TRANSFER LEARNING FOR ONE-CLASS RECOMMENDATION BASED ON MATRIX FACTORIZATION

by

RUIMING XIE

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Master of Philosophy
in Computer Science and Engineering

June 2013, Hong Kong

Authorization

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

RUIMING XIE

18 June 2013

TRANSFER LEARNING FOR ONE-CLASS RECOMMENDATION BASED ON MATRIX FACTORIZATION

by

RUIMING XIE

This is to certify that I have examined the above M.Phil. thesis

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by

the thesis examination committee have been made.

PROF. QIANG YANG, THESIS SUPERVISOR

PROF. MOUNIR HAMDI, HEAD OF DEPARTMENT

Department of Computer Science and Engineering

18 June 2013

ACKNOWLEDGMENTS

The past two years research would be much harder without the guidance of my supervisor, the assistance from my friends and the support of my parents.

First of all, I would like to express my deepest gratitude to my supervisor Prof. Qiang Yang for his kindly instructions on both my research and my life. Without the supervision of Prof. Yang, I would never have been able to explore and to exploit in the area of recommender system.

I would like to show my appreciation to Dr. Wei Xiang, who gave me great help while developing my research topic on selective knowledge transfer. I would also like to thank Yin Zhu, Erheng Zhong, Weike Pan, Lili Zhao, Nathan Liu and Sinno Pan, who have been working closely with me in the past two years and gave valuable suggestions on my study. In addition, I want to thank Dr. Hao Hu, Dr. Vincent Zheng, Kaixiang Mo, Ben Tan, Bin Wu, Naiyan Wang and other fellow students for their companionship.

I am grateful to Prof. Dit-Yan Yeung and Prof. Chi-Wing Wong for serving in the thesis examination committee. And many thanks to the CSE staffs, Mr. Isaac Ma and Ms. Connie Lau, for their administration work.

Finally, I am thankful to my parents for their unreserved support all these years.

TABLE OF CONTENTS

Title Page i

Authorization Page ii

Signature Page iii

Acknowledgments iv

Table of Contents v

List of Figures vii

List of Tables viii

Abstract ix

Chapter 1 Introduction **1**

1.1 Motivation 1

1.2 Contributions 3

1.3 Thesis Outline 4

Chapter 2 Background **5**

2.1 Collaborative Filtering 5

2.1.1 Memory-based CF 6

2.1.2 Model-based CF 7

2.1.3 Hybrid models 8

2.2 Transfer Learning 8

2.2.1 Transfer Learning for Collaborative Filtering 9

2.2.2 Negative Transfer 10

2.2.3 Large Scale Transfer Learning 11

Chapter 3 Gaussian Probabilistic Latent Semantic Analysis **13**

3.1 Problem Settings 13

3.2 Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis (GPLSA) 15

3.3 Transfer Learning for Collaborative Filtering (TLCF) 16
3.3.1 TLCF via Transferred GPLSA (TGPLSA) 16
3.4 Parallelization of the models 19
3.4.1 Parallelization of TGPLSA 19

Chapter 4 Selective Transfer Learning for Collaborative Filtering 21
4.1 Illustration of the STLCF settings 21
4.2 Logic behind STLCF 22
4.3 Selective Transfer Learning for Collaborative Filtering 23
4.3.1 Algorithm for STLCF 23
4.3.2 Derivation of STLCF 24
4.3.3 STLCF framework as a Whole 27

Chapter 5 Experiments 28
5.1 Data Sets and Experimental Settings 28
5.2 STLCF and Baselines Methods 30
5.3 Experimental Results 31
5.3.1 Performance Comparisons 31
5.3.2 Results on Long-Tail Users 33
5.3.3 STLCF with Multiple Source Domains 34
5.3.4 Parameters Analysis of STLCF 36
5.3.5 Convergence and Overfitting Test 36

Chapter 6 Conclusion and Future work 41

References 43

LIST OF FIGURES

3.1 Illustration of Selective Transfer Learning with multiple sources. 14

3.2 The graphical model for TGPLSA with one source domain and one target domain. 17

3.3 A demonstration of how the matrices are accessed during the computations of the GPLSA model 20

5.1 Change of the RMSEs with different τs. (Douban book to Movie) 36

5.2 Change of the RMSEs with different τs. (Netflix to Douban Movie) 37

5.3 Change of the RMSEs with different γs. (Douban book to Movie) 37

5.4 Change of the RMSEs with different γs. (Netflix to Douban Movie) 38

5.5 Change of the RMSEs when more and more weak learners join in the committee. 39

5.6 Change of αs when more and more weak learners join in the committee. 39

5.7 Change of the RMSEs with different numbers of latent topics. 40

LIST OF TABLES

2.1 Overview of STLCF in a big picture of Collaborative Filtering. 5

5.1 Datasets in our experiments. 29

5.2 Prediction performance of STLCF and the baselines. 30

5.3 Prediction performance of STLCF for Long-Tail Users on the D2 to D3 task. 33

5.4 Prediction performance of STLCF with multiple source domains containing much irrelevant information. 34

5.5 Prediction performance of STLCF with multiple source domains (Douban). 34

TRANSFER LEARNING FOR ONE-CLASS RECOMMENDATION BASED ON MATRIX FACTORIZATION

by

RUIMING XIE

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

ABSTRACT

One Class Recommender System aims at predicting users' future behaviors according to their historical actions. In these problems, the training data usually contain only binary data reflecting the behavior is happened or not. Thus, the data is sparser than traditional rating prediction problems. Recently, there are two ways to tackle the problem. 1, using knowledge transferred from other domains to mitigate the data sparsity problem. 2, providing methods to distinguish negative data and unlabeled data. However, it's not easy to simply transfer knowledge from source domain to target domain since their observations may be inconsistent. And without data from external source, distinguising negative and unlabeled data is sometimes infeasible.

In this paper, we propose a novel matrix tri-factorization method to transfer the useful information from source domain to target domain. Then we embed this method to a cluster-based SVD(singular value decomposition) framework. In several real-world datasets, we show our method achieve better prediction precision than other state-of-the-art methods. The cluster-based SVD method has been online for 2 months in a online shopping site, and its performance is among the best.

CHAPTER 1

INTRODUCTION

1.1 Motivation

Recommendation systems attempt to recommend items (e.g., movies, music, books, news, images, web pages, etc.) that are likely to be of interest to users. As a state-of-the-art technique for recommendation systems, collaborative filtering aims at predicting users' ratings on a set of items based on a collection of historical user-item preference records. In the real-world recommendation systems, although the item space is often very large, users usually rate only a small number of items. Thus, the available rating data can be extremely sparse for each user, which is especially true for newly started online services. Such data sparsity problem may make CF models overfit the limited observations and result in low-quality predictions.

In recent years, different transfer learning techniques have been developed to improve the performance of learning a model via reusing some information from other relevant systems for collaborative filtering [18, 38]. And with the increasing understandings of auxiliary data sources, some works (such as [11, 31]) start to explore the data from multiple source domains to achieve more comprehensive knowledge transfer. However, these previous methods over-trust the source data and assume that the source domains follow very similar distributions with the target domain, which is usually not true in the real-world applications, especially under the cross domain CF settings. For example, in a local music rating web site, natives may give trustful ratings for the traditional music; while in an international music rating web site, the ratings on those traditional music could be diverse due to the culture differences: those users with good culture background would constantly give trustful ratings, others could be inaccurate. If the target domain task is the music recommendation of a startup local web site, obviously we do not want all the international web site's data as source domain without selection. To better tackle the cross domain CF problems, we face the challenge to tell how consistent the data of target and source domains are and to adopt only those consistent source domain data while transferring knowledge.

Several research works (such as [6]) have been proposed to perform instance selection across domains for classification tasks based on empirical error. But they cannot be adopted to solve CF problems directly. Especially when the target domain is sparse, because of the limited observations of user's ratings on the items in the target domain, getting a low empirical error occasionally in the target domain does not mean the source domains are truly helpful in building a good model. In other words, the inconsistent knowledge from source domains may dominate the target domain model building and happen to fit the majority of the limited observations in the target domain, which is not desirable.

We take careful analysis on this problem and in our observation on the music rating example, some users, such as the domain experts, follow standard criteria to rate and hence share a consistent distribution over the mutual item set across domains. And further, we find this consistency can be better described by adding the variance of empirical error produced by the model. The smaller the variance of empirical error on predictions for a user, the more likely this user is consistent with those from other domains. And, we would like to adopt those who are more likely to share consistent preferences to perform knowledge transfer across domains. Based on this observation, we propose a new criterion using both the empirical error and its variance to capture the consistency between the source and the target domains. As an implementation, we embed this criterion into a boosting framework and propose a novel selective transfer learning approach for collaborative filtering (STLCF) [20]. STLCF works in an iterative way to adjust the importance of source instances, where those source data with low empirical error as well as low variance will be selected to help build target models.

1.2 Contributions

Our main contributions are summarized as follows:

- First, we find that selecting consistent auxiliary data for the target domain is important for the cross-domain collaborative filtering, while the consistency between source and target domains is influenced by multiple factors. To describe these factors, we propose a novel criterion to measure the consistency between source and target domains, based on both empirical error and its variance.
- Second, we propose a *selective* transfer learning framework for collaborative filtering - an extension of the boosting-based transfer learning algorithm that takes the above criterion into consideration while performing knowledge transfer, so that the sparseness issue in the CF problems can be better tackled.
- Third, the proposed framework is general, where different base models can be embedded. We propose an implementation based on Gaussian probability latent semantic analysis, which demonstrates the proposed framework can solve the sparseness problem on various real-world applications.
- Fourth, we investigate the distributed techniques and designed our proposed STLCF to well fit into them. Therefore, our work can be classified as an instance of large scale transfer learning. The parallel implementation demonstrates the power to handle the real-world tasks.

1.3 Thesis Outline

The rest of the thesis is organized as follows: we first provide the background of the research on Selective Transfer Learning for Cross Domain Recommendation, together with a very brief survey of the field in Chapter 2. Then, we discuss the technique grounds of the proposed framework in Chapter 3. We present the details of our proposed STLCF framework in Chapter 4 and the experiments in Chapter 5. Finally, we share our thoughts of possible future work and conclude the thesis in Chapter 6.

CHAPTER 2

BACKGROUND

In this chapter, we would like to give a brief review of the related literatures. We classify our work to be most related to the works in the areas of collaborative filtering.

In Table 2.1, we summarize the related works under the collaborative filtering context. To the best of our knowledge, no previous work for collaborative filtering has ever focused on the fine-grained analysis of knowledge transfer between source domains and the target domain, i.e., the selective transfer learning.

In the following, we would like to discuss the state-of-the-art methods for both Collaborative Filtering and Transfer Learning.

Table 2.1: Overview of STLCF in a big picture of Collaborative Filtering.

	Selective	Non-Selective
Transfer Learning	***STLCF***	RMGM [18], CMF [33], TIF [23], etc.
Non-Transfer Learning	–	MMMF [27], GPLSA [14], PMF [29], etc.

2.1 Collaborative Filtering

As an intelligent component in recommendation systems, Collaborative Filtering (CF) has gained extensive interest in both academia and industry. Generally speaking, CF is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that, if a person A has the same opinion as B on an issue, A is more likely to have B's opinion on a different issue x than to have the opinion on x of a randomly chosen person. For example, a collaborative filtering recommendation system for television tastes could make predictions about which television show a user should like given a partial list of this user's tastes (likes or dislikes, ratings, etc).

There are three types of CF: memory-based, model-based and hybrid.

2.1.1 Memory-based CF

This mechanism uses user rating data to compute the similarity between users or items. The similarity is then used for making recommendations. The memory-based method is used in many commercial systems, because it is easy to implement and is effective given plenty of records. Typical examples of this mechanism are neighborhood based CF and item-based/user-based top-N recommendations[34].

The advantages of this approach include:

- The explainability of the results, which is an important aspect of recommendation systems.
- It is easy to setup and use.
- New data can be added easily and incrementally.
- It need not consider contents of the items being recommended.
- The mechanism scales well with co-rated items.

However, there are several disadvantages with this approach:

- It requires plenty of human ratings.
- Its performance decreases when data gets sparse, which is a common phenomenon with web related items.
- Although it can efficiently handle new users, adding new items becomes more complicated since that representation usually relies on a specific vector space. That would require to include the new item and re-insert all the elements in the structure. This prevents the scalability of this approach.

2.1.2 Model-based CF

Models are developed using data mining, machine learning algorithms to find patterns based on training data. This approach has a more holistic goal to uncover latent factors that explain observed ratings. Most of the models are based on creating a classification or clustering technique to identify the users in the test set. Various models have been proposed, including factorization models [17, 23, 24, 26], probabilistic mixture models [15, 16], Bayesian networks [25] and restricted Boltzman machines [30].

There are several advantages with this paradigm:

- It handles the sparsity better than memory based ones.
- This helps with scalability with large data sets.
- It improves the prediction performance.
- It gives an intuitive rationale for the recommendations.

The disadvantage of this approach is the expensive model building. On the one hand, the modern recommendation system usually have petabytes of records as input; On the other hand, the convergence of most models requires intensive computation. One needs to have a tradeoff between prediction performance and scalability.

Given the accuracy of model-based CF, how to overcome the scalability issue has attracted much concern. With the rapid development of parallel computation, researchers have been exploring the use of parallel system to speed up the complex model building. For example in [5], the authors showed that a variety of machine learning algorithms including k-means, logistic regression, naive Bayes, SVM, PCA, gaussian discriminant analysis, EM and backpropagation (NN) could be speeded up by Google's map-reduce [8] paradigm.

Being aware of the computational infeasibility for most useful CF models in the real world settings, we would like to adopt the parallel computation framework in our implementation and experiments. We classified our Selective Transfer Learning for CF as model-based CF and use parallel computing to well handle the real world data and complex modeling.

2.1.3 Hybrid models

A number of applications [7] combine the memory-based and the model-based CF algorithms. These overcome the limitations of native CF approaches. It improves the prediction performance. Importantly, it overcomes the CF problems such as sparsity and loss of information. However, they have increased complexity and are expensive to implement.

2.2 Transfer Learning

Pan and Yang [22] surveyed the field of transfer learning. A major assumption in many machine learning and data mining algorithms is that the training and future data must be in the same feature space and have the same distribution. However, in many real-world applications, this assumption may not hold. For example, we sometimes have a classification task in one domain of interest, but we only have sufficient training data in another domain of interest, where the latter data may be in a different feature space or follow a different data distribution. In such cases, knowledge transfer, if done successfully, would greatly improve the performance of learning by avoiding much expensive data-labeling effort.

Recently, researchers propose the MultiSourceTrAdaBoost [37] to allow automatically selecting the appropriate data for knowledge transfer from multiple sources. The newest work TransferBoost [9] was proposed to iteratively construct an ensemble of classifiers via re-weighting source and target instance via both individual and task-based boosting. Moreover, EBBoost [32] suggests weight the instance based on the empirical error as well as its variance.

2.2.1 Transfer Learning for Collaborative Filtering

Some works on transfer learning are in the context of collaborative filtering. Mehta and Hofmann [21] consider the scenario involving two systems with shared users and use manifold alignment methods to jointly build neighborhood models for the two systems. They focus on making use of an auxiliary recommender system when only part of the users are aligned, which does not distinguish the consistency of users' preferences among the aligned users. Li *et al.* [19] designed a regularization framework to transfer knowledge of cluster-level rating patterns, which does not make use of the correspondence between the source and the target domains.

Our work is the first to systematically study *selective* knowledge transfer in the settings of collaborative filtering. Besides, we propose the novel factor - variance empirical error that is shown to be of much help in solving the real world CF problems.

2.2.2 Negative Transfer

Negative transfer happens when the source domain data and task contribute to the reduced performance of learning in the target domain. This could happen in the context of recommendation system. In the traditional transfer learning framework, if two users have common interests on the light music, we tend to believe they share the similar opinions on books. However, two users who have common interests on the light music may have quite different tastes on the books. In this case, if the transfer learning techniques are applied anyway, we can expect the bad performance. We call this phenomenon negative transfer.

Despite the fact that how to avoid negative transfer is a very important issue, little research work has been published on this topic. Rosenstein et al. [28] empirically showed that if two tasks are too dissimilar, brute-force transfer may hurt the performance of the target task. Some works have been exploited to analyze relatedness among tasks and task clustering techniques, such as [3] and [2], which may help provide guidance on how to avoid negative transfer automatically. Bakker and Heskes [2] adopted a Bayesian approach in which some of the model parameters are shared for all the tasks and others more loosely connected through a joint prior distribution that can be learned from the data. Thus, the data are clustered based on the task parameters, where tasks in the same cluster are supposed to be related to each other. Argyriou et al. [1] considered situations in which the learning tasks can be divided into groups. Tasks within each group are related by sharing a low-dimensional representation, which differs among different groups. As a result, tasks within a group can find it easier to transfer useful knowledge.

In our work, we investigate the negative transfer issue in the context of the recommendation system scenario. The key is to identify the items which would cause either much uncertainty in knowledge transfer or reduction in performance. Subsequently, we proposed a selective transfer learning framework for the model-based collaborative filtering tasks.

2.2.3 Large Scale Transfer Learning

So far, transfer learning has been mostly considered in the off-line learning settings, which do not emphasize the scalability and computation speed. Due to the rapid development of storage technique and flourish of internet services, the real world problems in recent recommendation systems are mostly based on some large data sets. Little work on large scale transfer learning has been published in previous literature, though it is badly desirable. To cope with the growing needs of today's recommendation system, we would like to investigate the parallel framework in our experiments. There are already some researchers working on the large scale machine learning, as you may find in a post from quora [1]. In our approach, we tried the Map-Reduce Framework and the Message Passing Interface (MPI).

Map-Reduce Framework

MapReduce is a framework for processing parallelizable problems in huge datasets using a large number of computers (nodes). A MapReduce program comprises a Map() procedure that performs filtering and sorting (such as sorting students by first name into queues, one queue for each name) and a Reduce() procedure that performs a summary operation (such as counting the number of students in each queue, yielding name frequencies).

- **"Map" step:** The master node takes the input, divides it into smaller sub-problems, and distributes them to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes the smaller problem, and passes the answer back to its master node.
- **"Reduce" step:** The master node then collects the answers to all the sub-problems and combines them in some way to form the output, i.e. the answer to the problem it was

[1] http://www.quora.com/What-are-some-software-libraries-for-large-scale-learning

originally trying to solve.

We will show that our methods can be plugged into the Map-Reduce framework for parallelization.

Message Passing Interface

Message Passing Interface (MPI) is a standardized and portable message-passing system designed by a group of researchers from academia and industry to function on a wide variety of parallel computers. Both point-to-point and collective communication are supported. As a dominant model used in high-performance computing, MPI's goals are high performance, scalability, and portability. There are several advantages of MPI:

- **Universality.** The message-passing model fits well on separate processors connected by a either fast or slow communication network.
- **Expressivity.** MPI exposes powerful interface to express the data parallel models.
- **Ease of debugging.** Because we write shared-memory (machine learning) models under MPI, the debugging process is relatively easier.
- **Performance.** As modern CPUs have become faster, management of their caches and the memory hierarchy has become the key to getting the most out of the machines. Message passing provides a way for the programmer to explicitly associate specific data with processers and thus allow the compiler and cache-management hardware to function fully.

Due to the flexibility of the protocol provided under MPI, it works well for the code development of learning algorithms. We will use MPI for implementation.

CHAPTER 3

GAUSSIAN PROBABILISTIC LATENT SEMANTIC ANALYSIS

3.1 Problem Settings

Suppose that we have a target task $\mathcal{D}$ where we wish to solve the rating prediction problem. Taking the regular recommendation system for illustration, $\mathcal{D}$ is associated with m_d users and n_d items denoted by $\mathcal{U}^d$ and $\mathcal{V}^d$ respectively. In this task, we observe a sparse matrix $\mathbf{X}^{(d)} \in \mathbb{R}^{m_d \times n_d}$ with entries x^d_{ui}. Let $\mathcal{R}^{(d)} = \{(u, i, r) : r = x^d_{ui}, \text{where } x^d_{ui} \neq 0\}$ denote the set of observed links in the system. For the rating recommendation system, r can either take numerical values, for example $[1, 5]$, or binary values $\{0, 1\}$. We aim to transfer knowledge from other N source domains $\mathcal{S} = \{S^t\}_{t=1}^{N}$ with each source domain S^t contains m^t_s users and n^t_s items denoted by $\mathcal{U}^{s^t}$ and $\mathcal{V}^{s^t}$. Similar to the target domain, each source domain S^t contain sparse matrices $\mathbf{X}^{(s^t)} \in \mathbb{R}^{m_{s^t} \times n_{s^t}}$ and observed links $\mathcal{R}^{(s^t)} = \{(u, i, r) : r = x^{s^t}_{ui},\ \text{where } x^{s^t}_{ui} \neq 0\}$.

The settings of STLCF are illustrated in Figure 3.1. As shown in the figure, the first row illustrates the case where items that are in the target domain also appear in the source domains. A real-world example is the rating prediction for the movies that appear in several web sites in various forms; the second row illustrates the case where users that are in the target domain also appear in the source domains. A real-world example is the Douban recommendation system [1], which provides music, book and movie recommendation for users. We adopt a setting commonly used in transfer learning for collaborative filtering: either the items or the users that

[1] `http://www.douban.com`

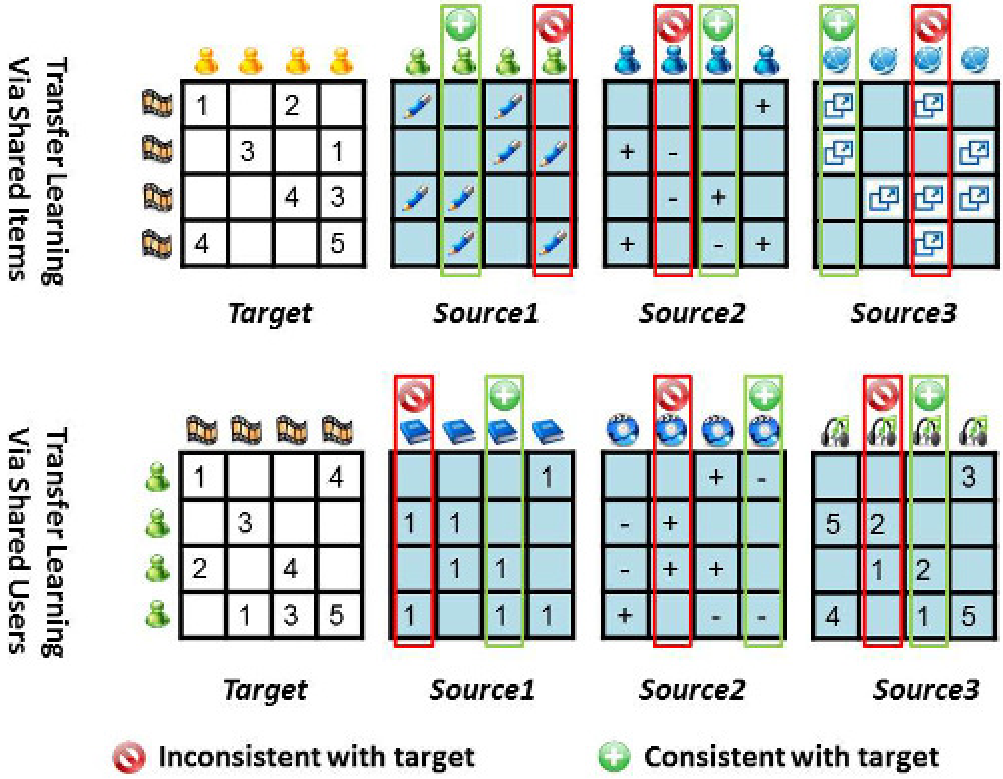

Figure 3.1: Illustration of Selective Transfer Learning with multiple sources.

are in the target domain also appear in the source domains. In the following derivation and description of our STLCF model, for the convenience of interpretation, we focus on the case that the user set is shared by both the target domain and the source domains. The case that the item set is shared can be easily tackled in a similar manner.

Under the assumption that the observation $\mathcal{R}^{(\{d,s^t\})}$ is obtained with u and i being independent, we formally define a co-occurrence model in both the target and the source domains to solve the collaborative filtering problem:

$$\begin{aligned}
Pr(x_{ui}^{\{d,s^t\}} = r, u, i) =& Pr(u)Pr(i \mid u)Pr(x_{ui}^{\{d,s^t\}} = r \mid u, i) \\
=& Pr(u)Pr(i)Pr(x_{ui}^{\{d,s^t\}} = r \mid u, i) \\
\propto& Pr(x_{ui}^{\{d,s^t\}} = r \mid u, i)
\end{aligned}$$

In the following, based on Gaussian probabilistic latent semantic analysis (GPLSA), we first briefly present a transfer learning model for collaborative filtering - transferred Gaussian probabilistic latent semantic analysis (TGPLSA) as an example, which is designed to integrate into our later proposed framework as a base model. After that, we present our selective transfer learning for collaborative filtering (STLCF) to perform knowledge transfer by analyzing the inconsistency between the observed data in target domain and the source domains. Careful readers shall notice that other than the TGPLSA example, STLCF is compatible to use various generative models as the base model.

3.2 Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis (GPLSA)

Following [14], for every user-item pair, we introduce hidden variables Z with latent topics z, so that user u and item i are rendered conditionally independent. With the observations of item set $\mathcal{V}$, user set $\mathcal{U}$ and rating set $\mathcal{R}$ in the source domain, we define:

$$Pr(x_{ui} = r|u, i) = \sum_{z} Pr(x_{ui} = r \mid i, z) Pr(z \mid u)$$

We further investigate the use of a Gaussian model for estimating $p(x_{ui} = r|u, i)$ by introducing $\mu_{iz} \in \mathcal{R}$ for the mean and σ_{iz}^2 for the variance of the ratings. With these, we define a Gaussian mixture model for a single domain as:

$$Pr(x_{ui} = r|u, i) = \sum_{z} Pr(z|u) Pr(r; \mu_{iz}, \sigma_{iz})$$

where $Pr(z|u)$ is the topic distribution over users, and $Pr(r; \mu_{iz}, \sigma_{iz})$ follows a Gaussian distribution:

$$Pr(r; \mu_{iz}, \sigma_{iz}) = \frac{1}{\sqrt{2\pi}\sigma_{iz}} \exp\left(-\frac{(r - \mu_{iz})^2}{2\sigma_{iz}^2}\right)$$

Maximum likelihood estimation amounts to minimize:

$$\mathcal{L} = -\sum_{r \in R} \sum_{x_{ui} \in \mathbf{X}} \log[Pr(x_{ui} = r \mid u, i; \theta)] \tag{3.1}$$

where θ refers to the parameters of a particular model. For CF problems with multiple ratings, we consider the loss function summing over all ratings R. Notice that the values of R are not limited in $[0, 1]$, it could also be in the range of $[1, 5]$.

Next, we extend GPLSA to the cross domain context to achieve transfer learning for collaborative filtering (TLCF).

3.3 Transfer Learning for Collaborative Filtering (TLCF)

When the target data $\mathbf{X}^{(d)}$ is sparse, it would be difficult to learn an accurate GPLSA model for rating prediction, as the model may overfit the rare observed data. An intuitive idea for solving the data sparsity problem is to borrow knowledge from other auxiliary systems, where plenty of the user behavior data are available. Thus, in this section, we first introduce the Transferred Gaussian Probabilistic Latent Semantic Analysis (TGPLSA) model, which is a nature extension of GPLSA. After that, we demonstrate how a family of boosting techniques can be adapted to help us to achieve *selective* knowledge transfer from multiple source domains.

3.3.1 TLCF via Transferred GPLSA (TGPLSA)

When the target data $\mathbf{X}^{(d)}$ is sparse, GPLSA may overfit the limited observed data. Following the similar idea in [36], we extend GPLSA to the Transferred Gaussian Probabilistic Latent Semantic Analysis (TGPLSA) model. Again we use s to denote index of the source domain where the knowledge come from, and d to denote the index of the target domain where the

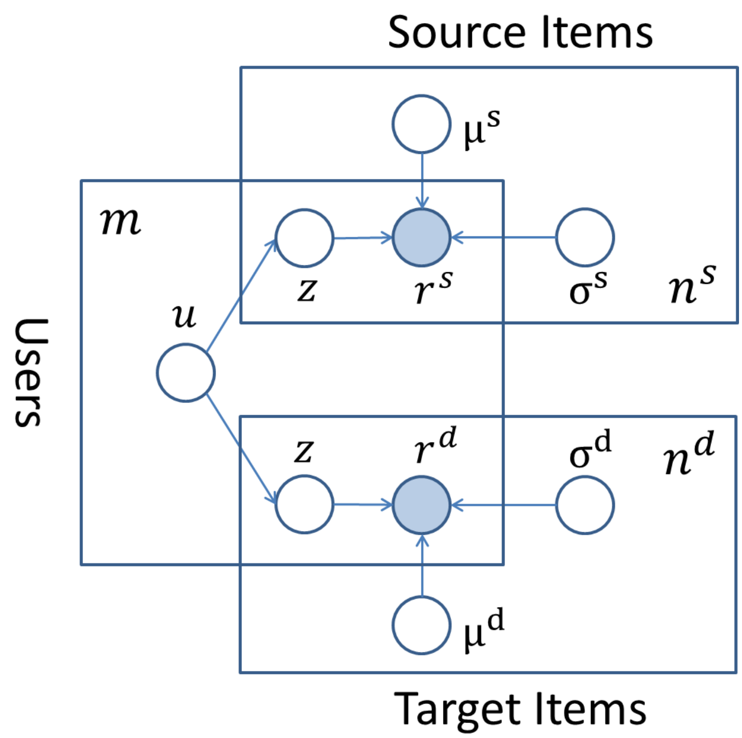

Figure 3.2: The graphical model for TGPLSA with one source domain and one target domain.

knowledge is received. The graphical representation of the proposed model TGPLSA is shown in Figure 3.2. For simplicity, we present the work with one source domain, and this model can be easily extended to multiple source domains. Moreover, we assume all the users appear in both the source domains and the target domain. Such scenarios are common in the real-world systems, like Douban[2].

TGPLSA jointly learn the two models for both the source domain and the target domain using a relative weight parameter[3] λ. Since the item sets $\mathcal{V}^s$ and $\mathcal{V}^d$ are different or even disjoint with each other, there could be inconsistency across domains as we discussed in Section 1. Clearly, the more consistent source and target domains are, the more help target task could get from source domain(s). We propose a weighted TLCF model (wTGPLSA) to further analyze this inconsistency in our work by learning item weight vectors $\mathbf{w}^s = \{w_i^s\}_{i=1}^{n^s}$ and $\mathbf{w}^d = \{w_i^d\}_{i=1}^{n^d}$ of the instances in source and target domain respectively. Then, the objective

[2]http://www.douban.com - a widely used online service in China, which provides music, book and movie recommendations.

[3]$\lambda \in (0, 1)$, which is introduced to represent the tradeoff of source and target information.

function in Eq.(3.1) can be extended as:

$$\mathcal{L} = -\sum_{r\in R}(\lambda \sum_{x^s_{ui}\in\mathbf{X}^s} \log(w^s_i \cdot Pr(x^s_{ui} = r \mid u^s, i^s; \theta^s))$$
$$+(1-\lambda) \sum_{x^d_{ui}\in\mathbf{X}^d} \log(w^d_i \cdot Pr(x^d_{ui} = r \mid u^d, i^d; \theta^d))) \tag{3.2}$$

In our setting, either $u^s = u^d$ or $i^s = i^d$. Besides, since we do not make any assumptions on the range of ratings in either source or target domains, the source domains and target domain may have different rating scales, as we show in the experiment in Section 5.3.3.

We adopt the expectation-maximization (EM) algorithm, a standard method for statistical inference, to find the maximum log-likelihood estimates of Eq. (3.2). For the convenience of presentation, in the following derivation, we denote $x^l_{ui}\!=\!r, u^l, i^l$ as x^l_{ui}.

In the E-step, we compute the posterior probabilities of the hidden variables, as follows:

$$Pr(z \mid x^s_{ui})\!=\!\frac{Pr(x^s_{ui} \mid z)Pr(z \mid u^s)}{\sum_{z'} Pr(x^s_{ui} \mid z')Pr(z' \mid u^s)}$$
$$Pr(z \mid x^d_{ui})\!=\!\frac{Pr(x^d_{ui} \mid z)Pr(z \mid u^d)}{\sum_{z'} Pr(x^d_{ui} \mid z')Pr(z' \mid u^d)} \tag{3.3}$$

Following the similar idea of TPLSA [36], we assume that a particular user's preferences on the latent topics are the same across different domains. Also notice that $Pr(z \mid u^d)$ and $Pr(z \mid u^s)$ reflect the users' preferences on the latent topics. Thus, we set $Pr(z \mid u^d)$ to be the same as $Pr(z \mid u^s)$ to build an information bridge for knowledge transfer across different systems.

In the M-step, we solve the maximization problem via the following equation:

$$Pr(z|u)\!=\!\frac{\sum_{l\in\{s,d\}}\lambda_l\sum_{r\in R}\sum_{x^l_{ui}\in\mathbf{X}^l_{u*}} w^l_i Pr(z|x^l_{ui})}{\sum_{z'}(\sum_{l\in\{s,d\}}\lambda_l\sum_{r\in R}\sum_{x^l_{ui}\in\mathbf{X}^l_{u*}} w^l_i Pr(z'|x^l_{ui}))} \tag{3.4}$$

And we obtain the following equations for the parameters of the Gaussian distribution:

$$\mu_{iz}^{l} = \frac{\sum_r \sum_{x_{ui}^l \in \mathbf{X}_{*i}^l} r \cdot Pr(z \mid x_{ui}^l)}{\sum_r \sum_{x_{ui}^l \in \mathbf{X}_{*i}^l} Pr(z \mid x_{ui}^l)} \tag{3.5}$$

$$(\sigma_{iz}^{l})^2 = \frac{\sum_r \sum_{x_{ui}^l \in \mathbf{X}_{*i}^l} (r - \mu_{iz})^2 Pr(z \mid x_{ui}^l)}{\sum_r \sum_{x_{ui}^l \in \mathbf{X}_{*i}^l} Pr(z \mid x_{ui}^l)} \tag{3.6}$$

After proceeding the EM algorithm by alternating E-step using Eq. (3.3) with M-step using Eq. (3.4), (3.5) and (3.6), finally we compute the expected result in the target domain as

$$\hat{x}_{ui}^{d} = E[r \mid u, i] = \int_R r Pr(r \mid u, i) dr \approx \sum_z Pr(z \mid u) \mu_{i,z}^{d} \tag{3.7}$$

3.4 Parallelization of the models

Due to the computational complexity as the readers may notice in previous sections, those latent feature models are hard to scale up on very large data sets. As we will discussed in Section 4, we would like to use the probabilistic latent feature models as base model. Therefore, we will need them to be applied to the real world datasets and converged fast. Thanks to the recent raised parallel computation techniques, such as Hadoop [35] and MPI [13], we could speed up the models mentioned in previous sections on some relatively large data sets.

3.4.1 Parallelization of TGPLSA

We present the logic behind the parallelization of TGPLSA. For the convenience of presentation, we show only the logic in a single domain. It is easy to extend it to multiple domains. As we can see in Figure 3.3, for a given user, only the values from a corresponding column in the user

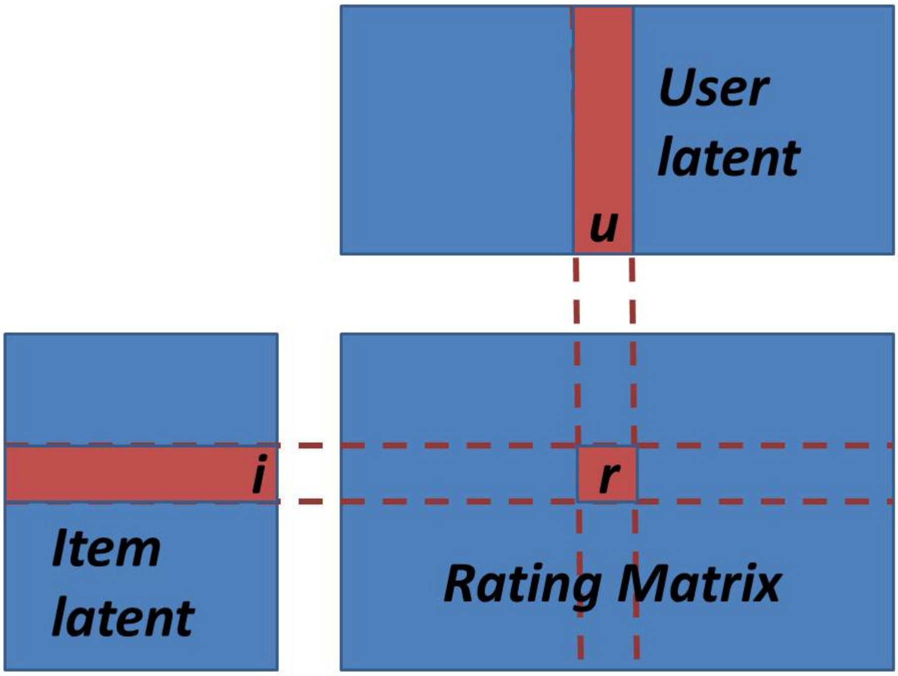

Figure 3.3: A demonstration of how the matrices are accessed during the computations of the GPLSA model

latent matrix and the value from the rating matrix are retrieved to calculate Equation 3.3 in the E-step. No other values are involved. Similar for the calculations in the M-step. The algorithm is therefore able to be plugged into the Map-Reduce framework.

We implement the parallel program under the MPI framework.

In the following chapters, we demonstrate how to adopt the boosting based framework and apply it to TGPLSA to achieve *selective* knowledge transfer from multiple source domains. Notice that other than TGPLSA, our proposed framework can be applied to other generative models to solve various problems.

CHAPTER 4

SELECTIVE TRANSFER LEARNING FOR COLLABORATIVE FILTERING

4.1 Illustration of the STLCF settings

A common assumption in previous transfer learning literature under the context of recommendation system is that the knowledge in source domain is always helpful in target domain tasks. In other words, they assume that all source domains have equally positive impacts on the target domain and the user or item preferences (depending on what is shared between the target domain and the source domains) are consistent in both source and target domains. Such an assumption is widely adopted in several works, such as Collective Matrix Factorization (CMF) [33] and its extensions.

Clearly, the above assumption is too strong for many real-world applications. Take the ratings of traditional Chinese music for example. In a Chinese local music rating web site, Chinese users may be critical for the traditional Chinese music; while in an international music rating web site, the ratings on those traditional Chinese music could be diverse due to the culture differences: those users with Chinese culture background would give trustful ratings, others could be inaccurate. Now, suppose there is a new Chinese local music rating web site, which is the target domain, and we try to transfer knowledge from both the existing Chinese local web site and the international web site to help the target domain rating prediction task. In this case, it would not be wise to treat the data from these two existing web sites equally. In addition, it would cause negative transfer if we use the entire data sets from the international Web sites without selection. Therefore, we should carefully analyze the source domain at both domain level and the instance level before applied to the target domain.

4.2 Logic behind STLCF

As we have discussed, using the source domain data without selection may harm the target domain learning. By proposing the selective knowledge transfer with the novel factors (empirical error and variance of empirical error), we come up with the details of Selective Transfer Learning framework for CF in this section.

As illustrated in the second example in Figure 3.1 where the domains have mutual user set, we would like to transfer knowledge of those items' records that consistently reflect the user's preferences. Because of our finding that the consistent records have small empirical error and variance, the selection shall consider these two factors. We embed these two factors into a boosting framework, where the source data with small empirical error and variance receive higher weights since they are consistent with the target data. This boosting framework models the cross-domain CF from two aspects:

- on one hand, we take more care of those mis-predicted target instances, following the traditional boosting logic;
- on the other hand, we automatically identify the consistency of the source domains during the learning process and selective use those source domains with more trustful information.

Based on the above consideration, our boosting based selective transfer learning framework for the collaborative filtering (STLCF) focuses on two levels: both domain level and instance level. At the domain level, STLCF boosts each source domain based on a notion of transferability proposed by Eaton *et al.* [10]. STLCF increases the weights of all instances from a source

Algorithm 1: Algorithm for Selective TLCF.

Input: $\mathbf{X}^d$, $\mathbf{X}^s$, T
$\mathbf{X}^d \in \mathbb{R}^{m \times n^d}$: the target training data
$\mathbf{X}^s \in \mathbb{R}^{m \times n^s}$: the auxiliary source data
G: the weighted TLCF model wTGPLSA
T: number of boosting iterations
Initialize: Initialize $\mathbf{w}^s : w_i^s \leftarrow \frac{1}{n^s}$, $\mathbf{w}^d : w_i^d \leftarrow \frac{1}{n^d}$
for $iter$ = 1 to T **do**
  Step 1: Apply G to generate a weak learner $G(\mathbf{X}^d, \mathbf{X}^s, \mathbf{w}^d, \mathbf{w}^s)$ that minimize Eq.(3.2)
  Step 2: Get weak hypothesis for both the d and s domains $h^{iter} : \mathbf{X}^d, \mathbf{X}^s \rightarrow \hat{\mathbf{X}}^d, \hat{\mathbf{X}}^s$
  Step 3: Calculate empirical error E^d and E^s using Eq.(4.4)
  Step 4: Calculate fitness weight β^{s_k} for each source domain s_k using Eq.(4.11)
  Step 5: Choose model weight α^{iter} via Eq.(4.7)
  Step 6: Update source item weight $\mathbf{w}^s$ via Eq.(4.10)
  Step 7: Update target item weight $\mathbf{w}^d$ via Eq.(4.9)
end for
Output: Hypothesis $Z = H(\mathbf{X}^{(d)}) = \Sigma_{t=1}^{T} \alpha^t h^t(\mathbf{X}^{(d)})$

domain if the source domain shows positive transferability, and decreases all weights if the source domain shows negative transferability. Simultaneously at instance level, STLCF adaptively increases the weight of mis-predicted instances to emphasize learning on those "tough" instances. Effectively, this adjusts the distribution of instance weights within each source domain. A formal description of the framework is given in Algorithm 1.

4.3 Selective Transfer Learning for Collaborative Filtering

4.3.1 Algorithm for STLCF

As shown in Algorithm 1, in each iteration, we apply base model TGPLSA over weighted instances to build a weak learner $G(\cdot)$ and hypothesis h^{iter}. Then to update the source and target item weights, domain level fitness weight β^{s_k} is chosen for each source domain s_k based on domain level consistency [10]. And α^{iter} for base model is also updated, considering empirical errors and variances. Accordingly, the weights of mis-predicted target items are increased and the weights of those less helpful source domains are decreased in each iteration. The final ensemble is given by an additive model, which gives larger weights to the hypotheses with lower errors. We provide a detailed derivation of STLCF in the rest of this section.

4.3.2 Derivation of STLCF

Target Function of a Single Weak Learner in STLCF

In previous works in collaborative filtering, the mean absolute error (MAE) is usually chosen as the loss function. In addition to the MAE loss, if we tolerate some prediction error τ, we define:

$$l_1(\mathbf{X}_{*i}, \hat{\mathbf{X}}_{*i}) = \begin{cases} -1, & \sum_{x_{ui} \in \mathbf{X}_{*i}} |\hat{x}_{ui} - x_{ui}| \leq \tau \cdot nnz(\mathbf{X}_{*i}) \\ +1, & \sum_{x_{ui} \in \mathbf{X}_{*i}} |\hat{x}_{ui} - x_{ui}| > \tau \cdot nnz(\mathbf{X}_{*i}) \end{cases} \tag{4.1}$$

where $nnz(\cdot)$ is the number of observed ratings. $\mathbf{X}_{*i}$ and $\hat{\mathbf{X}}_{*i}$ denote the true values and predictions respectively. We may also define the item level MAE error for target domain with respect to τ as:

$$\epsilon_i^d = l_1(\mathbf{X}_{*i}^d, \hat{\mathbf{X}}_{*i}^d) \tag{4.2}$$

To facilitate the optimization, we consider the following exponential loss for empirical risk minimization:

$$l_2(i) = l_2(\mathbf{X}_{*i}^d, \hat{\mathbf{X}}_{*i}^d) = e^{\epsilon_i^d} \tag{4.3}$$

As stated in previous section, the lower variance of empirical errors can provide more confident consistency estimation, we combine these factors and reformulate the loss function:

$$\mathcal{L} = \sum_{i=1}^{n^d} l_2(i) + \gamma \sqrt{\sum_{i>j}^{n^d} (l_2(i) - l_2(j))^2} \tag{4.4}$$

Above all, the model minimize the above quantity for some scalar $\gamma > 0$:

Additive Ensemble of Weak Learners

Assume that the function of interest $\mathcal{H}$ for prediction is composed of the hypothesis h^t from each weak learner. The function to be output would consist of the following additive model over the hypothesis from the weak learners:

$$\hat{x}_{ui}^d = f(x_{ui}^d) = \sum_{t=1} \alpha^t h^t(x_{ui}^d) \tag{4.5}$$

where $\alpha^t \in \mathbb{R}^+$.

Since we are interested in building an additive model, we assume that we already have a function $h(\cdot)$. Subsequently, we derive a greedy algorithm to obtain a weak learner $G^t(\cdot)$ and a positive scalar α^t such that $f(\cdot) = h(\cdot) + \alpha^t G^t(\cdot)$.

Derivation of Weak Learners

In the following derivation, for the convince of presentation, we omit the model index t, and use G to represent G^t, α to represent α^t.

By defining $\gamma_1 = (1 + (n-1)\gamma)$, $\gamma_2 = (2 - 2\gamma)$, α, $w_i^d = e^{l_1(h(\mathbf{X}_{*i}^d), \mathbf{X}_{*i}^d)}$ and $G_i^d = l_1(G(\mathbf{X}_{*i}^d), \mathbf{X}_{*i}^d)$, Eq.(4.4) can be equivalently posed as optimizing the following loss with respect to α:

$$\begin{aligned}\mathcal{L} \;=\;& \gamma_1\Big(\sum_{i\in I}(w_i^d)^2 e^{2\alpha} + \sum_{i\in J}(w_i^d)^2 e^{-2\alpha}\Big) + \gamma_2 \sum_{i>j:i,j\in I}^{n^d} w_i^d w_j^d e^{2\alpha} \\ & + \gamma_2 \sum_{i>j:i,j\in J}^{n^d} w_i^d w_j^d e^{-2\alpha} + \gamma_2 \sum_{i>j:i\in I,j\in J or i\in J,j\in I}^{n^d} w_i^d w_j^d\end{aligned} \tag{4.6}$$

For brevity, we define the following sets of indices as $I = \{i : G_i^d = +1\}$ and $J = \{i : G_i^d = -1\}$. Here J denotes the set of items whose prediction by $G(\cdot)$ falls into the fault tolerable range, while I denotes the rest set. By making the last transformation in Eq.(4.6) equal to zero, we get:

$$\alpha = \frac{1}{4}\log\left(\frac{(1-\gamma)(\sum_{i\in I} w_i^d)^2 + \gamma n^d \sum_{i\in I}(w_i^d)^2}{(1-\gamma)(\sum_{i\in J} w_i^d)^2 + \gamma n^d \sum_{i\in J}(w_i^d)^2}\right) \tag{4.7}$$

If we set $\gamma = 0$, then it is reduced to the form of AdaBoost:

$$\alpha = \frac{1}{4}\log\left(\frac{(\sum_{i\in I} w_i^d)^2}{(\sum_{i\in J} w_i^d)^2}\right) = \frac{1}{2}\log\left(\frac{(\sum_{i\in I} w_i^d)}{(\sum_{i\in J} w_i^d)}\right) \tag{4.8}$$

Finally, the updating rule for w_i^d is

$$w_i^d \leftarrow w_i^d e^{(-\alpha G_i^d)} \tag{4.9}$$

And for the instance weight w_i^d in the source domain, we can also adopt the similar updating rule in Eq.(4.9).

Other than the instance level selection discussed above, we also want to perform the domain level selection to penalize those domains that are likely to be irrelevant, so that the domains with more relevant instances speak loudly. Following the idea of task-based boosting [9], we further introduce a re-weighting factor β for each source domain to control the knowledge transfer. So we formulate the updating rule for w_i^s to be:

$$w_i^s \leftarrow w_i^s e^{(-\alpha G_i^s - \beta)} \tag{4.10}$$

where β can be set greedily in proportion to the performance gain of the single source domain transfer learning:

$$\beta = \frac{\sum w_i^d(\varepsilon_i - \vec{\varepsilon}_i)}{||\mathbf{w}^d||_1} \tag{4.11}$$

where ε_i is the training error of the transfer learning model, and $\vec{\varepsilon}_i$ is the training error of the non-transfer learning model, which utilizes only the observed target domain data.

4.3.3 STLCF framework as a Whole

To sum up, STLCF is a boosting based ensemble method, using certain generative single models as weak learner. We have seen an instance of generative model (TGPLSA) in Chapter 3. The ensemble function has been given as an weighted sum of several weak learners. We also presented the detailed update rules for each weak learner in Section 4.3.2.

CHAPTER 5

EXPERIMENTS

5.1 Data Sets and Experimental Settings

We evaluate the proposed method on four data sources: Netflix[1], Douban, IMDB[2], and Wikipedia[3] user editing records. The Netflix rating data contains more than 100 million ratings with values in $\{1, 2, 3, 4, 5\}$, which are given by more than 4.8×10^5 users on around 1.8×10^4 movies. Douban contains movie, book and music recommendations, with rating values also in $\{1, 2, 3, 4, 5\}$. IMDB hyperlink graph is employed as a measure of similarity between movies. In the graph, each movie builds links to its 10 most similar movies. The Wikipedia user editing records provide a $\{0, 1\}$ indicator of whether a user concerns or not about a certain movie.

The data sets used in the experiments are described as follows. For Netflix, to retain the original features of the users while keeping the size of the data set suitable for the experiments, we sampled a subset of $10,000$ users. In Douban data sets, we obtained 1.2×10^6 ratings on 7.5×10^3 music, 5.8×10^5 ratings on 3.5×10^3 books, and 1.4×10^6 ratings on 8×10^3 movies, given by 1.1×10^4 users. For both the IMDB data set and the Wikipedia data set, we filtered them by matching the movie titles in both the Netflix and the Douban Movie data sets. After pre-processing, the IMDB hyperlink data set contains $\sim 5 \times 10^3$ movies. The Wikipedia user editing records data set has 1.1×10^6 editing logs by 8.5×10^3 users on the same $\sim 5 \times 10^3$

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

[2] http://www.imdb.com

[3] http://en.wikipedia.org

Table 5.1: Datasets in our experiments.

Notation	Data Set	Data Type	Instances No.
D1	Douban Music	Rating [1,5]	1.2×10^6
D2	Douban Book	Rating [1,5]	5.8×10^5
D3	Douban Movie	Rating [1,5]	1.4×10^6
D4	Netflix	Rating [1,5]	1.8×10^4
D5	Wikipedia	Editing Log	1.1×10^6
D6	IMDB	Hyperlink	5.0×10^3

movies as IMDB data set. To present our experiments, we use the shorthand notations listed in Table 5.1 to denote the data sets.

We evaluate the proposed algorithm on five cross-domain recommendation tasks, as follows:

- The first task is to simulate the cross-domain collaborative filtering, using the Netflix data set. The sampled data is partitioned into two parts with disjoint sets of movies but identical set of users. One part consists of ratings given by $8,000$ movies with 1.6% density, which serves as the source domain. The remaining $7,000$ movies are used as the target domain with different levels of sparsity density.
- The second task is a real-world cross-domain recommendation, where the source domain is Douban Book and the target domain is Douban Movie. In this setting, the source and the target domains share the same user set but have different item sets.
- The third task is on Netflix and Douban data. We extract the ratings on the $6,000$ shared movies from Netflix and Douban Movie. Then we get 4.9×10^5 ratings from Douban given by 1.2×10^4 users with density 0.7%, and 10^6 ratings from Netflix given by 10^4 users with density 1.7%. The goal is to transfer knowledge from Netflix to Douban Movie. In this task, item set is identical across domains but user sets are totally different.
- The fourth task is to evaluate the effectiveness of the proposed algorithm under the context of multiple source domains. It uses both Douban Music and Douban Book as the source

Table 5.2: Prediction performance of STLCF and the baselines.

Datasets	Source sparseness	Target sparseness	Non-TL		Non-Selective TL		**Selective TL**	
			GPLSA	PMF	TGPLSA	CMF	STLCF(E)	STLCF(EV)
D4(Simulated) to D4(Simulated)	1.6%	0.1%	1.0012	0.9993	0.9652	0.9688	0.9596	**0.9533**
		0.2%	0.9839	0.9814	0.9528	0.9532	0.9468	**0.9347**
		0.3%	0.9769	0.9728	0.9475	0.9464	0.9306	**0.9213**
D2 to D3	1.5%	0.1%	0.8939	0.8856	0.8098	0.8329	0.7711	**0.7568**
		0.2%	0.8370	0.8323	0.7462	0.7853	0.7353	**0.7150**
		0.3%	0.7314	0.7267	0.7004	0.7179	0.6978	**0.6859**
D4 to D3	1.7%	0.1%	0.8939	0.8856	0.8145	0.8297	0.7623	**0.7549**
		0.2%	0.8370	0.8323	0.7519	0.7588	0.7307	**0.7193**
		0.3%	0.7314	0.7267	0.7127	0.7259	0.6982	**0.6870**

domains and transfer knowledge to Douban Movie domain.

- The fifth task varies the type of source domains. It utilizes the Wikipedia user editing records and IMDB hyperlink graph, together with Douban Movie as the source domains to perform rating predictions on the Netflix movie data set.

For evaluation, because we adopt the error rate style objective function in our optimization, we calculate the Root Mean Square Error (RMSE) on the heldout $\sim 30\%$ of the target data:

$$RMSE = \sqrt{\sum_{(u,i,x_{ui})\in T_E} (x_{ui} - \hat{x}_{ui})^2/|T_E|}$$

where x_{ui} and $\hat{x}_{ui}$ are the true and predicted ratings, respectively, and $|T_E|$ is the number of test ratings.

5.2 STLCF and Baselines Methods

We implement two variations of our STLCF method. STLCF(E) is an STLCF method that only take training error into consideration when performing selective transfer learning. STLCF(EV) not only considers training error, but also utilizes the empirical error variance.

To demonstrate the significance of our STLCF, we selected the following baselines:

- **PMF** [29] is a recently proposed method for missing value prediction. Previous work showed that this method worked well on the large, sparse and imbalanced data set.
- **GPLSA** [14] is a classical non-transfer recommendation algorithm.
- **CMF** [33] is proposed for jointly factorizing two matrices. Being adopted as a transfer learning technique in several recent works, CMF has been proven to be an effective cross-domain recommendation approach.
- **TGPLSA** is an uniformly weighted transfer learning model, which utilize all source data to help build the target domain model. It is used as one of the baselines because we adopt it as the base model of our boosting-based selective transfer learning framework.

Parameters for these baseline models are fine-tuned via cross validation.

5.3 Experimental Results

5.3.1 Performance Comparisons

We test the performance of our STLCF methods against the baselines. The results of the collaborative filtering tasks under three different target domain sparseness are shown in Table 5.2.

First, we observe that the non-transfer methods, i.e. GPLSA and PMF, fail to give accurate predictions, especially when the target domain is severely sparse. With the help of source domains, the (non-selective) transfer learning methods with equally weights on the source domains, like TGPLSA and CMF, can increase the accuracy of the rating predictions. And our selective transfer learning methods (i.e., STLCF(E) and STLCF(EV)) can do even better. The fact that our STLCF outperforms others is expected because by performing the *selective* knowledge transfer, we use the truly helpful source domain(s), which is designed to handle the sparseness issue in CF problems.

Second, comparing the two non-selective TLCF methods with the other two selective TLCF methods, we observe that on the last two real world tasks (D2 to D3 and D4 to D3) when the target domain is extremely sparse (say 0.1%), the improvement of accuracy achieved by our STLCF methods against the non-selective transfer learning methods is more significant than it does on the simulation data set based on Netflix (D4 to D4). Notice that the inconsistency of the target domain and the source domains on the simulation data sets is much smaller than that on the real-world cases. The experiment results show that our STLCF algorithm is effective in handling the inconsistency between the sparse target domain and the source domains.

Third, we notice that some factors, like empirical error variance, may affect the prediction. In Table 5.2, we compare our two STLCF methods, i.e., STLCF(E) and STLCF(EV) when the target domain sparsity is 0.1%. We can find that on the task "D2 to D3", i.e., Douban Book to Movie, STLCF(EV) is much better than STLCF(E). But on the task "D4(Simulated) to D4(Simulated)", the improvement of STLCF(EV) is not so significant against STLCF(E). These observations may be due to the domain consistency. For the tasks "D4(Simulated) to D4(Simulated)", both the source and target entities are movie ratings from Netflix data set, while the task "D2 to D3" tries to transfer the knowledge from a book recommendation system to the movie recommendation system, which may contain some domain specific items. When the target domain is very sparse, i.e. the user's ratings on the items are rare, there are chances to get high prediction accuracy occasionally on the observed data with a bad model on the source domains that are inconsistent with target domain. In this case, it is important to consider the variance of empirical error as well. Comparing to STLCF(E), STLCF(EV), which punishes the large variance, can better handle the domain inconsistency in transfer learning, especially when the target domain is sparse.

Table 5.3: Prediction performance of STLCF for Long-Tail Users on the D2 to D3 task.

Ratings per user	Non-TL	Non-Selective TL		**Selective TL i.e. STLCF**	
	GPLSA	TGPLSA	CMF	(E)	(EV)
1-5	1.1942	0.9294	0.9312	0.8307	**0.8216**
6-10	0.9300	0.7859	0.7929	0.7454	**0.7428**
11-15	0.8296	0.7331	0.7390	**0.7143**	0.7150
16-20	0.7841	0.7079	0.7113	**0.7042**	0.7050
21-25	0.7618	0.6941	0.6947	0.6942	**0.6910**
26-30	0.7494	0.6918	0.6884	0.6917	**0.6852**
31-35	0.7370	0.6909	0.6911	0.6915	**0.6818**
36-40	0.7281	0.6896	0.6856	0.6907	**0.6776**
41-45	0.7219	0.6878	0.6821	0.6890	**0.6740**
46-50	0.7187	0.6881	0.6878	0.6800	**0.6734**

5.3.2 Results on Long-Tail Users

To better understand the impact of STLCF with the help of the source domain, we conduct a fine-grained analysis on the performance improvement on Douban data sets, with Douban Book as source domain and Douban Movie as target domain. The results on different user groups in the target domain are shown in Table 5.3. In the table, method STLCF(E) does not punish the large variance of empirical error, while STLCF(EV) does. First, we observe that the STLCF models, i.e., STLCF(E) and STLCF(EV) can achieve better results on those long-tail users who have very few ratings in historical logs. Such fact implies that our STLCF methods could handle the long-tail users that really need a fine-grained analysis when performing knowledge transfer from source domains. Current CF models without any fine-grained analysis on the specific

Table 5.4: Prediction performance of STLCF with multiple source domains containing much irrelevant information.

Source Domain:		None	D3	D3 & D5	D3 & D6	D5 & D6	D3 & D5 & D6
Target	0.1%	0.9983	0.9789	0.9747	0.9712	0.9923	**0.9663**
(D4)	0.2%	0.9812	0.9625	0.9583	0.9572	0.9695	**0.9505**
sparseness	0.3%	0.9703	0.9511	0.9409	0.9464	0.9599	**0.9383**

Table 5.5: Prediction performance of STLCF with multiple source domains (Douban).

Source Domain:		None	D1	D2	D1 & D2
Target	0.1%	0.8856	0.7521	0.7568	**0.7304**
(D3)	0.2%	0.8323	0.7163	0.7150	**0.6904**
sparseness	0.3%	0.7267	0.6870	0.6859	**0.6739**

users usually fail to capture the preferences of the long-tail users, while our STLCF methods work well because they can selectively augment the weight of the corresponding source domain instances with respect to those long-tail cases at both instance level and domain level. Second, STLCF(EV) works better than STLCF(E) on those non-long-tail users, i.e., with more than 25 ratings per user in the historical log. This is expected because users with more ratings can benefit more from the error variance analysis to avoid negative knowledge transfer.

5.3.3 STLCF with Multiple Source Domains

We apply STLCF(EV) on the extremely sparse target movie domain, with two sets of source domains: one is composed of Douban Music and Douban Book, the other is composed of Douban Movie, IMDB hyperlink graph and Wikipedia user editing records. The results are in Table 5.5 and Table 5.4 respectively. We demonstrate our STLCF method can utilize multiple source domains of various types by handling the inconsistency between the target and the source domains.

First, for the Douban experiments shown in Table 5.5, we observe that comparing to only using either Douban Book or Douban Music as source domain, there are significant improvements when both of them are used. The result is expected because each of the source domains has its own parts of effective information for the target domain. For example, a user who show much interests in the movie “The Lord of the Rings” may have consistent preferences in its novel. In this case, with the help of more auxiliary sources, better results are expected.

Second, we explore the generalization of the choices of source domains by introducing domains like Wikipedia user editing records and IMDB hyperlink graph, which are not directly related to the target domain but still contain some useful information in helping the target task (Netflix rating prediction). The results are shown in Table 5.4. Comparing the results of the experiment that uses no source domain (non-transfer) to those that use source domains D5 & D6, we observe that although the Wikipedia user editing records or IMDB hyperlink graph is not closely related to the target domain and can hardly be adopted as source domains by previous transfer learning techniques, our STLCF method can still transfer useful knowledge successfully.

In addition, comparing the results of the experiment that uses single source domain D3 to those that use source domains D3 & D5, D3 & D6, or D3 & D5 & D6, we find that the Wikipedia user editing records or IMDB hyperlink graph could provide some useful knowledge that is not covered by the related movie source domains. Despite of the noise and heterogeneous setting, our STLCF method can still utilize these source domains to help the target domain tasks. As we have discussed in Chapter 4, our STLCF performs selective transfer learning at both domain level and instance level. On one hand, the domain level selective transfer can block the noisy information globally. As we can see, D5 & D6 are noisy and therefore contain much data that are inconsistent with the observed data in the target domain, therefore the overall transfer of D5 & D6 is penalized. On the other hand, the instance level selective transfer learning can further eliminate the affections of those irrelevant source instances.

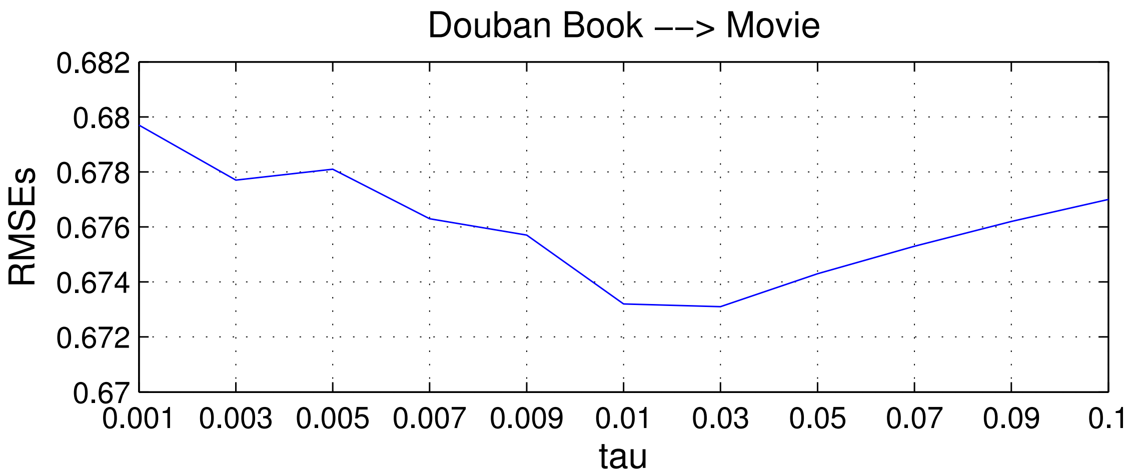

Figure 5.1: Change of the RMSEs with different τs. (Douban book to Movie)

Above all, our STLCF is highly adaptive to utilize source domains that are relatively inconsistent with the target domain, even when the target domain is rather sparse.

5.3.4 Parameters Analysis of STLCF

There are two parameters in our STLCF, i.e., the prediction error threshold τ and the empirical error variance weight γ. Since τ and γ are independent, we fix one and adjust another.

We fix the empirical error variance weight to be $\gamma = 0.5$ and adjust the parameter τ. Based on our results shown in Figure 5.1 and Figure 5.2 for two different transfer learning tasks, the model has good performance when τ is of order 10^{-2}. We also tuned the parameter γ, which balances the empirical error and its variance. We fix the prediction error threshold to be $\tau = 0.03$ in tuning γ. As shown in Figure 5.3 and Figure 5.4, when we vary the parameter γ from 0 to 1 in two settings, the best choices of γ are found to be around $0.4 - 0.5$.

5.3.5 Convergence and Overfitting Test

Figure 5.5 shows the RMSEs of STLCF(EV) as the number of weak learners changes on the Douban Book to Movie task. From the figure, we observe that STLCF(EV) converges well after

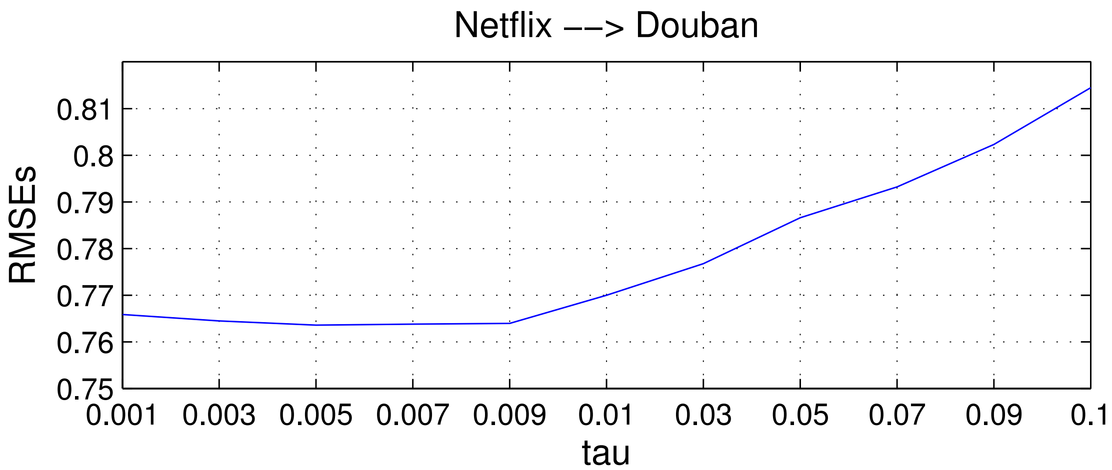

Figure 5.2: Change of the RMSEs with different τs. (Netflix to Douban Movie)

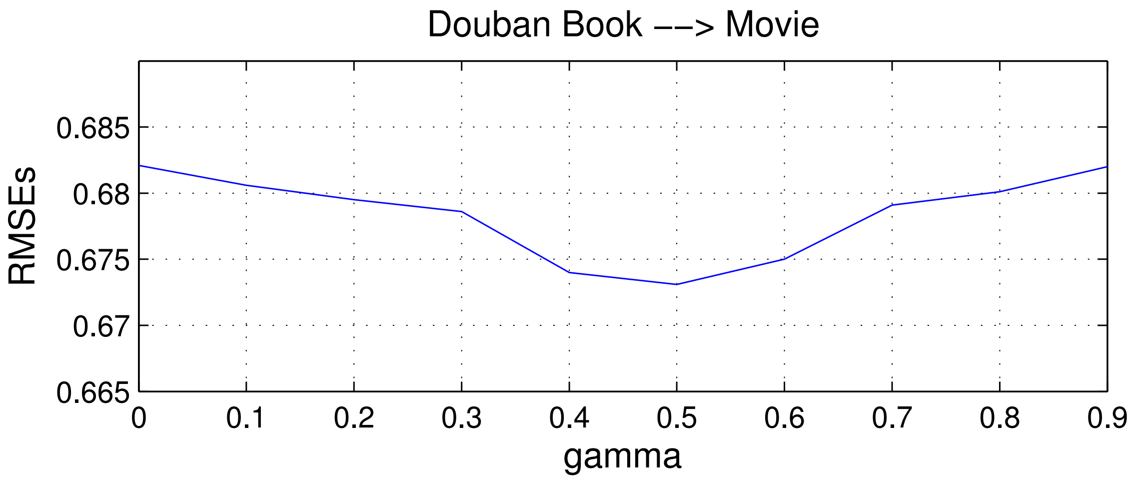

Figure 5.3: Change of the RMSEs with different γs. (Douban book to Movie)

40 iterations. In Figure 5.6, we can find that the corresponding α also converge to around 0.68 after 40 iterations as well. Empirically, we find STLCF converges in less than 50 iterations.

The number of latent topics of the base learner TGPLSA reflects the model's ability to fit training data. When we keep increasing the number of latent topics, the model tends to better fit the training data. But if the number of latent topics is too large, the model may suffer from overfitting.

We investigate the overfitting issue by plotting the training and testing RMSEs of the non-transfer learning model GPLSA, the non-selective transfer learning model TGPLSA and our selective transfer learning model STLCF(EV) over different numbers of latent topics in Figure 5.7. The data sparsity for the target domain is around 0.3%.

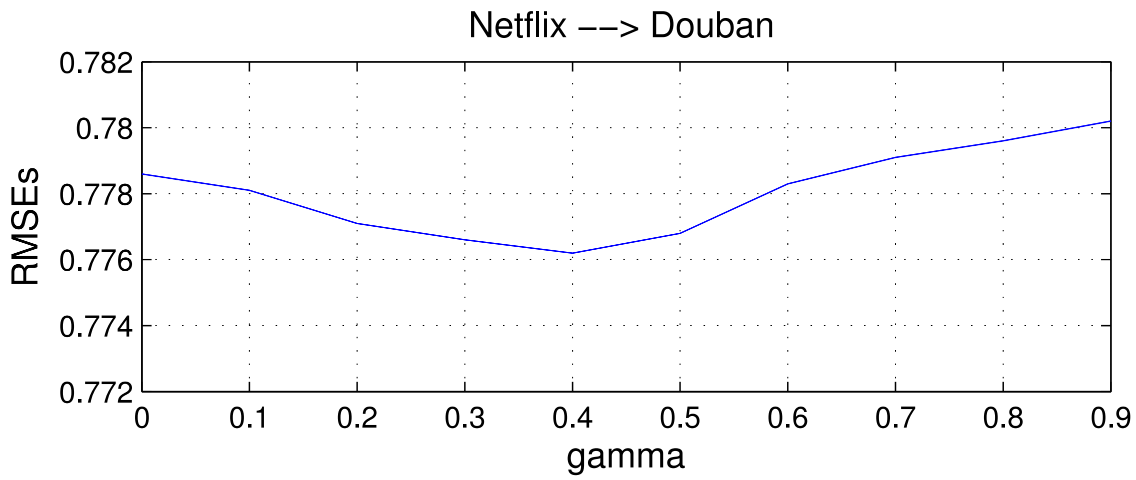

Figure 5.4: Change of the RMSEs with different γs. (Netflix to Douban Movie)

We can observe that comparing to our STLCF, the training RMSEs of GPLSA and TGPLSA decrease faster, while their testing RMSEs go down slower. When k is about 50, the testing RMSEs of GPLSA start to go up. And for TGPLSA, its testing RMSEs also go up slightly when k is larger than 75. But the testing RMSEs of our STLCF keep decreasing until $k = 125$ and even when k is larger than 125, the raise of our STLCF's testing RMSEs is not obvious. Clearly when the target domain is very sparse, our STLCF method is more robust against the overfitting, by inheriting the advantage from boosting techniques and the fine-grained selection on knowledge transfer.

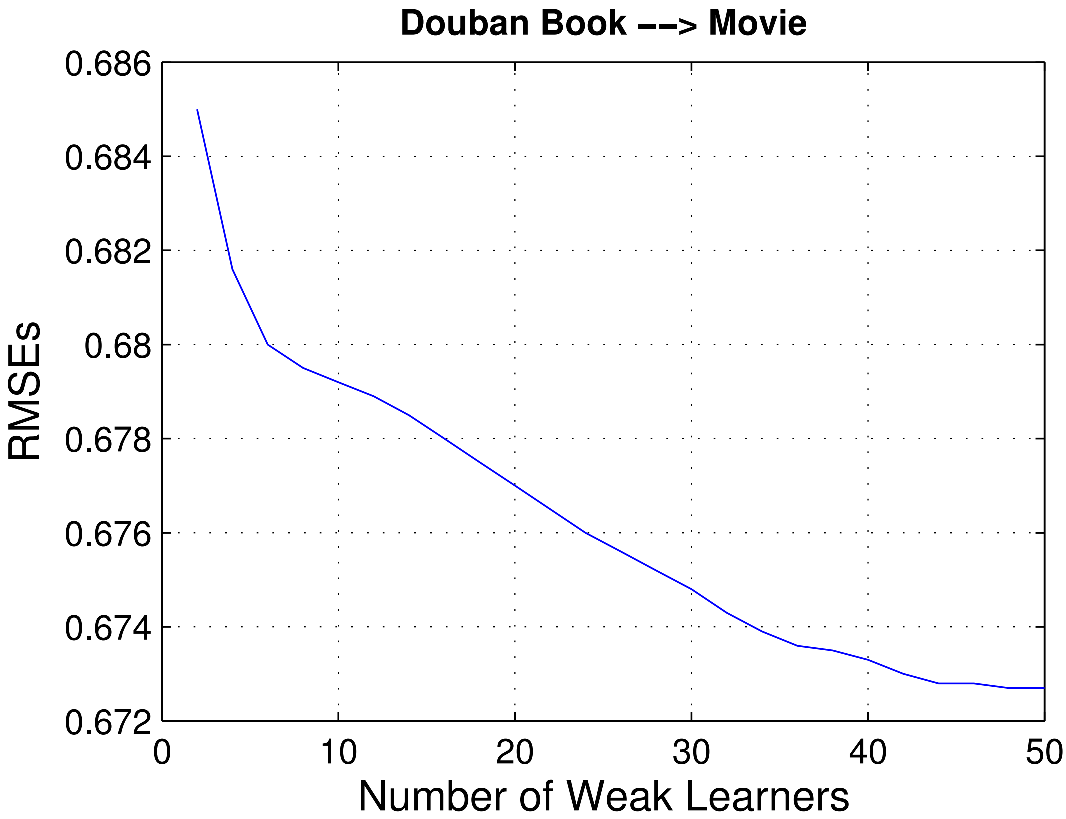

Figure 5.5: Change of the RMSEs when more and more weak learners join in the committee.

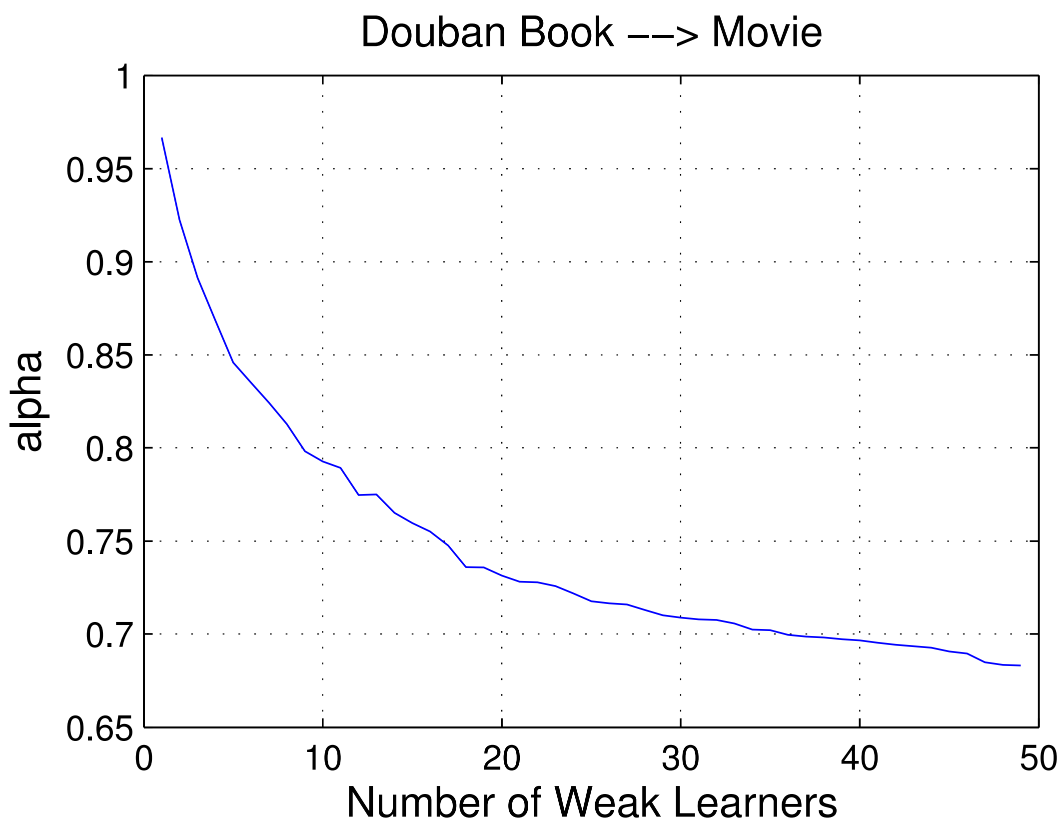

Figure 5.6: Change of αs when more and more weak learners join in the committee.

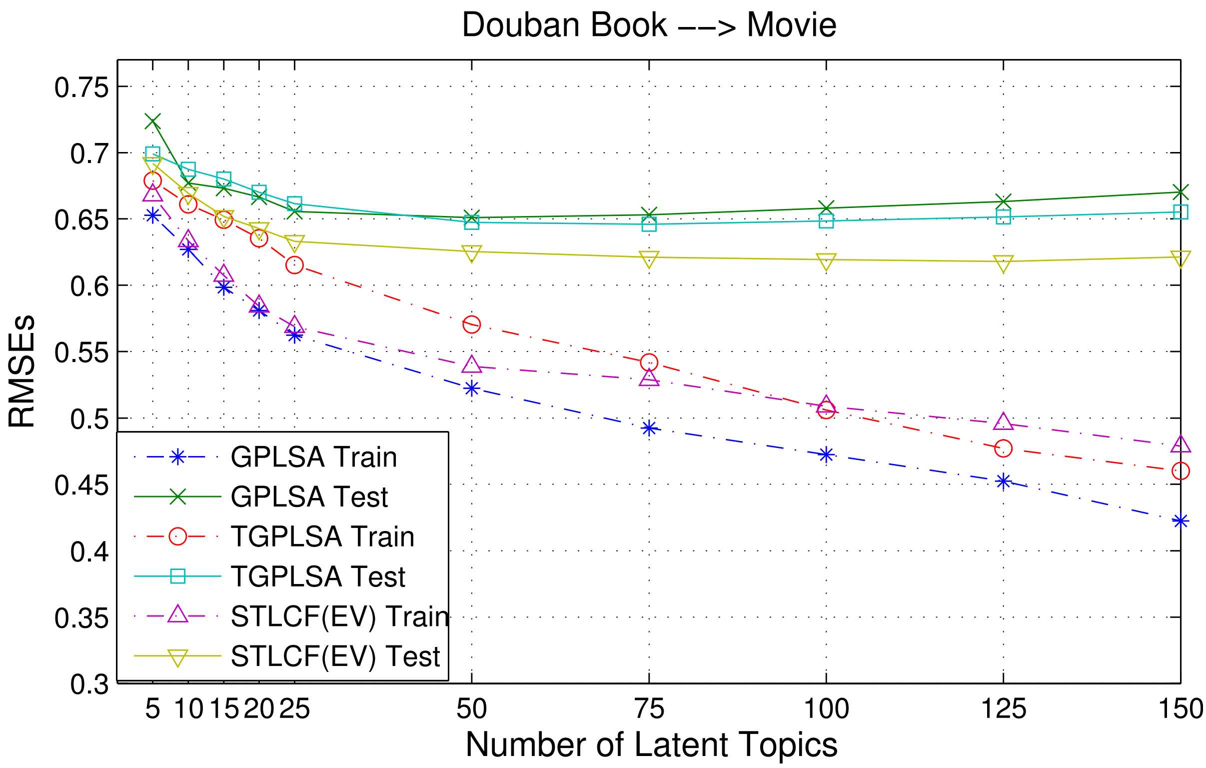

Figure 5.7: Change of the RMSEs with different numbers of latent topics.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, we proposed to perform *selective* knowledge transfer for CF problems and came up with a systematical study on how the factors such as variance of empirical errors could leverage the selection. We found although empirical error is effective to model the consistency across domains, it would suffer from the sparseness problem in CF settings. By introducing a novel factor - variance of empirical errors to measure how trustful this consistency is, the proposed criterion can better identify the useful source domains and the helpful proportions of each source domain. We embedded this criterion into a boosting framework to transfer the most useful information from the source domains to the target domain. The experimental results on real-world data sets showed that our selective transfer learning solution performs better than several state-of-the-art methods at various sparsity levels. Furthermore, comparing to existing methods, our solution works well on long-tail users and is more robust to overfitting.

However, we notice that there are limitations in the work. First, in STLCF, the knowledge transfer is item-based. That is, each item / user is evaluated independently. Therefore, the implicit relationships between items / users are omitted. Second, The computational cost of STLCF is expensive, even though the parallel implementation makes it possible to run on large clusters. Third, we require the full correspondence on either user set or the item set as a bridge for the knowledge transfer. This requirement limits the applications of STLCF in the real world, because most of the real system will not be able to provide the full correspondence information. Fourth, we are aware that although the STLCF performs well on the long-tails (target domain tasks with very limited observations, for example the experiment in Section 5.3.2), it still can not handle the case where no record of target domains is exist.

We believe the Selective Transfer Learning has practical applications in the real world and would be a promising research topic. STLCF is our initial attempt on this topic. In the future to make Selective Transfer Learning be more robust, we propose the following approaches:

- **Model-based Selective Transfer Learning.** Instead of item-based knowledge transfer, we would like to explore the model-based transfer. That is, the domain information is first generalized as model and then be applied to later tasks. This will improve the universality of the source domain information and reduce the storage demand, as the information is generalized by models.

- **Relationship Regularized Selective Transfer Learning.** On the one hand, with the rapid growing of social networks in the internet, we have access to plenty of online user relationship. On the other hand, previous researches on taxonomy have made it possible to build relationship between items. Relationship regularized selection of helpful knowledge is naturally the next work.

- **Selection of Domain Correspondence.** Due to either record corruption or the absence of data in the industry, it is not always possible to obtain the full correspondence between the source domains and the target domain for knowledge transfer. To make Selective Transfer Learning be practical, we want to research on the selection of correspondence between domains when only part of it could be helpful. For example, in the settings where the user set is shared among the source and the target domains, we would like to select parts of the users during the transfer learning processes.

- **Boosting in Multi-Dimension.** The technique in this article can be viewed as boosting over either the item or user dimension. Can we extend it to multi-dimensional boosting? For example, would the interest of a user towards certain items evolute over time? With the evolution of user interests, is it possible to make a better prediction on the future ratings?

REFERENCES

[1] Andreas Argyriou, Andreas Maurer, and Massimiliano Pontil. An algorithm for transfer learning in a heterogeneous environment. In *Machine Learning and Knowledge Discovery in Databases*, pages 71–85. Springer, 2008.

[2] Bart Bakker and Tom Heskes. Task clustering and gating for bayesian multitask learning. *The Journal of Machine Learning Research*, 4:83–99, 2003.

[3] Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Learning Theory and Kernel Machines*, pages 567–580. Springer, 2003.

[4] Wolfram Burgard and Dan Roth, editors. *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*. AAAI Press, 2011.

[5] Cheng Chu, Sang Kyun Kim, Yi-An Lin, YuanYuan Yu, Gary Bradski, Andrew Y Ng, and Kunle Olukotun. Map-reduce for machine learning on multicore. *Advances in neural information processing systems*, 19:281, 2007.

[6] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. In Ghahramani [12], pages 193–200.

[7] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.

[8] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[9] Eric Eaton and Marie desJardins. Selective transfer between learning tasks using task-based boosting. In Burgard and Roth [4].

[10] Eric Eaton, Marie desJardins, and Terran Lane. Modeling transfer relationships between learning tasks for improved inductive transfer. In *The European Conference on Machine Learning*, pages 317–332, 2008.

[11] Hoda Eldardiry and Jennifer Neville. Across-model collective ensemble classification. In Burgard and Roth [4].

[12] Zoubin Ghahramani, editor. *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*. ACM, 2007.

[13] William Gropp, Ewing L Lusk, and Anthony Skjellum. *Using MPI-: Portable Parallel Programming with the Message Passing Interface*, volume 1. MIT press, 1999.

[14] Thomas Hofmann. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *SIGIR*, pages 259–266, 2003.

[15] Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.

[16] Rong Jin, Luo Si, Chengxiang Zhai, and Jamie Callan. Collaborative filtering with decoupled models for preferences and ratings. In *Proceedings of CIKM 2003*, 2003.

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

[18] 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*, pages 2052–2057, 2009.

[19] Bin Li, Qiang Yang, and Xiangyang Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *International Conference on Machine Learning*, volume 26, 2009.

[20] Zhongqi Lu, Erheng Zhong, Lili Zhao, Evan Xiang, Weike Pan, and Qiang Yang. Selective transfer learning for cross domain recommendation. In *SIAM International Conference on Data Mining*, 2013.

[21] Bhaskar Mehta and Thomas Hofmann. Cross system personalization and collaborative filtering by learning manifold alignments. In *KI 2006: Advances in Artificial Intelligence*, pages 244–259. 2007.

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

[23] Weike Pan, Evan W. Xiang, and Qiang Yang. Transfer learning in collaborative filtering with uncertain ratings. In *AAAI*, 2012.

[24] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. *Proceedings of KDD Cup and Workshop*, 2007.

[25] David M. Pennock, Eric Horvitz, Steve Lawrence, and C. Lee Giles. Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In *Proc. of UAI*, pages 473–480, 2000.

[26] Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (ACM TIST)*, 3:19:1–19:22, May 2012.

[27] Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pages 713–719. ACM, 2005.

[28] 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.

[29] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *NIPS*, 2007.

[30] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey E. Hinton. Restricted boltzmann machines for collaborative filtering. In Ghahramani [12], pages 791–798.

[31] Xiaoxiao Shi, Jean-François Paiement, David Grangier, and Philip S. Yu. Learning from heterogeneous sources via gradient boosting consensus. In *SDM*, pages 224–235. SIAM / Omnipress, 2012.

[32] Pannagadatta K. Shivaswamy and Tony Jebara. Empirical bernstein boosting. *Journal of Machine Learning Research - Proceedings Track*, 9:733–740, 2010.

[33] Ajit Paul Singh and Geoffrey J. Gordon. Relational learning via collective matrix factorization. In *KDD*, pages 650–658, 2008.

[34] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:4, 2009.

[35] Tom White. *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2012.

[36] Gui-Rong Xue, Wenyuan Dai, Qiang Yang, and Yong Yu. Topic-bridged plsa for cross-domain text classification. In Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong, editors, *SIGIR*, pages 627–634. ACM, 2008.

[37] Yi Yao and Gianfranco Doretto. Boosting for transfer learning with multiple sources. In *CVPR*, pages 1855–1862, 2010.

[38] Yu Zhang, Bin Cao, and Dit-Yan Yeung. Multi-domain collaborative filtering. In *UAI*, pages 725–732, 2010.