<1 %
15	Ming He, Jiuling Zhang, Peng Yang, Kaisheng Yao. "Robust Transfer Learning for Cross-domain Collaborative Filtering Using Multiple Rating Patterns Approximation", Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018 Publication	<1 %
16	Ashish Kumar Sahu, Pragya Dwivedi. "User profile as a bridge in cross-domain recommender systems for sparsity reduction", Applied Intelligence, 2019 Publication	<1 %
17	Lecture Notes in Computer Science, 2015. Publication	<1 %
18	link.springer.com Internet Source	<1 %
19	repository.ntu.edu.sg Internet Source	<1 %

20	Ming He, Jiuling Zhang, Jiang Zhang. "MINDTL: Multiple incomplete domains transfer learning for information recommendation", China Communications, 2017 Publication	<1 %
21	Lecture Notes in Computer Science, 2014. Publication	<1 %
22	cris.brighton.ac.uk Internet Source	<1 %
23	research-information.bris.ac.uk Internet Source	<1 %
24	Pan, Weike. "A survey of transfer learning for collaborative recommendation with auxiliary data", Neurocomputing, 2016. Publication	<1 %
25	Yi Zhen, Wu-Jun Li, Dit-Yan Yeung. "TagiCoFi", Proceedings of the third ACM conference on Recommender systems - RecSys '09, 2009 Publication	<1 %
26	medium.com Internet Source	<1 %
27	"Database Systems for Advanced Applications", Springer Science and Business Media LLC, 2019 Publication	<1 %

28	"Machine Learning and Knowledge Discovery in Databases", Springer Science and Business Media LLC, 2013 Publication	<1 %
29	"Science of Cyber Security", Springer Science and Business Media LLC, 2019 Publication	<1 %
30	deepai.org Internet Source	<1 %
31	"Availability, Reliability, and Security in Information Systems and HCI", Springer Science and Business Media LLC, 2013 Publication	<1 %
32	Hongwei Zhang, Xiangwei Kong, Yujia Zhang. "Enhanced knowledge transfer for collaborative filtering with multi-source heterogeneous feedbacks", Multimedia Tools and Applications, 2021 Publication	<1 %
33	Lili Zhao, Sinno Jialin Pan, Qiang Yang. "A unified framework of active transfer learning for cross-system recommendation", Artificial Intelligence, 2017 Publication	<1 %
34	Qian Zhang, Dianshuang Wu, Jie Lu, Feng Liu, Guangquan Zhang. "A cross-domain recommender system with consistent	<1 %

information transfer", Decision Support Systems, 2017
Publication

35	Submitted to University of Hyderabad, Hyderabad Student Paper	<1 %
36	viztales.com Internet Source	<1 %
37	Abinash Pujahari, Dilip Singh Sisodia. "Pair-wise Preference Relation based Probabilistic Matrix Factorization for Collaborative Filtering in Recommender System", Knowledge-Based Systems, 2020 Publication	<1 %
38	Submitted to College of Engineering Trivandrum Student Paper	<1 %
39	Ayangleima Laishram, Vineet Padmanabhan. "Discovery of user-item subgroups via genetic algorithm for effective prediction of ratings in collaborative filtering", Applied Intelligence, 2019 Publication	<1 %
40	Hongwei Zhang, Xiangwei Kong, Yujia Zhang. "Selective Knowledge Transfer for Cross-domain Collaborative Recommendation", IEEE Access, 2021 Publication	<1 %

41	Submitted to Tennessee Technological University Student Paper	<1 %
42	"Knowledge Science, Engineering and Management", Springer Science and Business Media LLC, 2018 Publication	<1 %
43	Submitted to The University of Manchester Student Paper	<1 %
44	Vikas Kumar, Arun K. Pujari, Sandeep Kumar Sahu, Venkateswara Rao Kagita, Vineet Padmanabhan. "Proximal maximum margin matrix factorization for collaborative filtering", Pattern Recognition Letters, 2017 Publication	<1 %
45	Yakun Li, Jiadong Ren, Jiaomin Liu, Yixin Chang. "Deep sparse autoencoder prediction model based on adversarial learning for cross-domain recommendations", Knowledge-Based Systems, 2021 Publication	<1 %
46	Bin Li. "Transfer learning for collaborative filtering via a rating-matrix generative model", Proceedings of the 26th Annual International Conference on Machine Learning - ICML 09 ICML 09, 2009 Publication	<1 %

47	Paolo Cremonesi, Massimo Quadrana. "Cross-domain recommendations without overlapping data", Proceedings of the 8th ACM Conference on Recommender systems - RecSys '14, 2014 Publication	<1 %
48	Recommender Systems Handbook, 2011. Publication	<1 %
49	Si-Thin Nguyen, Hyun-Young Kwak, Seok-Hee Lee, Gwang-Yong Gim. "Using Stochastic Gradient Decent Algorithm For Incremental Matrix Factorization In Recommendation System", 2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2019 Publication	<1 %
50	Vikas Kumar, Arun K. Pujari, Sandeep Kumar Sahu, Venkateswara Rao Kagita, Vineet Padmanabhan. "Collaborative filtering using multiple binary maximum margin matrix factorizations", Information Sciences, 2017 Publication	<1 %
51	Weike Pan, Zhong Ming. "Interaction-Rich Transfer Learning for Collaborative Filtering with Heterogeneous User Feedback", IEEE Intelligent Systems, 2014 Publication	<1 %

52	K. G. Srinivasa, N. Pramod, K. R. Venugopal, L. M. Patnaik. "Effects of Information Filters", International Journal of Information Retrieval Research, 2012 Publication	<1 %
53	Pan, Weike, and Qiang Yang. "Transfer learning in heterogeneous collaborative filtering domains", Artificial Intelligence, 2013. Publication	<1 %
54	"User Modeling, Adaption and Personalization", Springer Science and Business Media LLC, 2011 Publication	<1 %
55	Ashish Kumar Sahu, Pragya Dwivedi. "Knowledge transfer by domain-independent user latent factor for cross-domain recommender systems", Future Generation Computer Systems, 2020 Publication	<1 %
56	Orly Moreno, Bracha Shapira, Lior Rokach, Guy Shani. "TALMUD", Proceedings of the 21st ACM international conference on Information and knowledge management - CIKM '12, 2012 Publication	<1 %
57	Zhetao Li, Pengpeng Qiao, Yuanxing Zhang, Kaigui Bian. "Adversarial Learning of Transitive Semantic Features for Cross-	<1 %

Domain Recommendation", 2019 IEEE Global Communications Conference (GLOBECOM), 2019
Publication

---

58 "Neural Information Processing", Springer Science and Business Media LLC, 2016 <1 %
Publication

---

59 Jie Lu, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, Guangquan Zhang. "Transfer learning using computational intelligence: A survey", Knowledge-Based Systems, 2015 <1 %
Publication

---

60 Jixu Chen, Xiaoming Liu. "Transfer learning with one-class data", Pattern Recognition Letters, 2014 <1 %
Publication

---

61 Submitted to University of Edinburgh <1 %
Student Paper

---

62 V. Sowmini Devi, Vineet Padmanabhan, Arun K. Pujari. "Chapter 10 A Matrix Factorization & Clustering Based Approach for Transfer Learning", Springer Science and Business Media LLC, 2017 <1 %
Publication

---

63 Xuejian Zhang, Zhongying Zhao, Chao Li, Yong Zhang, Jianli Zhao. "An Interpretable and <1 %

Scalable Recommendation Method Based on Network Embedding", IEEE Access, 2019
Publication

---

64 Zhou Fang, Sheng Gao, Bo Li, Juncen Li, Jianxin Liao. "Cross-Domain Recommendation via Tag Matrix Transfer", 2015 IEEE International Conference on Data Mining Workshop (ICDMW), 2015 <1 %
Publication

---

65 nozdr.ru <1 %
Internet Source

---

66 Ashish Kumar Sahu, Pragya Dwivedi. "Aligned Intrinsic User Factors Knowledge Transfer for Cross-domain Recommender Systems", Procedia Computer Science, 2020 <1 %
Publication

---

67 Qian Zhang, Wenhui Liao, Guangquan Zhang, Bo Yuan, Jie Lu. "A Deep Dual Adversarial Network for Cross-domain Recommendation", IEEE Transactions on Knowledge and Data Engineering, 2021 <1 %
Publication

---

68 Saraswat, Mala, and Shampa Chakraverty. "Exploiting rich side information sources of user and items for cross domain collaborative recommendation", 2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI), 2015. <1 %

Publication

69	"Intelligent Systems Design and Applications", Springer Science and Business Media LLC, 2020 Publication	<1 %
70	Wen Zhang, Qiang Wang, Taketoshi Yoshida, Jian Li. "RP-LGMC: Rating prediction based on local and global information with matrix clustering", Computers & Operations Research, 2021 Publication	<1 %
71	dspace.library.uvic.ca:8080 Internet Source	<1 %
72	"Neural Information Processing", Springer Science and Business Media LLC, 2018 Publication	<1 %
73	"Web Information Systems Engineering – WISE 2018", Springer Science and Business Media LLC, 2018 Publication	<1 %
74	"Web Technologies and Applications", Springer Science and Business Media LLC, 2013 Publication	<1 %
75	Mihai Datcu, Gottfried Schwarz, Corneliu Octavian Dumitru. "Chapter 7 Deep Learning Training and Benchmarks for Earth	<1 %

Observation Images: Data Sets, Features, and Procedures", IntechOpen, 2020
Publication

76	Zhaohong Deng, Kup-Sze Choi, Yizhang Jiang, Shitong Wang. "Generalized Hidden-Mapping Ridge Regression, Knowledge-Leveraged Inductive Transfer Learning for Neural Networks, Fuzzy Systems and Kernel Methods", IEEE Transactions on Cybernetics, 2014 Publication	<1 %
77	Danny Bickson. "Peer-to-peer secure multi-party numerical computation facing malicious adversaries", Peer-to-Peer Networking and Applications, 06/10/2009 Publication	<1 %
78	Submitted to Imperial College of Science, Technology and Medicine Student Paper	<1 %
79	Lecture Notes in Electrical Engineering, 2016. Publication	<1 %
80	Submitted to Macquarie University Student Paper	<1 %
81	Pan, Sinno Jialin, and Qiang Yang. "A Survey on Transfer Learning", IEEE Transactions on Knowledge and Data Engineering, 2010. Publication	<1 %

82	eprints.maynoothuniversity.ie Internet Source	<1 %

Exclude quotes	On	Exclude matches	< 14 words
Exclude bibliography	On		