Design and Evaluation of Factorization-Based Algorithms for Recommendation Systems

Yali Du

Faculty of Engineering and Information Technology

University of Technology Sydney

This thesis is submitted for the degree of

Doctor of Philosophy

July 2019

I would like to dedicate this thesis to my loving parents

for letting me pursue my dream

for so long

so far away from home

Certificate of Original Authorship

I, Yali Du declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney. This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. This document has not been submitted for qualifications at any other academic institution. This research is supported by the Australian Government Research Training Program.

Acknowledgements

First and foremost, I am greatly thankful for my adviser, Dacheng Tao, for his consistent support and guidance during my PhD, and for providing me invaluable advice on many topics. His high standard in research helps me to inspect and improve my work all the time. This thesis would not have been completed without his support. I am also thankful for my undergraduate advisers Shenggui Zhang and Yufeng Nie for encouraging me to pursue a career in research.

I would like to extend my gratitude to Dr. Yong Luo for introducing me to the research career in artificial intelligence, Dr. Jinfeng Yi, for introducing me to adversarial machine learning research. For the work in this thesis, I enjoyed working with Chang Xu, Meng Fang, Jinfeng Yi, Jun Cheng, Rao Kotagiri. I appreciate their brilliant work and timely support. Especially, I would like to thank Dr. Chang Xu for his consistent efforts and fruitful discussions for improving this thesis.

I am grateful to have the opportunity of working in a group with many intelligent minds. I benefit a lot from discussing and working with them. They are Maoying Qiao, Liu Liu, Xiyu Yu, Huan Fu, Chaoyue Wang, Zhongwen Xu, Tongliang Liu, Mingming Gong, Baosheng Yu, Shaoli Huang, Zhe Chen, Zhibin Hong, Jiang Bian, Jiankang Deng, Jiayan Qiu, Jue Wang, Changxing Ding, Ruxin Wang; and faculty members, Professor Ivor Tsang, Professor Yi Yang, Dr. Liang Zheng, Dr. Lin Chen.

I am also grateful for all the other friends who made this journey unforgettable: Xun Yang, Chen Gong, Hongshu Chen, Hua Zuo, Yuangang Pan, Jianfeng Dong, Erkun Yang, Shan You, Yuxuan Du, Sujuan Hou, Yiliao Song, Xiaoqing Yin, Liping Xie, Long Lan, Xianye Ben, Wankou Yang, Shigang Liu, Qiang Li, Tao Lei. I am also thankful for my

housing roommates Tim Harper, Faris, Firas, Yuta, Ann, Vishal, Annia, Eric Wang, Cassie Maye, Pingo, David, Sahil who distracted me from my work and made my housing life happier than I can imagine. I would like to especially thank Donna Xu, Qian Zhang, Xinyuan Chen, and Yan Yan, who have given me support during both cheery and stressful times.

Finally, this thesis is dedicated to my family, for everything else, and Moe, for always being there.

Abstract

Recommendation systems (RecSys) are valuable for both industry and customers in many fields, including e-commerce and social media. Despite the great demand for such effective systems, many challenges still exist. A major obstruction is the sparsity and poor quality of data that hinder the learning of a satisfactory RecSys. Another obstacle lies in the open nature of RecSys: this poses a threat to their safety in applications.

In this thesis, we work towards meeting these two challenges. To improve RecSys performance, we study how to exploit information from user reviews, constraints on user behaviors and user/items demographic features. Three approaches are proposed: 1) we develop a privileged matrix factorization model that exploits reviews for the learning of both user/item factors; 2) we build a collaborative allocation model that investigates the geometric constraint on the user-preference matrix; 3) given that the features might be noisy in reality, we propose an approach to identifying noisy information and selecting useful side features.

Driven by concern for the security of RecSys, our first consideration is to develop an evaluation method for testing the robustness of target models before proposing an approach to improve their resistance to malicious attacks. The target model is evaluated by measuring the minimal number of features required to mis-predict a user's preference. To enhance the robustness of target models, we inject noise in the training phase to enforce resistance to perturbations. Target models are further guided by standard networks through the distillation of generalized knowledge to avoid performance degeneration. This way, the target model becomes more resistant to adversarial perturbations while still achieving similar performances to standard models.

We conclude the thesis by outlining main contributions and indicating primary results.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

With the rapid development in e-commerce, consumer choices have drastically grown. Recommendation systems (RecSys) have become indispensable to both industry and customers. For customers, an effective recommendation system helps to narrow down a set of choices, discover new things and explore various options. For industry, an accurate recommendation to customers can increase sales and click-through rates, offer personalized service and create opportunities for promotion.

In spite of the great importance of RecSys, many challenges for building a satisfactory recommendation model remain unaddressed. On one hand, the continuing rapid growth of online service, combined with the sparsity and inferior quality of data, make it difficult to determine functional patterns from user behaviors. On the other hand, due to the open nature of RecSys, adversaries permeate the Internet widely, allowing fraudsters to easily inject noisy features into recommendation systems. The desire and need to build trustworthy recommendation models have combined to generate considerable industry interest.

This thesis is directed towards meeting these two challenges. We begin by constructing effective RecSys by exploiting review comments, user behaviors and noisy side information. From there we propose our methods for evaluating the resilience of RecSys to adversarial noise before presenting an efficient way to improve this robustness in deep learning-based recommendation systems.

The following (Sections 1.1 and 1.2) presents the existing background for these two challenges and our contributions towards addressing them. A broad summary is given in Section 1.3.

1.1 Designing RecSys with Features

One natural way of improving RecSys performance is to incorporate explicit features such as review texts, side information, etc. How to use these features effectively remains a great challenge, in both exploiting useful information and identifying noisy parts. This introduction will discuss drawbacks of existing methods and present our contributions for alternatives.

1.1.1 Learning representations from reviews

Nowadays, many online review systems contain users' reviews of an item along with its ratings. For example, in online shopping website Amazon [1], users are encouraged to write down their comments about the product they have purchased via the website and simultaneously provide an overall rating. These comments reflect why a user rates an item. The power of this review information has been exploited already by existing works [1–3]. Mcauley [1] used a topic distribution to describe all reviews from an item or user perspective. Almahairi et al. [2] explored representative word distributions for reviews from different items. Both contributions use topic or word distributions of reviews to represent item vectors. An obvious deficiency is that they ignore the salient fact that review comments are simultaneously associated with both users and items. In practice, different users may have distinct comments on the same items. Meanwhile, different items can receive disparate comments from the same user. It is necessary, therefore, to exploit fully the value of review comments in collaborative filtering from both user and item points of view.

In Chapter 2 we treat review comments as privileged information for each rating. We represent each review as a feature vector in privileged feature space, determining a privileged function on these feature vectors that describes the discrepancies between predictions and ground truth ratings. This work appeared previously as [4].

1.1.2 Exploiting rating allocation for users

Despite the success of collaborative filtering in diverse applications, their effectiveness is mainly seen in situations where items are rated independently by a user without consideration of their relationships. In practice, a user may be required to allocate limited credits to a set of items. In other words, the limited credits owned by the user are spread across all items. In cumulative voting, for example, a voter is given an exact number of points (or votes) to distribute among candidates. In family financing, however, a user distributes limited money among different financial products based on preference or confidence. We refer to this type of user as "energy-limited". However, existing algorithms cannot be applied to collaborative rating allocation problems, since they do not consider the relationship among ratings. Observations of the limited budget of a user's ratings inform and motivate us to consider users' rating vectors lying on the simplex, i.e., the multinomial manifold [5].

In Chapter 3, to preserve the intrinsic constraints of the preference vector, which indicates distributions of a user's limited credits, we deploy our method from two perspectives. First, we deploy a non-linear pull-back metric from a sphere to investigate the geometric properties of the data on the simplex. Second, we embed histogram data into Euclidean space and then measure Euclidean distance between predicted ratings and ground truth ratings. The resulting objective function can be efficiently optimized using a Riemannian conjugate gradient method. Experimental results on real-world datasets demonstrate the effectiveness of the proposed method of performing rating allocation for users. This work was previously published as [6, 7].

1.1.3 Learning from noisy side information

With the availability of side information, i.e., descriptors for users and items, there is considerable interest in the advantages of its use in improving the effectiveness of RecSys [8–11]. In practice, side information is not always perfect due to noise or outliers. There are several trials [12, 13] which eliminate the adverse effect of noise in an inductive matrix completion framework through non-linear mapping of side information or matrix decomposition. Re-

gardless of the certainty that selecting relevant side features for rating prediction is essential, it remains a point to be investigated.

In Chapter 4, we revisit the common low-rank principle in inductive matrix completion, where the low-rank matrix aligns users and items' side features with observed rating data. We then propose our method which we title, "Robust Inductive Matrix Completion", (RIMC) to identify noisy feature dimensions in user/item side information and suppress their negative influence on rating prediction.

1.2 Evaluating and Improving the Robustness of RecSys

In the last five years, security concerns regarding machine learning models have been brought out by adversarial examples. Due to the open nature of RecSys, the adversaries exist extensively on the Internet where fraudsters can easily inject noisy features into RecSys. How to evaluate and improve the robustness recommendation models become vital steps. Here we discuss our contributions in resolving these two issues.

1.2.1 Evaluating the robustness of RecSys

Currently, deep learning-based RecSys have been the standard paradigm in the industry. Despite the great importance of building safe RecSys, people's understanding of their robustness is limited. One can evaluate the robustness of a neural network by constructing attacks that demonstrate an upper bound [14]. Previously, O'Mahony et al. [15], Mobasher et al. [16], Lam and Riedl [17] studied attack models aimed at promoting or demoting a specific set of items. Li et al. [18] aimed to decrease the general performance of collaborative filtering models based on full knowledge of the learning algorithm and training data. Adversarial attacks on deep learning-based RecSys, however, are yet to be studied.

In Chapter 5, we propose a new threat model to evaluate the robustness of deep learning-based RecSys with contextual features. The major challenge exists in dealing with the natural discrete features in RecSys. Since continuous perturbations are not available, we study the minimal number of features required to mis-predict a user's preference. We introduce a

two-stage approach: we first optimize over the embedding vectors of a set of input features using $l_{2,1}$ norm until it succeeds; then we map the perturbed embedding vectors back to the embedding table and shrink the attacked feature set by ignoring the embedding vector that is minimally changed. Then we demonstrate the state-of-the-art recommendation models' robustness boundary. For example, for Wide&Deep on MovieLens dataset, perturbing one feature in the input can decrease the model's accuracy almost to zero. Finally, our empirical results suggest that a larger feature space leads to more vulnerability to adversarial attacks. It is also suggested that cost is important in customers' decisions.

1.2.2 Improving the robustness of RecSys

Aware of their vulnerability, how to enhance the robustness of RecSys takes on great importance. A close established line of research studies robust collaborative filtering based on matrix completion [19–21] in which a few entries or columns are randomly perturbed. As modern RecSys are increasingly sophisticated, adversaries' activities become more diversified and require an equally diverse response. How to improve the robustness of modern deep learning based RecSys remains an open question.

In Chapter 6, we address this question by developing a training paradigm for improving the robustness of target models. Specifically, we inject noise in the training phase to enforce the target models' resistance to perturbations. Target models are further guided by standard well-performed models through generalized knowledge distillation to avoid performance degeneration. This way, the target model will be more resistant to adversarial perturbations while achieving a performance similar to standard models. This work appeared previously as [22].

1.3 Summary

In the rest of the thesis,Chapters 2,3,4 contribute to developing useful RecSys. Chapters 5, 6 are devoted to understanding the robustness of RecSys and potential strategies for resisting adversarial attacks. Each chapter covers one of the five topics discussed above. In Chapter 7,

we reiterate key findings and conclude the thesis. Note that parts of the work reported in this thesis have previously been presented in earlier papers: [4, 6, 7, 22].

1.4 Publications

My publications include recommendation systems [4, 6, 7], adversarial machine learning [22, 23], and reinforcement learning [24]. A list of my publications is as follows:

1. **Yali Du**, Chang Xu, Dacheng Tao. Privileged Matrix Factorization for Collaborative Filtering. In International Joint Conference on Artificial Intelligence (IJCAI), 2017

2. **Yali Du**, Chang Xu, Dacheng Tao. Collaborative Rating Allocation. In International Joint Conference on Artificial Intelligence (IJCAI), 2017.

3. **Yali Du**, Chang Xu, Dacheng Tao. Matrix Factorization for Collaborative Budget Allocation. IEEE Transactions on Automation Science and Engineering (TASE), 2018.

4. **Yali Du***, Meng Fang*, Jinfeng Yi, Jun Cheng, Dacheng Tao. Towards Query Efficient Black-box Attacks: An Input-free Perspective. ACM Conference on Computer and Communications Security Workshop on Artificial Intelligence and Security (AISec@CCS), 2018. (*equal contribution)

5. **Yali Du**, Meng Fang, Jinfeng Yi, Jun Cheng, Dacheng Tao. Enhancing the Robustness of Neural Collaborative Filtering Systems Under Malicious Attacks. IEEE Transactions on Multimedia (TMM), 2019.

6. Lei Han*, Peng Sun*, **Yali Du***, Jiechao Xiong, Qing Wang, Xinghai Sun, Han Liu, Tong Zhang. Grid-Wise Control for Multi-Agent Reinforcement Learning in Video Game AI. International Conference On Machine Learning (ICML), 2019. (*equal contribution)

Chapter 2

Learning Representations from Reviews

Collaborative filtering plays a crucial role in reducing excessive information in online consuming by suggesting products to customers that fulfill their potential interests. Observing that users' reviews on their purchases are often in companion with ratings, recent works exploit the review texts in modeling user or item factors and have achieved prominent performance. Although effectiveness of reviews has been verified, one major defect of existing works is that reviews are used in justifying the learning of either user or item factors without noticing that each review associates a pair of user and item concurrently. To better explore the value of review comments, this chapter presents the privileged matrix factorization method that utilizes reviews in the learning of both user and item factors. By mapping review texts into the privileged feature space, a learned privileged function compensates the discrepancies between predicted ratings and ground truth values rating-wisely. Thus by minimizing discrepancies and prediction errors, our method harnesses the information present in the review comments for the learning of both user and item factors. Experiments on five real datasets testify the effectiveness of the proposed method.

2.1 Introduction

Recommendation systems are valuable for both providers and customers in many fields such as e-commerce, social media, and entertainment. For customers, an efficient recommendation

system helps to narrow down the set of choices, discover new things and explore various options. For providers, a trustworthy recommendation to customers can increase sales or click-through rates, offer personalized service and create opportunities for promotion.

The huge demand for online shopping has encouraged plentiful works to exploit users' ratings on products and hence make predictions of their future intent. Given a rating matrix composed of users' preferences on various items, recommendation systems aim to predict a user's preferences on the items that are not consumed. Content-based filtering and collaborative filtering methods are two popular methods for recommendation systems and have achieved prominent performance in the past. Content-based methods [25] make a recommendation according to a user's past interests by matching up this user's preferences with the item features. On the other hand, the rooted observation for collaborative filtering is that similar users have similar interests on the same item, and collaborative filtering methods [26] recommend items to users that are liked by their "similar" users based on the preference connections, which can be discovered by the collaborative filtering model. Low-rank matrix factorization is one of the most popular collaborative filtering algorithms. The sparse rating matrix is factorized into the product of two low-rank matrices, user matrix, and item matrix respectively, and the missing ratings are estimated by a dot product of a user vector and an item vector.

One of the main challenges in current recommendation tasks is the data sparsity, for instance, in the Amazon review dataset [1], up to 99.9% of ratings are missing. In order to enhance performance, many recommendation systems start to pay attention to auxiliary information accompanying the ratings. These days, a lot of online review systems contain users' reviews of an item along with its ratings, for example in Amazon review system, users are encouraged to write down their comments about the product they have purchased from this website and give an overall rating at the same time. A user might give a product lower rating because of its high price, and explains why his overall ratings are different from a spendthrift user who focuses very little on the price of products. The users' opinion might be expressed in their reviews such as "so high price!".

The powerfulness of text-based side information has been exploited by several state-of-the-art methods. Hidden factor as topics (HFT) [1] modeled multinomial topic distributions of review comments by Latent Dirichlet Allocation and casts rating prediction as a matrix factorization problem while topic distribution variables are linked to user vector or item vector. BoWLF [2] used a bag of words to represent review comments and maximized joint probabilities of these words along with minimizing prediction error of matrix factorization model for collaborative prediction. While HFT used a topic distribution to describe all reviews from an item or user, BoWLF explores representative word distributions for reviews from different items. Overall, both of them use item vectors to represent topic distributions or word distributions of reviews from an item.

Existing matrix factorization based methods commendably utilize the review information for collaborative filtering. However, an obvious deficiency is that they explore review comments to describe either users or items, but ignore the phenomenon that review comments are simultaneously associated with both users and items. In practice, different users may have distinct comments on the same items. Meanwhile, different items can receive disparate comments from the same user. Thus it is necessary to maximize the values of review comments in collaborative filtering from both user and item aspects.

In this chapter, we propose privileged matrix factorization method (PriMF for short) that utilizes extra information in the learning of both user and item vectors for collaborative filtering. We take the review comments as privileged information which are corresponding to rating values. After representing each review as a feature vector in privileged feature space, we learn a privileged function on these feature vectors that describe the discrepancies between predictions and ground truth ratings. By minimizing the prediction error and the discrepancies depicted by the privileged function, the proposed PriMF benefits the learning of latent factors for both users and items. Experiments on five real datasets show the effectiveness of proposed privileged matrix factorization method.

In Section 2.2, we discuss related works. We introduce the formalization of our method in Section 2.3, and give the optimization procedure in Section 2.4. Empirical verifications are presented in Section 2.5, followed by conclusions in Section 2.6.

2.2 Related Work

Recommendation systems can be generally classified into two types: content-based filtering [25] and collaborative filtering (CF). Content-based filtering algorithms recommend items to users according to their previous interests. Specifically, a rating is estimated by matching up item features according to users' preferences. Collaborative filtering aligns with the assumption that users who share similar interests on an item in the past are more likely to hold similar opinions on other items compared with a randomly chosen user.

While collaborative filtering has become one of the most used recommendation approaches, it has two important subclasses: memory-based and model-based approaches [26]. Memory-based methods [27] make use of the rating matrix and make recommendations by the relationship between the queried user and item and the known ratings. Model-based filtering algorithms fit a parametric model to the sparse rating matrix and then issue recommendations using the fitted model. For example, Bayesian networks [28, 29] train a Bayesian classifier based on the given rating matrix, and clustering-based methods [30] split the set of users or items on the given rating matrix thus taking them as the basis for future predictions. Recently, low-rank matrix factorization methods are successfully applied into recommendation systems [31, 32]. Under the low-rank assumption, a rating matrix can be expressed as a product of two low-rank matrices [33]. By fitting the two low-rank matrices to the given ratings, prediction of a user's interests on a new item can be made by multiplying corresponding user vector and item vector. Other recommendation approaches leverage rating data and side information to enhance performance [34].

To improve the performance for collaborative filtering, some recent works pay attention to the review text; the review text helps describe the user/item factors. Mcauley [1] modeled the rating prediction problem as matrix factorization. When the latent dimension of user vectors (or item vectors) and the number of hidden topics are linked, the topic distribution variable one user's reviews are represented by the user vector through softmax normalization. Ling et al. [3] proposes a combined approach of content-based and collaborative filtering which harness information from ratings and reviews together. Almahairi et al. [2] presents a

distributed Bag-of-Words method which infers the probability distribution of BoW by affine transformation on item representation with softmax normalization.

2.3 Privileged Matrix Completion

In the following, we give the formulation of our method and discuss its advantages in detail.

2.3.1 Problem statement

Considering a database composed of (user, item, rating) tuples from n users' preferences on m items, this database can be compactly denoted as a preference matrix R, which is usually sparse. The elements in the matrix R represent corresponding user's liking of the item. In real applications, the values are usually integer ratings settled in $[1, L]$. A recommendation system manages to predict a user's preference of an unconsumed item which corresponds an unseen entry in the rating matrix R.

2.3.2 Privileged matrix factorization

Given the rating matrix R, the common assumption is that R is low-rank and a matrix factorization method fits R with a low-rank matrix $X = UV^\top$. To predict ordinal values of R with real-valued X, a set of thresholds $\{\theta_1, ..., \theta_{L-1}\}$ are needed. In the hard-margin settings, X would be required as:

$$\theta_{R_{ij-1}} + 1 \leq X_{ij} \leq \theta_{R_{ij}} - 1, \tag{2.1}$$

where θ_0 and θ_L are $-\infty$ and $+\infty$ respectively. The hard-margin setting is often known to be powerless in non-separable case. Therefore to tame this problem, slack variables ξ_{ij} are introduced, and Eq. (2.1) is relaxed to

$$\theta_{R_{ij-1}} + 1 - \xi_{ij} \leq X_{ij} \leq \theta_{R_{ij}} - 1 + \xi_{ij},\ \xi_{ij} > 0. \tag{2.2}$$

To avoid the predictions that cross rating-boundaries, not only two immediate constraints $X_{ij} \leq \theta_{R_{ij}} - 1$ and $X_{ij} \geq \theta_{R_{ij-1}} + 1$ are penalized, but also $X_{ij} \leq \theta_l - 1, \forall l > R_{ij}$ and $X_{ij} \geq \theta_l + 1, \forall l < R_{ij}$. By introducing indicating variable T:

$$T_{ij}^l = \begin{cases} +1 & \text{for } l \geq R_{ij}, \\ -1 & \text{for } l < R_{ij}, \end{cases}$$

Eq. (2.2) is reformulated to the following compact form:

$$T_{ij}^l \cdot (\theta_l - X_{ij}) \geq 1 - \xi_{ij};\ \forall (i,j) \in \Omega, 0 \leq l \leq L. \tag{2.3}$$

We denote $\Phi = \{U, V\}$ as the variable set and $\mathscr{R}(\Phi)$ as the regularization term on variables in Φ to control the model complexity. Under the above constraints in Eq. (2.3), the optimization problem is formulated as:

$$\begin{aligned} \min_{\Phi,\theta} \quad & \mathscr{R}(\Phi) + C \sum_{(i,j)\in\Omega} \xi_{ij} \\ \text{s.t.} \quad & T_{ij}^l \cdot (\theta_l - X_{ij}) \geq 1 - \xi_{ij}. \end{aligned} \tag{2.4}$$

Observing that a rating indicating the user's preference is often accompanied with a review text, it is believed that a piece of comment text contains rich information about the user's opinions on an item. In real world, it is obvious that people need less training data when they learn a new task than that needed by training a machine on the same task. The rooted reason behind this phenomenon is that we are taught by teachers usually. A teacher has the experience on this task and has her own trained system in her mind. Although she cannot transfer her decision mechanism directly into your brain, she can teach you by generating privileged information that reflects her belief in the development of this mechanism. For example, when it comes to predicting a user's preference over two movie genres like "comedy" and "thriller", the training data are two movies with genre "comedy|drama" and "drama|thriller" and both movies are rated as 5. In this case, matrix factorization model might be confused at the ratings. But from the comments that "A great

comedy for family time" and "I don't like the kind of plots, but Thomas's acting is exciting", it is more confident to conclude that this consumer prefers "comedy" more than "thriller". This is exactly the role that we think the review comments can play in the training process. It reflects a user's thoughts in the decision procedure, and thus explains discrepancies between predictions and ground truth ratings.

Review information is from a different resource compared to rating data and lies in another feature space. The feature vector which is denoted as Z_{ij} corresponds to rating R_{ij}. Now we consider a rating prediction problem based on the quaternions (user, item, rating, review). Considering review texts as privileged information of corresponding ratings, the privileged function is defined as $\phi(\boldsymbol{Z}_{ij}, w, b) = w\boldsymbol{Z}_{ij} + b, w \in \mathbb{R}^d$. To ease the presentation, we append the bias term b to the end of the weight vector w and still denote it as w, $w \leftarrow [w, b]$. Similarly, we append 1 to the end of every feature vector as $Z_{ij} \leftarrow [Z_{ij}, 1]$. The slack variables in Problem (2.4) can be reformulated as $\xi_{ij} = \phi(\boldsymbol{Z}_{ij}, w), \forall (X_{ij}, \boldsymbol{Z}_{ij})$. Since we can learn different privileged functions for different items, we use W to denote coefficient variables in ϕ.

Adding W to the variable set $\Phi = \{U, V, W\}$, $\mathscr{R}(\Phi)$ takes the form as:

$$\mathscr{R}(\Phi) = \frac{1}{2}(\|U\|_F^2 + \|V\|_F^2 + \gamma\|W\|_F^2). \tag{2.5}$$

Our objective function thus can be written as:

$$\begin{aligned} \min_{\Phi,\theta} \quad & \mathscr{R}(\Phi) + \alpha \sum_{(i,j)\in\Omega} [\phi(\boldsymbol{Z}_{ij}, W)]_+ \\ \text{s.t.} \quad & T_{ij}^l \cdot (\theta_{jl} - X_{ij}) \geq 1 - [\phi(\boldsymbol{Z}_{ij}, W)]_+, \end{aligned} \tag{2.6}$$

where $[y]_+ = \max\{0, y\}$.

The variable ξ_{ij} in Problem (2.4) is introduced to represent slacks triggered by rating data that are not linearly separable. With privileged information in hand, the slacks in non-separable situation can be learned by a correcting function that is dependent on the additional information and has low Vapnik–Chervonenkis (VC) dimension. This learning paradigm is initially applied to support vector machine in classification problem [35, 36]. In

collaborative filtering, this slack margin can be triggered by various possibilities, such as users' special appetite on an item that is not consistent with item features, or users' sudden change in one criterion for rating. Moreover, it is even possible that rating data is corrupted when it is collected. Therefore, one linear correcting function $\phi(Z_{ij}, W)$ is not representative enough for correcting the non-linearly separable ratings.

To reinforce the model's tolerance to outliers or noises, we represent ξ_{ij} in Problem (2.4) by a linear privileged function and a tolerance term ζ_{ij}. The final model takes the form:

$$\begin{aligned} \min_{\Phi,\theta} \quad & \mathscr{R}(\Phi) + \alpha \sum_{(i,j)\in\Omega} [\phi(\boldsymbol{Z}_{ij}, W)]_+ + \beta \sum_{(i,j)\in\Omega} \zeta_{ij} \\ \text{s.t.} \quad & T_{ij}^l \cdot (\theta_{jl} - X_{ij}) + [\phi(\boldsymbol{Z}_{ij}, W)]_+ \geq 1 - \zeta_{ij}, \\ & \zeta_{ij} \geq 0. \end{aligned} \tag{2.7}$$

Thus the offsets for hard-margins in Problem (2.4) are split into a discrepancy term represented by privileged function and a tolerance term that allow the existence of outliers. The above problem is equivalent to the following problem:

$$\begin{aligned} \min_{\Phi,\theta} f(\Phi,\theta) = \quad & \mathscr{R}(\Phi) + \alpha \sum_{(i,j)\in\Omega} [\phi(\boldsymbol{Z}_{ij}, W)]_+ \\ & + \beta \sum_{(i,j)\in\Omega} h(T_{ij}^l \cdot (\theta_{jl} - X_{ij}) + [\phi(\boldsymbol{Z}_{ij}, W)]_+). \end{aligned} \tag{2.8}$$

where $h(u)$ is the hinge loss and it can be, for example, $h(u) = \max\{0, 1-u\}$.

After Problem (2.8) is addressed, we get the optimal solution of Problem (2.8) that $\Phi^* = \{U^*, V^*, W^*, \theta^*\}$. Denote $X^* = U^* V^{*T}$, we infer the ratings of each item as:

$$R_{*j}^* = 1 + \max\{l | X_{*j}^* \geq \theta_{jl}^*, l = 0, 1, ..., L-1\}, \tag{2.9}$$

where R^* is the optimal output of our algorithm.

2.4 Optimization

In the following, we present our optimization method. Let U_{i*} and V_{j*} denote the i-th and j-th row of U and V respectively. For simplicity, the independent variable of function $h(u)$ is defined as $u_{ij}^l = T_{ij}^l(\theta_{jl} - X_{ij}) + [\phi(\boldsymbol{Z}_{ij}, W)]_+$. Taking derivatives w.r.t. U in Problem (2.8), we obtain:

$$\frac{\partial f}{\partial U_{i*}} = U_{i*} - \beta \sum_{jl} T_{ij}^l V_{j*} \cdot \frac{\partial h}{u}|_{u=u_{ij}^l}, \tag{2.10}$$

and the partial derivative w.r.t. V is analogous. Let $\mathbf{1}(x > 0)$ denote the indicator function. The partial derivative about w_j is:

$$\frac{\partial f}{\partial w_j} = w_j + \alpha \sum_i Z_{ij} \cdot \mathbf{1}(w_j^T Z_{ij}) + \beta \sum_{il} Z_{ij} \cdot \mathbf{1}(w_j Z_{ij}) \frac{\partial h}{u}|_{u=u_{ij}^l}. \tag{2.11}$$

And the partial derivative w.r.t. θ_{jl} element-wisely is:

$$\frac{\partial f}{\partial \theta_{jl}} = \beta \sum_i T_{ij}^l \cdot \frac{\partial h}{u}|_{u=u_{ij}^l}. \tag{2.12}$$

With these gradients in hand, we can turn to gradient descent methods for solving Problem (2.8).

For optimization of U, V, W, and θ we choose conjugate gradient descent method [37]. For simplicity of presentation, we denote y as concatenated vector of variables U, V, W and θ, and denote $g(y) = f(\Phi, \theta)$. We have the general form of conjugate gradient as

$$y_{t+1} = y_t + \eta_t \cdot p_t, \tag{2.13}$$

where η_t is a sequence of step sizes generated by a line search algorithm that can guarantee the decrease in objective function. p_t is the descent direction generated by the following rule:

$$p_t = -\nabla g_t + \tau_{t-1} p_{t-1}, \tag{2.14}$$

and ∇g_t is the abbreviation of $\nabla g(y_t)$. τ_t is the conjugate gradient parameter and it is chosen to ensure the conjugacy between conjugate gradient directions.

Several options of τ_k can be used in conjugate gradient algorithm. Here we choose the Polak-Ribiere (PR+) [37] method for calculating the conjugate gradient update parameter:

$$\tau_t = \max\{\frac{\langle \nabla g_t, \nabla g_t - \nabla g_{t-1} \rangle}{\langle \nabla g_{t-1}, \nabla g_{t-1} \rangle}, 0\}. \tag{2.15}$$

Note that τ_t is chosen to ensure that p_t is orthogonal to all previous search directions by a symmetric positive definite matrix. When two consecutive gradients are far away from orthogonal or conjugate directions are almost orthogonal to gradients, we restart the searching process by reset the exploration direction to the steepest descent direction. The restart condition takes the form: $\langle \nabla g_t, \nabla g_{t-1} \rangle / \langle \nabla g_t, \nabla g_t \rangle \geq \nu$ or $\langle p_{t+1}, \nabla g_{t+1} \rangle \geq 0$. Here ν is set to 0.1 as recommended by [38].

After selecting the search direction, strong Wolfe conditions [38] are applied to choose step size η_k that can guarantee sufficient decrease on objective function. The conditions take the form as:

$$g(y_t + \eta_t p_t) \leq g(y_t) + c_1 \eta_t \nabla g_t^T p_t, \tag{2.16a}$$

$$|\nabla g(y_t + \eta_t p_t)| \leq c_2 |\nabla g_t^T p_t|. \tag{2.16b}$$

A secant method is used to find the root of directional derivatives. The overall algorithm is summarized in Algorithm 1.

Due to the undifferentiability of standard hinge loss at zero point, the smooth hinge loss is adopted in [31]:

$$h_u = \begin{cases} 0 & \text{for } u \geq 1, \\ \frac{(1-u)^2}{2} & \text{for } 0 \leq u < 1, \\ \frac{1}{2} - u & u < 0. \end{cases}$$

The smooth hinge loss does not continuously reward the correct predictions, meanwhile, it is differentiable at zero point.

Strong Wolfe conditions guarantee the proposed optimization to converge. However, due to the non-convexity of the objective function, the solution may fall into the local minimum.

Algorithm 1 Conjugate Gradient Method for Problem (2.8)

Input: R: Incomplete matrix, ε: stopping criteria, k: latent dimension, T: max iteration
Output: R^*: the optimal approximation of R

Init: $y_0 = [U_0(:); V_0(:); \theta_0(:); W_0]$;
eval: $g_0 = g(y_0),\ \nabla g_0 = \nabla g(y_0)$;
set: $p_0 = -\nabla g_0, t \leftarrow 0$;
while $t < T$ **and** $\|\nabla g_t\| > \varepsilon$ **do:**
choose step size: η_t satisfies Cond. (2.16);
update $y_{t+1} = y_t + \eta_t p_t$;
eval: ∇g_{t+1};
get CG direction:
$\tau_{t+1}^{PR} \leftarrow \frac{\langle \nabla g_{t+1}, \nabla g_{t+1} - \nabla g_t \rangle}{\langle \nabla g_{t+1}, \nabla g_{t+1} \rangle}$;
$\tau_{t+1}^{PR+} \leftarrow \max\{\tau_{t+1}^{PR}, 0\}$;
$p_{t+1} \leftarrow -\nabla g_{t+1} + \tau_{t+1}^{PR+} p_t$;
$t \leftarrow t + 1$;
check restart condition:
if $\frac{\langle \nabla g_k, \nabla g_{k-1} \rangle}{\langle \nabla g_k, \nabla g_k \rangle} \geq \nu$ **or** $\langle p_{t+1}, \nabla g_{t+1} \rangle \geq 0$:
reset: $p_{t+1} = \nabla g_{t+1}$;
end if
end while
Return: R^* predicted by Eq. 2.9.

In experiments we will repeat the training process several times to avoid spurious local minimum.

2.5 Experiments

We verify the proposed method on five datasets from Amazon reviews [1]. These data were collected from the Amazon website with the years spanning from 1998 to 2013. The five datasets refer to five categories of products, watches, musical instruments, industrial and scientific, gourmet foods and books. Every rating in the datasets is associated with a piece of review comment. The detailed information for these datasets are in Table 2.1.

Table 2.1 Dataset Information

Dataset	users	items	review	words	words/review	reviews/item
Watches	62,041	10,318	68,356	5,436,671	79.53	6.62
Musical Instruments	67,007	14,182	85,405	7,442,294	87.14	6.02
Industrial Scientific	29,590	22,622	137,042	6,920,151	50.50	6.06
Gourmet Foods	112,544	23,476	154,635	10,542,984	68.18	6.59
Books	2,588,991	929,264	12,886,488	1,613,603,531	125.22	13.87

2.5.1 Experimental settings

We follow the settings used in [1, 2]. For each dataset, 80% ratings are randomly selected for training and the remaining 20% are evenly split into validation and test. The review information is only available in the training phase.

To extract feature vectors from reviews of different lengths, we adopt a neural network language model, Paragraph Vector [39] for help. As an extension of Word Vector [40], it can preserve semantics of words as well.

The size of the privileged feature vector is fixed to $k = 10$ which is consistent with the setting in HFT. The rank r of X is determined by cross validation. In the experiments, the optimum r on different datasets is around $r = 50$. Baseline methods have similar parameter settings. The measurement used is mean square error (MSE):

$$\text{MSE} = \frac{\|P_\Omega(R^* - R)\|_F^2}{|\Omega|}. \tag{2.17}$$

Where R^* is the optimal estimation of the ground truth rating matrix R and P_Ω is the sampling operator.

2.5.2 Baseline methods

We compare the proposed PriMF with four baseline methods.

- **FM3F:** Fast Maximum Margin Matrix Factorization [31] is a basic collaborative filtering model that minimizes the discrepancies between erroneous predictions and ground truth ratings.

- **HFT:** Hidden Factors as Topics [1] is a combined approach of matrix factorization and LDA. It models multinomial topic distributions by using either user factors or item factors which means that it takes all reviews by a particular item as a document or that by a particular user as a document and maximizes the joint probabilities of the text corpus.

- **BoWLF:** Bag-of-Words latent factor model [2] takes all reviews by an item as a document. For different document, the word distributions in the vocabulary are modeled by a weight matrix-vector product with the item correspondingly. By representing each review as a bag of words, BoWLF maximize the joint probabilities of document corpus by multiplying over all words in each document.

- **LMLF:** This is a recurrent neural network language model [2] that takes a sequence of words as input and preserve their order. It models different items to have their respective word distributions. The probability of current word conditioned on previous words is modeled by an affine transformation of a recurrent function and LSTM is adopted as its recurrent function module.

2.5.3 Evaluation results

For both the comparative methods and PriMF, we randomly initialize the training for five times and select parameters that have the lowest error on validation set. We select α, β and γ for PriMF from $\{10^{-5}, 10^{-4}, ..., 10^{4}\}$. For baseline methods, parameters are validated in the recommended range in original papers. The averaged MSE and standard deviation on test sets are reported in Table 2.2.

PriMF achieves a remarkable improvement over FM3F after introducing privileged information and gains consistent improvement on other three methods. In general PriMF and comparative methods can all be classified into matrix factorization category and all of them make predictions by multiplying user vector and item vector respectively. The difference exists in the way of using reviews in the training phase. While HFT, BoWLF and LMLF

Table 2.2 Mean square error comparisons with baseline methods on different datasets. Values in brackets indicate standard deviation error.

	(a) FM3F	(b) HFT	(c) BoWLF	(d) LMLF	(e) PriMF	PriMF improvement over (a)	(b)	(c)	(d)
Watches	1.571(0.02)	1.468 (0.03)	1.466 (0.03)	1.473 (0.03)	1.451(0.02)	7.62%	1.13%	1.00%	1.47%
Musical Instrument	1.448(0.03)	1.382 (0.02)	1.375 (0.02)	1.388 (0.02)	1.355(0.01)	6.42%	1.95%	1.45%	2.37%
Industrial	0.354(0.01)	0.354 (0.01)	0.352 (0.01)	0.356 (0.01)	0.324(0.01)	8.47%	8.47%	7.95%	8.99%
Gourmet Foods	1.732(0.01)	1.486 (0.02)	1.464 (0.02)	1.478 (0.02)	1.454(0.02)	16.07%	2.18%	0.71%	1.65%
Books	1.381(0.01)	1.141 (0.00)	1.10 (0.02)	1.110 (0.01)	1.09(0.01)	20.80%	4.15%	0.57%	1.47%
Overall	1.297	1.166	1.151	1.161	1.136	12.46%	2.63%	1.38%	2.19%

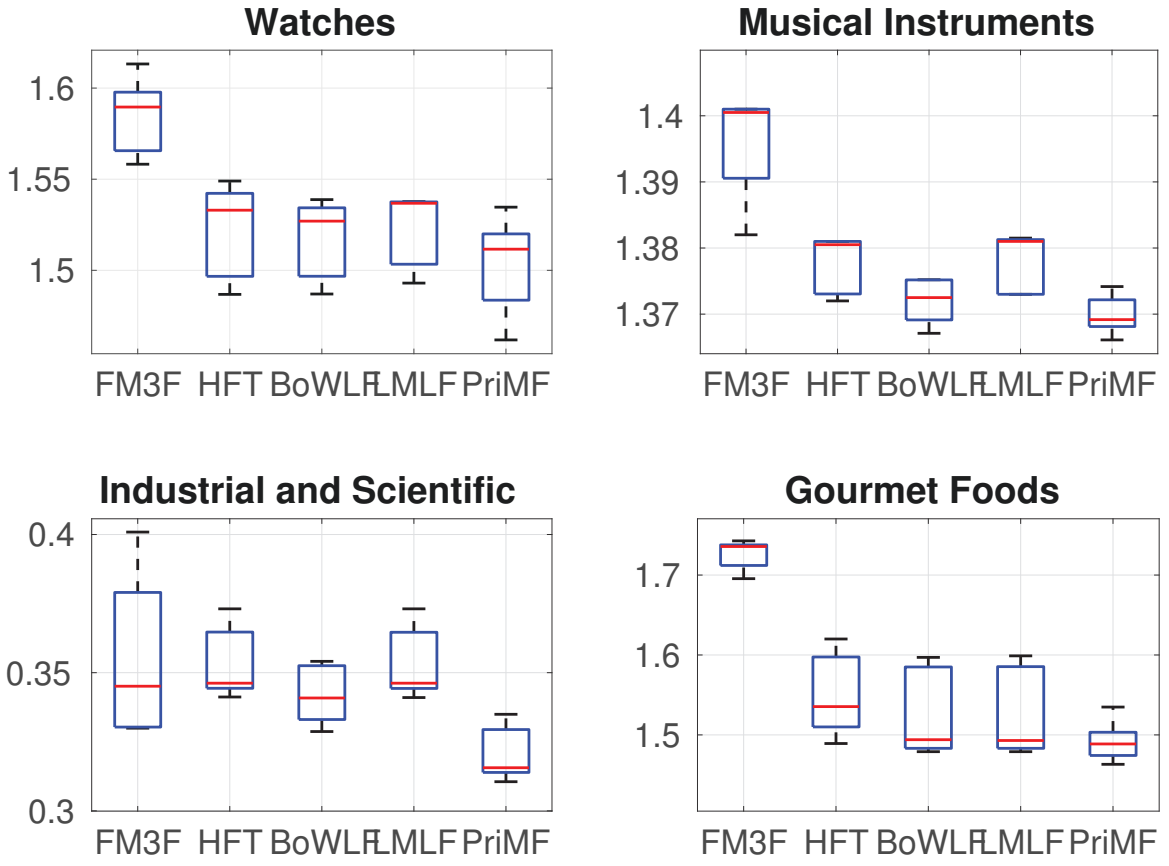


Fig. 2.1 Box and whisker plot of different random split of data. Center line represents median. Box extents show first quarter and third quarter. Whisker extents illustrate maximum and minimum values

model topic or word distributions of review contents by item vectors without considering user vectors, their performance sacrifices.

To test the performance of these approaches under different splits of data, we repeat random splits of data for five times and keep the percentage used in previous experiments. In every split, these methods are trained on a different set of ratings and review comments. The

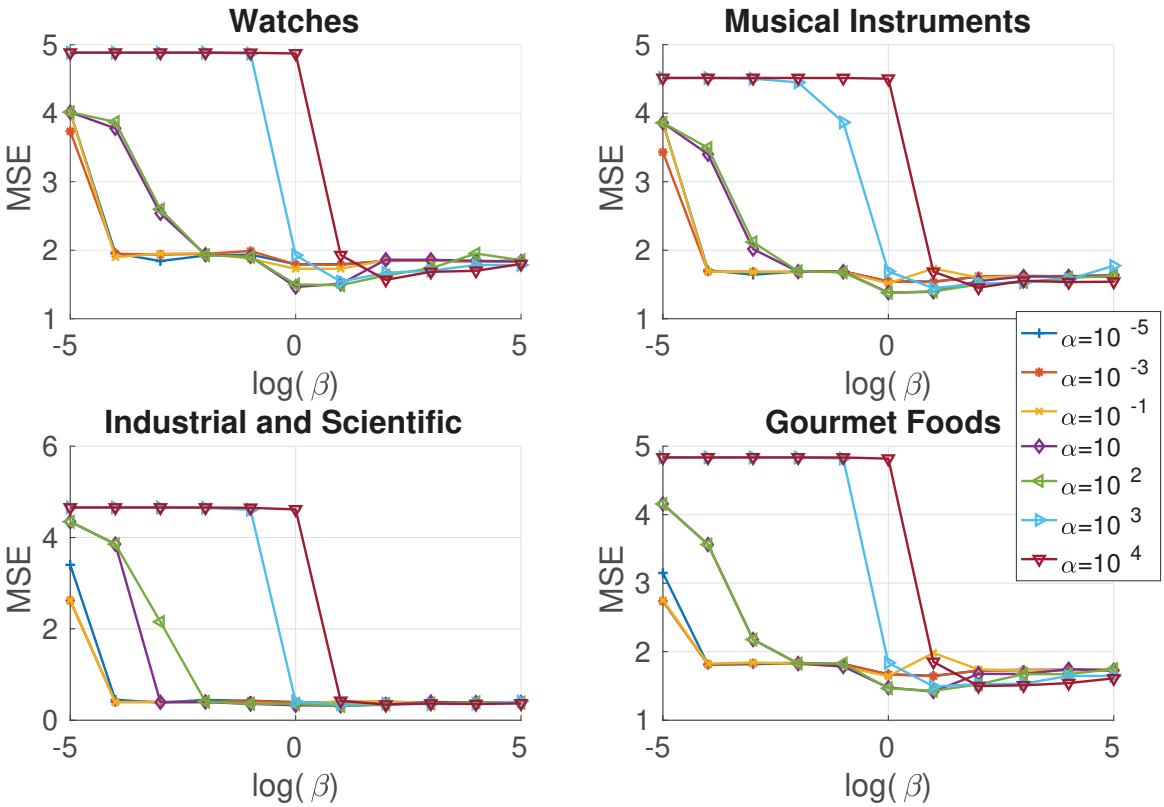


Fig. 2.2 Mean square error of PriMF changing with α and β on different datasets

parameters are chosen on the validation set and the performance of different approaches on test set are reported in Fig. 2.1. From Fig. 2.1, we find that the performance of each method under different splits varies dramatically, but the relative performance across five methods remains consistent. This observation indicates that the results tested under different splits of data are not straightforwardly comparable, but each single random split can be used to evaluate and select models as long as the split is kept the same for all evaluated methods.

2.5.4 Parameter analysis

In this section, we analyze the influence of two hyperparameters α and β in optimization problem (2.8). α controls the weight of privileged function that models discrepancies between predictions and groundtruth values, while β constrains prediction loss. We report the performance of PriMF under different choices of α and β in Fig. 2.2. When α is fixed, we can choose the value of β from a large range that leads to consistent good performance

of PriMF, and vice versa. We conclude that our method is stable with respect to different choices of parameters.

2.6 Conclusions

In this chapter, we study a privileged matrix factorization method for collaborative filtering. We utilize review texts that are in companion with rating values to assist the learning of user and item factors. In contrast to existing approaches that use review texts to describe either user or item factors, we notice that a review comment is related to a pair of user and item concurrently, and utilize it in the learning of both user and item factors. We take review texts as privileged information for matrix factorization and express them as feature vector through a Paragraph Vector transformation. The discrepancies between predictions and ground truth ratings are modeled by a privileged function dependent on review vector. We testify our model on several challenging Amazon review datasets and achieve better performance. While the remarkable strength of our method is discussed above, the weakness shared with the other collaborative filtering methods is the implicit assumption on the complete review comments. In practice, not every rating comes with a piece of review comment. In the future, we are considering the scenario in which some reviews are missing which is a common issue as well.

Chapter 3

Exploiting Rating Allocation for Users

This chapter studies the collaborative budget allocation problem in which users are not isolated in collaborative consumption of goods or services when available goods or services are limited. Different from existing methods which treat each user independently, we investigate the geometric properties of users consumption or preference on services and design a matrix completion framework on the simplex. In this framework, an item's allocation vector indicating how available services are allocated to users is estimated by the combination of user profiles as basis points on the simplex. Instead of using Euclidean distance directly, we specify Riemannian distance on the simplex or project histogram data on simplex to Euclidean space. To intensify our model's stability, we relax the exact recovery constraint to make robust collaborative prediction. The resulting objective function is then efficiently optimized by a Riemannian conjugate gradient method on the simplex. Experiments on real-world data sets demonstrate our model's competitiveness versus other collaborative budget prediction methods. Comparisons of different distance metrics for histogram data are shown and discussed.

3.1 Introduction

Collaborative prediction is a kind of peer-based activity which aims to predict users' preference based on their previous choices and connections with other users. In sharing economy

system, the preference can be interpreted as the potential amount of goods or service that a user is prone to consume. Collaborative filtering has wide applications in industry; for example, in an Amazon book recommendation system, if the system inputs are user ratings on previously read books, predictions of a user's preference of new and unread books are accomplished using patterns discovered from the partially observed rating matrix.

Existing collaborative prediction algorithms are effective and perform well in diverse applications including book, movie, and music recommendation systems. However, their success is mainly seen in isolated situations where a user's consumption on items is not constrained. In sharing economy system, users' activities are influenced by their peers' behavior since the generally available amount of service or goods in the environment is controlled to be limited [41–43]. In this circumstance, a user's excessive consumption on services or goods will probably reduce other people's access to these services or goods. Another example which shows the drawback of the current collaborative prediction algorithms is when users have limited credits to be distributed among items. Specifically, a user may be required to allocate limited credits to a set of items. In other words, the limited credits owned by the user are split among all items. For example, in cumulative voting, a voter is given an explicit number of points (or votes) to distribute among candidates, while in family financing, a user distributes his (or her) limited money among different financial products based on his (or her) preference or confidence [44]. In these scenarios, either a user has limited credits to be shared among items or an item has limited supplies that can be offered to users. Existing collaborative prediction algorithms do not apply to this budget allocation problem, which additionally involves constraints among preference values. To study the most beneficial allocation strategy for allocating limited services among peers while respect users preference is of great economical and social value.

We name the problem discussed above as a collaborative budget allocation problem. For the ease of presentation, in the following, we discuss the collaborative budget allocation under the constrained credits or ratings of users. For a budget-limited user, N different items will occupy parts of the budget and together they will use up all the user's budget such as money, votes. The deep-seated truth in this scenario is that the elements in the N-dimensional

allocation vector are not independent. With the observation of the limited budget on a user's credits or limited resources, we are motivated to consider the allocation vectors lying on the simplex, i.e., the multinomial manifold [5].

In this chapter, we propose to accomplish the collaborative budget allocation problem based on matrix factorization approach. To preserve the intrinsic constraints among the preference vector which indicates distributions of a user's limited credits, we deploy our method from two perspectives. First, we deploy a non-linear pull-back metric from sphere to investigate the geometric properties of the data on the simplex. Second, we embed histogram data into Euclidean space and then measure Euclidean distance between predicted ratings and ground truth ratings. To achieve stable matrix completion, we consider the possible noise on the rating observations and propose the robust collaborative rating allocation model which allows a certain amount of discrepancies between observations and predictions. The resulting objective function can be efficiently optimized using a Riemannian conjugate gradient method. Experimental results on real-world datasets demonstrate the effectiveness of the proposed method on excavating data's geometric properties.

The rest of the chapter is structured as follows. In Section 3.2, we review the line of related works on collaborative filtering. Section 3.3 presents the formulation for general collaborative budget allocation. We present the different distance metrics for histogram data on Section 3.4. The optimization method is proposed in Section 3.5, followed by the experimental results in Section 3.6. Finally, this chapter is concluded in Section 3.7.

3.2 Related Work

In the recent decades, factorization-based methods have become popular for collaborative filtering where the incomplete preference matrix is considered approximately low-rank [45, 46]. The rating matrix is factorized into a product of two low-rank matrices, thus filling the missing entries [31, 32, 47–49]. Despite the success of these methods, their effectiveness is limited to the isolated situation when the supply of items and demand of users are not restricted, which is not the best choice in the resource-constrained environment [42, 43], such

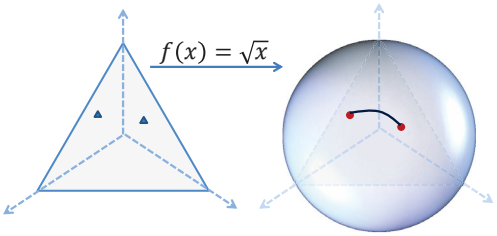


Fig. 3.1 An illustration of the distance on the simplex.

as budget allocation of histogram data. To capture important relationships among data, many researchers turn to study the effective metrics on histogram data. Kedem et al. [50] proposed to employ a linear mapping on the histogram data and learn the metric based on χ^2 distance. Cuturi and Avis [51] studied ground metric learning based on given labeled histogram data, i.e., discovering the optimal transport (meta-distances) between two examples. Le and Cuturi [52] adopted Aitchison transformations on simplex data and proposed to learn the parameters for Aitchison embedding by contrastive divergence approach. Apart from the metric learning approach on histogram data, Le and Cuturi [53] learn Euclidean mappings from simplex data to Euclidean space. Thus Euclidean distance measurement can be deployed.

In this chapter, we consider the geometric constraints on the rating vector, and propose to use pull-back metric from the sphere by Hellinger mappings to specify the distance measurement on the simplex, which is a special form of the metric discussed in [52]. Moreover, when observations are contaminated by noises [54], we extend our collaborative budget allocation to noise case where a small amount of noise can be allowed in observations.

Notations: Throughout this chapter, we consider histograms of dimension $n+1$. We use bold upper case letters $\mathbf{X}$ for matrices, and bold lower case letters $\mathbf{x}$ for vectors. Lower case letters stand for scalars in $\mathbb{R}$.

3.3 Problem Formulation

Take m users and their preference such as rating values on $n+1$ items as a matrix $\mathbf{Y} \in \mathbb{R}^{(n+1)\times m}$, where the j-th column represents user j's ratings on $n+1$ items and the i-th row corresponds to the ratings on item i from m users with $i \leq n+1$ and $j \leq m$. Indices of

observed entries constitute a set Ω. Valid ratings usually settle within a range $[0,k]$. In the rating allocation problem, the budget of a user's ratings on all items is fixed. Hence, the goal of collaborative rating allocation can be interpreted as filling an incomplete matrix $\mathbf{Y}$ with a matrix $\mathbf{X}$ whose columns $\mathbf{X}_{\cdot j}$ lie on the simplex. An $n-$simplex $\mathbb{P}_n$ is defined as

$$\mathbb{P}_n = \{\mathbf{x} \in \mathbb{R}^{n+1} | \ \forall i, \mathbf{x}_i \geq 0 \text{ and } \sum_{i=1}^{n+1} \mathbf{x}_i = 1\}.$$

The simplex $\mathbb{P}_n$ is described as a subset of $\mathbb{R}^{n+1}$, but it is actually an $n-$dimensional manifold. In fact, any point on the simplex is a parameter vector $\mathbf{x}$, which itself happens to be a probability vector.

The incomplete matrix $\mathbf{Y}_{(n+1)\times m}$ can be recovered through a matrix $\mathbf{X}_{(n+1)\times m}$ with a lower latent dimension

$$\begin{aligned} \min_{\mathbf{X}} \quad & \sum_{j=1}^{m} \mathbf{dist}(\mathbf{X}_{\cdot j}, \mathbf{Y}_{\cdot j}) \\ \text{s.t.} \quad & P_\Omega(\mathbf{X} - \mathbf{Y}) = 0, \mathbf{X}_{\cdot j} \in \mathbb{P}_n, \end{aligned} \tag{3.1}$$

where $\mathbf{dist}(\cdot,\cdot)$ is a distance measurement and P_Ω is the sampling operator defined as:

$$[P_\Omega(\mathbf{Y})] = \begin{cases} \mathbf{Y}_{ij} & \text{if } (i,j) \in \Omega, \\ 0 & \text{else.} \end{cases}$$

Considering the maximal latent dimension of $\mathbf{X} \in \mathbb{R}^{(n+1)\times m}$ as r, we can factorize $\mathbf{X}$ into two factors where $\mathbf{X} = \mathbf{UV}$, $\mathbf{U} \in \mathbb{R}^{(n+1)\times r}$ acts as the basis on the simplex and $\mathbf{V} \in \mathbb{R}^{r\times m}$ is the corresponding coefficient matrix. We give the constraints on $\mathbf{U}$ as

$$\mathbf{U}_{\cdot k}^T \mathbf{1}_{n+1} = 1, \forall k,$$

where $\mathbf{1}_{n+1}$ is an (n+1)-dimensional column matrix of all 1s.

Since each individual column of $\mathbf{X}_{(n+1)\times m}$ lies on the simplex $\mathbb{P}_n$ and $\mathbf{X}_{\cdot j} = \mathbf{UV}_{\cdot j}$, we get

$$(\mathbf{UV}_{\cdot j})^T \mathbf{1}_{n+1} = \mathbf{V}_{\cdot j}^T \mathbf{U}^T \mathbf{1}_{n+1} = \mathbf{V}_{\cdot j}^T \mathbf{1}_r = 1,$$

which implies $\mathbf{V}_{\cdot j}^T \mathbf{1}_r = 1, \forall j$. Hence, Eq. (3.1) can be reformulated as:

$$\begin{aligned} \min_{\mathbf{X},\mathbf{U},\mathbf{V}} \quad & \sum_{i=1}^{m} \mathbf{dist}_{\mathscr{M}}(\mathbf{X}_{\cdot j}, \mathbf{U}\mathbf{V}_{\cdot j}) \\ \text{s.t.} \quad & P_{\Omega}(\mathbf{X}-\mathbf{Y}) = 0, \mathbf{X}_{\cdot j}^T \mathbf{1}_{n+1} = 1, \\ & \mathbf{U}_{\cdot k}^T \mathbf{1}_{n+1} = 1, \mathbf{V}_{\cdot j}^T \mathbf{1}_r = 1, \end{aligned} \tag{3.2}$$

where $\mathbf{dist}_{\mathscr{M}}(\cdot,\cdot)$ refers to the distance on the simplex.

In many real world applications, one can only observe a small fraction of entries corrupted by some noises. For example, in cumulative voting, one is not sure about his votes; in family financing, people may hop between investments on two financial products. To enhance the robustness of collaborative allocation problem, we argue that the predicted matrix is not necessary to exactly match the observations. Considering the scenario in which observations are contaminated by noises, the exact match between observations and predictions may mislead the algorithm to suboptimal solutions. One single entry could render the estimated $\mathbf{X}$ arbitrarily far from the true $\mathbf{Y}_0$. To build a robust collaborative budget allocation system, one natural idea is to relax the constraint in Eq. (3.2) to avoid model's brittleness against grossly corrupted observations. Suppose we observe that $\mathbf{Y}_{ij} = \mathbf{X}_{ij} + v_{ij}$, the noise can be constrained by a parameter δ depending on the specific problem:

$$\sum_{(i,j)\in\Omega} v_{ij}^2 \leq \delta.$$

Based on above discussion, we propose our robust collaborative budget allocation model

$$\begin{aligned} \min_{\mathbf{X},\mathbf{U},\mathbf{V}} \quad & \sum_{i=1}^{m} \mathbf{dist}_{\mathscr{M}}(\mathbf{X}_{\cdot j}, \mathbf{U}\mathbf{V}_{\cdot j}) \\ \text{s.t.} \quad & \|P_{\Omega}(\mathbf{X}-\mathbf{Y})\|_F^2 \leq \delta, \mathbf{X}_{\cdot j}^T \mathbf{1}_{n+1} = 1, \\ & \mathbf{U}_{\cdot k}^T \mathbf{1}_{n+1} = 1, \mathbf{V}_{\cdot j}^T \mathbf{1}_r = 1. \end{aligned} \tag{3.3}$$

An $n-$simplex $\mathbb{P}_n$ is in fact a subset of $\mathbb{R}^{n+1}$, but Euclidean metric is not suitable to measure the distance between histogram data on the simplex, since it cannot depict the geometric

relationship between histograms. In the following section, we will study feasible metrics on histogram data.

3.4 Distance Metric on Histogram Data

In this section, we provide distance metrics for histogram data from two perspectives. On the one hand, we can specify a distance metric on the simplex directly by using pull-back metric from a sphere. On the other hand, we can map the histogram data into Euclidean space and measure the Euclidean distance between predictions and observations.

3.4.1 Pullback metric from Sphere

In information geometry, Fisher information metric is a particular Riemannian metric defined on the simplex. Given the diffeomorphism mapping $H : \mathbb{P}_n \mapsto \mathbb{S}_n^+$, we consider the pull-back metric from the positive orthant from the sphere $\mathbb{S}_n^+$, where

$$H(\mathbf{x}) = \sqrt{\mathbf{x}},$$

$$\mathbb{S}_n^+ = \{\mathbf{x} \in \mathbb{R}^{n+1} | \forall i, \mathbf{x}_i \geq 0 \text{ and } \sum_{i=1}^{n+1} \mathbf{x}_i^2 = 1\}.$$

The pull-back Fisher information metric (inner product) on the simplex $\mathbb{P}_n$ takes the form

$$g_{\mathbf{x}}(\xi_{\mathbf{x}}, \eta_{\mathbf{x}}) = \sum_{i=1}^{n+1} (\xi_{\mathbf{x}})_i (\eta_{\mathbf{x}})_i / \mathbf{x}_i,$$

which derives the geodesic distance on the simplex

$$d(\mathbf{x}, \mathbf{z}) = \cos^{-1}(\langle H(\mathbf{x}), H(\mathbf{z}) \rangle) = \cos^{-1}(\sum_{i=1}^{n+1} \sqrt{\mathbf{x}_i \mathbf{z}_i}).$$

Here $\langle \cdot, \cdot \rangle$ is the Euclidean inner product (more details about simplex can be found in [5]).

Based on the aforementioned knowledge on the simplex, we can proceed to specialize Eq. (3.2) to the simplex $\mathbb{P}_n$ to achieve the following model:

$$\begin{aligned} \min_{\mathbf{X},\mathbf{U},\mathbf{V}} \quad & \sum_{j=1}^{m} \cos^{-1}\Big(\sum_{i=1}^{n+1} \sqrt{\mathbf{X}_{ij}} \cdot \sqrt{\mathbf{U}_{i\cdot}\mathbf{V}_{\cdot j}}\Big) \\ \text{s.t.} \quad & P_{\Omega}(\mathbf{X}-\mathbf{Y}) = 0, \mathbf{X}_{\cdot j} \in \mathbb{P}_n, \forall j, \\ & \mathbf{U}_{\cdot k} \in \mathbb{P}_n, \forall k, \mathbf{V}_{\cdot j} \in \mathbb{P}_{r-1}, \forall j. \end{aligned} \tag{3.4}$$

By solving the above objective function, we expect to obtain user profiles $\mathbf{U}$ and item profiles $\mathbf{V}$, which then produce $\mathbf{X}$ to approximate unseen entries in the partially observed $\mathbf{Y}$. In contrast to existing matrix completion methods, the proposed model thoroughly investigates the geometric properties of the data, and thus it can lead to an optimal solution for the challenging collaborative rating allocation problem.

3.4.2 Euclidean maps of histogram data

We can investigate the embeddings that map probability simplex to appropriate Euclidean space with a suitable dimension. [55] claims that the information of histogram data is reflected in their relative value rather than the absolute value. Thus Euclidean distance will not be suitable here. Consider two data points on the simplex $\mathbf{x}, \mathbf{z} \in \mathbb{P}_n$, Aitchison explicitly explores the logarithmic ratio between $\mathbf{x}$ and $\mathbf{z}$,

$$\log \frac{\mathbf{x}_i}{\mathbf{z}_i} = \log \mathbf{x}_i - \log \mathbf{z}_i.$$

Additive log-ratio embedding

The first mapping proposed by Aitchison [56] is the additive log-ratio embedding that transforms a simplex data point $\mathbf{x}$ from $\mathbb{P}_n$ to $\mathbb{R}^n$,

$$\mathbf{alr}(\mathbf{x}) \stackrel{\text{def}}{=} \begin{bmatrix} \log \frac{\mathbf{x}_1+\varepsilon}{\mathbf{x}_{n+1}+\varepsilon} \\ \log \frac{\mathbf{x}_2+\varepsilon}{\mathbf{x}_{n+1}+\varepsilon} \\ \vdots \\ \log \frac{\mathbf{x}_n+\varepsilon}{\mathbf{x}_{n+1}+\varepsilon} \end{bmatrix} \in \mathbb{R}^n$$

where ε is a small regularization parameter. The **alr** can be reshaped by matrix operations in the following:

$$\mathbf{alr}(\mathbf{x}) = \mathbf{A}\log(\mathbf{x}+\varepsilon\mathbf{1}_{n+1}), \mathbf{A} = \begin{bmatrix} 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & -1 \end{bmatrix}. \tag{3.5}$$

Here $\mathbf{A} \in \mathbb{R}^{n\times(n+1)}$, $\mathbf{1}_{n+1} \in \mathbb{R}^{n+1}$ is the vectors with all its elements equal to ones, and $\log \mathbf{x}$ denote the element wisely logarithm of $\mathbf{x}$. The **alr** mapping is associated with the logistic-norm distribution on $\mathbb{P}_n$ [57, 58], and it defines an isomorphism between $(\mathbb{P}_n, \oplus, \otimes)$, and $(\mathbb{R}^n, +, \times)$, where $\oplus, \otimes$ are the perturbation and power on the multinomial manifold respectively [59]. The **alr** embedding is not isometric since it can not maintain the distance between $\mathbb{P}_n$ and $\mathbb{R}^n$. After the **alr** embedding, we are able to measure the distance of two data points, namely $\mathbf{alr}(\mathbf{x})$ and $\mathbf{alr}(\mathbf{z})$ by the Euclidean distance metric as L_2 norm. We have the following distance:

$$\begin{aligned} d_{\mathbf{alr}}(\mathbf{x},\mathbf{z}) &\stackrel{\text{def}}{=} d(\mathbf{alr}(\mathbf{x}), \mathbf{alr}(\mathbf{z})) \\ &= \|\mathbf{A}\log(\mathbf{x}+\varepsilon\mathbf{1}_{n+1}) - \mathbf{A}\log(\mathbf{z}+\varepsilon\mathbf{1}_{n+1})\|_2 \\ &= \|\log(\frac{\mathbf{x}+\varepsilon\mathbf{1}_{n+1}}{\mathbf{z}+\varepsilon\mathbf{1}_{n+1}})\|_{\mathbf{A}^T\mathbf{A}}. \end{aligned} \tag{3.6}$$

The objective for optimization will be formulated into this form:

$$\begin{aligned} \min_{\mathbf{X},\mathbf{U},\mathbf{V}} \quad & \frac{1}{2}\sum_{j=1}^{m} \| \log(\frac{\mathbf{X}_{\cdot j}+\varepsilon \mathbf{1}_{n+1}}{\mathbf{U}\mathbf{V}_{\cdot j}+\varepsilon \mathbf{1}_{n+1}}) \|^2_{\mathbf{A}^T\mathbf{A}} \\ \text{s.t.} \quad & \|P_{\Omega}(\mathbf{X}-\mathbf{Y})\|_F^2 \leq \delta,\ \mathbf{X}_{\cdot j}^T \mathbf{1}_{n+1} = 1, \\ & \mathbf{U}_{\cdot k}^T \mathbf{1}_{n+1} = 1, \mathbf{V}_{\cdot j}^T \mathbf{1}_r = 1. \end{aligned} \tag{3.7}$$

Here $\|\cdot\|_{\mathbf{O}}$ denotes the Mahalanobis norm with $\|\cdot\|_{\mathbf{O}} \stackrel{\text{def}}{=} \sqrt{\cdot^T \mathbf{O} \cdot}$. The optimization will be given in Section V.

Centered log-ratio embedding

The second mapping proposed by Aitchison [59] is centered log-ratio embedding (**clr**) which takes logarithmic ratio of $\mathbf{x}_i$ with the geometric mean of all its coordinates,

$$\mathbf{clr}(\mathbf{x}) \stackrel{\text{def}}{=} \begin{bmatrix} \log \frac{\mathbf{x}_1+\varepsilon}{\sqrt[n+1]{\prod_{j=1}^{n+1} \mathbf{x_j}+\varepsilon}} \\ \log \frac{\mathbf{x}_2+\varepsilon}{\sqrt[n+1]{\prod_{j=1}^{n+1} \mathbf{x_j}+\varepsilon}} \\ \vdots \\ \log \frac{\mathbf{x}_{n+1}+\varepsilon}{\sqrt[n+1]{\prod_{j=1}^{n+1} \mathbf{x_j}+\varepsilon}} \end{bmatrix} \in \mathbb{R}^{n+1}.$$

This embedding can be reformulated into matrix operations as follow:

$$\mathbf{clr}(\mathbf{x}) = (\mathbf{I}_{n+1} - \frac{\mathbf{1}_{(n+1)\times(n+1)}}{n+1}) \log(\mathbf{x}+\varepsilon \mathbf{1}_{n+1}). \tag{3.8}$$

The transformation **clr** is both isomorphism and isometry between $\mathbb{R}^{n+1}$ and multinomial manifold $\mathbb{P}_n$. By setting the matrix $\mathbf{A} = \mathbf{I}_{n+1} - \frac{\mathbf{1}_{(n+1)\times(n+1)}}{n+1}$, the collaborative rating allocation problem can be formulated in the same form as Eq. (3.7).

Isometric log-ratio embedding

[60] proposed to transform the image of **clr** embeddings onto $\mathbb{R}^n$, which is named isometric log-ratio embedding (**ilr**). The **ilr** embedding is defined as:

$$\mathbf{ilr}(\mathbf{x}) \stackrel{\text{def}}{=} \mathbf{B}(\mathbf{I}_{n+1} - \frac{\mathbf{1}_{(n+1)\times(n+1)}}{n+1}) \log(\mathbf{x} + \varepsilon \mathbf{1}_{n+1}) \in \mathbb{R}^n, \tag{3.9}$$

where $\mathbf{B} \in \mathbb{R}^{n\times(n+1)}$ and its row vectors act as a base of null space of $\mathbf{1}_{n+1}^T$. **ilr** mapping is also isometric between $\mathbb{P}_n$ and $\mathbb{R}^n$

Generalized Aitchison embedding

Besides the off-the-shelf Euclidean transformations that can be deployed, we can learn the embedding based on provided data [53]. Based on the form of previous embeddings, rather than setting up a specific weight matrix and defining an arbitrary regularization constrant ε, we can leverage instead these parameters to define a flexible Aitchison transformation based on training data. Let $\mathbf{P} \in \mathbb{R}^{d\times(n+1)}$ and $\mathbf{b} \in \mathbb{R}^{n+1}$ denote the weight matrix and regularization vector, the generalized Aitchison mapping $\alpha(\mathbf{x})$ is defined as

$$\alpha(\mathbf{x}) \stackrel{\text{def}}{=} \mathbf{P} \log(\mathbf{x} + \mathbf{b}) \in \mathbb{R}^d. \tag{3.10}$$

Thus the distance between two data points $\mathbf{x}$ and $\mathbf{z}$ is as follows:

$$\begin{aligned} d_\alpha(\mathbf{x}, \mathbf{z}) &\stackrel{\text{def}}{=} d(\alpha(\mathbf{x}), \alpha(\mathbf{z})) \\ &= \|\mathbf{P}\log(\mathbf{x}+\mathbf{b}) - \mathbf{P}\log(\mathbf{z}+\mathbf{b})\|_2 \\ &= \|\log(\frac{\mathbf{x}+\mathbf{b}}{\mathbf{z}+\mathbf{b}})\|_{\mathbf{Q}}, \end{aligned} \tag{3.11}$$

where $\mathbf{Q}$ is the positive semi-definite matrix with $\mathbf{Q} = \mathbf{P}^T\mathbf{P}$.

$$\begin{aligned} \min_{\mathbf{X},\mathbf{U},\mathbf{V},\mathbf{Q},\mathbf{b}} \quad & \frac{1}{2}\sum_{j=1}^{m} \|\log(\frac{\mathbf{X}_{\cdot\mathbf{j}}+\mathbf{b}}{\mathbf{U}\mathbf{V}_{\cdot\mathbf{j}}+\mathbf{b}})\|_{\mathbf{Q}}^2 \\ \text{s.t.} \quad & \|P_\Omega(\mathbf{X}-\mathbf{Y})\|_F^2 \leq \delta, \mathbf{X}_{\cdot j}^T\mathbf{1}_{n+1} = 1, \\ & \mathbf{U}_{\cdot k}^T\mathbf{1}_{n+1} = 1, \mathbf{V}_{\cdot j}^T\mathbf{1}_r = 1, \\ & \mathbf{Q} \succeq 0, \mathbf{b} > \mathbf{0}_{n+1}. \end{aligned} \tag{3.12}$$

Here $\mathbf{Q} \succeq 0$ is the abbreviation for $\mathbf{Q} \in \mathscr{S}^+$, which means that $\mathbf{Q}$ is a positive semi-definite matrix.

3.5 Optimization

In this section, we present the optimization procedure for loss functions under both Riemannian distance and Euclidean distance for Aitchison embeddings. Note that in both models, $\mathbf{U}, \mathbf{V}$ are required to be located on the $\mathbb{P}_n$ and $\mathbb{P}_{r-1}$ respectively. To preserve this geometric constraint, we will perform Riemannian conjugate gradient optimization for both methods. For Eq. (3.4), the general optimization procedure is split into two subproblems: (i) optimize $\mathbf{U}$ and $\mathbf{V}$ while $\mathbf{X}$ is fixed; and (ii) optimize $\mathbf{X}$ while $\mathbf{U}$ and $\mathbf{V}$ are fixed. For Eq. (3.12), two more parameters $\mathbf{Q}$ and $\mathbf{b}$ need to be included. This method refers to the alternating direction method (see [61] for further details). In Section 3.5.1, 3.5.2, 3.5.3, we introduce the optimization for $\mathbf{U}, \mathbf{V}$ and embedding parameters followed by updating rules for $\mathbf{X}$ in Section 3.5.4.

3.5.1 Euclidean gradients with metric on Simplex

During the optimization of the objective Eq. (3.4) when $\mathbf{X}$ is fixed, Eq. (3.4) degenerates to the following subproblem

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V}} \quad & F(\mathbf{U},\mathbf{V}) = \sum_{j=1}^{m} \cos^{-1}\Big(\sum_{i=1}^{n+1} \sqrt{\mathbf{X}_{ij}} \cdot \sqrt{\mathbf{U}_{i\cdot}\mathbf{V}_{\cdot j}}\Big) \\ \text{s.t.} \quad & \mathbf{U}_{\cdot k} \in \mathbb{P}_n, \forall k, \mathbf{V}_{\cdot j} \in \mathbb{P}_{r-1}, \forall j. \end{aligned} \tag{3.13}$$

Eq. (3.13) is a non-convex function with regard to $\mathbf{U}$ and $\mathbf{V}$, so we proceed to update $\mathbf{U}$ and $\mathbf{V}$ alternately. The objective function Eq. (3.13) can be decomposed over $\mathbf{U}_{i\cdot}$ row-wise and $\mathbf{V}_{\cdot j}$ column-wise. Taking derivatives to $\mathbf{U}_{i\cdot}$ and $\mathbf{V}_{\cdot j}$, we get

$$\frac{\partial F}{\partial \mathbf{U}_{i\cdot}} = -\frac{1}{2} \sum_{j=1}^{m} \frac{\sqrt{\mathbf{X}_{ij}}/\sqrt{\mathbf{U}_{i\cdot}\mathbf{V}_{\cdot j}}}{\sqrt{1-\mathbf{s}_j^2}} \cdot \mathbf{V}_{\cdot j}^T, \tag{3.14a}$$

$$\frac{\partial F}{\partial \mathbf{V}_{\cdot j}} = -\frac{1}{2\sqrt{1-\mathbf{s}_j^2}} \sum_{i=1}^{n+1} \frac{\sqrt{\mathbf{X}_{ij}}}{\sqrt{\mathbf{U}_{i\cdot}\mathbf{V}_{\cdot j}}} \cdot \mathbf{U}_{i\cdot}^T, \tag{3.14b}$$

where $\mathbf{s} \in \mathbb{R}^m, \mathbf{s}_j = \sum_{i=1}^{n+1} \sqrt{\mathbf{X}_{ij}} \cdot \sqrt{\mathbf{U}_{i\cdot}\mathbf{V}_{\cdot j}}, j = 1,2,...,m$. Since both $\mathbf{X}_{\cdot j}$ and $\mathbf{U}\mathbf{V}_{\cdot j}$ are on $\mathbb{P}_n$, $\mathbf{s}_j$ is guaranteed to be not bigger than 1.

We can optimize $\mathbf{V}_{\cdot j}$ column-by-column on the simplex $\mathbb{P}_{r-1}$. As for $\mathbf{U}_{i\cdot}$, however, row-by-row optimization may violate the constraints among the rows; for example, $\mathbf{U}_{ij}$ and $\mathbf{U}_{kj}$ with $i \neq k$ are not independent given $\mathbf{U}_{\cdot j}^T \mathbf{1}_{n+1} = 1$. In this case, we cannot optimize $\mathbf{U}_{i\cdot}$ and $\mathbf{U}_{k\cdot}$ independently. To tackle this problem, we consider the product manifold of the multiple simplex manifold

$$\mathbb{P}_n^r = \{\mathbf{U} = [\mathbf{U}_{ik}] \in \mathbb{R}^{(n+1)\times r}) : \mathbf{U}_{ik} \geq 0, \mathbf{U}^T \mathbf{1}_{n+1} = \mathbf{1}_r\}.$$

We consider stacking the derivatives of $\frac{\partial F}{\partial \mathbf{U}_{i\cdot}}$, obtain the derivative of $\mathbf{U}$

$$\frac{\partial F}{\partial \mathbf{U}} = ((\frac{\partial F}{\partial \mathbf{U}_{1\cdot}})^T, (\frac{\partial F}{\partial \mathbf{U}_{2\cdot}})^T, ..., (\frac{\partial F}{\partial \mathbf{U}_{n+1\cdot}})^T)^T,$$

and then perform manifold optimization on the product manifold.

This approach is convenient because: (i) optimizing on the product manifold $\mathbb{P}_n^r$ intrinsically implies the constraints among the rows of $\mathbf{U}$; and (ii) optimizing the rows of $\mathbf{U}$ jointly can parallelize computation and improve numerical efficiency. Although we do not need to jointly optimize $\mathbf{V}_{\cdot j}$, we process it similar to the optimization $\mathbf{U}$ to exploit numerical efficiency.

3.5.2 Euclidean gradients with Aitchison mappings

Considering Eq. (3.12), denote $\mathbf{W} = \log(\frac{\mathbf{X}+\mathbf{b}}{\mathbf{UV}+\mathbf{b}})$, $\mathbf{W}_{\cdot j} = \log(\frac{\mathbf{X}_{\cdot \mathbf{j}}+\mathbf{b}}{\mathbf{UV}_{\cdot \mathbf{j}}+\mathbf{b}}) = \log(\mathbf{X}_{\cdot \mathbf{j}}+\mathbf{b}) - \log(\mathbf{UV}_{\cdot \mathbf{j}}+\mathbf{b}), \mathbf{W} \in \mathbb{R}^{(n+1)\times m}, j = 1,...,m$, and $\mathbf{Q} = \mathbf{A}^T\mathbf{A}$, we have that

$$F(\mathbf{U},\mathbf{V}) = \sum_{j=1}^{m} \frac{1}{2}\mathbf{W}_{\cdot j}^T \mathbf{Q}\mathbf{W}_{\cdot j}.$$

By taking derivatives of $F(\mathbf{U},\mathbf{V})$ with regard to $\mathbf{U},\mathbf{V}$ respectively, we obtain

$$\frac{\partial F}{\partial \mathbf{U}} = \sum_{j=1}^{m} (\mathbf{Q}\mathbf{W}_{\cdot j} \circ \frac{-1}{\mathbf{UV}_{\cdot j}+\mathbf{b}}) \cdot \mathbf{V}_{\cdot j}^T, \tag{3.15a}$$

$$\frac{\partial F}{\partial \mathbf{V}_{\cdot j}} = \mathbf{U}^T \cdot (\mathbf{Q}\mathbf{W}_{\cdot j} \circ \frac{-1}{\mathbf{UV}_{\cdot j}+\mathbf{b}}). \tag{3.15b}$$

Here $\circ$ is the Schur product. By reorganizing the terms, we have that

$$\frac{\partial F}{\partial \mathbf{U}} = (\mathbf{Q}\mathbf{W} \circ \frac{-1}{\mathbf{UV}+\mathbf{b}}) \cdot \mathbf{V}^T, \tag{3.16a}$$

$$\frac{\partial F}{\partial \mathbf{V}} = \mathbf{U}^T \cdot (\mathbf{Q}\mathbf{W} \circ \frac{-1}{\mathbf{UV}+\mathbf{b}}). \tag{3.16b}$$

Follow the Riemannian conjugate gradient method, Eq. (3.7) can be optimized efficiently.

Considering Eq. (3.12), we have two more parameters $\mathbf{Q}$ and $\mathbf{b}$. We propose to alternately optimize $\mathbf{X},\mathbf{U},\mathbf{V},\mathbf{Q},\mathbf{b}$. While $\mathbf{Q},\mathbf{b},\mathbf{X}$ are fixed, $\mathbf{U},\mathbf{V}$ can still be updated by Eq. (3.16).

Taking derivatives to variable $\mathbf{Q}$ and $\mathbf{b}$, we have that:

$$\frac{\partial F}{\partial \mathbf{b}} = \sum_{j=1}^{m} \mathbf{Q}\mathbf{W}_{\cdot j} \circ \left(\frac{1}{\mathbf{X}_{\cdot j} + \mathbf{b}} - \frac{1}{\mathbf{U}\mathbf{V}_{\cdot j} + \mathbf{b}}\right), \tag{3.17a}$$

$$\frac{\partial F}{\partial \mathbf{Q}} = \sum_{j=1}^{m} \mathbf{W}_{\cdot j}\mathbf{W}_{\cdot j}^{T}. \tag{3.17b}$$

By reorganizing terms, we have the partial derivatives of $\mathbf{Q}, \mathbf{b}$ in the following form:

$$\frac{\partial F}{\partial \mathbf{b}} = \left(\mathbf{Q}\mathbf{W} \circ \left(\frac{1}{\mathbf{X} + \mathbf{b}} - \frac{1}{\mathbf{U}\mathbf{V} + \mathbf{b}}\right)\right) \cdot \mathbf{1}_{m \times 1}, \tag{3.18a}$$

$$\frac{\partial F}{\partial \mathbf{Q}} = \mathbf{W}\mathbf{W}^{T}. \tag{3.18b}$$

Thus the updating rules for $\mathbf{b}_t$ and $\mathbf{Q}_t$ can be formulated into the following form:

$$\mathbf{b}_t = \Pi_{+}\left(\mathbf{b}_{t-1} - \frac{t_0}{\sqrt{t}} \frac{\partial F}{\mathbf{b}}(\mathbf{Q}_{t-1}, \mathbf{b}_{t-1})\right), \tag{3.19a}$$

$$\mathbf{Q}_t = \Pi_{\mathscr{S}^{+}}\left(\mathbf{Q}_{t-1} - \frac{t_0}{\sqrt{t}} \frac{\partial \mathscr{F}}{\partial \mathbf{Q}}(\mathbf{Q}_{t-1}, \mathbf{b}_{t-1})\right), \tag{3.19b}$$

where Π_{+} and $\Pi_{\mathscr{S}^{+}}$ are projection operators that project variables to positive space in $\mathbb{R}^{n+1}$ and cone of positive semi-definite matrix respectively.

As $\mathbf{Q}$ is a positive semi-definite matrix which can be decomposed into $\mathbf{P}^{T}\mathbf{P}$, we may consider optimizing $\mathbf{P}$ directly. The subgradient of F with respect to $\mathbf{P}$ is

$$\frac{\partial F}{\partial \mathbf{P}} = 2\mathbf{P}\frac{\partial F}{\mathbf{Q}} \in \mathbb{R}^{d \times (n+1)}. \tag{3.20}$$

Thus the problem can be solved by alternately optimizing $\mathbf{X}, \mathbf{U}, \mathbf{V}, \mathbf{P}, \mathbf{b}$.

Optimization on a Riemannian manifold is comparable to that on a Euclidean space, that is: (i) finding a descent direction; and (ii) performing a line search to obtain a sufficient decrease and to ensure convergence. In the following section, we discuss the optimization of $\mathbf{U}$ in detail to illustrate our optimization method on the simplex. The optimization of $\mathbf{V}$ is similar, which is omitted here.

3.5.3 Conjugate gradients with the Armijo line search method

The conjugate gradient method is chosen to solve the model in Eq. (4) due to its well-recognized efficiency when applied to large-scale problems. The conjugate gradient method incorporates gradients at the current point with descent directions from previous points to obtain a new direction. Let $f(\mathbf{U})$ and $g(\mathbf{V})$ denote $F(\mathbf{U},\mathbf{V})$ with $\mathbf{V}$ and $\mathbf{U}$ fixed respectiveely, in Euclidean space, a conjugate gradient method generates a sequence $\mathbf{U}_k$ by the following recurrence:

$$\mathbf{U}_{k+1} = \mathbf{U}_k + \alpha_k \xi_{\mathbf{U}_k}, \tag{3.21}$$

where α_k is the step size generated by a line search algorithm, and in Euclidean space $\xi_{\mathbf{U}_k}$ is the descent direction generated by the following rule:

$$\xi_{\mathbf{U}_k} = -\frac{\partial f}{\partial \mathbf{U}} + \beta_{k-1}\xi_{\mathbf{U}_{k-1}}.$$

On a Riemannian manifold, the descent direction is computed on the tangent space. This space varies smoothly as one moves along the manifold. At a point $\mathbf{U}$, the tangent space $T_{\mathbf{U}}\mathbb{P}_n^r$ is a vector space and is actually a subspace of its embedded Euclidean space,

$$T_{\mathbf{U}}\mathbb{P}_n^r = \{\xi_{\mathbf{U}} \in R^{(n+1)\times r} : \xi_{\mathbf{U}}^T \mathbf{1}_{n+1} = \mathbf{0}_r\}. \tag{3.22}$$

Given a descent direction $\xi_{\mathbf{U}} \in T_{\mathbf{U}}\mathbb{P}_n^r$, the line-search is performed along a smooth curve on the manifold. The derivative of this curve at $\mathbf{U}$ equals the descent direction $\xi_{\mathbf{U}}$. After one step movement on tangent space $T_{\mathbf{U}}$, we need to retract it back to the manifold. The retraction operation is defined as a mapping from the tangent space to the manifold. The updating step is concluded by $\mathbf{U}_{k+1} = R_{\mathbf{U}_k}(\alpha_k \xi_{\mathbf{U}_k})$.

Unfortunately, the tangent space $T_{\mathbf{U}}\mathbb{P}_n^r$ is defined locally at point $\mathbf{U}$, which means $\xi_{\mathbf{U}_k} \in T_{\mathbf{U}_k}\mathbb{P}_n^r$ and $\xi_{\mathbf{U}_{k+1}} \in T_{\mathbf{U}_{k+1}}\mathbb{P}_n^r$ are in different tangent spaces and might not coincide. This property makes the direct addition of tangent vector $\xi_{\mathbf{U}_k}$ and $\xi_{\mathbf{U}_{k+1}}$ impossible. To adapt the conjugate gradient method to manifolds, vector transport needs to be defined to transport tangent vectors from one tangent space to another.

To specify this recurrence to the simplex, we need to define: (i) a projection of the matrix in ambient space into the tangent space; and (ii) a retracting mapping from tangent space to the manifold $R_{\mathbf{U}_k}(\alpha_k \xi_{\mathbf{U}_{k+1}})$.

(i) Projection: the linear operation that projects a matrix $\mathbf{Z} \in R^{n \times r}$ into the tangent space $T_{\mathbf{U}}\mathbb{P}_n^r$, is defined as

$$\Pi_{\mathbf{U}}(\mathbf{Z}) = \mathbf{Z} - \mathbf{1}_{n+1}\mathbf{1}_{n+1}^T \mathbf{Z} \otimes \mathbf{U}, \tag{3.23}$$

where $\otimes$ is the element-wise matrix multiplication operation.

(ii) Retraction: the mapping that locates the next iteration to the manifold along the tangent vector $R_{\mathbf{U}} : T_{\mathbf{U}}\mathbb{P}_n^r \to \mathbb{P}_n^r$ is

$$\begin{aligned} \mathbf{U}_+ &= R_{\mathbf{U}}(\xi_{\mathbf{U}}) \\ &= \mathbf{U} \otimes \exp(\xi_{\mathbf{U}}\mathbf{U}^*) \cdot (\mathbf{1}_{n+1}\mathbf{1}_{n+1}^T (\mathbf{U} \otimes \exp(\xi_{\mathbf{U}}\mathbf{U}^*)))^* \end{aligned} \tag{3.24}$$

where ()* denotes the element-wise matrix inversion and $\exp(\cdot)$ denotes the element-wise exponential operator.

Let $\mathrm{grad} f(\mathbf{U})$ denote the Riemannian gradient in tangent space $T_{\mathbf{U}}\mathbb{P}_n^r$ at $\mathbf{U}$; it is calculated by projecting a normalized Euclidean gradient into the tangent space $T_{\mathbf{U}}\mathbb{P}_n^r$,

$$\mathrm{grad} f(\mathbf{U}) = \Pi_{\mathbf{U}}(\frac{\partial f}{\partial \mathbf{U}} \otimes \mathbf{U}). \tag{3.25}$$

The vector transport from the previous tangent space to the current tangent space

$$\mathscr{T}_{\mathbf{U}_k \to \mathbf{U}_{k+1}} : T_{\mathbf{U}_k}\mathbb{P}_n^r \to T_{\mathbf{U}_{k+1}}\mathbb{P}_n^r, \xi_{\mathbf{U}_k} \mapsto \Pi_{\mathbf{U}_{k+1}}(\xi_{\mathbf{U}_k}) \tag{3.26}$$

is specified by the projection defined in Eq.(3.23).

Hence the conjugate gradient method can be specified to the simplex

$$\mathbf{U}_{k+1} = R_{\mathbf{U}_k}(\alpha_k \xi_{\mathbf{U}_k}) \tag{3.27}$$

with $\xi_{\mathbf{U}_k} = -\mathrm{grad} f(\mathbf{U}_k) + \beta_{k-1}\mathscr{T}_{\mathbf{U}_{k-1} \to \mathbf{U}_k}\xi_{\mathbf{U}_{k-1}}$.

Algorithm 2 Alternating Direction Method for Eq. (3.4)

Input: $\mathbf{Y}$: Incomplete matrix, Ω: sampling set, ε: stopping criteria, k: latent dimension, T: max iteration

Output: $\mathbf{X}$: the optimal approximation of $\mathbf{Y}$

1: Initialize $\mathbf{U}_0, \mathbf{V}_0$ and $\mathbf{X}_0 = O_{(n+1)\times m}$
2: **for** $t = 1, 2, ..., T$ **do**
3: Update $\mathbf{X}_k$ according to Eq. (3.31)
4: Update $\mathbf{U}_t$:
4a: choose a proper step size τ_t from Eq.(3.29)
$\mathbf{U}_{init} = -\tau_t \text{grad} f(\mathbf{U}_{t-1})$
4b: use $\mathbf{U}_{init}$ as an initial guess and call RSCG
$\mathbf{U}_t = \text{RSCG}(\mathbf{U}_{init})$
5: Update $\mathbf{V}_t$:
5a: choose a proper step size υ_t from Eq.(3.29)
$\mathbf{V}_{init} = -\upsilon_t \text{grad} g(\mathbf{V}_{t-1})$
5b: use $\mathbf{U}_{init}$ as an initial guess and call RSCG
$\mathbf{V}_t = \text{RSCG}(\mathbf{V}_{init})$
6: If 1-$F(\mathbf{U}_t, \mathbf{V}_t)/F(\mathbf{U}_{t-1}, \mathbf{V}_{t-1}) < \varepsilon$; break; end
7: **end for**

Conjugate Gradients Update Parameter

Several options of β_k can be used in the conjugate gradient algorithm. Here we choose the Polak-Ribiere(PR+) method to calculate the CG update parameter. Within the Riemannian optimization framework, β_k takes the following form:

$$\beta_k = \frac{\langle \text{grad} f(\mathbf{U}_k), \text{grad} f(\mathbf{U}_k) - \mathscr{T}_{\mathbf{U}_{k-1} \to \mathbf{U}_k}(\text{grad} f(\mathbf{U}_{k-1})) \rangle}{\langle \text{grad} f(\mathbf{U}_{k-1}), \text{grad} f(\mathbf{U}_{k-1}) \rangle}.$$

Note that the PR+ updating rule has the auto restart property when the descent direction is almost orthogonal to the gradient direction. Specifically, the algorithm will abandon the previous CG descent direction and initialize the CG algorithm according to the current gradient direction.

Armijo Step Size

The step size α_k is chosen with the Armijo line search algorithm. Given the initialized step size $\overline{\alpha}$ and $t, \sigma \in (0,1)$, the condition is given by:

$$f(\mathbf{U}) - f(R_{\mathbf{U}}(t^m \overline{\alpha} \xi)) \geq -\sigma \langle \text{grad} f(\mathbf{U}), t^m \overline{\alpha} \xi \rangle_{\mathbf{U}},$$

where m is the number of backtrackings in finding the current Armijo point. Thus the step size is given by $\alpha = t^m \overline{\alpha}$.

The initialization of step size $\overline{\alpha}$ is very important since it can greatly enhance the efficiency of the line search algorithm. In our method, we observe that an exact minization on the tangent space alone without retraction is a good initial guess. For $\mathbf{U}$, the exact minimization takes the form

$$\min_t \frac{1}{2} \|\mathbf{X} - (\mathbf{U} + t\xi)\mathbf{V}\|_F^2, \tag{3.28}$$

$\mathbf{V}$ takes a similar form so it is omitted here. The minimizer of Eq. (3.28) takes a closed-form solution:

$$t_* = \langle \mathbf{X} - \mathbf{UV}, \xi \mathbf{V} \rangle / \langle \xi \mathbf{V}, \xi \mathbf{V} \rangle. \tag{3.29}$$

As ξ is the search direction of $\mathbf{U}$, it is always non-zero when the algorithm has not converged. Therefore, there are no worries about the denominator being zero.

We name the proposed algorithm the "Riemannian Simplex Conjugate Gradient" (RSCG). The convergence of RSCG can be guaranteed by [62, 37] when β is chosen by the PR+ rule and α is chosen by the Armijo condition.

3.5.4 Updating X on Simplex

$\mathbf{X}$ is used to approximate true rating values in $\mathbf{Y}$ with $\mathbf{X} = \mathbf{UV}$ and $P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{Y})$. In a typical collaborative filtering algorithm, we have:

$$\mathbf{Z}_k = \mathbf{U}_k \mathbf{V}_k + P_\Omega(\mathbf{Y} - \mathbf{U}_k \mathbf{V}_k).$$

Since $\mathbf{Y}$ lies on the simplex, above updating step might violate this constraint. To ensure this, a projection is needed to project it back to the constraint set [63], which is defined as follows:

$$\mathbf{Z} = \mathbf{Z} \oslash (\mathbf{1}_n \mathbf{1}_n^T \mathbf{Z}),$$

where $\oslash$ is the point-wise division operator. Considering the constraint on matching observed values in $\mathbf{Y}$, updating rule for $\mathbf{X}$ in Eq. (3.4) is as follows:

$$\mathbf{X}_k = P_{\Omega^c}(\mathbf{Z}_k) \oslash \mathbf{1}_n \mathbf{1}_n^T P_{\Omega^c}(\mathbf{Z}_k) \otimes [1 - \mathbf{1}_n \mathbf{1}_n^T P_{\Omega}(\mathbf{Z}_k)] + P_{\Omega}(\mathbf{Z}_k), \tag{3.30}$$

where Ω^c is the complement set of Ω. In relaxed constraints in Eq. (3.12), exact match between predictions and true values on observed is not the best choice. We relax this constraint during projection by introducing α that balances observations and predictions during optimization:

$$\mathbf{X}_k = P_{\Omega^c}(\mathbf{Z}_k) \oslash \mathbf{1}_n \mathbf{1}_n^T P_{\Omega^c}(\mathbf{Z}_k) \otimes [1 - \mathbf{1}_n \mathbf{1}_n^T P_{\Omega}(\alpha \mathbf{Z}_k)] + P_{\Omega}(\alpha \mathbf{Z}_k). \tag{3.31}$$

α is a hyper parameter and is determined by validation. The overall optimization algorithm for Eq. (3.4) is summarized in Algorithm 2.

3.6 Experiments

In this section, we evaluate the proposed CBA-Simplex on three real-world datasets and compare it with PMF [32], NPCA [47], LGeomCG [64] and LMaFit [65]. PMF is a probabilistic matrix factorization model for collaborative filtering that models users' preference by a conditional distribution on user and item matrices; LMaFit performs basic matrix factorization while employing non-linear successive over-relaxation algorithm for advancing the effectiveness; NPCA refers to a non-parametric probabilistic PCA model; and LGeomCG performs matrix completion on the manifold of fixed-rank matrices. The RSCG algorithm is implemented based on Manopt [63].

Table 3.1 Dataset information

Dataset	#Samp.	#Class	Feature	Rep	#Dim
MIT Scene	2080	8	SIFT	BoF	600
UIUC Scene	3000	15	SIFT	BoF	600

Table 3.2 Comparisons of different collaborative filtering methods in terms of RMSE ($\times 10^{-3}$) and NMAE ($\times 10^{-3}$)

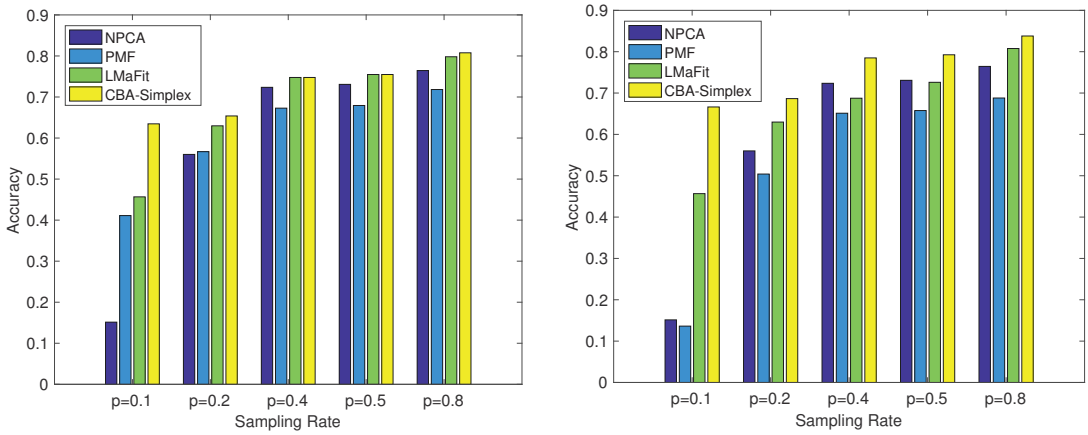
Methods	MIT Scene		UIUC Scene	
	W. RMSE	S. RMSE	W. RMSE	S. RMSE
LMaFit	10.897	8.804	8.802	8.238
LGeomCG	9.691	—-	11.415	—-
PMF	9.271	9.129	8.508	8.508
NPCA	4.028	8.926	3.224	6.587
CBA-Simplex	2.929	4.194	2.673	4.201
	W. NMAE	S. NMAE	W. NMAE	S. NMAE
LMaFit	2.078	2.383	2.175	1.833
LGeomCG	1.561	—-	1.738	—-
PMF	2.476	3.185	2.426	2.426
NPCA	1.155	2.648	0.830	2.305
CBA-Simplex	1.197	1.721	1.310	1.929

Following Marlin's setup [66], we test our algorithm under "weak" and "strong" generalization. The weak generalization is a single step process that involves directly predicting missing data in the rating matrix. The strong generalization refers to a two-stage process where the models are trained on one set of users and tested on another disjoint set of users. Since LGeomCG cannot explicitly factorize a rating matrix into user profiles and item profiles, it is not applicable for strong generalization. The measurements for matrix recovery are root mean square error (RMSE) and normalized mean absolute error (NMAE):

$$\text{RMSE} = \|P_\Omega(\mathbf{Y}-\mathbf{X})\|_F/\sqrt{|\Omega|},$$

$$\text{NMAE} = \|P_\Omega(\mathbf{Y}-\mathbf{X})\|_1/(\mathbf{Y}_{\max}-\mathbf{Y}_{\min})|\Omega|,$$

where $\mathbf{Y}$ represents ground truth ratings and $\mathbf{X}$ shows estimated values.



(a) MIT Scene (b) UIUC Scene

Fig. 3.2 Classification results vary with sampling fraction on MIT Scene and UIUC Scene.

3.6.1 Image data

We consider two image datasets MIT Scene and UIUC Scene whose properties and parameters are displayed in Table 3.1. We extract SIFT descriptors on 16×16 patches in images and then convert them to bag-of-features (BoF). The number of visual words in the codebook, $|V|$, is set to be 600 and the implementation is based on *vlfeat toolbox* [67].

The BoF representation of each image is the histogram of visual words. Each dimension of a histogram represents the relative frequency of a visual word in the descriptors of an image and the sum of the histogram equals 1. It is an estimation of the probability distribution and is exactly on a $(|V|-1)$-simplex.

We first extract the BoF features of an image dataset. Define sampling fraction $p = \frac{|\Omega|}{(n+1)m}$ as the ratio of observed entries to total entries. By setting $p = 0.5$, we randomly keep 50% of features for fitting collaborative filtering models. Then we split the dataset into training, validation and test sets. In "weak" generalization, we train the collaborative filtering algorithms on training sets and recover the full matrix. Then we train multi-class Support Vector Machine classifiers on the recovered training set by one-vs-all strategy. In "strong" generalization, we train the collaborative filtering algorithms on test sets by using the basis matrix learned from training sets and hence estimate missing entries in test sets. At last we evaluate the performance of SVM classifier on recovered test sets.

Evaluation results

The number of basis points on the simplex r is set to be the number of classes. We randomly start the training of collaborative filtering algorithms for five times and report averaged RMSE and NMAE in Table 3.2. We find that CBA-Simplex outperforms baselines on two datasets under both "weak" and "strong" generalization. The BoF feature vectors lie exactly on the $(|V|-1)$-simplex. While comparative algorithms ignore this property, we subtly make use of data's geometric properties and constrain the optimization of each user's rating vector on the simplex, which leads to prominent performance of CBA-Simplex. The corresponding classification performance is illustrated in Fig. 3.2 with different sampling fractions.

When sampling fraction $p = 1$, which means no feature is missing, the averaged classification accuracies of MIT Scene and UIUC Scene are 88.75% and 86.87% respectively. Based on the collaborative filtering results under different sampling fraction p, we repeat the training of SVM for five times, and report averaged accuracies in Fig. 3.2. With the growing of sampling fractions, all methods witness a rising performance. Under different sampling fractions of feature matrix, our method CBA-Simplex achieves consistently better performance on UIUC Scene and shows promising efficacy especially when the sample size is small. This indicates that our collaborative filtering method implements better approximation to the ground truth feature matrix.

3.6.2 Recommendation data

We apply NMAE and the normalized mean square error (NMSE) to measure the recovery error which is defined as

$$\text{NMSE} = \frac{\sum_{(i,j)\in\Omega}(\mathbf{Y}_{ij} - \mathbf{X}_{ij})^2}{(\mathbf{Y}_{\max} - \mathbf{Y}_{\min})^2|\Omega|}.$$

We next explore the proposed CBA-Simplex's performance on movie recommendation datasets in collaborative filtering, namely, MovieLens 100K (ML-100K), MovieLens 1M (ML-1M) [68] and EachMovie [69]. The input ratings are integers between 1 and 5 for the MovieLens datasets and between 1 to 6 for EachMovie. The dataset information is summarized in Table 3.3.

MovieLens 100K consists of 100 thousands of ratings from 943 users on 1682 movies. Each user has rated at least 20 movies, so we include all users in our experiments. We randomly select 800 users for weak generalization and the other 143 users for strong generalization. MovieLens 1M contains 1,000,209 anonymous ratings of 3,952 movies by 6,040 MovieLens users. 5,000 users are selected for weak generalization and 1,040 users for strong generalization. EachMovie contains 2.8 million ratings from 72,916 users on 1,648 different movies. By discarding users with less than 20 ratings, we keep 36,656 users with a total of 2,579,985 ratings. Of these, 30,000 users are used for weak generalization and 6656 users for strong generalization. For all algorithms, the latent dimension is set to 100 for EachMovie [31], 150 for MovieLens 1M and 10 for MovieLens 100K, respectively.

The input ratings are integers between 1 and 5, and the budget of user j's ratings on items is fixed, i.e., $y = \sum_{i=1}^{n+1} \mathbf{Y}_{ij}$. We can normalize the data through $\mathbf{Y}'_{ij} = \mathbf{Y}_{ij}/y,\ i = 1,2,...,n+1$. The pre-processed ratings of user i now lie on the simplex satisfying $\sum_{i=1}^{n+1} \mathbf{Y}'_{ij} = 1$. By assuming that sampling of Ω is random and unbiased, the budget of one user's ratings on n items is approximated by $y \approx (\sum_{(i,j)\in\Omega} P_\Omega(\mathbf{Y}))(n+1)/\Omega$. The test and validation sets are created by reserving one rating from each user respectively. This process is repeated three times. We compare the averaged NMAE on three random partitions of weak and strong generalization and report the results in Table 3.5. After the preprocessing, users' rating vectors lie on the simplex and our method CBA-Simplex can preserve this geometric property during the learning of user and item matrices, while baseline methods optimize in the free space and hence gain deficient performance.

Tables 3.4, 3.5, and 3.6 give the performance comparisons of different algorithms. We find that CBA-Simplex outperforms baselines on three datasets under both weak and strong generalization. This manifests the significance of employing simplex constraints on each user's rating vector and the effectiveness of our pull-back metric from the sphere. Moreover, better performance under strong generalization affirms CBA-Simplex's superiority in learning coefficient matrix which can be treated as item profiles inherited from weak generalization.

Table 3.3 Dataset Information

Datasets	Dimension	$\|\Omega\|$	Sampling ratio
ML-100K	1682×943	100,000	6.305%
ML-1M	3952×6040	1,000,209	4.190%
EachMovie	1648×36656	2,579,985	4.271%

Table 3.4 MovieLens 100K: Comparisons with baselines in terms of NMSE and NMAE

Methods	Weak NMAE	Strong NMAE
LMaFit	0.6495±0.0033	0.6463±0.0187
LGeomCG	0.6299±0.0089	—
PMF	0.2357±0.0025	0.2548±0.0138
NPCA	0.2278±0.0023	0.2472± 0.013
CBA-Simplex	0.2152±0.0023	0.2410±0.0118
	Weak NMSE	Strong NMSE
LMaFit	0.4988±0.0233	0.4986± 0.0090
LGeomCG	0.4743±0.0034	—
PMF	0.0783±0.0082	0.0907±0.0092
NPCA	0.0749± 0.0099	0.0871±0.0129
CBA-Simplex	0.0722±0.0004	0.0745± 0.0062

Table 3.5 MovieLens 1M: Comparisons with baselines in terms of NMSE and NMAE

Methods	Weak NMAE	Strong NMAE
LMaFit	0.6808± 0.001	0.6740±0.0048
LGeomCG	0.6275±0.0032	—
PMF	0.2339±0.0008	0.2339±0.0036
NPCA	0.2267±0.0009	0.2266±0.0043
CBA-Simplex	0.2128±0.0022	0.2143±0.0025
	Weak NMSE	Strong NMSE
LMaFit	0.5388±0.0107	0.5277±0.0251
LGeomCG	0.4723±0.0102	—
PMF	0.0769±0.0113	0.0763±0.0166
NPCA	0.0736±0.0041	0.0731±0.0076
CBA-Simplex	0.0702±0.0027	0.0714±0.0122

Latent Dimensionality and Efficiency

To test the response of CBA-Simplex to increasing latent dimension, we report our testing NMSE and NMAE on MovieLens 1M under latent dimension from 10 to 1000. In Fig. 3.3, both NMSE and NMAE show obvious decreasing trends with the increasing of latent

Table 3.6 EachMovie: Comparisons with baselines in terms of NMSE and NMAE

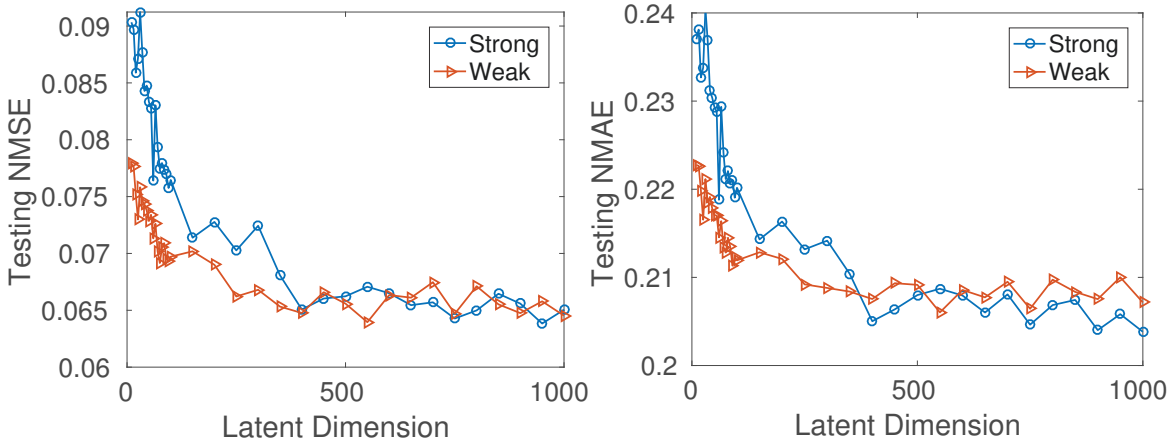
Methods	Weak NMAE	Strong NMAE
LMaFit	0.6259± 0.001	0.6245±0.0045
LGeomCG	0.4983±0.0034	—
PMF	0.2450±0.0004	0.2447±0.0018
NPCA	0.2419±0.0004	0.2419±0.0018
CBA-Simplex	0.2167±0.0097	0.2134±0.0147
	Weak NMSE	Strong NMSE
LMaFit	0.4859±0.0032	0.4843±0.0169
LGeomCG	0.3583±0.0046	—
PMF	0.0949±0.0038	0.0954±0.0079
NPCA	0.0911±0.0052	0.0917±0.0063
CBA-Simplex	0.0815±0.0047	0.0778±0.0125

dimension and they do not go up while the latent dimension is bigger than 100, which indicates that CBA-Simplex is immune to overfitting.

To verify the efficiency of CBA-Simplex, changes of NMSE and NMAE with iterations under different latent dimensionality in training process are reported in Fig. 3.4. We find that in weak generalization, CBA-Simplex reduces NMSE and NMAE by every iteration and stays stable after 10 iterations while in strong generalization, CBA-Simplex stays stable after only three iterations. We owe this efficiency to the representative model learned from weak generalization. The model's representativeness indicates CBA-Simplex's capability in excavating the geometric properties of rating vectors on the simplex.

3.6.3 Comparisons of different metrics

Denote the Aitchison mappings **alr, clr, ilr** and Generalized Aitchison embedding for collaborative budget allocation as CBA-**alr**, CBA-**clr**, CBA-**ilr** and CBA-**GenAit** respectively. We report the recovery performance of collaborative budget allocation algorithm under different embeddings in Table 3.7. For the ease of comparison, we list the recovery error of CBA-Simplex in Table 3.7 as well. From Table 3.7, we observe that CBA-**GenAit** achieves better results than CBA-Simplex, which implies that learning the metric based on training data is semantically superior to the arbitrarily specified distance on the Simplex. While CBA-**clr**



(a) NMSE (b) NMAE

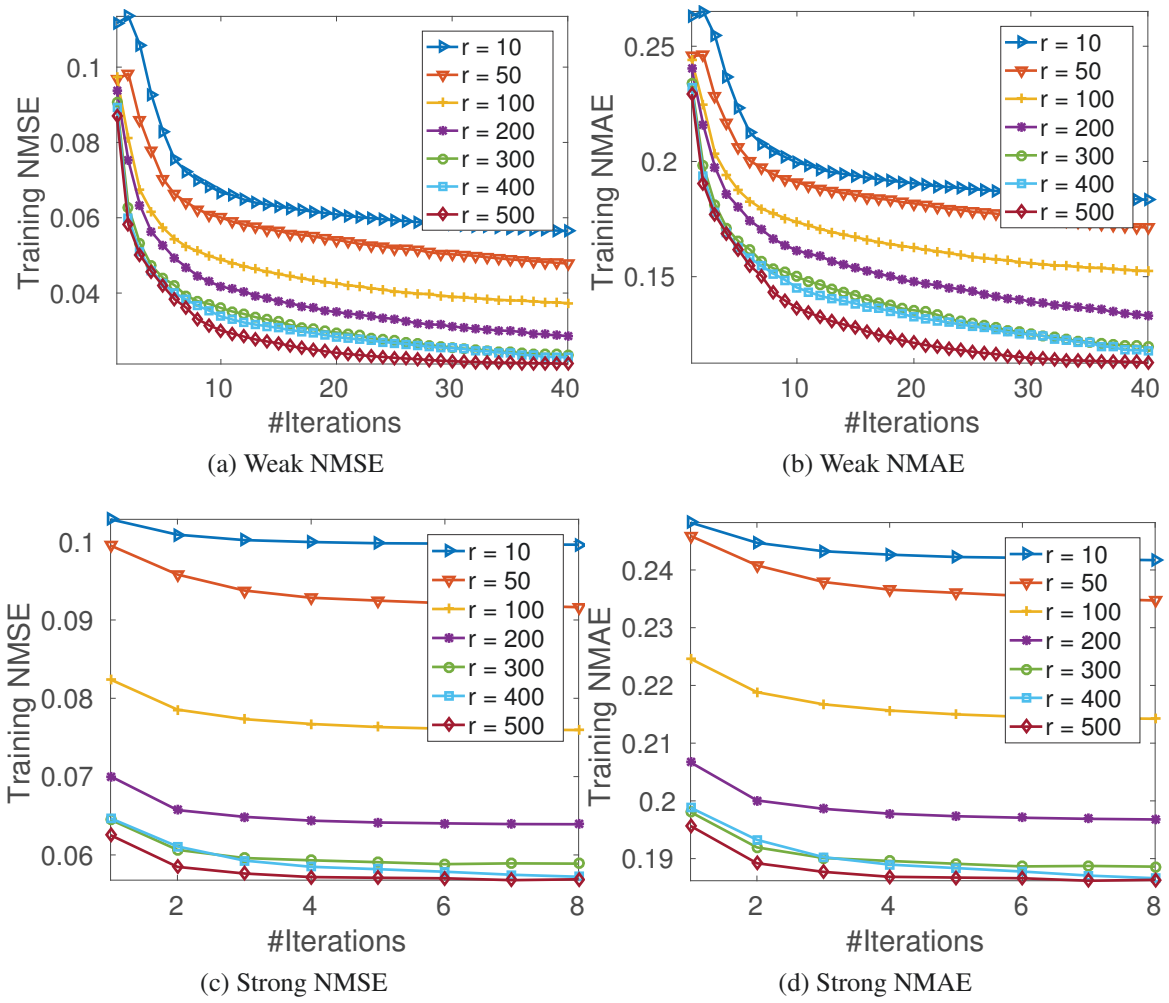
Fig. 3.3 MovieLens 1M: NMSE and NMAE changing with latent dimensions between 10 and 1000 under weak and strong generalization.

Table 3.7 Comparisons of different Aitchison embeddings for histogram data in terms of RMSE ($\times 10^{-3}$) and NMAE ($\times 10^{-3}$)

Methods	MIT Scene		UIUC Scene	
	W. RMSE	S. RMSE	W. RMSE	S. RMSE
CBA-**alr**	3.897	6.804	2.802	4.823
CBA-**clr**	2.910	4.011	2.415	4.301
CBA-**ilr**	3.071	4.129	2.508	4.508
CBA-**GenAit**	2.828	3.926	2.224	3.587
CBA-Simplex	2.929	4.194	2.673	4.201
	W. NMAE	S. NMAE	W. NMAE	S. NMAE
CBA-**alr**	2.078	2.383	1.715	2.833
CBA-**clr**	1.261	1.753	1.387	2.213
CBA-**ilr**	1.276	1.852	1.426	2.426
CBA-**GenAit**	1.155	1.648	1.283	1.835
CBA-Simplex	1.197	1.721	1.310	1.929

and CBA-**ilr** have similar performance to CBA-Simplex, CBA-**alr**'s results deteriorate. This indicates that the artificially specified metrics cannot always be effective, but learning the metric from provided data will usually have superior performance.

To examine performance under different quality of observations, we test these methods under noisy data that is corrupted by noises. Specifically, we generate noise that follows Gaussian distribution $\mathcal{N}(0, \rho_f^2)$ where ρ_f stands for standard deviation and is the noise level.



(a) Weak NMSE (b) Weak NMAE

(c) Strong NMSE (d) Strong NMAE

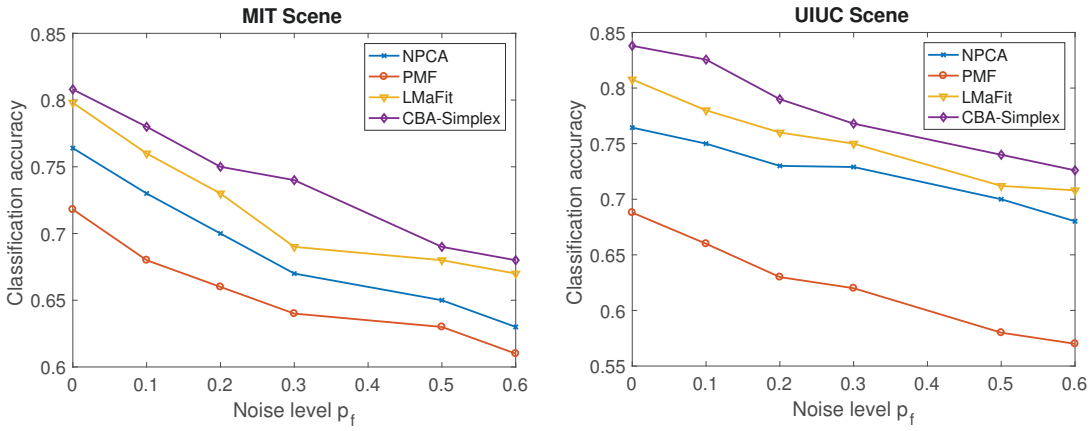
Fig. 3.4 MovieLens 1M: NMSE and NMAE as objective decreasing with iterations under weak and strong generalization. r refers to latent factors of 10, 50, 100, 200, 300, 400, 500.

Under sampling rate $p = 0.8$ and noise level $\rho_f = 0.1$, we report the classification accuracy for data recovered by our budget allocation methods under different embeddings in Table 3.8. It can be observed that the relaxed recovery constraints lead to more stable prediction performance. Thus this verifies the effectiveness of proposed robust model for collaborative budget allocation approach.

Furthermore, we compare CBA-Simplex with baselines under different noise levels. We generate matrix recovery data with different noise levels and test prediction methods under noise level ρ_f={0.1, 0.2, 0.3, 0.5 0.6} and report the obtained classification accuracy in Fig. 3.5. Both baseline methods and CBA-Simplex decrease with the noises while CBA-Simplex

Table 3.8 Comparisons of classification accuracy under different Aitchison embeddings with exact and relaxed recovery constraints

Methods	MIT Scene		UIUC Scene	
	Exact rec.	Relaxed rec.	Exact rec.	Relaxed rec.
CBA-**alr**	62.90%	65.43%	68.66%	73.17%
CBA-**clr**	67.26%	73.09%	73.43%	80.69%
CBA-**ilr**	66.48%	70.38%	75.87%	79.39%
CBA-**GenAit**	78.23%	82.68%	80.21%	83.51%
CBA-Simplex	72.19%	78.20%	75.32%	81.76%



(a) MIT Scene (b) UIUC Scene

Fig. 3.5 Classification results vary with sampling fraction on MIT Scene and UIUC Scene.

achieves more smooth performance. This verifies the robustness of the proposed relaxed recovery model.

3.7 Conclusions

In this chapter, we studied the collaborative budget allocation problem in sharing economy system. In order to measure the distance between two histogram data points, we propose to use either Riemannian metric on simplex or Aitchison mappings to preserve the geometric relationship between histogram data points. To intensify the model's robustness in real applications, a relaxed recovery constraints are studied. The proposed methods are optimized by Riemannian conjugate gradients on the simplex. We verify our models' superiority over

baseline methods, and compare the difference between distance metrics introduced in this chapter. We conclude that (i) to learn the metric based on the training data tends to have better performance; (ii) to relax the exact recovery constraint can improve the performance of budget allocation when preference value observations are corrupted by noise.

Chapter 4

Learning from Noisy Side Information

Matrix recovery from few observations is a significant problem and has wide applications in practice. Inductive matrix completion approach recovers the ground truth matrix under minimal rank constraint with side information which is often used in recommendation systems to characterize users and items, and, has been proven effective in various applications. However, side information is likely to compromise recommendation systems when there is noise. Identifying side information noise and selecting related side features are important for rating prediction but are usually omitted in existing methods. To solve this problem and hence enhance the recommendation system's robustness, here we propose the robust inductive matrix completion (RIMC) method, which recovers the underlying rating matrix and simultaneously identifies noisy dimensions. Specifically, we revisit the low-rank principle that aligns user and item side features with rating predictions and introduce a group sparse part to identify noisy side features and eliminate their negative influence. Our theoretical analysis shows that RIMC obtains a unique optimizer with high probability when side information is perfect and achieves ε-recovery with similar sampling complexity compared to other methods that utilize side information. Besides, when side information is useless, RIMC guarantees a lower upper bound on the sampling complexity for ε-recovery compared to state-of-the-art methods. Experiments on real data confirm the effectiveness of the proposed method.

4.1 Introduction

Matrix completion is one of the most popular collaborative filtering methods. In matrix completion, users' preferences for items are taken as elements in a rating matrix, and missing entries are recovered using the observed entries in the matrix. A low-rank constraint is usually applied over the recovered rating matrix to prevent the trivial solution, and a series of works [70, 71] employed the trace norm to approximate the low-rank constraint. Recently, factorization-based approaches for estimating rating matrix become popular [31, 72]. The partially observed matrices are decomposed into two low-rank components, i.e., the user factor and the item factor, and successfully applied to recommendation tasks [46].

To improve matrix completion method performance for recommendation tasks, in addition to conventional (user, item, preference) triplets, the side information characterizing users or items have recently been exploited. These types of side information have been incorporated into matrix completion in different ways. Chen et al. [8] designed a feature-based collaborative filtering framework in which side information such as browsing history, neighborhood relationships, and movie reviews was incorporated into factorization models by assigning discrete weights to different dimensions of user and item vectors. Natarajan and Dhillon [9] incorporated rich side features such as biological sources and various disease-related text into an inductive matrix completion framework to predict gene-disease associations. The effectiveness of side information has been demonstrated both experimentally [8, 9] and theoretically [11, 10].

In practice, side information is not always perfect due to the presence of noise or outliers. For example, users sometimes falsify their demographic data such as age or sensitive occupations. Hence, exploiting the positive effects of side information while avoiding its flaws is essential for further improving collaborative filtering. However, only a few studies have addressed the problem of noisy side information. Recently, Chiang et al. [13] proposed dirtyIMC to simultaneously balance noisy features and rating observations. Specifically, dirtyIMC decomposed rating matrices into two parts: the first described by side features and the other part a sparse matrix to capture information that side features failed to describe. Si et al. [12] adopted nonlinear transformation to rectify the influence of noisy side features, thus

preserving their benefits. However, these approaches do not identify noise in side features or eliminate their negative influence on the underlying rating matrix. Besides, selecting relevant side features for rating prediction is important yet has not been investigated by these methods. While [73] and [19] studies the noise on rows (columns), the noise that usually occurs in several user and item attributes are not investigated yet.

In this work, we address the inductive matrix completion problem when user (item) side information is noisy. We revisit the common low-rank principle in inductive matrix completion in which the low-rank matrix aligns users and items' side features with observed rating data; and we propose a method to identify noisy feature dimension in user (item) side information and suppress their negative influence on rating prediction, namely Robust Inductive Matrix Completion (RIMC). The main contributions are summarized as follows:

- We propose to use a group sparse matrix to identify noisy features in the user side information. Moreover, this matrix further plays a role to suppress the negative influence of noisy entries.
- We reformulate the proposed RIMC into an augmented Lagrangian function and adopt the alternating direction method to solve it.
- Theoretical analyses of the sampling complexity for RIMC are provided. When the side information is perfect, RIMC obtains the unique optimizer under similar sampling complexity comparing to existing works. When the side information is noisy, we provide the sampling complexity for ε-recovery. However, when the side information is useless, our model provides a lower upper bound on the sampling complexity for ε-recovery.
- Experiments conducted on three benchmark datasets demonstrate that RIMC better predicts rating values than baseline methods and can identify feature noise.

The remainder of the chapter is structured as follows. In Section 4.2, we review related works on matrix completion. Section 4.3 presents the formulation for the matrix completion problem and the proposed RIMC. The alternating direction method for optimizing the

objective is presented in Section 4.4, followed by analysis of sampling complexity for exact recovery and ε-recovery in Section 4.5. Experimental results are presented in Section 4.6, and we conclude in Section 4.8.

4.2 Related Work

Matrix completion aims to estimate the unseen values based on a small number of provided matrix entries. Many existing research works focus on a convex relaxation of rank minimization on the underlying rating matrix, i.e., minimizing the trace norm [71, 74]. Another approach fills in the incomplete matrix using factorization-based methods [31, 32, 47, 75], for which the rating matrix is factorized into two parts: the user profile and the item profile, with unseen entries, then completed by corresponding multiplication of user and item profiles. The first theoretical guarantee for matrix completion is provided in [71, 70] that, with high probability, exact recovery of the incomplete rating matrix can be achieved under certain conditions by minimizing the nuclear norm of the hidden rating matrix. Based on the strength of these works, Shamir and Shalev-Shwartz [76] studied the recovery of the distribution-free matrix completion problem with the regularized trace norm, while [77] studied the recovery of an ill-posed rating matrix whose entries were variably contaminated; the recovery error was proportional to the noise level. The sampling complexity of factorization-based methods for standard recovery is less well understood, but recent geometric analysis suggests there are no spurious local minima and all local minima are also global optima [78–80].

In addition to these matrix completion methods based solely on ratings, some recent works have exploited the effectiveness of side features, e.g., a user's gender or a movie's genre, to enhance matrix completion performance. Natarajan and Dhillon [9] regarded profile data as the user and item side information and applied it to a low-rank matrix to generate ratings. Chen et al. [8] studied a feature-based collaborative filtering framework in which side information including a user's browsing history and neighborhood relationships were utilized to construct weights applied to each dimension of user and item vectors, respectively. In multi-label learning, Xu et al. [11] required that the side features of rows (customers) and

columns (items) were both orthonormal. Recently, Lu et al. [81] studied explicit interactions between side features and incomplete observations without orthonormal constraints. On the strength of [9], a sparse constraint on the underlying low-rank matrix was employed to control the degree to which the user interacted with the item side features.

In reality, side information can be noisy. Chiang et al. [13] balanced noisy side features with observations. Specifically, they decomposed the ground truth rating matrix into two parts, the first recovered by applying user and item side feature to a low-rank matrix and the second a sparse matrix that captured the information lying outside the given side features. Si et al. [12] constructed a nonlinear mapping that efficiently embedded noisy side information, thus ensuring that the embedded side features were aligned with the task and retained the useful benefits of side information. These methods either balance the rating observations with noisy side features or directly transform noisy side features into another unknown space. However, these methods cannot locate the noisy dimensions and control their influence on the interactions between the user and item side features. [73, 19] study the recovery when rows or columns of the observations are contaminated.

In the following section, we present our method, which works effectively with both perfect and noisy side features.

4.3 Problem Formulation

Matrix completion has been widely applied in recommendation systems due to its simplicity and strong theoretical guarantees. Given customers and products in a recommendation problem, the observed interactions between customers and products are set as corresponding values in a rating matrix. Then, matrix completion can be deployed to predict the future interactions between customers and products. Denoting rating matrix $\mathbf{R} \in \mathbb{R}^{n \times m}$, a typical form of matrix completion method based on incomplete observations is as follows:

$$\min_{\mathbf{Z} \in \mathbb{R}^{n \times m}} \quad \text{rank}(\mathbf{Z}) \quad \text{s.t.} \quad P_{\Omega}(\mathbf{Z}) = P_{\Omega}(\mathbf{R}), \tag{4.1}$$

where Ω denotes the index set of observed rating values, and Z is a low-rank estimation of the rating matrix and implies that users or items can be grouped into a few distinctive sets. However, rank minimization is an NP-hard problem, and no algorithms are available to solve it in a tractable time [82]. [71] proposes a convex relaxation of the Equation (4.1) using trace norm:

$$\min_{\mathbf{Z}\in\mathbb{R}^{n\times m}} \quad \|\mathbf{Z}\|_{tr} \quad \text{s.t. } P_{\Omega}(\mathbf{Z}) = P_{\Omega}(\mathbf{R}). \tag{4.2}$$

To solve this problem, singular value thresholding [83] is a first-order algorithm that requires little storage and has low computational cost. By relaxing this problem to a least squares problem regularized by the nuclear norm by fliping the constraints to the objective function, alternating direction method and linearized augmented Lagrangian methods [84] can be adopted for their convergence guarantees.

As well as the rating matrix, side information is often available to describe users or items. For instance, in a movie recommendation system, two movies of the same genre may have the similar underlying descriptors, while users rate the movie according to their characteristics and interests. For example, a female user may have different tastes to a male user, and influence of using the gender as a user side feature will be reflected on the different ratings submitted by male and female users. Denote $\mathbf{X} \in \mathbb{R}^{n\times d_x}$ and $\mathbf{Y} \in \mathbb{R}^{m\times d_y}$ as user and item side features, where d_x and d_y are the side-feature dimensions. When the side features $\mathbf{X}$ spans the row space of $\mathbf{R}$ and $\mathbf{Y}$ spans the column space of $\mathbf{R}$, one can recover $\mathbf{R}$ by a smaller matrix such that $\mathbf{X}\mathbf{M}\mathbf{Y}^{\top} = \mathbf{R}$. This formulation is called inductive matrix completion [10, 9], and the objective function is formulated as

$$\min_{\mathbf{M}} \|\mathbf{M}\|_{tr} \quad \text{s.t. } P_{\Omega}(\mathbf{X}\mathbf{M}\mathbf{Y}^{\top}) = P_{\Omega}(\mathbf{R}), \tag{4.3}$$

where $\mathbf{M}$ is the matrix that needs to be estimated. The matrix $\mathbf{M}$ aligns user and item side features with the observed ratings.

The inductive matrix completion model in Equation (4.3) provides an effective approach to investigating user and item side information in the recommendation problem. However, these side information does not always perfectly describe users or items in practice. The

matrix $\mathbf{M}$'s performance can be easily downgraded by noise in side features $\mathbf{X}$ and $\mathbf{Y}$, but there is no systematic technique to identify noisy side features and correct for their negative influence. Observing that noise usually occurs in some particular side information attributes, we introduce a group sparse matrix restrained by the $l_{2,1}$ norm. The group sparse part can identify noisy features in either user or item side information, i.e., columns in $\mathbf{X}$ and $\mathbf{Y}$, and control their influence on prediction performance. The resulting objective function can be written as:

$$\begin{aligned} \min_{\mathbf{W},\mathbf{S},\mathbf{Z}} \quad & \|\mathbf{W}\|_{tr} + \lambda_S \|\mathbf{S}\|_{2,1} + \lambda_Z \|\mathbf{Z}\|_{tr} \\ \text{s.t.} \quad & P_\Omega(\mathbf{Z}) = P_\Omega(\mathbf{R}), \\ & \mathbf{X}(\mathbf{W}+\mathbf{S})\mathbf{Y}^\top = \mathbf{Z}, \end{aligned} \tag{4.4}$$

where $\mathbf{Z} \in \mathbb{R}^{n\times m}$. We call the proposed method Robust Inductive Matrix Completion model (RIMC for short). We introduce the bridging matrix $\mathbf{Z}$ to directly estimate the underlying rating matrix $\mathbf{R}$. $\mathbf{W}$ can be interpreted as the embedding matrix that aligns the given user and item profiles with the latent vectors that generate the ratings. $\mathbf{S}$ is restricted to be group sparse and can identify the noisy dimensions in the side information. Through the interaction with $\mathbf{S}$, the negative influence of noise in $\mathbf{X}$ and $\mathbf{Y}$ is reduced. When side information is useless, the usage of $\mathbf{Z}$ still guarantees a satisfactory performance. This is discussed in Theorem 2.

4.4 Optimization

We use the adaptive alternating direction method (ADM) to solve Equation (4.4). The augmented Lagrangian function of Equation (4.4) is

$$\begin{aligned} \mathscr{L}(\mathbf{Z},\mathbf{W},\mathbf{S},\mathbf{M_1},\mathbf{M_2},\beta) =& \|\mathbf{W}\|_{tr} + \lambda_S \|\mathbf{S}\|_{2,1} + \lambda_Z \|\mathbf{Z}\|_{tr} \\ & + \langle \mathbf{M}_1, P_\Omega(\mathbf{Z}-\mathbf{R})\rangle + \langle \mathbf{M}_2, \mathbf{Z}-\mathbf{X}(\mathbf{W}+\mathbf{S})\mathbf{Y}^\top\rangle \\ & + \frac{\beta}{2}\|P_\Omega(\mathbf{Z}-\mathbf{R})\|_F^2 + \frac{\beta}{2}\|\mathbf{Z}-\mathbf{X}(\mathbf{W}+\mathbf{S})\mathbf{Y}^\top\|_F^2, \end{aligned} \tag{4.5}$$

where $\mathbf{M}_1$ and $\mathbf{M}_2$ are Lagrange multipliers, and $\beta > 0$ is the penalty for augmented constraints. In general, we follow the alternating direction theme, and update each variable in turn:

$$\begin{aligned}
\mathbf{W}^{k+1} &= \arg\min_{\mathbf{W}} \mathscr{L}(\mathbf{Z}^k, \mathbf{W}, \mathbf{S}^k, \mathbf{M}_2^k, \beta^k), \\
\mathbf{S}^{k+1} &= \arg\min_{\mathbf{S}} \mathscr{L}(\mathbf{Z}^k, \mathbf{W}^{k+1}, \mathbf{S}, \mathbf{M}_2^k, \beta^k), \\
\mathbf{Z}^{k+1} &= \arg\min_{\mathbf{Z}} \mathscr{L}(\mathbf{Z}, \mathbf{W}^{k+1}, \mathbf{S}^{k+1}, \mathbf{M}_1^k, \mathbf{M}_2^k, \beta^k), \\
\mathbf{M}_1^{k+1} &= \mathbf{M}_1^k + \beta^k P_\Omega(\mathbf{Z}^{k+1} - \mathbf{R}), \\
\mathbf{M}_2^{k+1} &= \mathbf{M}_2^k + \beta^k(\mathbf{Z}^{k+1} - \mathbf{X}(\mathbf{W}^{\mathbf{k+1}} + \mathbf{S}^{\mathbf{k+1}})\mathbf{Y}^\top).
\end{aligned} \tag{4.6}$$

We next solve the three sub-problems and derive their closed-form solutions.

Updating W

The subproblem related to variable $\mathbf{W}$ takes the form

$$\min_{\mathbf{W}} \|\mathbf{W}\|_{tr} + \langle \mathbf{M}_2, \mathbf{Z}^k - \mathbf{X}(\mathbf{W} + \mathbf{S}^k)\mathbf{Y}^\top \rangle + \frac{\beta}{2}\|\mathbf{Z}^k - \mathbf{X}(\mathbf{W} + \mathbf{S}^k)\mathbf{Y}^\top\|_F^2. \tag{4.7}$$

By adding a constant term $\langle \mathbf{M}_2^k/\beta^k, \mathbf{M}_2^k/\beta^k \rangle$, we have

$$\min_{\mathbf{W}} \|\mathbf{W}\|_{tr} + \frac{\beta^k}{2}\|\mathbf{B}_1^k - \mathbf{X}\mathbf{W}\mathbf{Y}^\top\|_F^2, \tag{4.8}$$

where $\mathbf{B}_1^k = \mathbf{Z}^k - \mathbf{X}\mathbf{S}^k\mathbf{Y}^\top + \mathbf{M}_2^k/\beta^k$. Equation (4.8) does not have a closed-form solution. Using the linearization technique, we have that

$$\frac{1}{2}\|\mathbf{B}_1^k - \mathbf{X}\mathbf{W}\mathbf{Y}^\top\|_F^2 \approx \frac{1}{2}\|\mathbf{X}\mathbf{W}^k\mathbf{Y}^\top - \mathbf{B}_1^k\|_F^2 + \langle f_1^k, \mathbf{W} - \mathbf{W}^k \rangle + \frac{\tau_A}{2}\|\mathbf{W} - \mathbf{W}^k\|_F^2, \tag{4.9}$$

where $f_1^k = \mathbf{X}^\top(\mathbf{X}\mathbf{W}^k\mathbf{Y}^\top - \mathbf{B}_1^k)\mathbf{Y}$ and τ_A is the proximal parameter with $\tau_A > 0$. Thus the subproblem with regard to $\mathbf{W}$ is approximated by

$$\min_{\mathbf{W}} \|\mathbf{W}\|_{tr} + \frac{\beta^k\tau_A}{2}\|\mathbf{W} - [\mathbf{W}^k - f_1^k/\tau_A]\|_F^2. \tag{4.10}$$

Here we adopt the singular value thresholding method in [83] and reach the closed form solution for Equation (4.7)

$$\mathbf{W}^{k+1} = SVT(\mathbf{W}^k - f_1^k/\tau_A, 1/(\beta^k\tau_A)), \tag{4.11}$$

where $SVT(*,*)$ is the singular value thresholding operator defined in [83].

Updating S

The sub-problem related to variable $\mathbf{S}$ takes the form

$$\min_{\mathbf{S}} \lambda_{\mathbf{S}}\|\mathbf{S}\|_{2,1} + \langle \mathbf{M}_2, \mathbf{Z}^k - \mathbf{X}(\mathbf{W}^k + \mathbf{S})\mathbf{Y}^\top \rangle + \frac{\beta^k}{2}\|\mathbf{Z}^k - \mathbf{X}(\mathbf{W}^k + \mathbf{S})\mathbf{Y}^\top\|_F^2. \tag{4.12}$$

We take the first-order derivative of the above subproblem and:

$$\lambda_S \mathbf{D}\mathbf{S} + \beta^k \mathbf{X}^\top(\mathbf{X}\mathbf{S}\mathbf{Y}^\top - \mathbf{Z} + \mathbf{X}\mathbf{W}\mathbf{Y}^\top - \mathbf{M}_2^k/\beta^k)\mathbf{Y} = 0, \tag{4.13}$$

where $\mathbf{D}$ is a diagonal matrix with the i-th diagonal element as

$$d_{ii} = \frac{1}{\|\mathbf{s}^i\|_2}, \mathbf{s}^i \text{ is the } i\text{-th row of } \mathbf{S}. \tag{4.14}$$

Then,

$$\mathbf{M}_2^k = \lambda_{\mathbf{S}}(\mathbf{X}\mathbf{D}^{-1}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{S}\mathbf{Y}^\top(\mathbf{Y}\mathbf{Y}^\top)^{-1}. \tag{4.15}$$

By substituting it into Equation (4.13) and using the relationship $\mathbf{Z} = \mathbf{X}(\mathbf{W}+\mathbf{S})\mathbf{Y}^\top$, we reach the closed-form solution for $\mathbf{S}$ as:

$$\mathbf{S} = \mathbf{D}^{-1}\mathbf{X}^\top(\mathbf{X}\mathbf{D}^{-1}\mathbf{X}^\top)^{-1}(\mathbf{Z} - \mathbf{X}\mathbf{W}\mathbf{Y}^\top)(\mathbf{Y}\mathbf{Y}^\top)^{-1}\mathbf{Y}. \tag{4.16}$$

Since $\mathbf{D}$ is dependent on $\mathbf{S}$, it is also unkonwn. We adopt the iterative algorithm proposed in [85], which has been shown to be efficient and to converge to the global optimum. Details are provided in Algorithm 3.

Algorithm 3 Alternating optimization of $\mathbf{S}$

Input: side features $\mathbf{X}, \mathbf{Y}$, ε: stopping criteria, d_x: latent dimension, T: max iteration
Output: $\mathbf{S} \in \mathbb{R}^{d_x \times d_y}$

1: Set $t = 0$. Initialize $\mathbf{D}_t \in \mathbf{R}^{d_x \times d_y}$ as an identity matrix;
2: **repeat**
Calculate
$\mathbf{S}_{t+1} = \mathbf{D}^{-1}\mathbf{X}^\top(\mathbf{X}\mathbf{D}^{-1}\mathbf{X}^\top)^{-1}(\mathbf{Z} - \mathbf{X}\mathbf{W}\mathbf{Y}^\top)(\mathbf{Y}\mathbf{Y}^\top)^{-1}\mathbf{Y}$
Calculate $\mathbf{D}_{t+1}$ with i-th diagonal entries as $\frac{1}{\|\mathbf{s}^i\|_2}$
$t = t + 1$
3: **until** *converges*.

Updating Z

The subproblem related to variable $\mathbf{Z}$ takes the form

$$\begin{aligned}\min_{\mathbf{Z}}\ & \lambda_Z\|\mathbf{Z}\|_{tr} + \langle \mathbf{M}_1, P_\Omega(\mathbf{Z} - \mathbf{R})\rangle + \langle \mathbf{M}_2, \mathbf{Z} - \mathbf{X}(\mathbf{W}^k + \mathbf{S}^k)\mathbf{Y}^\top\rangle \\ & + \frac{\beta_k}{2}\|P_\Omega(\mathbf{Z} - \mathbf{R})\|_F^2 + \frac{\beta_k}{2}\|\mathbf{Z} - \mathbf{X}(\mathbf{W}^k + \mathbf{S}^k)\mathbf{Y}^\top\|_F^2,\end{aligned} \tag{4.17}$$

which can be reformulated as

$$\min_{\mathbf{Z}} \lambda_Z\|\mathbf{Z}\|_{tr} + \frac{\beta_k}{2}\|P_\Omega(\mathbf{Z} - \mathbf{B}_3^k)\|_F^2 + \frac{\beta_k}{2}\|\mathbf{Z} - \mathbf{B}_4^k\|_F^2. \tag{4.18}$$

Here $\mathbf{B}_3^k = P_\Omega(\mathbf{R} - \mathbf{M}_1^k/\beta_k)$, $\mathbf{B}_4^k = \mathbf{X}(\mathbf{W}^k + \mathbf{S}^k)\mathbf{Y}^\top - \mathbf{M}_2^k/\beta_k$.

To obtain the closed-form solution, we linearize the above sub-problem with regard to $\mathbf{Z}$, thus we have

$$\begin{aligned}&\frac{1}{2}\|P_\Omega(\mathbf{Z} - \mathbf{B}_3^k)\|_F^2 \approx \frac{1}{2}\|P_\Omega(\mathbf{Z}^k - \mathbf{B}_3^k)\|_F^2 + \langle f_3^k, \mathbf{Z} - \mathbf{Z}^k\rangle + \frac{\tau_B}{2}\|\mathbf{Z} - \mathbf{Z}^k\|_F^2, \\ &\frac{1}{2}\|\mathbf{Z} - \mathbf{B}_4^k\|_F^2 \approx \frac{1}{2}\|\mathbf{Z}^k - \mathbf{B}_4^k\|_F^2 + \langle f_4^k, \mathbf{Z} - \mathbf{Z}^k\rangle + \frac{\tau_B}{2}\|\mathbf{Z} - \mathbf{Z}^k\|_F^2.\end{aligned} \tag{4.19}$$

Here, $f_3^k = P_\Omega(\mathbf{Z}^k - \mathbf{B}_3^k), f_4^k = \mathbf{Z}^k - \mathbf{B}_4^k$ and τ_B is the proximal parameter with $\tau_B > 0$. Thus, the subproblem with regard to $\mathbf{Z}$ is approximated by

$$\min_{\mathbf{Z}} \lambda_Z\|\mathbf{Z}\|_{tr} + \frac{\beta_k\tau_B}{2}\|\mathbf{Z} - (\mathbf{Z}^k - f_3^k/\tau_B)\|_F^2 + \frac{\beta_k\tau_B}{2}\|\mathbf{Z} - (\mathbf{Z}^k - f_4^k/\tau_B)\|_F^2. \tag{4.20}$$

Adopting a similar strategy in solving the sub-problem with regard to $\mathbf{W}$ in Equation (4.7), we reach the closed-form solution for $\mathbf{Z}$ based on singular value thresholding:

$$\mathbf{Z}^{k+1} = SVT(\mathbf{Z}^k - (f_3^k + f_4^k)/(2\tau_B), \lambda_{\mathbf{Z}}/(2\beta_k\tau_B)). \tag{4.21}$$

Now we have the updating rules for the three sub-problems and two multipliers $\mathbf{M}_1, \mathbf{M}_2$; we alternately update these five variables until convergence.

4.5 Recovery Analysis

The proposed model is compatible with both perfect side information and noisy side features. If user features lie in the space spanned by columns of rating matrix, and item features are in the space spanned by rows of rating matrix, the side information is considered perfect [13], and the proposed inductive matrix completion model in Equation (4.4) can recover ground truth $\mathbf{Z}_0, \mathbf{W}_0$, and $\mathbf{S}_0$ with high probability. When the side features are not consistent with the spaces spanned by the rating matrix, e.g., there exists the influence from noise, $\mathbf{XWY}^\top$ cannot guarantee to perfectly recover the partially observed $\mathbf{R}$. However, an additional sparse matrix $\mathbf{S}$ can be employed to compensate for the influence of noisy side features. The expected l-risk of our recovery model can be bounded by the level of noise in $O(s^{1/2})$, with s denoting the noise level.

4.5.1 Sampling complexity for exact recovery

In this section, we discuss exact recovery with perfect side information. Before presenting the theorem, we provide some definitions used in the analysis. Denoting $col(\mathbf{A})$ and $row(\mathbf{A})$ as the space spanned by the column vectors and row vectors of $\mathbf{A}$ respectively, the perfect side features are those satisfying the following rules:

$$col(\mathbf{R}) \subseteq col(\mathbf{X}),\ row(\mathbf{R}) \subseteq col(\mathbf{Y}).$$

Consider the underlying rating matrix $\mathbf{R}$ as $\mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ and $\mathbf{X} = \mathbf{U}_X\mathbf{\Sigma}_X\mathbf{V}_X, \mathbf{Y} = \mathbf{U}_Y\mathbf{\Sigma}_Y\mathbf{V}_Y$. The orthogonal projections onto $\mathbf{U}$ and $\mathbf{V}$ are defined as:

$$P_{\mathbf{U}} = \mathbf{U}\mathbf{U}^\top \in \mathbb{R}^{n\times n}; P_{\mathbf{V}} = \mathbf{V}\mathbf{V}^\top \in \mathbb{R}^{m\times m}. \tag{4.22}$$

As suggested by [71], if a matrix is equal to zero in almost all of its entries, it would be impossible to recover it unless we know all of its entries. Thus, the coherence measurements for $\mathbf{R}$, $\mathbf{X}$, and $\mathbf{Y}$ are required. A higher coherence of $\mathbf{R}$ means that its singular vectors are closer to the standard basis vectors and the required sampling complexity increases. Given standard basis vectors $\mathbf{e}_i$, the coherence measurements for $\mathbf{U}, \mathbf{V}$ are defined as follows:

$$\begin{aligned} &\mu_{\mathbf{U}} \equiv \frac{n}{r}\max_{1\le i\le n}\|P_{\mathbf{U}}\mathbf{e}_i\|^2, \mu_{\mathbf{V}} \equiv \frac{m}{r}\max_{1\le j\le n}\|P_{\mathbf{V}}\mathbf{e}_j\|^2, \\ &\mu_{\mathbf{U}\mathbf{V}^\top} \equiv \max_{i,j}\frac{mn}{r}([\mathbf{U}\mathbf{V}^\top]_{ij})^2. \end{aligned} \tag{4.23}$$

The coherence measurement for side features $\mathbf{X}$ and $\mathbf{Y}$ is defined as:

$$\mu_{\mathbf{X}} = \max_{1\le i\le n}\frac{m\|\mathbf{x}_i\|^2}{d_x}, \mu_{\mathbf{Y}} = \max_{1\le j\le n}\frac{n\|\mathbf{y}_j\|^2}{d_y}, \tag{4.24}$$

where $\mathbf{x}_i$ and $\mathbf{y}_j$ stand for the i-th row of $\mathbf{X}$ and the j-th row of $\mathbf{Y}$, respectively.

We denote $\mu = \max\{\mu_{\mathbf{U}}, \mu_{\mathbf{V}}, \mu_{\mathbf{X}}, \mu_{\mathbf{Y}}\}$, and $\sigma = \max\{\|\Sigma_{\mathbf{X}}^{-1}\|_{tr}, \|\Sigma_{\mathbf{Y}}^{-1}\|_{tr}\}$. Given a certain guarantee of the sampling complexity in the ground truth matrix and perfect side features, we present our analytical result over the recovery problem in Equation (4.4) in Theorem 1.

Theorem 1. *Let* $N = \max(m,n), q_0 = \frac{1}{2}(1+\log d - \log r)$, $T_0 = \frac{128}{3}p\sigma\mu\max(\mu_{\mathbf{U}\mathbf{V}^\top}, \mu)r(d_x + d_y)\log N$ *and* $T_1 = \frac{8}{3}p\sigma^2\mu^2(d_xd_y + r^2)\log N$*, where* p *is a constant. Assume* $T_1 \ge q_0T_0$*,* $\mathbf{X}$ *and* $\mathbf{Y}$ *are both full column rank. For any* $p \ge 1$*, with a probability at least* $1 - 4(q_0+1)N^{-p+1} - 2q_0N^{-p+2}$*, Problem* (4.4) *has the unique optimizer* $(\mathbf{W}_0, \mathbf{S}_0, \mathbf{Z}_0)$ *satisfying* $\mathbf{S}_0 = 0, \mathbf{Z}_0 = \mathbf{X}\mathbf{W}_0\mathbf{Y}^\top$*, with necessary sampling rate as low as* $O(r\log N)$*. More precisely, the sampling size* $|\Omega|$ *should satisfy that* $\Omega \ge \frac{64}{3}p\sigma\mu\max(\mu_{\mathbf{U}\mathbf{V}^\top}, \mu)(1+\log d - \log r)r(d_x+d_y)\log N$.

Proof sketch. To prove that $(\mathbf{W}_0, \mathbf{S}_0, \mathbf{Z}_0)$ is the unique optimizer to Equation (4.4), we turn to prove that other solutions indicated by $(\mathbf{W}_0 + \Delta\mathbf{W}, \mathbf{S}_0 + \Delta\mathbf{S}, \mathbf{Z}_0 + \Delta\mathbf{Z})$ lead to worse

objective values. This equals to prove that the following inequality does not hold under any circumstance:

$$\begin{aligned} &\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr}+\lambda_S\|\mathbf{S}_0+\Delta\mathbf{S}\|_{2,1}+\lambda_Z\|\mathbf{Z}_0+\Delta\mathbf{Z}\|_{tr} \\ \geq &\|\mathbf{W}_0\|_{tr}+\lambda_Z\|\mathbf{Z}_0\|_{tr}+\lambda_S\|\mathbf{S}_0\|_{2,1}. \end{aligned} \tag{4.25}$$

We first contract the term $\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr}+\lambda_S\|\mathbf{S}_0+\Delta\mathbf{S}\|_{2,1}+\lambda_Z\|\mathbf{Z}_0+\Delta\mathbf{Z}\|_{tr}$ based on inequalities of matrix norms and the property of optimizer solution that $P_\Omega(\mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top)=0$. This results in an equivalent form to Equation (4.25):

$$\lambda_Z\langle\mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top,\mathbf{U}\mathbf{V}^\top+\mathbf{U}_\perp\mathbf{V}_\perp^\top-\mathbf{H}\rangle+\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr}\geq\|\mathbf{W}_0\|_{tr}.$$

Then we show that Equation (4.26) does not hold when $\mathbf{W_0}+\Delta\mathbf{W}$ lies either inside or outside the neighborhood of $\mathbf{W}$. Details of the proof are presented in Section 4.7.

From Theorem 1, we can see that the sampling complexity required for exact recovery is $O(rlogN)$, which is smaller than the $O(rNlog^2(N))$ required by generic matrix completion without side information in [70, 71]. When μ is small, the singular vectors of $\mathbf{R}$ are sufficiently spread, and it is easy to recover the ground truth $\mathbf{R}$. Conversely, when μ is large, the singular vectors $\mathbf{U}$ and $\mathbf{V}$ are more correlated with the identity matrix, making it hard to recover. This is intuitive; for example, if $\mathbf{U}$ and $\mathbf{V}$ are both identity matrices, then the rating matrix only has non-zero values in its first $\min(d_x,d_y)$ diagonal elements. If the sampling does not cover most of these entries, it is impossible to correctly recover the true rating matrix.

When side information is presented, Xu et al. [11] proved a similar bound, but it requires orthogonality constraint on both $\mathbf{X}$ and $\mathbf{Y}$. Chiang et al. [13] proposed a more advanced model, but the exact recovery is not discussed. Lu et al. [81]'s work reaches a similar sampling rate compared to us, but we are motivated differently. In the following, we will show that under the worst case, our method provides a lower upper bound on the sampling complexity for a satisfactory recovery.

4.5.2 Sampling complexity for ε-recovery

We next consider the optimization problem where the perfect feature matrices $\mathbf{X}$ and $\mathbf{Y}$ are corrupted by $\Delta\mathbf{X}(\|\Delta\mathbf{X}\|_F \leq s_1)$ and $\Delta\mathbf{Y}(\|\Delta\mathbf{Y}\|_F \leq s_2)$, respectively. We investigate the following optimization problem:

$$\begin{aligned} \min_{\mathbf{W},\mathbf{S}} \quad & \|P_\Omega((\mathbf{X}+\Delta\mathbf{X})(\mathbf{W}+\mathbf{S})(\mathbf{Y}+\Delta\mathbf{Y})-\mathbf{R})\|_F^2 \\ \text{s.t.} \quad & \mathbf{Z}-\mathbf{X}\mathbf{W}\mathbf{Y}^\top \in B(0,\phi), \|\mathbf{Z}\|_{tr} \leq \eta \\ & \|\mathbf{W}\|_{tr} \leq \alpha, \|\mathbf{S}\|_{2,1} \leq \beta. \end{aligned} \tag{4.26}$$

Here, $B(0,\phi) \subset \mathbb{R}^{m\times n}$ is a ball with radius ϕ and center at 0, which indicate that the Frobenius norm of $\mathbf{Z}-\mathbf{X}\mathbf{W}\mathbf{Y}^\top$ is restricted to be smaller than ϕ. The matrix $\mathbf{R}_{ij}$ is assumed to be sampled i.i.d. from an index set $\{(i_\alpha, j_\alpha)\}_{\alpha=1}^s$ under an unknown distribution.

Consider the variable set $\Theta = \{(\mathbf{Z},\mathbf{W},\mathbf{S}) \mid \|\mathbf{Z}\|_{tr} \leq \eta, \|\mathbf{W}\|_{tr} \leq \alpha, \|\mathbf{S}\|_{2,1} \leq \beta, \mathbf{Z} = \mathbf{X}\mathbf{W}\mathbf{Y}^\top\}$ as the feasible solution set and $\theta = (\mathbf{Z},\mathbf{W},\mathbf{S}) \in \Theta$ as any feasible solution. Let $F_\theta(i,j) = \mathbf{x}_i(\mathbf{W}+\mathbf{S})\mathbf{y}_i^\top$ be the estimation function for $\mathbf{R}_{ij}$ with θ as the parameters, and $F_\Theta = \{f_\theta \mid \theta \in \Theta\}$ be the set of feasible functions. The loss function l is defined as $l(f_\theta(i,j),\mathbf{R}_{ij}) = P_\Omega(\mathbf{X}(\mathbf{W}+\mathbf{S})\mathbf{Y}^\top - \mathbf{R})_{ij}^2$. Then, we introduce two risks: the expected l-risk

$$\mathscr{R}_l(f) = \mathbb{E}_{(i,j)}[l(f_\theta(i,j),\mathbf{R}_{ij})] \tag{4.27}$$

and the empirical l-risk

$$\widehat{\mathscr{R}}_l(f) = \frac{1}{s}\sum_{t=1}^{s}[l(f_\theta(i_t,j_t),\mathbf{R}_{ij})]. \tag{4.28}$$

Our model looks for θ that parameterizes $f^* = \arg\min_{f\in F_\Theta} \widehat{\mathscr{R}}_l(f)$.

Given $\Phi = \mathbf{Z}-\mathbf{X}\mathbf{W}\mathbf{Y}^\top$ and $k_\phi = rank(\Phi)$, we have the following theorem.

Theorem 2. *Let $\|\mathbf{Z}\|_{tr} \leq \eta, \|\mathbf{W}\|_{tr} \leq \alpha, \|\mathbf{S}\|_{2,1} \leq \beta, \|\Phi\|_F \leq \phi$, and the perfect side feature matrices (spanning latent space of $\mathbf{R}$) are corrupted with $\triangle\mathbf{X}$ and $\triangle\mathbf{Y}$ where $\|\triangle\mathbf{X}\|_F \leq s_1$ and $\|\triangle\mathbf{Y}\|_F \leq s_2$ and $\kappa = \max(s_1,s_2)$. Considering the ε-recovery $\mathbf{R}$ that the expected loss $\mathbb{E}[l(f,\mathbf{R})] \leq \varepsilon$ for a given arbitrary small $\varepsilon \geq 0$, $O(\min\{(\eta+\sqrt{k_\phi}\phi^2)\log N, (\alpha+$*

$\sqrt{d}\beta)\kappa^2\sqrt{N}\}/\varepsilon^2)$ *observations are sufficient for our model when corrupted factors of side information are bounded.*

Proof sketch. First, we apply the following lemma to bound the expected l-risk.

Lemma 1. *(Bound on expected l-risk [86]) Let l be a loss function with Lipschitz constant L_l bounded by B with respect to its first argument, and δ be a constant where $0 < \delta < 1$. Let $\mathfrak{R}(F_\Theta)$ be the Rademacher complexity of the function class F_Θ (w.r.t. Ω and associated with l) defined as:*

$$\mathfrak{R}(F_\Theta) = \mathbb{E}_\sigma[\sup_{f \in F_\Theta} \frac{1}{s} \sum_{t=1}^{s} \sigma_t l(f(i_t, j_t), F_{i_t, j_t})], \tag{4.29}$$

where each σ_t follows Rademacher distribution and takes values $\{\pm 1\}$ with equal probability. Then, with probability at least $1-\delta$, for all $f \in F_\Theta$ we have:

$$\mathscr{R}_l(f) \leq \widehat{\mathscr{R}}_l(f) + 2\mathbb{E}_\Omega[\mathfrak{R}(F_\Theta)] + B\sqrt{\frac{\log \frac{1}{\delta}}{2s}}. \tag{4.30}$$

To upper bound the expected l-risk $\mathscr{R}_l(f)$, both empirical risk $\widehat{\mathscr{R}}_l(f)$ and model complexity demonstrated by Rademacher complexity $\mathbb{E}_\Omega[\mathfrak{R}(F_\Theta)]$ need to be upper-bounded. Through the careful selection of hyperparameters, $\widehat{\mathscr{R}}_l(f^*)$ is close to zero. By upper bounding $\mathfrak{R}(F_\Theta)$, we obtain the upper bound of $\mathscr{R}_l(f)$. By controlling the expected $l-$risk by ε, we obtain the bound provided in Theorem 2. The detailed analysis of how to bound $\mathfrak{R}(F_\Theta)$ is presented in Section 4.7.

Theorem 2 suggests that $O(\log N)$ observations are sufficient for ε-recovery, which is smaller than the $O(N \log N)$ required for both perfect [71] and ε recovery [87] in previous matrix recovery analyses without side features. Hence, although side information might not be perfect in practice, it provides some useful information for model learning. Our model's expected risk is affected by nuclear norms of $\mathbf{Z}$ and $\mathbf{W}$, and the $l_{2,1}$ norm of $\mathbf{S}$. If the rating matrix has a higher rank, more observations are required for an accurate recovery. When the side information is useless, the matrix $\mathbf{Z}$ plays an important role in recovering the underlying matrix $\mathbf{R}$. In this case, the sampling complexity for ε-recovery would be given by $O(\alpha\sqrt{N})$

and is upper bounded by $O(\min(d_x, d_y)\sqrt{N})$. While the upper bound provided by [81] is $O(N^{3/2})$, our result is better since $d_x \leq N$ always holds.

4.6 Experiments

In this section, we demonstrate the efficacy of our method on real-world data sets. We use two different datasets: MovieLens 100K for user preference predictions and NCI-DREAM challenge for drug sensitivity estimation. Let p denote the percentage of observed entries used for training, and the remaining $1-p$ observed entries are evenly split into validation and testing sets. The parameters λ_S, λ_Z, and β are tuned between 10^{-3} and 10^4 on the validation set. We use the relative mean square error (RMSE) metric, defined as:

$$\text{RMSE} = \frac{\|P_\Omega(\mathbf{R}^* - \mathbf{R})\|_2^2}{\|P_\Omega(\mathbf{R})\|_2^2}. \tag{4.31}$$

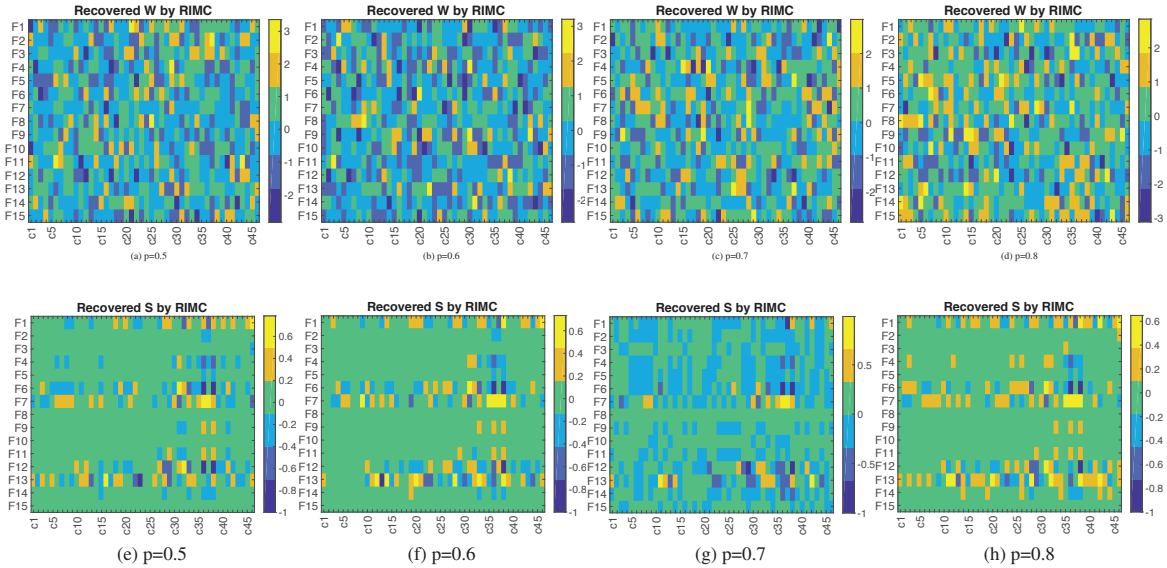
where $\mathbf{R}^*$ is the optimal estimation of the ground truth rating matrix $\mathbf{R}$ and P_Ω is the sampling operator.

4.6.1 Baseline methods

- **IMC** is a basic inductive matrix completion method [10] which assumes that the rating matrix is attained by multiplying user and item feature vectors with a low rank matrix. IMC further decomposes the underlying low rank matrix into two low-dimensional matrices to simplify optimization.

- **Maxide** is another representative inductive matrix completion method [11]. It has the same assumption to that in IMC but regularizes the unknown low-rank matrix with the nuclear norm. Hence, the singular value thresholding-based algorithm is deployed to optimize the objective.

- **dirtyIMC** is an inductive matrix completion method considering dirty side information [13]. This approach directly balances rating observations and dirty side features by



(e) p=0.5 (f) p=0.6 (g) p=0.7 (h) p=0.8

Fig. 4.1 Recovered W and S under different sampling ratio p of NCI Challenge dataset

using a low rank matrix to capture the information that is failed to be described by side features. Our method is different from it since we utilize a group sparse matrix to eliminate the influence of noisy side features.

- **SIIMC** [81] adopts explicit modeling of interactions between the user and item features by introducing the sparse formulation on the unknown hidden matrix. A bridging matrix is used to secure recovery performance to combat the existence of Gaussian noise

4.6.2 Evaluation on real data

We test our method on different collaborative filtering applications. NCI-DREAM data is used to predict breast cancer cell line sensitivity to different drugs, and the MovieLens datasets are about recommending movies to potential users.

NCI-DREAM challenge

This dataset contains expression data for 18,633 genes in 46 breast cancer cell lines and their sensitivity to 26 drugs [88]. Following the setting in [81], we used 14 features to

Table 4.1 Comparisons of relative MSE of different methods under different splits of NCI Challenge dataset

Method	50%	60%	70%	80%
SIIMC	0.190	0.145	0.139	0.181
IMC	0.637	0.557	0.489	0.437
dirtyIMC	0.632	0.551	0.475	0.432
Maxide	0.288	0.255	0.240	0.268
RIMC	**0.054**	**0.040**	**0.042**	**0.039**

Table 4.2 Comparisons of relative MSE of different methods under different splits of the MovieLens 100K dataset

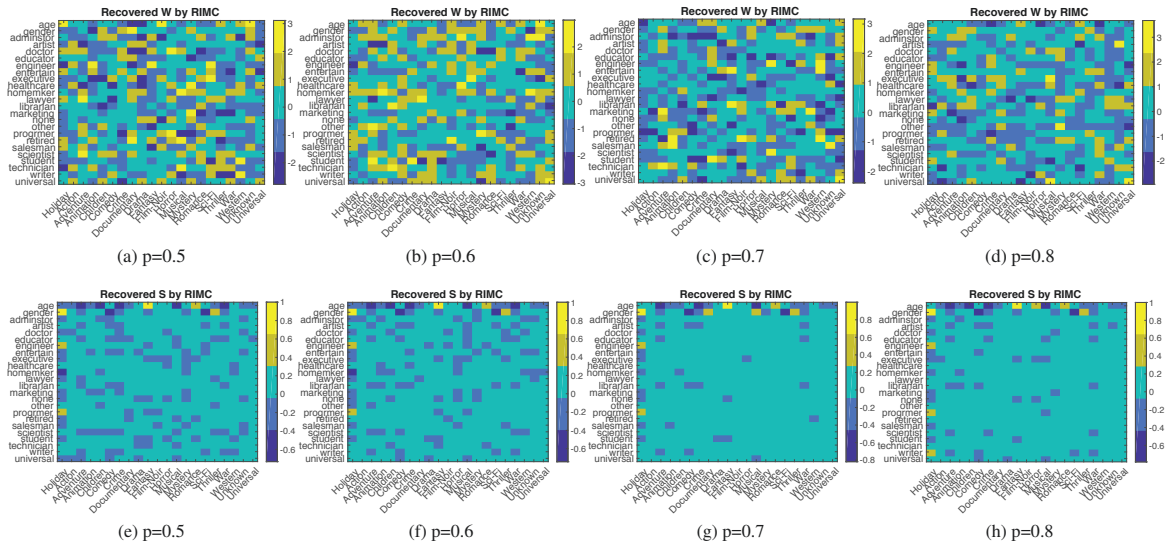
Method	50%	60%	70%	80%
SIIMC	0.292	0.282	0.279	0.276
IMC	0.959	0.945	0.943	0.935
dirtyIMC	0.814	0.775	0.738	0.705
Maxide	0.421	0.419	0.425	0.424
RIMC	**0.105**	**0.094**	**0.095**	**0.092**

describe properties of drugs, such as hydrogen bond donor count and molecular weight ((http://pubchem.ncbi.nlm.nih.gov/). For breast cancer cell lines, principle component analysis was used to extract the most important 45 principal components. We test our method with different training set proportions with $p = \{50\%, 60\%, 70\%, 80\%\}$. The RMSE results obtained by the baselines [13, 11, 10] are quoted from [81].

Comparisons with four baseline methods are shown in Table 4.1. RIMC achieves consistently better performance than the baseline methods. We show the recovered $\mathbf{W}$ and $\mathbf{S}$ in Figure 4.1. From Figure 4.1, it appears that features F1, F6, F7, F12, F13, and F14 are noisy or not informative for final predictions. This observation well aligns with current biological knowledge, as these features are known to be less important drug descriptors. Detailed information about these drug features can be found in [81].

MovieLens 100K

MovieLens 100K consists of 100,000 ratings with 943 users and 1682 movies. Each user has at least 20 ratings on different movies. Users are associated with 23 features such as "age", "gender", "occupation" etc., and 19 features discriminate different movie genres. For



(a) p=0.5 (b) p=0.6 (c) p=0.7 (d) p=0.8

(e) p=0.5 (f) p=0.6 (g) p=0.7 (h) p=0.8

Fig. 4.2 Recovered W and S under different sampling ratio p of MovieLens 100K dataset

fair comparisons, we take the movie release date into consideration following the setting in [81] and add one more universal dimension to the user and item feature vectors. The feature vector lengths for users and items are thus 24 and 21, respectively. We verify our method with different percentages of training data $p = \{50\%, 60\%, 70\%, 80\%\}$, and report the RMSEs in Table 4.2. We quote the RMSE results obtained by the baselines [13, 11, 10] from [81].

The recovered latent matrices **W** and **S** are shown in Figure 4.2. The RMSE decreases with more training data on all methods, but RIMC achieves consistently better performance when predicting unobserved entries under different sampling ratios. In Figure 4.2, (a)-(d) correspond to the recovered **W** and (e)-(h) represent the estimated sparse matrix **S**. The rows of **W** and **S** are related to user features, and the columns are related to item features. Under the different sampling ratios in (e)-(h), we observe that the heating entries in **S** barely change, meaning that the noisy features identified by RIMC remain unchanged with different sampling ratios. In (e)-(h) the heating rows in **S** are "age", "gender", and the heating column is release date. This indicates that these dimensions are noisy and not well aligned with the ground truth side features. By interacting with **S**, their negative influence is decreased, leading to better predicting performance with RIMC.

4.7 Proof Details

In this section, we present the proof details of theoretical assertions presented in previous sections. For Theorem 1, we prove that other solutions will lead to worse objective function values comparing to $(\mathbf{W}_0, \mathbf{S}_0, \mathbf{Z}_0)$. For Theorem 2, we derive the error bound of the expected recovery error.

4.7.1 Proof of Theorem 1

To prove that $(\mathbf{W}_0, \mathbf{S}_0, \mathbf{Z}_0)$ is the unique optimizer to Equation (4.4) under certain conditions, we consider the contraction that

$$\begin{aligned} &\|\mathbf{W}_0 + \Delta\mathbf{W}\|_{tr} + \lambda_S\|\mathbf{S}_0 + \Delta\mathbf{S}\|_{2,1} + \lambda_Z\|\mathbf{Z}_0 + \Delta\mathbf{Z}\|_{tr} \\ \geq &\|\mathbf{W}_0\|_{tr} + \lambda_S\|\mathbf{S}_0\|_{2,1} + \lambda_Z\|\mathbf{Z}_0\|_{tr} \end{aligned} \tag{4.32}$$

The assumption that we hold for the proof is the bound of trace norm on $\mathbf{W}$.

Here we define four linear operators that will be used later, $P_{\mathbf{X}} \in \mathbb{R}^{n\times n}, P_{\mathbf{Y}} \in \mathbb{R}^{m\times m}; P_T, P_{T^\perp}(\mathbf{A}) : \mathbb{R}^{n\times m} \mapsto \mathbb{R}^{n\times m}$:

$$\begin{aligned} P_{\mathbf{X}} &= \mathbf{U}_{\mathbf{X}}\mathbf{U}_{\mathbf{X}}^\top = \mathbf{X}\mathbf{V}_{\mathbf{X}}\Sigma_{\mathbf{X}}^{-2}\mathbf{V}_{\mathbf{X}}^\top\mathbf{X} \\ P_{\mathbf{Y}} &= \mathbf{U}_{\mathbf{Y}}\mathbf{U}_{\mathbf{Y}}^\top = \mathbf{Y}\mathbf{V}_{\mathbf{Y}}\sigma_{\mathbf{Y}}^{-2}\mathbf{V}_{\mathbf{Y}}^\top\mathbf{Y} \\ P_T(\mathbf{A}) &= P_{\mathbf{U}}\mathbf{A}P_{\mathbf{Y}} + P_{\mathbf{X}}\mathbf{A}P_{\mathbf{V}} - P_{\mathbf{U}}\mathbf{A}P_{\mathbf{V}} \\ P_{T^\perp}(\mathbf{A}) &= (P_{\mathbf{X}} - P_{\mathbf{U}})\mathbf{A}(P_{\mathbf{Y}} - P_{\mathbf{V}}) = P_{\mathbf{X}^\perp}\mathbf{A}P_{\mathbf{Y}^\perp} \end{aligned} \tag{4.33}$$

To prove Theorem 1, we introduce Lemmas 2 and 3 followed by an assumption on $\|\mathbf{W_0}\|_{tr}$.

Lemma 2. *[81] For any $\mathbf{M} \neq 0, \mathbf{M} \in \mathbb{R}^{m\times n}$ satisfying $P_\Omega(\mathbf{M}) = 0$ and $\mathbf{M} = P_X\mathbf{M}P_Y$, with a probability at least $1 - 4N^{-p+1}$ we have*

$$\|P_T\|_F \leq \zeta\|P_{T^\perp}(\mathbf{M})\|_F \tag{4.34}$$

where $\zeta \leq \sqrt{\frac{d}{2r}}$, if $T_0 \leq |\Omega| \leq T_1$ and $p > 1$.

Lemma 3. *[81] Assume that there exists a matrix* $\mathbf{H} \in \mathbb{R}^{n\times m}$ *such that* $P_\Omega(\mathbf{H}) = \mathbf{H}$*, then with a probability of* $1-2q(N+1)N^{-p+1}$*, it is satisfied that*

$$\|P_T(\mathbf{H}) - \mathbf{U}\mathbf{V}^\top\|_F \leq \sqrt{\frac{r}{2d}}, \|P_{T^\perp}(\mathbf{H})\|_{sp} < \frac{1}{2}, \tag{4.35}$$

if $q \geq q_0$ and $p > 1$.

Assumption 1.

$$\|\mathbf{W}_0\|_{tr} = \alpha < C\lambda_Z(\frac{1}{2} - \zeta\sqrt{\frac{r}{2d}}), \tag{4.36}$$

where r is the rank of underlying rating matrix $\mathbf{R}$*, and d is the dimension of side features.* ζ *is a parameter with* $\zeta \leq \sqrt{\frac{d}{2r}}$.

Next we start the proof of Theorem 1.

Proof of Theorem 1. If we assume that the solution of Equation (4.4) is not unique, there might exist another solution $\mathbf{W}_0 + \Delta\mathbf{W}, \Delta\mathbf{S}, \mathbf{Z}_0 + \Delta\mathbf{Z}$ with $\Delta\mathbf{W}, \Delta\mathbf{S}, \Delta\mathbf{Z} \neq 0$.

If $(\mathbf{W}_0, \mathbf{S_0}, \mathbf{Z_0})$ is the unique optimizer of Equation (4.4) with a high probability, then the contraction in Equation (4.32) will hold as well. We have the following inequalities:

$$\begin{aligned}
&\|\mathbf{W}_0 + \Delta\mathbf{W}\|_{tr} + \lambda_S\|\mathbf{S}_0 + \Delta\mathbf{S}\|_{2,1} + \lambda_Z\|\mathbf{Z}_0 + \Delta\mathbf{Z}\|_{tr} \\
=&\|\mathbf{W}_0 + \Delta\mathbf{W}\|_{tr} + \lambda_S\|\Delta\mathbf{S}\|_{2,1} \\
&+ \lambda_Z\|\mathbf{X}(\mathbf{W}_0 + \Delta\mathbf{W} + \Delta\mathbf{S})\mathbf{Y}^\top\|_{tr}\|\mathbf{U}\mathbf{V}^\top + \mathbf{U}_\perp\mathbf{V}_\perp^\top\| && (4.37) \\
\geq&\lambda_Z\langle\mathbf{X}\mathbf{W}_0\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top\rangle + \langle\mathbf{X}\mathbf{W}_0\mathbf{Y}^\top, \mathbf{U}_\perp\mathbf{V}_\perp^\top\rangle \\
&+ \langle\mathbf{X}(\Delta\mathbf{W} + \Delta\mathbf{S})\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top + \mathbf{U}_\perp\mathbf{V}_\perp^\top\rangle \\
&+ \|\mathbf{W}_0 + \Delta\mathbf{W}\|_{tr} + \lambda_S\|\Delta\mathbf{S}\|_{2,1} && (4.38) \\
\geq&\lambda_Z\|\mathbf{Z}_0\|_{tr} + \langle\mathbf{X}(\Delta\mathbf{W} + \Delta\mathbf{S})\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top + \mathbf{U}_\perp\mathbf{V}_\perp^\top - \mathbf{H}\rangle \\
&+ \|\mathbf{W}_0 + \Delta\mathbf{W}\|_{tr} && (4.39)
\end{aligned}$$

Here (4.38) holds because of $\|\mathbf{A}\|_{tr}\|\mathbf{B}\|_{sp} \geq \langle\mathbf{A}, \mathbf{B}\rangle$; (4.39) comes from $col(\mathbf{R}) \subseteq col(\mathbf{X}), row(\mathbf{Y}) \subseteq col(\mathbf{R}), \mathbf{R} = \mathbf{U}\Sigma\mathbf{V}^\top$; and $P_\Omega(\mathbf{X}(\Delta\mathbf{W} + \Delta\mathbf{S})\mathbf{Y}^\top) = 0$. Next for Equation (4.39), we need to

prove that

$$\lambda_Z\langle \mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top+\mathbf{U}_\perp\mathbf{V}_\perp^\top-\mathbf{H}\rangle + \|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr} \geq \|\mathbf{W}_0\|_{tr}.$$

Thus the proof of contraction in Equation (4.32) is now equivalent to proving either one of the follow conditions hold, where $\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr} \geq 0$ always holds:

Condition 1.

$$\langle \mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top+\mathbf{U}_\perp\mathbf{V}_\perp^\top-\mathbf{H}\rangle \geq 0 \tag{4.40a}$$

$$\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr} \geq \|\mathbf{W}_0\|_{tr}, \tag{4.40b}$$

Condition 2.

$$\lambda_Z\langle \mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top+\mathbf{U}_\perp\mathbf{V}_\perp^\top-\mathbf{H}\rangle \geq \|\mathbf{W}_0\|_{tr} = \alpha. \tag{4.41}$$

In the following, we discuss when these two conditions hold. Our strategy is to prove that when $\mathbf{W} = \mathbf{W}_0+\Delta\mathbf{W}$ lies within the neighbor of $\mathbf{W}_0$, i.e., ε-ball $B_\varepsilon(\mathbf{W}_0)$, **Condition 1** hold, and when it is isolated from $\mathbf{W}_0$, **Conditions 2** holds.

Condition 1 : minimizer in the neighborhood

For the condition in Equation (4.40b), we consider its contrary case. Let us assume $\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr} \leq \|\mathbf{W}_0\|_{tr}$. Firstly, if the possible minimizer $\mathbf{W}$ exists in the small ε-ball $B_\varepsilon(\mathbf{W})$ as a continuous neighbour of $\mathbf{W}_0$, such that $\varepsilon \leq \min_{i,j}\mathbf{W}_{ij}$, then for each $\mathbf{W}_t = \{\mathbf{W}_0+\Delta\mathbf{W} \in B_\varepsilon(\mathbf{W}_0)\}$, it satisfies that

$$\begin{aligned}
\|\mathbf{W}_t\|_{tr} &\geq \|\mathbf{W}_0\|_{tr} + d_x d_y \varepsilon^2 - 2\|\mathbf{W}_0\|_1 \varepsilon \\
&\geq \|\mathbf{W}_0\|_{tr} + d_x d_y \varepsilon^2 - 2\|\mathbf{W}_0\|_{tr}\sqrt{d}\varepsilon \\
&\geq \|\mathbf{W}_0\|_{tr} + d_x d_y \varepsilon^2 - 2\sqrt{d}\alpha\varepsilon
\end{aligned} \tag{4.42}$$

Since ε is arbitrary, $\|\mathbf{W}_0+\Delta\mathbf{W}\|_{tr} \geq \alpha \geq \|\mathbf{W}_0\|_{tr}$, which leads to the contradiction. Thus Condition 1 in Equation (4.40b) is satisfied.

For Condition 1 in Equation (4.40a), we prove that it holds under three cases: (i) $\Delta\mathbf{W} \neq 0, \Delta\mathbf{S} = 0$, (ii) $\Delta\mathbf{W} = 0, \Delta\mathbf{S} \neq 0$, (iii) $\Delta\mathbf{W} \neq 0, \Delta\mathbf{S} \neq 0$

Case (i): $\Delta\mathbf{W} \neq 0, \Delta\mathbf{S} = 0$. Assume $\mathbf{U}_\perp, \mathbf{V}_\perp$ are singular vectors of $P_{T^\perp}(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top)$, and $[\mathbf{U}, \mathbf{U}_\perp]$ and $[\mathbf{V}, \mathbf{V}_\perp]$ are unitary matrices,

$$\begin{aligned}
&\langle \mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top + \mathbf{U}_\perp\mathbf{V}_\perp^\top - \mathbf{H}\rangle \\
=&\langle P_T(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top), \mathbf{U}\mathbf{V}^\top - P_T(\mathbf{H})\rangle + \langle P_{T^\perp}\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top), \mathbf{U}_\perp\mathbf{V}_\perp^\top - P_{T^\perp}(\mathbf{H})\rangle \\
\geq&\|P_{T^\perp}(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top)\|_{tr} - \|P_T(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top)\|_F\|\mathbf{U}\mathbf{V}^\top - P_T(\mathbf{H})\|_F \\
&- \|P_{T^\perp}(\mathbf{H})\|_F\|P_{T^\perp}(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top)\|_{tr} \\
\geq&\|P_{T^\perp}\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top\|_{tr}(\frac{1}{2} - \zeta\sqrt{\frac{r}{2d}}) \geq 0.
\end{aligned} \tag{4.43}$$

The equality holds because of $\langle P_T(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top), \mathbf{U}_\perp\mathbf{V}_\perp^\top\rangle = 0$, and $\langle P_{T^\perp}(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top), \mathbf{U}\mathbf{V}^\top\rangle = 0$. The first inequality results from $\langle \mathbf{A}, \mathbf{B}\rangle \geq -\|\mathbf{A}\|_F\|\mathbf{B}\|_F$ and $\langle \mathbf{A}, \mathbf{B}\rangle \leq \|\mathbf{A}\|_{tr}\|\mathbf{B}\|_{sp}$. The last inequality comes from Lemmas 1 and 2, and $\|\mathbf{A}\|_F \leq \|\mathbf{A}\|_{tr}$.

Case (ii): $\Delta\mathbf{W} = 0, \Delta\mathbf{S} \neq 0$. Assume $\mathbf{U}_\perp, \mathbf{V}_\perp$ are singular vectors of $P_{T^\perp}(\mathbf{X}\Delta\mathbf{S}\mathbf{Y}^\top)$, and $[\mathbf{U}, \mathbf{U}_\perp], [\mathbf{V}, \mathbf{V}_\perp]$ are unitary matrices. By using the same contraction rules to case (i), we have that

$$\begin{aligned}
&\langle \mathbf{X}\Delta\mathbf{S}\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top + \mathbf{U}_\perp\mathbf{V}_\perp^\top - \mathbf{H}\rangle \\
\geq&\|P_{T^\perp}(\mathbf{X}\Delta\mathbf{S}\mathbf{Y}^\top)\|_{tr} - \|P_T(\mathbf{X}\Delta\mathbf{S}\mathbf{Y}^\top)\|_F\|\mathbf{U}\mathbf{V}^\top - P_T(\mathbf{H})\|_F \\
&- \|P_{T^\perp}(\mathbf{H})\|_F\|P_{T^\perp}(\mathbf{X}\Delta\mathbf{S}\mathbf{Y}^\top)\|_{tr} \\
\geq&\|P_{T^\perp}\mathbf{X}\Delta\mathbf{S}\mathbf{Y}^\top\|_{tr}(\frac{1}{2} - \zeta\sqrt{\frac{r}{2d}}) \geq 0.
\end{aligned} \tag{4.44}$$

Case (iii): $\Delta\mathbf{W} \neq 0, \Delta\mathbf{S} \neq 0$. Assume $\mathbf{U}_\perp, \mathbf{V}_\perp$ are singular vectors of $P_{T^\perp}(\mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top)$, and $[\mathbf{U}, \mathbf{U}_\perp]$ and $[\mathbf{V}, \mathbf{V}_\perp]$ are unitary matrices. Similarly, using the same contraction rules to

case (i), we have that

$$\begin{aligned}&\langle \mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top, \mathbf{U}\mathbf{V}^\top+\mathbf{U}_\perp\mathbf{V}_\perp^\top-\mathbf{H}\rangle + \langle P_{T^\perp}\mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top), \mathbf{U}_\perp\mathbf{V}_\perp^\top - P_{T^\perp}(\mathbf{H})\rangle \\ \geq &\|P_{T^\perp}\mathbf{X}(\Delta\mathbf{W}+\Delta\mathbf{S})\mathbf{Y}^\top\|_{tr}(\frac{1}{2}-\zeta\sqrt{\frac{r}{2d}}) \geq 0. \end{aligned} \tag{4.45}$$

Combining these three cases, we have that **Condition 1** in Equation (4.40a) hold with high probability. Thus **Condition 1** holds.

Condition 2: isolated minimizer

Consider the minimizer $\mathbf{W}$ exists outside of the small ε-ball $B_\varepsilon(\mathbf{W})$, which means $\mathbf{W}_0$ is an isolated minimizer. We also consider the three cases discussed in Condition 1, which are case (i) $\Delta\mathbf{W}\neq 0, \Delta\mathbf{S}=0$, (ii) $\Delta\mathbf{W}=0, \Delta\mathbf{S}\neq 0$, and (iii) $\Delta\mathbf{W}\neq 0, \Delta\mathbf{S}\neq 0$. In the following we prove that under case (i), **Condition 2** holds, and the results that it holds under case (ii) and (iii) can easily be derived.

For case (i) $\Delta\mathbf{W}\neq 0, \Delta\mathbf{S}=0$, let us assume that the following inequality holds:

$$\|P_{T^\perp}\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top\|_F \geq C\lambda_Z \geq 0, \tag{4.46}$$

From Assumption 1, we have that

$$\begin{aligned}&\|P_{T^\perp}(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top)\|_{tr}(\frac{1}{2}-\zeta\sqrt{\frac{r}{2d}})-\alpha \\ \geq &\|P_{T^\perp}(\mathbf{X}\Delta\mathbf{W}\mathbf{Y}^\top)\|_F(\frac{1}{2}-\zeta\sqrt{\frac{r}{2d}})-\alpha \\ \geq &C\lambda_Z(\frac{1}{2}-\zeta\sqrt{\frac{r}{2d}})-\alpha \\ \geq &0 \end{aligned} \tag{4.47}$$

By carefully choosing $\lambda_\mathbf{Z}$, we can always find a C satisfying Equation (4.46). Thus **Condition 2** holds. Together with **Condition 1**, the contraction stated in Equation (4.32) is proved, and thus Theorem 1 holds. □

4.7.2 Proof of Theorem 2

To prove Theorem 2, we need to upper bound $\mathfrak{R}(F_\Theta)$. An important lemma is first provided as below:

Lemma 4. *Let* $S_w = \{\mathbf{W} \in \mathbb{R}^{n\times n} \mid \|\mathbf{W}\|_{tr} \leq w\}$ *and* $a = \max_i \|\mathbf{A}_i\|_F$, *where* $\{\mathbf{A}_i | \mathbf{A}_i \in \mathbb{R}^{n\times n}\}_{i=1}^m$ *is an arbitrary set, then:*

$$\mathbb{E}_\sigma[\sup_{\mathbf{W}\in S_w} \frac{1}{m}\sum_{i=1}^m \sigma_i tr(\mathbf{W}\mathbf{A}_i)] \leq 2aw\sqrt{\frac{\log 2n}{m}} \tag{4.48}$$

We use two different techniques to bound the Rademacher complexity $\mathfrak{R}(F_\Theta)$. The first bound of $\mathfrak{R}(F_\Theta)$ is given in Lemma 5.

Lemma 5. *let* $\mathscr{X} = \max_i \|\mathbf{X}_{i\cdot}\|_2, \mathscr{Y} = \max_j \|\mathbf{Y}_{j\cdot}\|_2$, *then*

$$\mathbb{E}_\Omega[\mathfrak{R}(F_\Theta)] \leq 2L_l(\sqrt{d_x d_y}\beta\mathscr{X}\mathscr{Y} + (\alpha + \sqrt{d_x d_y}\beta)\Delta)\sqrt{\frac{\log 2d}{s}} + 2L_l(\eta + \sqrt{k_\phi}\phi)\sqrt{\frac{\log 2N}{s}}$$

where $\Delta = s_1\mathscr{Y} + s_2\mathscr{X} + s_1 s_2$.

Proof.

$$\begin{aligned}
\mathfrak{R}(F_\Theta) =& \mathbb{E}_\sigma[\sup_{f\in F_\Theta} \frac{1}{s}\sum_{t=1}^s \sigma_t l(f(i_t, j_t), \mathbf{R}_{i_t,j_t})] \\
=& \mathbb{E}_\sigma[\sup_{\mathbf{W},\mathbf{S}} \frac{1}{s}\sum_{t=1}^s \sigma_t \mathbf{X}(\mathbf{W}+\mathbf{S})\mathbf{Y}^\top - \mathbf{R})^2_{i_t j_t}] \\
\leq& \frac{L_l}{s}\mathbb{E}_\sigma[\sup_{\mathbf{W},\mathbf{S}} \sum_{t=1}^s \sigma_t(\mathbf{x}_{i_t} + \Delta\mathbf{x}_{i_t})(\mathbf{W}+\mathbf{S})(\mathbf{y}_{j_t} + \Delta\mathbf{y}_{j_t})^\top] \\
=& \frac{L_l}{s}\mathbb{E}_\sigma[\sup_{\|\mathbf{W}\|_{tr}\leq\alpha} \sum_{t=1}^s \sigma_t \mathbf{x}_{i_t}\mathbf{W}\mathbf{y}_{j_t}^\top + \sup_{\|\mathbf{S}\|_{2,1}\leq\beta} \sum_{t=1}^s \sigma_t \mathbf{x}_{i_t}\mathbf{S}\mathbf{y}_{j_t}^\top + \sup_{\mathbf{W},\mathbf{S}} \sum_{t=1}^s \sigma_t \Delta\mathbf{x}_{i_t}(\mathbf{W}+\mathbf{S})\mathbf{y}_{j_t}^\top \\
&+ \sup_{\mathbf{W},\mathbf{S}} \sum_{t=1}^s \sigma_t \mathbf{x}_{i_t}(\mathbf{W}+\mathbf{S})\Delta\mathbf{y}_{j_t}^\top + \sup_{\mathbf{W},\mathbf{S}} \sum_{t=1}^s \sigma_t \Delta\mathbf{x}_{i_t}(\mathbf{W}+\mathbf{S})\Delta\mathbf{y}_{j_t}^\top]
\end{aligned}$$

$$
\begin{aligned}
&\leq L_l \mathbb{E}_\sigma [\frac{1}{s}(\sup_{\|\mathbf{Z}\|_{tr}\leq\eta} \sum_{t=1}^{s} \sigma_t tr(\mathbf{Z}_{i_t,j_t}\mathbf{e}_{j_t}\mathbf{e}_{i_t}) + \sup_{\|\Phi\|_F\leq\phi} \sum_{t=1}^{s} \sigma_t tr(\Phi_{i_t,j_t}\mathbf{e}_{j_t}\mathbf{e}_{i_t}))\\
&\quad + 2t\sqrt{\frac{\log 2d}{s}}[\max_{i,j} \|\mathbf{y}_j \Delta\mathbf{x}_i^\top\|_2 + \max_{i,j} \|\Delta\mathbf{y}_j \mathbf{x}_i^\top\|_2 + \max_{i,j} \|\Delta\mathbf{y}_j \Delta\mathbf{x}_i^\top\|_2]\\
&\quad + \frac{1}{s}(\sup_{\|\mathbf{S}\|_{2,1}\leq\beta} \sum_{t=1}^{s} \sigma_t \mathbf{x}_{i_t}\mathbf{S}\mathbf{y}_{j_t}^\top + \sup_{\mathbf{S}} \sum_{t=1}^{s} \sigma_t \Delta\mathbf{x}_{i_t}\mathbf{S}\mathbf{y}_{j_t}^\top + \sup_{\mathbf{S}} \sum_{t=1}^{s} \sigma_t \mathbf{x}_{i_t}\mathbf{S}\Delta\mathbf{y}_{j_t}^\top\\
&\quad + \sup_{\mathbf{S}} \sum_{t=1}^{s} \sigma_t \Delta\mathbf{x}_{i_t}\mathbf{S}\Delta\mathbf{y}_{j_t}^\top)]\\
&\leq 2L_l[(\eta + \sqrt{k_\phi}\phi)\sqrt{\frac{\log 2N}{s}} + t\sqrt{\frac{\log 2d}{s}}\Delta + \frac{1}{s}(\sup_{\|\mathbf{S}\|_{2,1}\leq\beta} \sum_{t=1}^{s} \sigma_t \mathbf{x}_{i_t}\mathbf{S}\mathbf{y}_{j_t}^\top\\
&\quad + \sup_{\mathbf{S}} \sum_{t=1}^{s} \sigma_t \Delta\mathbf{x}_{i_t}\mathbf{S}\mathbf{y}_{j_t}^\top + \sup_{\mathbf{S}} \sum_{t=1}^{s} \sigma_t \mathbf{x}_{i_t}\mathbf{S}\Delta\mathbf{y}_{j_\alpha}^\top + \sup_{\mathbf{S}} \sum_{t=1}^{s} \sigma_t \Delta\mathbf{x}_{i_t}\mathbf{S}\Delta\mathbf{y}_{j_t}^\top)]\\
&\leq 2L_l[(\eta + \sqrt{k_\phi}\phi)\sqrt{\frac{\log 2N}{s}} + \alpha\sqrt{\frac{\log 2d}{s}}\Delta + \sqrt{d_x d_y}\beta\sqrt{\frac{\log 2d}{s}}(\Delta + \mathscr{X}\mathscr{Y})],
\end{aligned}
$$

which concludes Lemma 5. □

However, it is claimed that the above bound will become too loose in some circumstances. Hence, we need to deal with these cases by introducing the second tighter bound on Rademacher complexity.

Lemma 6. *let* $\mathscr{X} = \|\mathbf{X}\|_F, \mathscr{Y} = \|\mathbf{Y}\|_F$, *then*

$$
\mathbb{E}_\Omega[\mathfrak{R}(F_\Theta)] \leq 2L_l(\alpha + \sqrt{d_x d_y}\beta)\mathscr{X}\mathscr{Y}\sqrt{\frac{\log 2d}{s}} + \sqrt{\frac{9BCL_l(\sqrt{m}+\sqrt{n})}{s}(\sqrt{d}\alpha + \sqrt{d_x d_y}\beta)\Delta} \tag{4.49}
$$

where $\Delta = s_1\mathscr{Y} + s_2\mathscr{X} + s_1 s_2$.

Proof. Denote $\mathbf{P} \in \mathbb{R}^{m\times n}$ with each entry $\mathbf{P}_{ij} = \sum_{t:i_t=i,j_t=j} \sigma_t$, which means the "hit-time" on the i,j-th element of Ω, then we can divide $\mathfrak{R}(F_\Theta)$ as:

$$
\mathfrak{R}(F_\Theta) = \mathbb{E}_\sigma[\sup_{f\in F_\Theta} \frac{1}{s}\sum_{(i,j)} \mathbf{A}_{ij} l(f(i,j),\mathbf{R}_{ij})] + \mathbb{E}_\sigma[\sup_{f\in F_\Theta} \frac{1}{s}\sum_{(i,j)} \mathbf{B}_{ij} l(f(i,j),\mathbf{R}_{ij})] \tag{4.50}
$$

where we define

$$\mathbf{A}_{ij} = \begin{cases} \mathbf{P}_{ij}, & \text{if } h_{ij} \geq p \\ 0, & \text{otherwise.} \end{cases} \quad \mathbf{B}_{ij} = \begin{cases} 0, & \text{if } h_{ij} \geq p \\ \mathbf{P}_{ij}, & \text{otherwise.} \end{cases}$$

Here $h_{ij} = |\{t, i_t = i, j_t = j\}|$ and p is a thresholding value. In the following, we separately bound two terms in Equation 4.50 and specify the value of p appropriately to get the expected result. Since $l(f(i,j), \mathbf{R}_{ij})$ is bounded by B, we have that:

$$\mathbb{E}_\sigma[\sup_{f \in F_\Theta} \frac{1}{s} \sum_{(i,j)} \mathbf{A}_{ij} l(f(i,j), \mathbf{R}_{ij})] \leq \frac{B}{s} \mathbb{E}_\sigma[\sum_{(i,j)} |\mathbf{A}_{ij}|] \leq \frac{B}{\sqrt{p}},$$

where the last inequality holds because of **Lemma 10** in [76].

On the other hand, for second term in Equation 4.50, we have that,

$$\begin{aligned}
&\mathbb{E}_\sigma[\sup_{f \in F_\Theta} \frac{1}{s} \sum_{(i,j)} \mathbf{B}_{ij} l(f(i,j), \mathbf{R}_{ij})] \\
\leq &\frac{L_l}{s} \mathbb{E}_\sigma[\sup_{\mathbf{W},\mathbf{S}} \sum_{ij} \mathbf{B}_{ij} (\mathbf{x}_i + \Delta\mathbf{x}_i)(\mathbf{W} + \mathbf{S})(\mathbf{y}_j + \Delta\mathbf{y}_j)^\top] \\
= &\frac{L_l}{s} \mathbb{E}_\sigma[\sup_{\|\mathbf{W}\|_{tr} \leq \alpha} \sum_{(i,j)} \mathbf{B}_{ij} \mathbf{x}_i \mathbf{W} \mathbf{y}_j^\top + \sup_{\|\mathbf{S}\|_{2,1} \leq \beta} \sum_{(i,j)} \mathbf{B}_{ij} \mathbf{x}_i \mathbf{S} \mathbf{y}_j^\top + \sup_{\mathbf{W},\mathbf{S}} \sum_{(i,j)} \mathbf{B}_{ij} \Delta\mathbf{x}_i (\mathbf{W} + \mathbf{S}) \mathbf{y}_j^\top \\
&+ \sup_{\mathbf{W},\mathbf{S}} \sum_{(i,j)} \mathbf{B}_{ij} \mathbf{x}_i (\mathbf{W} + \mathbf{S}) \Delta\mathbf{y}_j^\top + \sup_{\mathbf{W},\mathbf{S}} \sum_{(i,j)} \mathbf{B}_{ij} \Delta\mathbf{x}_i (\mathbf{W} + \mathbf{S}) \Delta\mathbf{y}_j^\top] \qquad (4.51)
\end{aligned}$$

For the first two terms, we have,

$$\begin{aligned}
&\frac{L_l}{s} \mathbb{E}_\sigma[\sup_{\|\mathbf{W}\|_{tr} \leq \alpha} \sum_{(i,j)} \mathbf{B}_{ij} \mathbf{x}_i \mathbf{W} \mathbf{y}_j^\top + \sup_{\|\mathbf{S}\|_{2,1} \leq \beta} \sum_{(i,j)} \mathbf{B}_{ij} \mathbf{x}_i \mathbf{S} \mathbf{y}_j^\top] \\
\leq &L_l \mathbb{E}_\sigma[\sup_{\|\mathbf{W}\|_{tr} \leq \alpha} \frac{1}{s} \sum_{t=1}^{s} \sigma_t \mathbf{x}_{i_t} \mathbf{W} \mathbf{y}_{j_t}^\top + \sup_{\|\mathbf{S}\|_{2,1} \leq \beta} \frac{1}{s} \sum_{t=1}^{s} \sigma_t \mathbf{x}_{i_t} \mathbf{S} \mathbf{y}_{j_t}^\top] \\
\leq &L_l [2\alpha \mathscr{X} \mathscr{Y} \sqrt{\frac{\log 2d}{s}} + 2\sqrt{d_y k_S} \beta \mathscr{X} \mathscr{Y} \sqrt{\frac{\log 2d}{s}}] \qquad (4.52)
\end{aligned}$$

For the rest three terms, we have

$$
\begin{aligned}
&\frac{L_l}{s}\mathbb{E}_\sigma[\sup_{\mathbf{W},\mathbf{S}}\sum_{(i,j)}\mathbf{B}_{ij}\Delta\mathbf{x}_i(\mathbf{W}+\mathbf{S})\mathbf{y}_j^\top+\sup_{\mathbf{W},\mathbf{S}}\sum_{(i,j)}\mathbf{B}_{ij}\mathbf{x}_i(\mathbf{W}+\mathbf{S})\Delta\mathbf{y}_j^\top+\sup_{\mathbf{W},\mathbf{S}}\sum_{(i,j)}\mathbf{B}_{ij}\Delta\mathbf{x}_i(\mathbf{W}+\mathbf{S})\Delta\mathbf{y}_j^\top]\\
\leq&\frac{L_l\mathbb{E}_\sigma[\|\mathbf{B}\|_{sp}]}{s}(\sup_{\|\mathbf{W}\|_{tr}\leq\alpha}\|\Delta\mathbf{X}\mathbf{W}\mathbf{Y}^\top\|_{tr}+\sup_{\mathbf{W}_{tr}\leq\alpha}\|\mathbf{X}\mathbf{W}\Delta\mathbf{Y}^\top\|_{tr}+\sup_{\|\mathbf{W}\|_{tr}\leq\alpha}\|\Delta\mathbf{X}\mathbf{W}\Delta\mathbf{Y}^\top\|_{tr}\\
&+\sup_{\|\mathbf{S}\|_1\leq\beta}\|\Delta\mathbf{X}\mathbf{S}\mathbf{Y}^\top\|_{tr}+\sup_{\mathbf{S}_1\leq\beta}\|\mathbf{X}\mathbf{S}\Delta\mathbf{Y}^\top\|_{tr}+\sup_{\|\mathbf{S}\|_1\leq\beta}\|\Delta\mathbf{X}\mathbf{S}\Delta\mathbf{Y}^\top\|_{tr})\\
\leq&\frac{L_l\mathbb{E}_\sigma[\|\mathbf{B}\|_{sp}]\sqrt{\min(d_x,d_y)}}{s}(\alpha(\|\Delta\mathbf{X}\|_F\|\mathbf{Y}^\top\|_F+\|\mathbf{X}\|_F\|\Delta\mathbf{Y}^\top\|_F+\|\Delta\mathbf{X}\|_F\|\Delta\mathbf{Y}^\top\|_F)\\
&+\sqrt{d_y}(\|\Delta\mathbf{X}\|_F\|\mathbf{Y}^\top\|_F+\|\mathbf{X}\|_F\|\Delta\mathbf{Y}^\top\|_F+\|\Delta\mathbf{X}\|_F\|\Delta\mathbf{Y}^\top\|_F))\\
\leq&\frac{L_l\mathbb{E}[\|\mathbf{B}\|_{sp}]}{s}(\sqrt{d_x}\alpha+\sqrt{d_xd_y}\beta)\Delta\quad(\text{or}(\sqrt{d_x}\alpha+\sqrt{d_xd_y}\beta))\\
\leq&\frac{2.2CL_l\sqrt{p}(\sqrt{m}+\sqrt{n})}{s}[(\sqrt{d_x}\alpha+\sqrt{d_xd_y}\beta)\Delta]
\end{aligned}
$$

Finally we set $p=sB/[2.2CL_l(\sqrt{m}+\sqrt{n})(\sqrt{d_x}\alpha+\sqrt{d_xd_y}\beta)\Delta]$, and combine the three items above to obtain

$$
\mathfrak{R}(F_\Theta)\leq 2L_l(\alpha+\sqrt{d_xd_y}\beta)\mathscr{X}\mathscr{Y}\sqrt{\frac{\log 2d}{s}}+\sqrt{\frac{9BCL_l(\sqrt{m}+\sqrt{n})}{s}(\sqrt{d_x}\alpha+\sqrt{d_xd_y}\beta)\Delta}. \tag{4.53}
$$

Thus Lemma 6 is proved. □

Proof of Theorem 2. By combining Lemma 5 and Lemma 6, the expected l-risk of an optimal solution $(\mathbf{Z}^*,\mathbf{W}^*,\mathbf{S}^*)$ will be bounded by

$$
\begin{aligned}
R_l(f^*)\leq&2L_l\sqrt{d_xd_y}\beta\mathscr{X}\mathscr{Y}\sqrt{\frac{\log 2d}{s}}\\
&+\min\{2L_l[(\alpha+\sqrt{d_xd_y}\beta)\Delta\sqrt{\frac{\log 2d}{s}}+(\eta+\sqrt{k_\phi}\phi)\sqrt{\frac{\log 2N}{s}}],\\
&2L_l\alpha\mathscr{X}\mathscr{Y}\sqrt{\frac{\log 2d}{s}}+\sqrt{\frac{9BCL_l(\sqrt{m}+\sqrt{n})}{s}(\sqrt{d_x}\alpha+\sqrt{d_xd_y}\beta)\Delta}\}\\
&+B\sqrt{\frac{\log\frac{1}{\delta}}{2s}}
\end{aligned} \tag{4.54}
$$

Let $R_l(f^*)$ be smaller than ε and re-organize terms, we reach the conclusion that $O(\min\{(\eta + \sqrt{k_\phi}\phi^2)\log N, (\alpha + \sqrt{d}\beta)\kappa^2\sqrt{N}\}/\varepsilon^2)$ is sufficient for ε-recovery. Thus Theorem 2 is proved.

□

4.8 Conclusions

We studied the robust inductive matrix completion method for recommendation problems in this work. Inductive matrix completion methods explicitly explore user side information such as "age" and "gender" and items such as "genre" and "release date" to boost recommendation performance. Considering that noise may be present in some attributes, how to eliminate its negative influence in recommendation algorithms is a key problem. We revisit the common low-rank principal in inductive matrix completion modeling and introduce a group sparse part to identify noisy attributes for users and items. We theorize that, with perfect side features, our RIMC method can recover the ground truth latent matrices and, with informative but not perfect side features, RIMC requires fewer observations to achieve ε-recovery than methods that do not use side information. We validate our model on the MovieLens and NCI Challenge datasets and achieve promising performance.

Chapter 5

Evaluating the Robustness of RecSys

Deep learning based recommendation systems (RecSys) have achieved significant performance gains in many applications. However, whether their prediction remains constant with the slight perturbations in the input features, or whether the model is robust or not, is not clearly understood. Though the robustness of models has raised concerns from practitioners in applications, there is little work studying it. In this chapter, we study the robustness of recommendation systems: what are the minimal number of features required to mispredict a user's preference. We design a threat model to evaluate RecSys' robustness by tackling the challenge of the discrete input features. We empirically demonstrate the robustness of some state-of-the-art recommendation models. By perturbing one or two features, the recommendation models' performance can downgrade dramatically; we break Wide&Deep from Google with 100% success rate on MovieLens dataset. Our experimental results suggest that large feature space leads to high vulnerability, and our results also point out potential directions for improving the RecSys' performance.

5.1 Introduction

Recommendation systems play an important role for both customers and sellers. Representation learning on explicit features with deep learning methods has become popular in various scenarios, such as apps and movie recommendation [89–91], click prediction [92]. In these

applications, contextual variables such as weather, date, daytime are usually available. For these categorical variables, a popular way is to embed them into a latent space, and apply different operations to connect the embedding vectors ensuing by regular neural network layers [89, 91]. However, their robustness [93] to perturbations is not understood. In other words, if one changes the input feature from **Monday** to **Tuesday**, will the prediction vary dramatically? This problem has not been studied in deep learning based RecSys. In real applications, the adversaries exist extensively on the Internet and fraudsters can easily inject noisy features into the recommendation systems. To understand their robustness becomes important.

The robustness of recommendation systems has raised attention from researchers and practitioner. However, there is little work that tries to understand this problem. In a closely established field, the robustness of neighbor-based and factorization-based collaborative filtering algorithms have been studied. Existing work [15–17] studied attacks of the neighbor-based collaborative filtering approaches by promoting or demoting a specific set of items. Seminario and Wilson [94] showed that item-based collaborative filtering is more robust under particular conditions. Li et al. [18] aimed to decrease the general performance of collaborative filtering models based on full knowledge of the learning algorithm and training ratings. In other related fields, many works study the robustness of deep neural networks in computer vision [95, 14, 96, 97]. When features are continuous, such as pixel values in images, one evaluate models' robustness by the minimal perturbation required to misclassify a sample [93]. However, there is no study on deep learning-based RecSys.

In this chapter, we bridge the gap and study the robustness of deep learning-based recommendation systems. The major challenge exists in how to deal with the natural discrete features in RecSys. Since continuous perturbations are not available, we study the minimal number of features required to mispredict a user's preference. We introduce a two-stage approach: we first optimize over the embedding vectors of a set of input features using $l_{2,1}$ norm until succeeding; then we map the perturbed embedding vectors back to the embedding table and shrink the attacked feature set by ignoring the embedding vector that is minimally changed.

We demonstrate that deep learning-based recommendation systems are easily manipulated; with one or two features perturbed, the prediction of user preference can vary dramatically. We consider two state-of-the-art models, Neural Factorization Machine (NFM) [89] and Wide&Deep [91] from Google. Both of the models are constructed based on discrete features such as user IDs, item IDs, and contextual features. The contextual features are sparse and represented as embedding vectors. By perturbing the features of the target model, we show that the state-of-the-art recommendation models' can be easily downgraded by our threat model. For example, by modifying one attribute on MovieLens dataset, the success rate reaches 100% on attacking Wide&Deep model; and by modifying two attributes on Frappe, the success rate reaches 98% on attacking NFM model.

Our contributions are threefold: i) we first propose a new threat model to evaluate the robustness of deep learning-based RecSys with contextual features; ii) we demonstrate the state-of-the-art recommendation models' robustness boundary. For example, for Wide&Deep on MovieLens dataset, perturbing one feature in the input can decrease the model's accuracy to 0%; iii) Our empirical results suggest that a larger feature space leads to more vulnerability in adversarial attacks. It is also suggested that the cost is important for customers to make decisions. This points out a direction of improving the RecSys' performance by including more cost-related features, such as price and payment method.

In Section 5.3 we introduce the proposed threat model. In Section 5.4 we present the experiment results, followed by related works in Section 5.2. Finally, we conclude in Section 5.5 and discuss potential directions for improving the current RecSys.

5.2 Related Work

Recently, due to the power of neural networks in representation learning, many works [98, 90, 99, 100] propose to learn the embedding of user/item and other discrete features. Despite their great success [89, 92, 91], there is little work studying the robustness of them. [15–17] studied attacks on the neighbor-based collaborative filtering approaches by promoting or demoting a specific set of items. Seminario and Wilson [94] showed that

item-based collaborative filtering is more robust under particular conditions. Li et al. [18] aimed to decrease the general performance of factorization-based collaborative filtering models based on full knowledge of the learning algorithm and training data. The optimal set of malicious users and their ratings can be reached by solving a min-max objective function to contaminate a collaborative filtering system's performance.

Our threat model is different from all of these works; while they require complete knowledge of the training data and can intervene in the training process, our model evaluates the target model at test phase only.

5.3 Proposed Framework

We first give an introduction to the framework of state-of-the-art RecSys. Then we introduce our threat model and a random attack approach as a baseline.

5.3.1 Preliminaries

We denote the input feature to a RecSys as $x = \{u, i, c_1, ..., c_n\}$ where u, i represent the IDs of users and items respectively, n is the number of categorical variables and $\{c_1, ..., c_n\}$ represent the list of features to be feed to the system. For each variable, we use D_j to denote its candidate values, and D_u, D_i to denote candidate values for user and item IDs. For example, in Frappe dataset, the variable "*weekday*" has seven values taken from {*Monday*, ...,*Sunday*}.[1] By arranging all sparse features including user and item IDs into a long array, the feature vector x is a list of integer numbers indicating the index of sparse features in the feature set.

We denote the scoring function as $F(x)$. Assume that the label y takes value from $\{-1, +1\}$, $F(x) > 0$ will lead to a positive label on x and vice versa. Following [89], one first denotes the connection layer of embedding vectors as $f(v_x)$; then the hidden layers are defined as:

$$z_l = \sigma_l(w_l f(v_x) + b_l), \;\; l = 1, ..., L \tag{5.1}$$

[1] http://baltrunas.info/research-menu/frappe

where L is the layer number and σ_l is the activation function, such as rectified linear units (ReLUs). w_l, b_l, z_l denote the weights, bias, and output of l-th layer. The final scoring function can be defined as:

$$F(x) = h^T \sigma_L(W_L^\top(...\sigma_1(W_1^\top z_1 + b_1)...) + b_L), \tag{5.2}$$

where h is the weight for the final prediction layer. The infrastructure of the introduced model is presented in Figure 5.1. This framework encompasses many state-of-the-art recommendation models. For example, He et al. [90] is a special case to Figure 5.1 when the input features contain only user and item IDs. NFM specify the connection layer by higher-order feature interaction. Wide&Deep combines a linear function on continuous features with a deep module structured as in Figure 5.1.

We aim to build a threat model on the learned prediction function $F(\cdot)$ and evaluate the models' performance under attacks.

5.3.2 The proposed threat model

The threat model considered is a white-box targeted attack model. The adversary tries to mislead the system to make wrong decisions about a user's preference on an item based on the manipulated features. Since the label y lies in $\{-1, +1\}$, our goal is to reverse the output of scoring function $F(x)$ from positive to negative or vise versa.

Since input x is composed of sparse features, the adversary makes changes to the features in x obtaining x' to reach its malicious goal. To produce confident adversarial examples, we set a threshold θ for adversarial examples. Specifically, let $|\cdot|$ denotes absolute value, only when x' satisfies the following condition, we label x' as a successful adversarial example:

$$F(x)F(x') < 0, \;\; |F(x')| > \theta. \tag{5.3}$$

Following this rule, the loss function for adversarial attack can be defined as:

$$\mathscr{L}(x') = \max\{0, y \cdot F(x') + \theta\}. \tag{5.4}$$

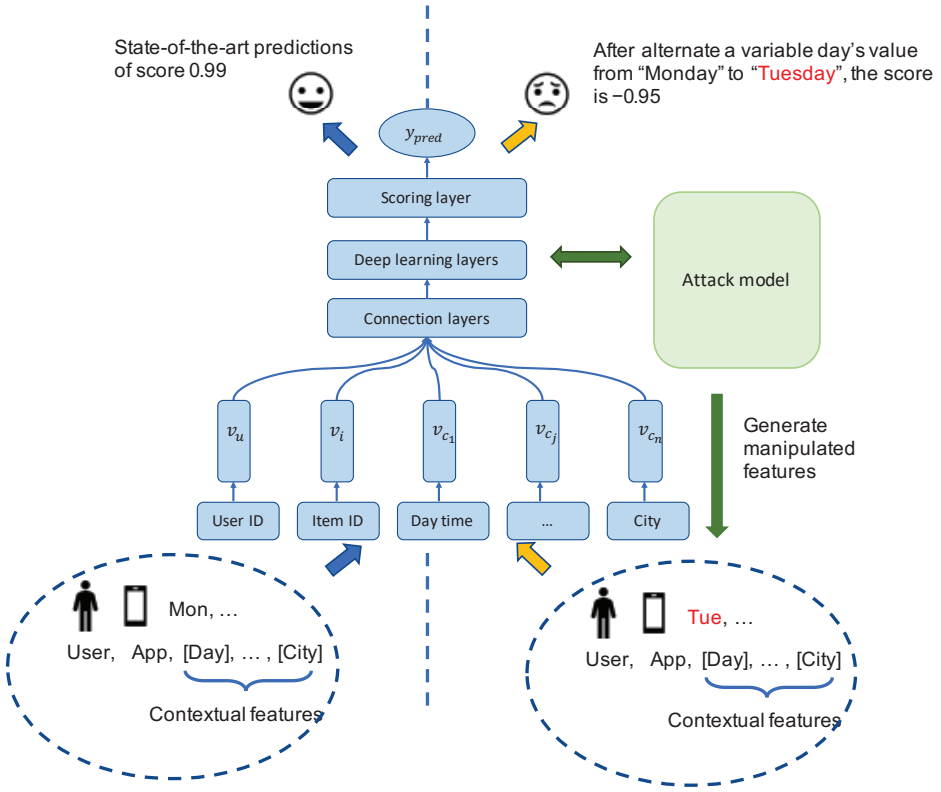


Fig. 5.1 Overview of a deep learning-based recommendation system based on Frappe dataset and its potential vulnerability to unnoticeable changes. The listed features including user ID, item ID are encoded into vectors of a fixed length. By alternating a feature from "Monday" to "Tuesday", the model's prediction changes dramatically.

θ controls the confidence of adversarial prediction.

Traditional white-box attacks such as [14] estimate the output's derivative for the input features and update the input towards the adversarial direction. However, infinitesimally optimizing over the input feature x to search for an adversarial example is impossible as the input features are sparse. One may perform attacks by exhaust every possible value of the variables in the input space. For example, on Frappe dataset, eight contextual variables are available, and each variable has many candidate values from 2 to 233. If an adversary wants to manipulate two features, there are 100 thousand choices in total to modify the output of an input.

Though the input feature x contains discrete features, the corresponding embedding vectors are continuous. Let $V = \{v_u, v_i, v_{c_1}, ..., v_{c_n}\}$ denote the embedding vectors corresponding to the features in the input x. Let $\widetilde{F}(V)$ denote the module in $F(x)$ after the embedding layer

of x. Let $\mathscr{L}(V') = \max\{0, y \cdot \widetilde{F}(V') + \theta\}$. To alter the output of the model $\widetilde{F}$, one would solve the following problem:

$$\begin{aligned} \min_{V'} \quad & \mathscr{L}(V') + \lambda \|\delta V\| \\ \text{s.t.} \quad & \delta V = V - V' \end{aligned} \tag{5.5}$$

Here δV is the adversarial noises added to the embedding vector and $\|\cdot\|$ is a regularization term to be specified, often L_1, L_2, and L_∞ norm. The embedding vectors lie in a continuous space and the gradient information of the scoring function $\widetilde{F}(V)$ with respect to the embedding vectors are easily obtained. By solving the Equation (5.5), it is highly possible to obtain adversarial embedding vectors with high success rate, since similar forms have been used in adversarial attacks of image recognition problems [14] and report nearly 100% success rate on attacking state-of-the-art image classification models.

One might change the embedding vectors as much as he needs, but it is not guaranteed to be mapped back to categorical features in practice. To keep the designed adversarial examples a valid input is the critical challenge in designing a threat model for embedding-based recommendation systems. On the one hand, the number of permissible attacked variables should be kept low. On the other hand, for the attacked variable, the modified embedding should be close to the candidate embedding vectors in the embedding table to make it easy to map embedding back to sparse features.

We consider using $l_{2,1}$ norm on the embedding modifiers. The $l_{2,1}$ norm is introduced to select important features for predictors originally by [85]. By adopting the $l_{2,1}$ norm regularization, the obtained modifiers would be group sparse, which means that only few of the columns are significantly modified while others remain close to zeros. The similar idea is also used in [101] for natural language processing tasks. We bring it into attacking recommendation systems. We obtain the following problem:

$$\begin{aligned} \min_{V'} \quad & \mathscr{L}(V') + \lambda_1 \|\delta V\|_{2,1} + \\ & \lambda_2 \sum_k \min_{D_k} \|v_k + \delta v_k - D_k\|_2 \\ s.t. \quad & v_k + \delta v_k \in D_k. \end{aligned} \tag{5.6}$$

Algorithm 4 A sketch of the evaluation process on the recommendation models

Input: a pre-trained model $\widetilde{F}$, input example $x = \{u, i, c_1, ..., c_{n_j}\}$, original prediction y, the size of the attacked variable set d, an initialized attacked variable set $\mathscr{S}$
Output: adversarial example x'

while True **do**
  Solve the problem in Eq. (5.7) on embedding vectors $\{v^*_{c_j} | j \in \mathscr{S}\}$ such that: $\widetilde{F}(V') \cdot y < 0, |\widetilde{F}(V')| > \theta$
  Select the vector $v^*_{c_j}$ that is minimally modified by: $\arg\min_j \|v^*_{c_j} - D_j\|_2$
  Map it back to ground-truth embedding vector by: $v^*_{c_j} \leftarrow v_{c_j}$ $\mathscr{S} = \mathscr{S} - j$
  if $|\mathscr{S}| = d$ **then**
    Map $v^*_{c_j}$ back to v'_{c_j} in embedding table by: $v' = \min_{v \in D_j} \|v^*_{c_j} - v\|_2$
    Obtain the corresponding features and update x'
    break
  end if
end while
return x'

Here k represents the index of all variables including user and items IDs. While optimizing over V' easily leads to success, it is hard to map the obtained adversarial embedding back to valid feature IDs. When we force the mapping adversarial embedding vectors to some IDs, the attack might be invalid.

When attacking real-world recommendation systems, one might encounter many constraints. For example, some features are not allowed to be modified, such as the user and item IDs. These two variables are both features and unique index of this input. Other features that may cause the developers' attention should be avoided to be attacked as well. To control the attacked variables, we define $\mathscr{S}$ as the set of permissible variables that can be modified.

Hence, we adopt a two-stage optimization strategy. We alternate between minimizing $\mathscr{L}(V')$ to obtain adversarial embedding and shrinking the set $\mathscr{S}$ by removing the embedding vector that is minimally changed during optimization until a satisfactory set $\mathscr{S}$ is reached. Finally, we project the perturbed embedding vectors back to sparse features to generate a perturbed example x'. Assume the maximum allowed number of variables is d, we propose

the following model:

$$\begin{aligned} \min_{V'} \quad & \mathscr{L}(V') + \lambda \sum_{j \in \mathscr{S}} \|\delta v_{c_j}\|_2 \\ \text{s.t.} \quad & v_{c_j} + \delta v_{c_j} \in D_i, \\ & |\mathscr{S}| \leq d. \end{aligned} \tag{5.7}$$

$\mathscr{S}$ can be randomly initialized to have a cardinality of d.

We summarize the overall process of the optimization of our attack model Equation (5.7) in Algorithm 4. In general, we gradually shrink the attacked set $\mathscr{S}$ until we reach a required set size k ($k \leq d$) of attacked variables; then we map V' back to the embedding table and obtain the perturbed feature x'. If x' still satisfies Eq. (5.3), we label it as a valid attack. Otherwise, the attack fails. In experiments, we use the most strict constraints by setting $d = 1$ or 2.

5.3.3 Random attack as a baseline

Since there is no previous work dealing with the attacks on deep learning-based recommendations, we construct a baseline by randomly alter the features in x to other eligible features, namely *random attack*. We denote the number of attacked variables as k. For Frappe, eight contextual variables are available for attack. Under the threshold θ, the procedures for random attack can be summarized into the following steps:

1. Randomly pick k variables from $x = \{u, i, c_1, ..., c_n\}$ and avoid selecting u, i;
2. For each variable, randomly select a candidate value to replace the input value and obtain x';
3. Repeat the above two steps until $F(x) \cdot F(x') < 0$ and $|F(x')| > \theta$, or, the maximum iterations m is exceeded.

The higher the maximum iterations are, the larger the probability is for the random attack to succeed. In experiments, we set the limitation of maximum iterations and test the attack performance with different numbers of variables to attack.

Table 5.1 Dataset information

Dataset	#Instance	#Feature	#User	#Item
Frappe	288,609	5,382	957	4,082
MovieLens	2,006,859	90,445	17,045	23,743

5.4 Experiments

First, we introduce the experimental settings and then evaluate two state-of-the-art models' resistance to attacks by the proposed threat model.

5.4.1 Experimental settings

Base Models

We evaluate the proposed threat model on two state-of-the-art deep learning models: Neural Factorization Machine [89] and Wide&Deep [91] by Google.

- **NFM** [89]. This model boosts the traditional linear factorization machine [102] with the power of deep neural networks. It encodes the sparse features by one-hot encoding and learns interactions between features automatically. This model benefits from strengths of factorization machine and deep learning in learning both high order and non-linear interactions.

- **Wide&Deep** [91]. Wide&Deep model combines a generalized linear model with a deep component to capture the feature interactions. We follow the structure of Wide&Deep reported in the paper which includes a three-layer MLP module with rectified linear units (ReLUs) as the activation function. The layer sizes are set to be {1024, 512, 256 }.

Datasets

Two datasets Frappe [103] and MovieLens [68] are used in training the models and evaluating the performance of the proposed threat model.

Table 5.2 RMSE of the evaluated models under different latent factors on Frappe and MovieLens on test set.

	Frappe					MovieLens				
Method	n=16	n=32	n=64	n=128	n=256	n=16	n=32	n=64	n=128	n=256
Wide&Deep	0.3890	0.3765	0.3744	0.3858	0.4002	0.5573	0.5469	0.535	0.5633	0.5284
NFM	0.3874	0.3453	0.3137	0.3138	0.3119	0.5669	0.5478	0.5211	0.4833	0.4535

Frappe Frappe is a dataset for mobile app recommendation under contextual information. 96,203 app usage logs are recorded in the dataset. Each usage log contains a list of ten variables, which are user ID, app ID and eight context variables such as day time (e.g., morning), weather, and country. Follow the processing in [89], every log containing ten variables are converted into a one-hot feature vector, which leads to a total of 5, 382 features. A label value of 1 indicates that under the context a user used the item.[2]

MovieLens We use the latest version of MovieLens data released by GroupLens in 2017. This dataset is organized to be used to study the higher-order feature interactions by [89]. The tagging data contains 17,045 users' tag applications on 23,743 items, resulting in a total of 668,953 logs with 49,657 distinct tags. 90,445 features are obtained in total by converting each log including userID, item ID and a tag, to a feature vector using one-hot encoding. For each log, a label value of 1 means that the user has pinned the tag to the item. [3]

For the negative examples, we follow [89] and construct two negative examples for each positive example. For Frappe, two apps that are not used under the contextual features are sampled as negative examples. For Movielens, two tags that are not labeled by the user about the item are selected as negative examples. The resulting information of the whole dataset is reported in Table 5.1.

Setup and baselines

For training, we follow the dataset segmentation used in [89] in which 70% of the whole dataset is used for training, 20% for validation and 10% for test. The evaluation of the

[2]http://baltrunas.info/research-menu/frappe
[3]http://grouplens.org/datasets/movielens/latest

model is based on the root mean square error (RMSE) on the test set. λ is selected from $\{10^{-3}, ..., 10^{3}\}$ based on the performance on the validation set.

Firstly, we train two sets of models on Frappe and MovieLens respectively following the settings used in [89]. The test error is reported in Table 5.2, which is consistent with the results reported in the original papers. Then we construct the set for robustness evaluation by randomly selecting 200 examples from the test set. We set $\theta = 0.5$ to perform confident attacks, which means that when attacking an example with a positive label, the score of the manipulated example less than -0.5 will be considered as a successful attack.

Baseline attack

The baseline attack algorithm is *random attack*. As introduced in Section 5.3.3, *Random attack* attacks the target model by randomly alternate the features in an input x until it succeeds or reaches the maximum iteration, which is set to 1000 throughout the experiments.

Evaluation We use the success rate to evaluate the effectiveness of the threat model which is defined as:

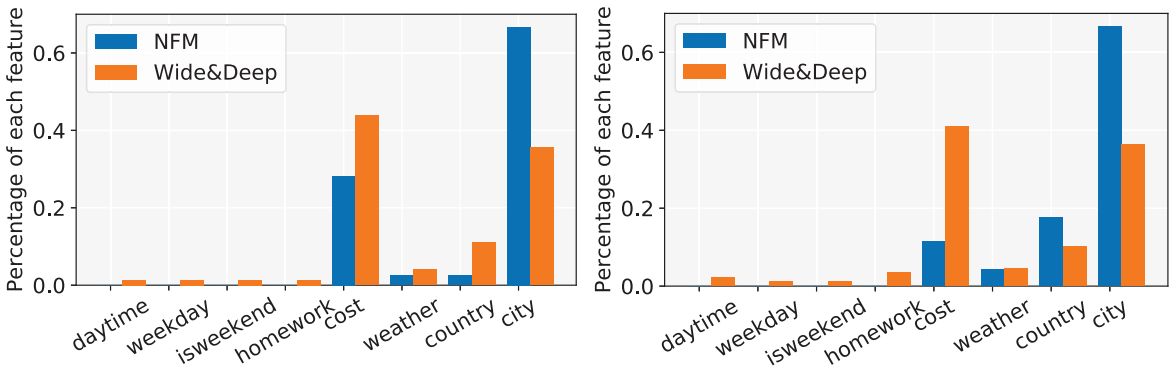
$$success\ rate = \frac{\text{\#total success}}{\text{\#total number of evaluations}}.$$

Success rate demonstrates the capacity of the threat model to generate adversarial examples to fool the target recommendation system.

To evaluate the RecSys' robustness, we apply the proposed threat model under a minimal number of attacked features to target models. We adapt two specific metrics: attack-1 success rate and attack-2 success rate. Attack-1 success rate is used for measuring the performance when the number of variables (features) allowed to attack is 1. Attack-2 success rate is for measuring the performance when the number of variables (features) allowed to attack is 2.

Table 5.3 Overall evaluation of base models on Frappe and Movielens. The number of permissible features are two for Frappe and one for Movielens. "#param" denotes the number of trainable parameters and "M" represents "one million". "Rand" represents *Random Attack*.

	Frappe						Movielens					
	n=128			n=256			n=128			n=256		
Target model	#param	success rate		#param	success rate		#param	success rate		#param	success rate	
		Our	Rand		Our	Rand		Our	Rand.		Our	Rand
Wide&Deep	2.66M	95.50%	25%	4.66M	85.50%	23%	12.72M	100%	30%	24.69M	100%	39%
NFM	0.71M	99%	15%	1.45M	95%	18%	11.68M	16%	16%	23.31M	86%	18%



(a) #feature=1 (b) #feature=2

Fig. 5.2 Bin frequencies for context variables under attack-1 and attack-2. In a), we plot the bin frequencies of the perturbed variables in successful attacks in attack-1. In b), each valid attack may contain one or two perturbed variables. We group all the perturbed variables and plot the bin frequencies.

5.4.2 Overall evaluation over different target systems

We first evaluate the effectiveness of the proposed attack, as well as the robustness of state-of-the-art models. The allowed variables to be changed are set to be two for Frappe and one for MovieLens. Under latent factors {128, 256}, we report the success rate of the proposed attack on Wide&Deep and NFM in Table 5.3. Higher success rate indicates lower robustness of target models. Our threat model achieves a high success rate on both target models and performs better than *Random Attack*. On Frappe dataset, NFM is more vulnerable than Wide&Deep. On MovieLens, the only feature is one tag, and Wide&Deep are entirely defeated by our threat model with 100% success rate.

Table 5.4 Comparison between *Random attack* and our attack on MovieLens dataset while attack two base models. The latent factors are set to be 64 for both models.

	NFM		Wide&Deep	
#feature	Rand. Att.	Our attack	Rand. Att.	Our attack
1	15%	64%	33%	100%

5.4.3 Analysis of vulnerable features

To understand which variable is more vulnerable to perturbations, we first collect successful attacks on Frappe. The number of options for eight variables **{daytime, weekday, isweekend, homework, cost, weather, country, city}** are {7, 7, 2, 3, 2, 9, 80, 233}. Under attack-1 and attack-2, we plot the bin frequencies for eight context variables respectively in Figure 5.2.

Among these variables, **city, country, weather** have relatively larger feature space compared to **daytime, weekday, isweekend, homework**, and they are more vulnerable to perturbations. We speculate that the threat model benefits from the large variable space for better exploration. For **cost**, although its feature space is only binary with values from {free, paid}, its influence on the prediction is significant. This validates the common knowledge that most customers care about the cost.

5.4.4 Attack under different number of permissible variables

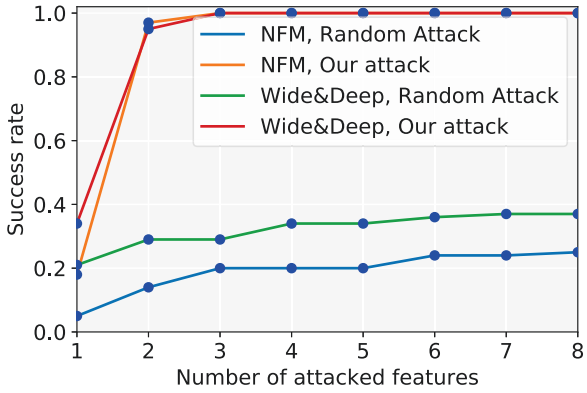
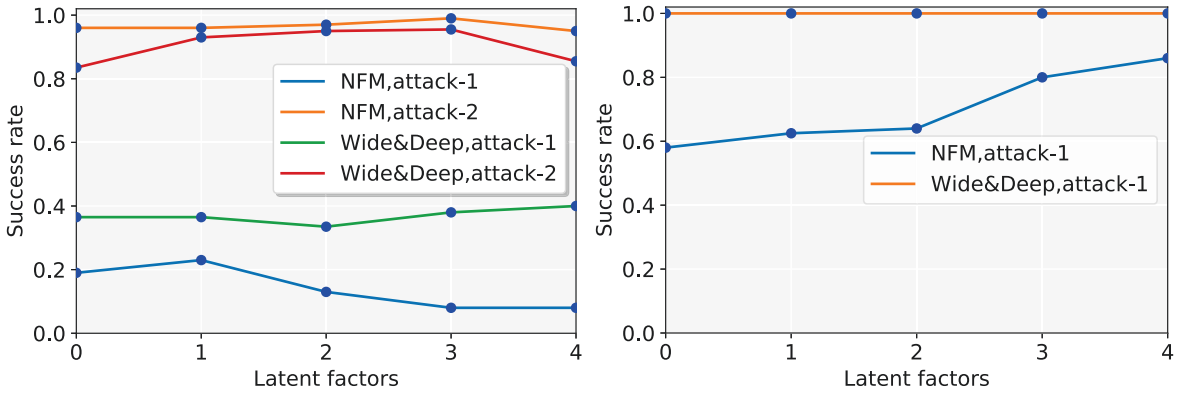


Fig. 5.3 Success rate of *Random Attack* and our threat model on Frappe dataset with different numbers of features being perturbed. The latent factors are set to be 64 for two target models.

When the number of attacked variables are different, we compare the success rate of applying the threat models to NFM and Wide&Deep. The latent factors for NFM and Wide&Deep are set to be 64. We show the results of Frappe dataset in Figure 5.3. The results show that with more attacked variables, the success rate of our attack increases dramatically and random attack increases very slowly. Our attack obtained 100% success rate when attacking three vari-



(a) Frappe (b) MovieLens

Fig. 5.4 Evaluation of the robustness of NFM and Wide&Deep by our proposed threat model under different latent factors.

ables on both NFM and Wide&Deep. The performance of *Random attack* does not vary significantly with the increase in the number of attacked features and maximum iterations. The more features we are allowed to alter, the higher the success rate will be for the threat models.

While the MovieLens dataset provides one feature only, we report the results in Table 5.4. The results show that our attack surpasses *Random Attack* by a large margin.

5.4.5 Attacks under different latent factors

To understand the target models' reaction to the threat model under different latent factors, we vary the size of embedding vectors in the set {16, 32, 64, 128, 256} and report the success rate in Figure 5.4. It shows that two base models are both vulnerable to attacks.

For Frappe dataset, we report the attack on one and two variables. When only one variable is permissible for an attack, the success rate of the threat model is low on NFM, but Wide&Deep are more easily broken down than NFM. When attacking two variable on Frappe, the proposed threat model report high success rate under different latent factors which are all larger than 95% on NFM and 83% on Wide&Deep.

We also crossly refer to Table 5.2 which shows the test error on different datasets using base models. We find that low test error on Wide&Deep leads to high success rate on the

attack. This phenomenon is also observed when attacking NFM on MovieLens. We speculate that the better-fitted model on training data is more fragile since the model might be tuned to identify the scarcely distinguishable examples. Hence its robustness is downgraded. One thing worth mentioning is that the attack on the MovieLens dataset is easier to succeed than on Frappe when attacking one variable. Note that Frappe has 343 contextual features in total and MovieLens provides $49,657$ tags. This fact shows that larger feature space leaves adversaries more chances to perform successful attacks.

5.5 Conclusions and Discussions

In this work, we have introduced an initial study on the robustness of deep learning-based recommendation systems. A two-stage method is proposed to tackle the challenge in discrete feature space in RecSys, which first optimizes over the continuous embedding space then projects the altered embedding vectors back to ground-truth to shrink the set of attacked categorical variables. Through our experiments, we show that state-of-the-art recommendation models are vulnerable to adversarial attacks. By altering one feature in the MovieLens data, our attack entirely defeats Wide&Deep model from Google with a 100% success rate.

The empirical results suggest that larger feature space leads to higher vulnerability in adversarial perturbations. We also find out that the cost is an important factor influencing customers' decisions. This points out a direction of improving the RecSys' performance by including more cost-related features, such as price and payment method.

Chapter 6

Enhancing the Robustness of RecSys Under Malicious Attacks

Recommendation system has become ubiquitous in online shopping in recent decades due to its power in reducing excessive choices of customers and industries. Recent collaborative filtering methods based on the deep neural network are studied and introduce promising results due to their power in learning hidden representations for users and items. However, it has revealed its vulnerabilities under malicious attacks. With the knowledge of a collaborative filtering algorithm and its parameters, the performance of this recommendation system can be easily downgraded. Unfortunately, this problem is not addressed well, and the study on defending recommendation systems is insufficient.

In this chapter, we aim to improve the robustness of recommendation systems based on two concepts: *stage-wise hints training* and *randomness*. To protect a target model, we introduce noise layers in the training of a target model to increase its resistance to adversarial perturbations. To reduce noise layers' influence on models' performance, we introduce intermediate layers' outputs as hints from a teacher model to regularize the intermediate layers of a student target model. We consider white box attacks under which attackers have the knowledge of the target model. The generalizability and robustness properties of our method have been analytically inspected in experiments and discussions, and the computation cost is comparable to training a standard neural network-based collaborative filtering model.

Through our investigation, the proposed defensive method can reduce the success rate of malicious user attacks and keep the prediction accuracy comparable to standard neural recommendation systems.

6.1 Introduction

In recent decades, recommendation systems have become a pivotal tool for both industries and customers due to the information explosion on the Internet. By using these recommendation algorithms, such as factorization-based approaches, the developers make an implicit assumption that they are secure. However, due to the open property of recommendation systems, recent works show that factorization based approaches are vulnerable to malicious user attacks [18]. Malicious users are threatening artifacts to the recommendation systems that can be crafted if the adversary knows the architecture of the recommendation system or legitimate user data [18, 15, 16]. Attackers, on the one hand, try to craft malicious user data that degrade the system's performance; on the other hand, they will keep their behaviors close to normal users to avoid detection. In attacking a recommendation system, malicious users' goals are diversified. By carefully exploit from training data and the structure of recommendation systems, adversaries can mislead the system to produce the target value that is far from the ground truth to achieve their malicious goals. The general purpose for an adversary falls into two categories: to promote (or demote) the popularity of a specific set of items and decrease the overall accuracy of the target recommendation systems to downgrade users' trust on it [17]. One important rule for crafting malicious examples for collaborative filtering is to keep the adversarial data close to the legitimate data, making them hard to be detected by the human.

Different attacks such as random attacks and push attacks have been investigated [15–17]. While [15, 16] are both built upon the neighbor-based collaborative filtering system. Li et al. [18] proposes a more aggressive data poisoning attack to decrease the general performance of collaborative filtering models which is based on full knowledge of the learning algorithm and training data. By solving a min-max objective function, the optimal set of malicious users can

be reached to contaminate collaborative filtering system's performance. Despite these attack strategies on collaborative filtering systems, the defense approaches for collaborative filtering are limited [18]. A close line of established research in robust collaborative filtering lies in robust matrix completion [19–21] in which a few entries or columns are randomly perturbed. However, none of them handles maliciously contaminated features for collaborative filtering.

In this chapter, we focus on designing an effective training paradigm for obtaining robust collaborative filtering models. We investigate the recent popular neural network-based collaborative filtering algorithms and design defensive methods for trustworthy collaborative filtering. Inspired by knowledge distillation [104, 105], we consider building our training paradigm by using knowledge from different layers of a teacher model to improve the training of student models. The standard collaborative filtering model is treated as a teacher. We use a similar structure to build a student collaborative filtering model. Extra noise layers are added before dense layers in a student model to improve its robustness to small perturbations. Note that the student model usually has the same or smaller size than the teacher model and the discrepancy between dimensionality of intermediate layers from teacher and student can be mitigated by a mapping function. By introducing intermediate representations from a teacher, we first make the student model to learn to predict the intermediate representations of the teacher, then train the whole student model. The noise layer enforces the student network to learn the neural layers that are robust to perturbations [106], and the hints training paradigm regularizes the student model to have similar performance to the teacher model.

We summarize the contributions of this chapter as follows:

- We propose an effective training strategy to obtain robust neural network-based collaborative filtering system. To the best of our knowledge, our model is one of the first works addressing this problem.

- We analytically demonstrate that the proposed framework can reduce the system's sensitivity to adversarial gradients exploited by malicious users and retain accuracy in predictions.

- We experimentally verify that our proposed neural collaborative filtering algorithm is less sensitive to the standard C&W attack than approaches that do not deploy our defensive training strategy and retain accuracy comparable to standard models.

The rest of this chapter is structured as follows. In Section 6.2, we discuss related works. We introduce preliminary knowledge of neural collaborative filtering approach and the attack strategy in Section 6.3. The formalization of our method is presented in Section 6.4 followed by analytical discussion in Section 6.5. Empirical verifications are presented in Section 6.6 and the chapter is concluded in Section 6.7.

6.2 Related Work

Recently, deep learning has been widely used in collaborative filtering; the embedding and prediction has become a standard paradigm for collaborative filtering. Representative works include [90, 98]. Specifically, these methods take paired user and item features as inputs. Through feature embedding layers, user and item features are projected into their respective latent space ensuing by different neural network layers and activation functions. [98] utilizes bilinear decoder to map the embedded user and item factors to ratings, and [90] uses a linear classifier with sigmoid activation function. While autoencoder-based approaches [100] take either item or user feature as inputs, they can be treated as a special case of standard NeuCF architecture. [99, 107] share similar architectures to autoencoder-based models but utilizes a stochastic approach. Instead of ReLU or logistic units, [107] utilizes stochastic units with a particular distribution such as binary of Gaussian, while [99] proposes a more advanced model capable of handling high dimensional binary vectors.

Although many algorithms have been developed to improve the performance of recommendation systems, their robustness has not been thoroughly studied. Below we review the existing work on both attack and defense techniques.

In [15–17], researchers study attacks on neighbor-based collaborative filtering. Li et al. [18] studies data poisoning attack on factorization-based recommendation systems, which attacks during the training phase of collaborative filtering. Seminario and Wilson [94] shows

that accuracy and robustness reaches a balance. While they achieve superior performance in impairing existing algorithm's performance, the investigation of the provable defense strategy is limited. A trending topic that attracts huge attention is the vulnerabilities of deep neural networks (DNN) [96, 108, 109, 23]. By adding some imperceptible perturbation to the image, the classification results DNN might vary a lot, owing to the finding in [96] that the smoothness assumption lying under many kernel methods does not hold. Different attacks claim overwhelming success in generating adversarial examples. To name a few, fast gradient sign method (FGSM) [108] generates adversarial perturbations in the direction of loss function gradients; Jacobian Saliency Map Attack (JSMA) [109] estimates the output's sensitivity to input feature dimensions and modifies the most impactive features until it varies the classification; and C&W attack solves an optimization problem to find malicious perturbed images. Through light modifications to these attacks, they can be applied to Neural collaborative filtering algorithm as well.

Compared to the adversarial attack, the defense is a much more challenging task, especially in the area of recommendation systems. Distillation [93] is proposed as a defense to adversarial perturbations through training a student network guided by a teacher [110]. By training a network with soft labels, the distilled network will be prone to avoid over-fitting on training data, and thus be robust to maliciously perturbed inputs lying on the "blind spots" of non-linear networks. It can significantly reduce the effectiveness of adversarial examples on DNNs. However, Carlini and Wagner propose a powerful attack (C&W attack) that can still acquire high success rate on DNNs trained with defensive distillation [14]. Several variations includes [111–113] follow a similar framework to C&W attack. The C&W attack has been recognized as the state-of-the-art white-box attack.

6.3 Preliminaries

We first revisit the matrix factorization (MF) and neural network-based collaborative filtering algorithms, and then review the existing work of adversarial attacks on recommendation systems. Throughout this chapter, we consider histograms of dimension $n+1$. We use bold

upper case letters $\mathbf{X}$ for matrices, and bold lower case letters $\mathbf{x}$ for vectors. Lowercase letters stand for scalars in $\mathbb{R}$.

6.3.1 Collaborative filtering

Consider a recommendation system with N_i users and N_j items and let i, j indicate i-th user and j-th item respectively, we denote the observed rating value as r_{ij} such that $i \in [N_i], j \in [N_j]$ which indicates the interactions between users and items. Let $r_{ij} \in \{1, 2, ..., L\}$ and Ω denote the index set of m observations with $(i, j) \in \Omega$ and $\mathbf{R}$ denote the underlying rating matrix. One would be interested in finding hidden factors of users and items and predict the interactions between them in the following factorization form:

$$r_{ij} = \mathbf{u}_i \mathbf{v}_j^\top, \tag{6.1}$$

where root mean square error is often applied to measure the recovery distance between ground truth ratings and predicted ones. Based on the side information of customers and products, inductive matrix completion (IMC for short) assumes that ratings are generated by multiplying feature vectors of users and items to an unknown low-rank matrix. Based on this fact, rating predictions can be generated by $r_{ij} = \mathbf{x}_i \mathbf{M} \mathbf{y}_j^\top$, where $x_i \in \mathbb{R}^{d_x}$ and $y_j \in \mathbb{R}^{d_y}$ denote the features for i-th user and j-th item respectively, d_x and d_y are the dimension of user and item profile features. $\mathbf{M} \in \mathbb{R}^{d_x \times d_y}$ is the underlying matrix that aligns user and item side features.

Neural collaborative filtering is an extension of classical collaborative filtering models. As shown in Figure 6.1, neural collaborative filtering takes user and item IDs p_i and q_j as input and encodes them to hidden representation as hidden factors. The embedding vectors for users and items are concatenated together and then feed to multi-layer perceptron layers. Denote the concatenated embeddings as π,

$$\pi(p_i, q_j) = \begin{bmatrix} \text{emb}(p_i) \\ \text{emb}(q_j) \end{bmatrix},$$

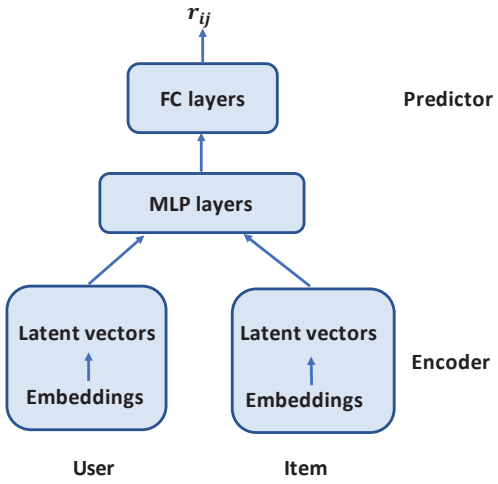


Fig. 6.1 Overview of a neural collaborative filtering architecture. It encodes user and item features, then uses multi-layer perceptrons to predict ratings.

the neural collaborative filtering algorithm can be formulated as:

$$f(p_i, q_j) = \phi_{out}(\phi_X(...\phi_2(\phi_1(\pi(p_i, q_j)))...)), \tag{6.2}$$

where $\phi(\cdot)$ is dense layer with respective activation functions. Since neural collaborative filtering aims to predict implicit feedbacks between users and items, the output layer is a sigmoid activation function and $f(p_i, q_j)$ output the probability of if p_i is relevant to q_j. In this work, we make slight modification to the input by including side information together with user and item IDs. Then the concatenated embedding of users and items becomes

$$\pi(p_i, q_j) = \begin{bmatrix} \text{emb}(p_i^{id}) \\ \text{emb}(p_i^{fea}) \\ \text{emb}(q_j^{id}) \\ \text{emb}(q_j^{fea}) \end{bmatrix}.$$

In the following, we will introduce how to attack user and item features to mislead recommendation systems.

6.3.2 Adversarial attacks

Previous works have shown that certain vulnerabilities of recommendation systems should be taken into consideration [18, 15, 16]. We consider the situation that adversary attacks a given neural collaborative filtering model to downgrade its overall performance. We use C&W attack [14], one of the strongest white-box attack methods, to show how adversary craft fictitious user or item data to contaminate the model performance in the testing phase.

Neural collaborative filtering systems take both user and item features as inputs. To jointly perturb both features, the adversary adds small perturbations $(\delta p_i, \delta q_j)$ into the input features such that $F(p_i + \delta p_i, q_j + \delta q_j) \neq r_{ij}$. C&W attack generates adversarial perturbations by solving an optimization problem. Due to the binary nature of implicit feedback prediction problem, we slightly change the adversarial goal of C&W attack to make the NeuCF model make less confident predictions, which is formulated as:

$$\begin{aligned} &\min_{\delta p_i, \delta q_j} && \|\delta p_i\| + \|\delta q_j\| \\ &\text{s.t.} && f(p_i + \delta p_i, q_j + \delta q_j) \leq 0.5. \end{aligned} \tag{6.3}$$

This problem is equivalent to the following form:

$$\min_{\delta p_i, \delta q_j} \|\delta p_i\| + \|\delta q_j\| + c \cdot f(p_i + \delta p_i, q_j + \delta q_j) \tag{6.4}$$

where c is a suitable constant. The norm in the objective is specified depending on the perturbation shapes that adversary chooses. The perturbations can be small variations on all dimensions or sufficient alternations on a selected group of feature dimensions. Therefore, L_1, L_2 and L_∞ norm can be employed on the objective in Equation (6.3). Since it is easier to perturb the user's features than the item's in practice, it will not be necessary to perturb p_i and q_j at the same time. Notice that C&W attack is a white-box attack algorithm in which both the architecture and weights in F are known. An adversary can thus solve Equation (6.4) by optimization methods.

6.4 Stage-wise Hints Training for Robust Neural Collaborative Filtering

Our approach contains two main components: stage-wise hints training and randomness. We design a teacher-student architecture based on knowledge transfer. The student model is similar to the structure of the teacher and but noise layers are deployed before each dense layer. First, we train basic collaborative filtering as a teacher. Second, we train a student model stage-wisely with different hints, which are intermediate representations from the teacher. Last, we train the whole student network with knowledge distillation.

6.4.1 Knowledge transfer from a teacher

Inspired by distillation network [104], we start our method with a basic model named as a teacher. We follow the same structure as used in [90] and include side information to construct a feature-based neural collaborative filtering network that takes user and item information as input and output implicit feedback of whether (user, item) pairs have interaction before.

At first, we choose a hidden layer of the encoder of the teacher and obtain the output of this layer and use this output as the knowledge for transferring. The knowledge is also called hints because it allows a student network to mimic it [105]. In this way, the student network is guided by the knowledge from the teacher. The knowledge is constructed by using the hints module, defined as

$$h_{k_i} = \mathrm{emb}_H(p_i, q_j, \theta_H), \tag{6.5}$$

where θ_H denotes the parameters for hints module which is a subnet of teacher network up to the respective hidden layer, and emb_H indicates the corresponding deep nested functions. h_{k_i} denotes hints knowledge and k_i is the index of intermediate layer.

We also have a student model, which has a similar architecture with the teacher but it can be a thinner and deeper network than the teacher. We transfer the knowledge h_{k_i} from the teacher to the student. We use this knowledge as the output of a hidden layer of the encoder of the student to control how to train the student. Thus the training of the student network is

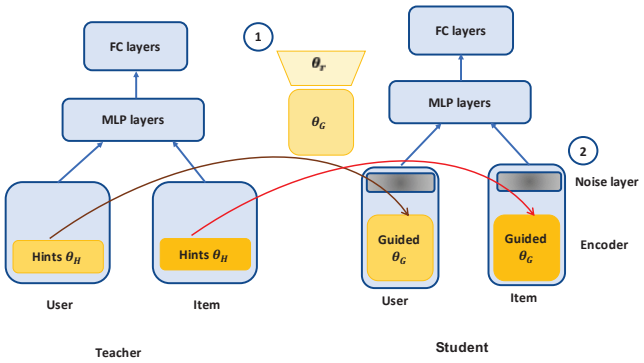


Fig. 6.2 The graph illustration of student training procedure: (i) Step 1 is the hints training process which aligns the output between intermediate layers of teachers and students; θ_r denotes the regression parameters that align the output dimension of teacher and students; (ii) Step 2 is the knowledge distillation training process with noise layers added before MLP layers.

guided by this knowledge. When training the student, there is a guided module in the student. This part is controlled by h_{k_i}, that is

$$\mathscr{L}(\theta_G, \theta_r) = \frac{1}{2}||h_{k_i} - \sigma(\theta_r \cdot \text{emb}_G(p_i, q_j, \theta_G) + b)||. \tag{6.6}$$

We add a regressor parameterized by θ_r on top of the guided layer of the student in case that the sizes of the student and teacher are different.

Figure 6.2 shows the hints transfer from embedding layers of a teacher to a student before feeding to Multi-layer perceptron (MLP). The teacher has a larger dimension for user and item's embeddings. Thus θ_r is introduced to map the student model's embedding to the space of the teacher model's embedding.

Finally, based on pre-trained parameters in guided layers, we train the parameters of the whole student network with knowledge distillation. The distilled model is trained by the following objective function:

$$\min_{\theta_F} -\frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} \sum_{l\in\{1,\ldots,L\}} F_l \log F_l^d(p_i, q_j), \tag{6.7}$$

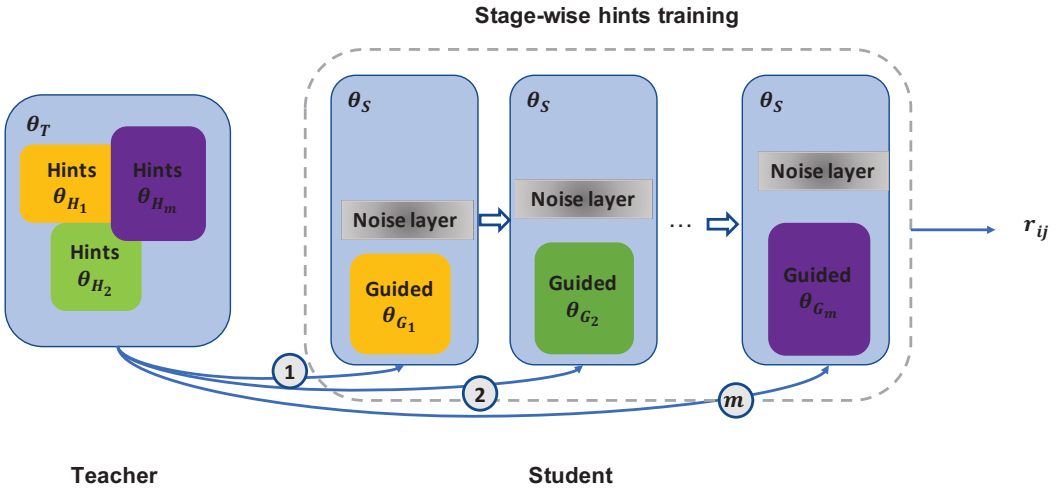


Fig. 6.3 The stage-wise hints training framework. Steps 1 to m are the hints training routine that is illustrated in Figure 6.2 but need to be repeated m times for m different hints modules from the teacher.

where F is the output of softmax layer of teacher model, F^d indicates that of student model and θ_F is the whole set of parameters for F. $\{1,...,L\}$ denotes the label set. As we predict implicit feedbacks, 1 indicates p_i is relevant to q_j and 0 otherwise [90]. We have binary output and the objective function for distillation is structured as following:

$$-\frac{1}{|\Omega|}\sum_{(i,j)\in\Omega}(1-F_1)\log(1-F_1^d(p_i,q_j))F_1\log F_1^d(p_i,q_j), \tag{6.8}$$

If the amplitude of adversarial gradients is high, executing cost of adversary will be low, since altering the input, the variations of outputs will vary a lot. To this end, a natural idea to protect the model from being attacked is to reduce the variations around inputs.

6.4.2 Stage-wise hints training of the student model

Rather than learn from only the softmax layer or an intermediate layer of the teacher, we train the student model stage-wisely based on hints from different layers of the teacher. We allow different hints obtained from the same teacher, which is different from the usage of distillation in previous works. Based on Equation (6.5), we choose different size of modules and then obtain different hints $h_1,\cdots,h_m$ corresponding to these modules respectively. For

Algorithm 5 A sketch of stage-wise hints training based on a pre-trained teacher model

Input: teacher network θ_T, student network θ_S, $g_1, ..., g_m$, $h_1, ..., h_m$
Output: student model acquainted of transferred knowledge θ_S^*

1: **for** m hints modules **do**
2: $\theta_H \leftarrow \{\theta_T^1, ..., \theta_T^{h_i}\}$
3: $\theta_G \Leftarrow \{\theta_S^1, ..., \theta_S^{g_i}\}$
4: initialize θ_r
5: Choose a intermediate layer from the teacher T and generate a hint h_m
6: Train the guided layers of the student using h_m
$\theta_G^* = \arg\min_{\theta_G} \mathscr{L}(\theta_G, \theta_r)$
7: $\{\theta_S^1, ..., \theta_S^{g_i}\} = \{\theta_G^{*1}, ..., \theta_G^{*g}\}$
8: **end for**

stage-wise training, the guided module in each stage in the student model is different and follows the rule from shallow to deep.

We summarize the stage-wise training regime in Algorithm 5. By setting a standard collaborative filtering model as the teacher, we choose m hints modules from teacher model and m guided modules from the student model. Then we update the student network by m stages. We train the student model from shallow to deep, which implies that $g_1 < g_2 < ... < g_m$ and $h_1 < h_2 < ... < h_m$. After stage-wise hints training, the student model is acquainted with knowledge from the teacher and can predict the intermediate representations of the teacher model.

6.4.3 Randomness by injecting noises

As mentioned in Section 6.4.1, to protect the model from being attacked one need to reduce the variations around inputs. This is not easy to be controlled during white-box attacks. On the contrary, we train the collaborative filtering model that is robust to variations of inputs. We consider improving the robustness of neural networks by adding randomness into the network structure. In particular, we introduce a new "noise layer" that fuses input vector with a randomly generated noise, defined as

$$\theta_S^i \leftarrow \theta_S^i + \varepsilon, \quad \varepsilon \sim \mathscr{N}(0, \delta^2), \tag{6.9}$$

where θ_S^i is i-th layer student model. If $\varepsilon = 0$, no noise is introduced. If $\varepsilon > 0$, through minimizing Equation (6.6), we are looking for a student model that spans the same space of activations of teacher model and is robust to perturbations as well.

The noise introduced in training time is crucial for increasing the target model's robustness to perturbations. Through hints training strategy, the guided layer will be robust to the perturbations as well as produce similar results to the hints layer. The final step is training the student model by knowledge distillation strategy. Figure 6.3 shows the framework of the proposed method. The teacher model generates several useful hints and distills the knowledge to student model by different stages.

6.4.4 Relation to other works

The stage-wise hints training with knowledge distillation can be seen as a particular form of curriculum learning [114]. By choosing a proper sequence of tasks to learning, curriculum learning is proved to accelerate the convergence and improve generalization. In this work, we use a teacher model as a guide and train the student model from simple to complex ones, which would considerably ease training [115]. [105] adopts a single stage hints training framework which can be seen as a special case of our method. While [105] uses hints to guide the training of a thinner and deeper network, we use hints training together with randomness to increase models' robustness to perturbations.

Except for the training of teacher model, our method does not introduce computation overhead on training student models. Though we train student models in multi-stages, the overall epochs of training are not increased much. If a well-performed neural collaborative filtering model has already existed, the cost for training our robust neural collaborative filtering model can be further reduced.

6.5 Analysis of Stage-Wise Hints Training Strategy

In this section, we first establish the *robustness* definition for neural collaborative filtering model and illustrate the vulnerability of NeuCF. Then we analytically investigate the impact

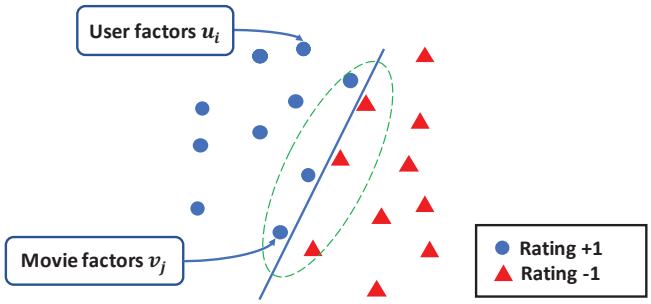


Fig. 6.4 The graph illustration of vulnerabilities of the collaborative filtering system

of our multiple hints training strategy to the neural collaborative filtering system from two perspectives: (i) model's sensitivity to adversarial perturbations, (ii) generalizability of neural collaborative filtering model.

6.5.1 Robustness of neural collaborative filtering system

The *robustness* of a neural collaborative algorithm to adversarial perturbations is defined as its capacity against perturbations. A robust neural collaborative filtering algorithm should (i) show good prediction accuracy in the training set and testing set as well as (ii) model a smooth neural collaborative filtering machine that can make stable predictions around the neighborhood of a legitimate user. The definition of *robustness* introduced in [93] can be slightly modified to the collaborative filtering here, which is:

$$\rho_{adv}(F) = E_{\mu}[\Delta_{adv}(p_i, q_j, F)] \tag{6.10}$$

where inputs (i, j) are drawn from an unknown distribution μ that neural collaborative filtering algorithm attempt to model with function F. $\Delta_{adv}(i, j, F)$ is defined as the minimal perturbations needed to wrongly predict sample (i, j):

$$\Delta_{adv}(i, j, F) = \arg\min_{\delta p_i, \delta q_j} \{\|\delta p_i\| + \|\delta q_j\| : F(p_i + \delta p_i, q_j + \delta q_j) \neq F(p_i, q_j)\} \tag{6.11}$$

where $\|\cdot\|$ is the norm that need to be specified according to systems' requirements. A robust neural collaborative filtering system requires higher average minimum perturbations to wrongly predict an example from distribution μ.

To illustrate the vulnerability of neural collaborative filtering algorithm, we consider using a bilinear decoder [98] to predict a user's preference on an item. Assume user and item features are embedded into $\mathbf{u}_i = \text{emb}_{user}(p_i), \mathbf{v}_j = \text{emb}_{item}(q_j)$, the prediction function is as follows:

$$p(r_{ij} = l) = \frac{e^{\mathbf{u}_i Q_l \mathbf{v}_j^\top}}{\sum_{l=1}^{L} e^{\mathbf{u}_i Q_l \mathbf{v}_j^\top}}. \tag{6.12}$$

If user feature factor U is fixed, the collaborative filtering problem can be taken as a linear classification problem for each item v_j. In Figure 6.4, we show a possible solution for given positive and negative ratings in red and blue color. If we treat item factor v_j as the classifier indicated by the straight line in Figure 6.4 and the blue dots and red triangles as user feature vectors, then predicting their interactions labeled as "+1" or "-1" can be treated as a binary classification problem. In Figure 6.4, although this is a correct solution which classifies both positive and negative ratings correctly, we can see that it is vulnerable to attacks since six examples in green eclipse are very close to the decision boundary. By slightly perturb the input user feature p_i or item features q_j, the six points in the green eclipse will be wrongly predicted.

Note that we consider a linear case to make the intuition simple to be understood. In other cases, the hyperplane would be more complexed. Figure 6.4 indicates that to let positively (negatively) labeled user have a larger neighborhood with same labels will lead to a more robust neural collaborative filtering system. During training the NeuCF model, our goal is to converge to a function F^* that is against adversarial perturbations and generalize well. According to the universality theorem proposed in [116] for neural networks, the existence of such F^* is guaranteed. The hints training technique is thus deployed in the network training and helping it converge to the optimal F^* without getting stuck into local optima.

6.5.2 Impact on model sensitivity to inputs

Through our hints training strategy, the model's sensitivity to malicious perturbations is reduced. We then derive the distilled model's sensitivity to input features. Denote $g(p_i, q_j) = \sum_{h=1}^{L} e^{z_h(p_i,q_j)/T}$, we then have that

$$\begin{aligned}
&\frac{\partial F_l(p_i,q_j)}{\partial p_{ik}} = \frac{\partial}{\partial p_{ik}}\left(\frac{e^{z_l/T}}{\sum_{h=1}^{L} e^{z_h/T}}\right) \\
=&\frac{1}{g^2(p_i,q_j)}\left(\frac{\partial e^{z_l/T}}{\partial p_{ik}} g(p_i,q_j) - e^{z_l/T}\frac{\partial g(p_i,q_j)}{\partial p_{ik}}\right) \\
=&\frac{1}{g^2(p_i,q_j)}\frac{e^{z_l/T}}{T}\left(\sum_{h=1}^{L}\frac{\partial z_l}{\partial p_{ik}} e^{z_h/T} - \sum_{h=1}^{L}\frac{\partial z_h}{\partial p_{ik}} e^{z_h/T}\right) \\
=&\frac{1}{T}\frac{e^{z_l/T}}{g^2(p_i,q_j)}\left(\sum_{h=1}^{L}\left(\frac{\partial z_l}{\partial p_{ik}} - \frac{\partial z_h}{\partial p_{ik}}\right) e^{z_h/T}\right).
\end{aligned} \tag{6.13}$$

The last equation shows that Jacobian decreases with increasing temperature T because Jacobian is inversely proportional to T, and outputs of last hidden layer are divided by T before exponentiated.

This analysis shows that by using high temperature T in the network, the amplitude of adversarial gradients change is reduced with the variation of inputs. It is important to notice that the high temperature T only changes the amplitude of logit but not the relative orders of these logits, which does not influence the prediction accuracy. At test time, temperature T is set to $T = 1$, which will lead to more discrete rating predictions.

If we have a look into the hints training formulated by Equation (6.6), hints from teacher network play as a regularization term that constrains the guided module make the correct prediction and being robust to random perturbations simultaneously.

The proposed training strategy has several superiorities. Firstly, the distillation technique only smooths the amplitude of logits but not change the relative order of different logits, thus does not reduce the model's accuracy. Secondly, the distillation technique has a low impact on the model's architecture while proposing a new architecture requires much effort in analyzing its effectiveness. Furthermore, the slight modification to the training procedure does not reduce the model's speed in the testing phase. By introducing the distilled model, a

low computation cost of training a teacher model is added to the training phase, but it is a fixed cost and is acceptable.

6.5.3 Impact on model generalizability

When training a student network with noise added to its guiding layer, we are optimizing the following problem:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2)}\mathscr{L}(F_{\theta,\varepsilon}(p_i,q_j),r_{ij}). \tag{6.14}$$

Based on the obtained parameters θ, we make predictions for ratings using the following form:

$$\hat{r}_{ij} = \arg\max \mathbb{F}_{\theta}(p_i,q_j). \tag{6.15}$$

To explain why minimizing Equation (6.14) is able to yield similar prediction to original network, according to [106], we show that minimizing Equation (6.14) is equal to minimize the upper bound of the inference loss. Specifically, we have:

$$\begin{aligned}
&\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2)}\mathscr{L}(F_{\theta,\varepsilon}(p_i,q_j),r_{ij}) \\
=&-\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2)}\log F_{\theta,\varepsilon}(p_i,q_j)\cdot[r_{ij}] \\
\geq&-\log\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2)}F_{\theta,\varepsilon}(p_i,q_j)\cdot[r_{ij}] \\
\geq&-\log\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2)}F_{\theta,\varepsilon}(p_i,q_j)\cdot[\hat{r}_{ij}],
\end{aligned} \tag{6.16}$$

where the first inequation holds because of Jensen's inequality and second inequation comes from Equation (6.15). This validates that minimizing Equation (6.14) is equal to minimizing the upper bound of $-\log F_{\theta,\varepsilon}(p_i,q_j)[\hat{r}_{ij}]$ which is the inference loss.

6.6 Experiments

In the following, we empirically evaluate the effectiveness of the proposed robust training strategy against adversarial attacks.

Table 6.1 Overview of architectures of the teacher and student model. The second last dense layer determines the latent factor for the neural collaborative filtering model which is 64 and 32 for teacher and student model in this table.

Layer Type	Teacher	Student
Embedding	64 units	32 units
Embedding	64 units	32 units
Dense	64 units	32 units
Dense	64 units	32 units
Concatenate layer	256 units	128 units
ReLU Dense Layer	128 units	64 units
ReLU Dense Layer	64 units	32 units
ReLU Dense Layer	1 unit	1 unit
Sigmoid Activation	1 unit	1 unit

6.6.1 Dataset information

Since reviewing items is an onerous process in real recommendation systems, items that have more ratings from users are implicitly more popular than these that own fewer ratings. To simplify the discussion of our robust training strategy, we focus on this binary implicit feedback prediction problem. In this case, the untargeted and targeted attacks are equal. The attack aims to downgrade the overall performance of the collaborative filtering system. We experiment with two publicly available dataset MovieLens-1M and MovieLens-100K [68]. MovieLens-1M[1] includes 6,040 users, 3,952 movies and 1,000,209 anonymous ratings among them. We follow the setting in [90] which use MovieLens to investigate the implicit signal from explicit rating feedback. The one million ratings are transformed into implicit feedback indicating if the user has rated the corresponding item. This dataset include information for both users and items, such as “gender”, “age”, “occupation” for users and “genre”, “release date” for movies.

MovieLens-100K[2] includes 100,000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies. This dataset contains the same kind of side information as MovieLens-1M, and we testify the collaborative filtering model’s performance on predicting implicit feedbacks between users and items.

[1] http://grouplens.org/datasets/movielens/1m/
[2] https://grouplens.org/datasets/movielens/100k/

Table 6.2 Overview of hyperparameters for training the model. For two stage hints training of student model, we train the model by 10 epochs in each stage and 10 epocs in final knowledge distillation training.

Parameter	Teacher	Student
Learning Rate	0.001	0.001
Optimizer	Adam	Adam
Batch Size	128	128
Total Epochs	20	30
Temperature	10	10

6.6.2 Overview of the experimental setup

To evaluate the proposed hints training strategy, we will compare its influence on model's generalizability to testing examples and its robustness to malicious attacks. For the neural collaborative filtering model, we develop a feature-based neural collaborative filtering model (FNCF) based on multi-layer percepetron model (MLP[3]). We follow a similar structure of that used in [90] and deploy a model with three dense layers. For latent factor as 64 and 32 of teacher and student model, the details of the architecture is shown in Table 6.1. The teacher and student model share similar structure but the teacher model has larger latent factor which is the input dimensionality of the last dense layer. Both of them takes four inputs, which are user IDs, item IDs, user side information and item side information. The temperature for distillation is set to 10 through all the experiments. The set of hyper-parameters are presented in Table 6.2.

Baseline setup

For the baseline of robust training strategy, we adopt *distillation as a defense* studied in [93]. The robust training strategy is listed as follows:

- FNCF-Distill trains a neural collaborative filtering model with knowledge distillation;
- FNCF-Single is a special case of our stage-wise hints training under which only the middle layer's representations of teacher model serve as hints to train the student model

[3] https://github.com/hexiangnan/neural_collaborative_filtering

which is 128 units of a ReLU dense layer in Table 6.1. Distillation training is deployed after single hint training.

- FNCF-Multi trains a model with two hints module and knowledge distillation. The hints modules produce output at first and second ReLU dense layer which is at length of 128 and 64 respectively as shown in Table 6.1.

To train the specified model, we use Keras [117] with Tensorflow [118] as the backend, which simplifies the implementation of architectures with multi-input and multi-output. The adversarial sample crafting is implemented by Tensorflow [118] based on pre-trained Keras models.

Evaluation criteria

To evaluate the performance of predictions on implicit feedbacks, we adopt the popular *leave-one-out* evaluation [90, 119] which construct the validation and test set by holding out one record for each user respectively. To judge the ranking list generated by collaborative filtering algorithms, we utilize Hit Ratio (HR) which is a recall-based metric and Normalized Discounted Cumulative Gain (NDCG) [120] with the rank list length set to be $K = 10$. HR is formulated by:

$$HR@K = \frac{Number\ of\ hits@K}{N}, \tag{6.17}$$

where N is the size of test set. Since HR@K is a recall-based metric, it does not reflect the model's capability of getting the top ranks correct. NDCG assigns higher score to results on top ranks, which is calculated by:

$$NDCG@K = Z_K \sum_{i=1}^{K} \frac{2^{r_i} - 1}{\log_2(i+1)}. \tag{6.18}$$

Adversarial example crafting

The adversarial goal of the adversary is to alter the input (p_i, q_j) of a neural collaborative filtering model F to make the model F's prediction deviate from its ground truth value.

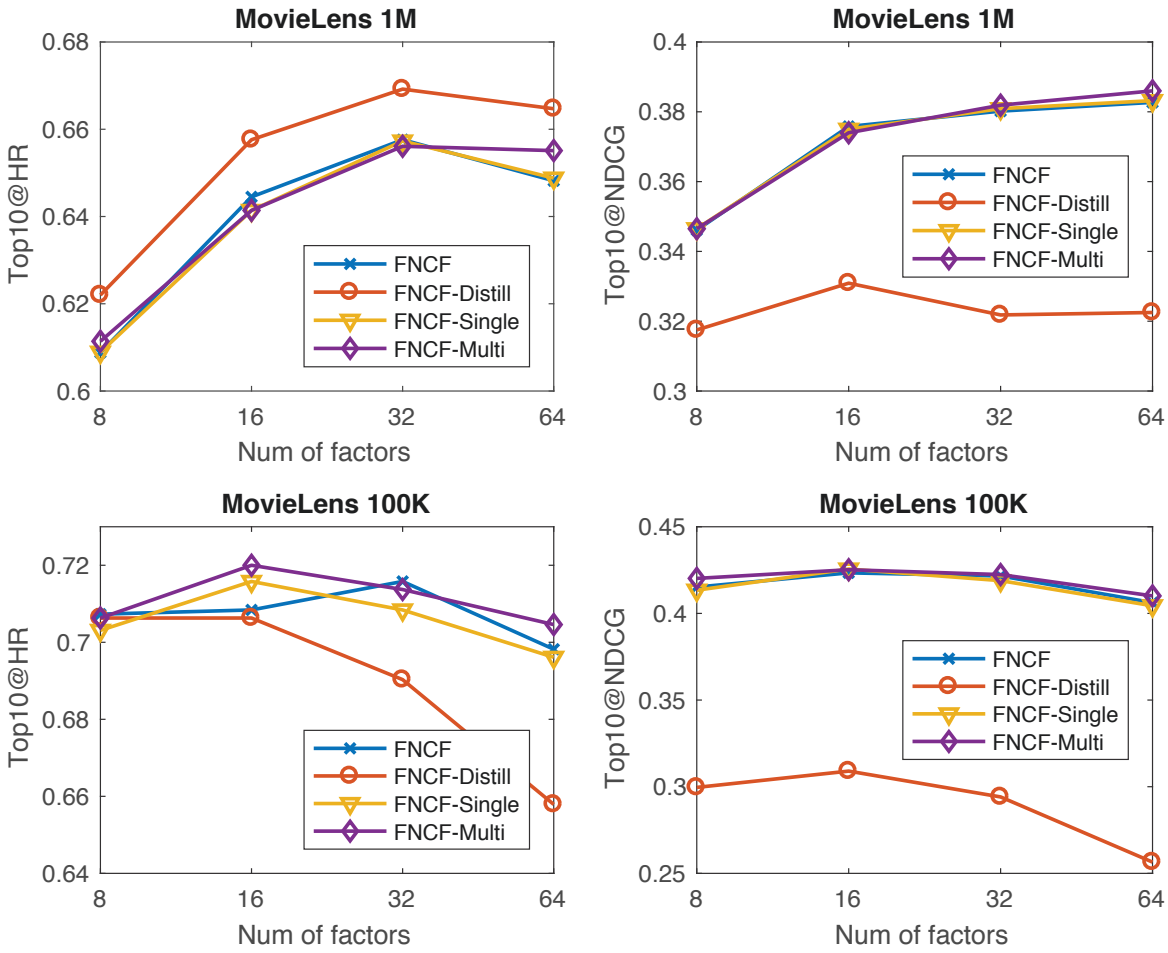


Fig. 6.5 Comparisons of FNCF with FNCF-Single, FNCF-Multi and FNCF-Distill at temperature $T = 10$ and noise level $\sigma = 0.05$.

We use the C&W attack[4], with the L_2 norm regularization on perturbation variable δ and make slight changes to adapt to our model with paired input. Note that as instructed by the C&W attack, we take the output of logits layer to implement attacks, which is possible to circumvent the distillation as the defense.

6.6.3 Influence on model's generalizability

In this set of experiments, we compare FNCF's performance with three training strategies under a different number of hidden factors. As shown in Figure 6.5, we observe that FNCF-Multi and FNCF-Single reach comparable performance to FNCF on both datasets. This

[4]https://github.com/carlini/nn_robust_attacks

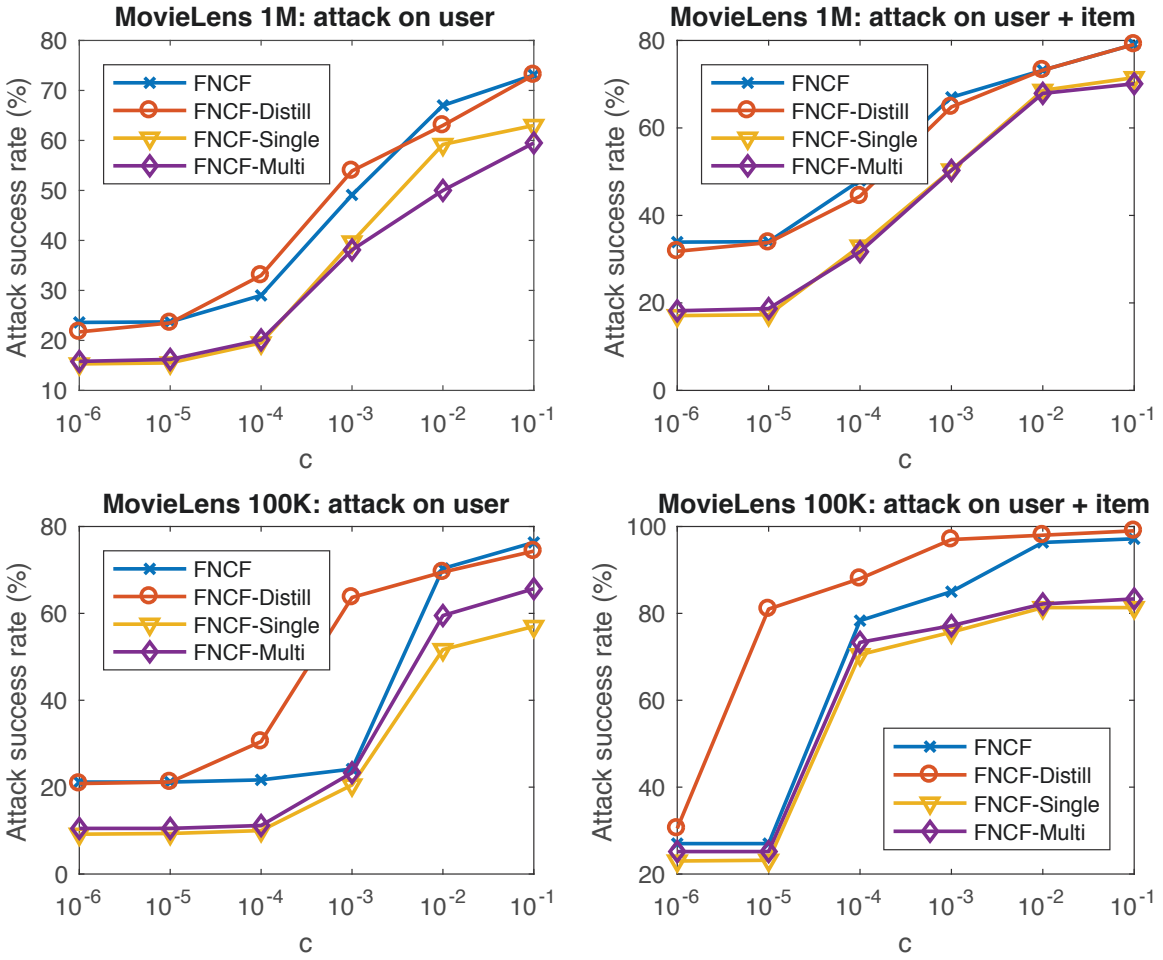


Fig. 6.6 Attack success rate on FNCF, FNCF-Single, FNCF-Multi and FNCF-Distill at temperature $T = 10$ and noise level $\sigma = 0.05$.

Table 6.3 Different models' performance under different noise levels at temperature $T = 10$.

	MovieLens 1M									
	Top10@HR					Top10@NDCG				
Noise level	0	0.01	0.05	0.1	0.2	0	0.01	0.05	0.1	0.2
FNCF	0.6576	0.6576	0.6576	0.6576	0.6576	0.3802	0.3802	0.3802	0.3802	0.3802
FNCF-Single	0.6579	0.6575	0.6575	0.6584	0.6591	0.3797	0.3814	0.3821	0.3818	0.3818
FNCF-Multi	0.6548	0.6538	0.6571	0.6573	0.6591	0.3805	0.3799	0.3824	0.3823	0.3835
	MovieLens 100K									
	Top10@HR					Top10@NDCG				
Noise level	0	0.01	0.05	0.1	0.2	0	0.01	0.05	0.1	0.2
FNCF	0.7158	0.7158	0.7158	0.7158	0.7158	0.4216	0.4216	0.4216	0.4216	0.4216
FNCF-Single	0.7116	0.7126	0.7137	0.7147	0.7126	0.4201	0.4189	0.4216	0.4203	0.4179
FNCF-Multi	0.7126	0.7137	0.7094	0.7147	0.7094	0.4200	0.4214	0.4188	0.4198	0.4195

Table 6.4 Model's robustness against adversarial perturbations under different noise levels at temperature $T = 10$.

	MovieLens 1M									
	Success rate					L_2 distortion				
Noise level	0	0.01	0.05	0.1	0.2	0	0.01	0.05	0.1	0.2
FNCF	0.7320	0.7320	0.7320	0.7320	0.7320	0.8415	0.8415	0.8415	0.8415	0.8415
FNCF-Distill	0.7210	0.7210	0.7210	0.7210	0.7210	0.9790	0.9790	0.9790	0.9790	0.9790
FNCF-Single	0.7870	0.7250	0.6730	0.6320	0.6000	0.8446	0.8869	0.8210	0.7715	0.7808
FNCF-Multi	0.7810	0.7930	0.6420	0.6160	0.6120	0.8676	0.9155	0.8502	0.7318	0.7341
	MovieLens 100K									
	Success rate					L_2 distortion				
Noise level	0	0.01	0.05	0.1	0.2	0	0.01	0.05	0.1	0.2
FNCF	0.7783	0.7783	0.7783	0.7783	0.7783	1.0716	1.0716	1.0716	1.0716	1.0716
FNCF-Distill	0.7653	0.7653	0.7653	0.7653	0.7653	1.1173	1.1173	1.1173	1.1173	1.1173
FNCF-Single	0.7300	0.7050	0.6183	0.5800	0.5750	1.1183	1.0831	0.9994	0.9746	1.0004
FNCF-Multi	0.7200	0.7100	0.7000	0.6583	0.6533	1.0821	1.0645	1.0622	1.0313	1.0379

verifies that the noise layer in FNCF-Multi and FNCF-Single will not introduce significant decreases in model's generalization. FNCF-Distill achieves disparate performance than the other three methods. Although FNCF-Distill has a significant improvement on HR@K on MovieLens-1M, it performs poorly in terms of NDCG. This reflects that FNCF-Distill cannot make confident predictions on users and items' relations.

Furthermore, we quantify the noise's influence on the model's generalizability. In Table 6.3, we vary the noise levels and observe that there are no significant variations between FNCF and FNCF-single, FNCF-Multi. Introducing noises into FNCF-Single renders a few variations to the prediction performance, which is acceptable. One can claim that the variations introduced by hints training process are moderate and can be neglected.

6.6.4 Impact on adversarial perturbations

In this set of experiments, we evaluate our training strategy's influence on malicious attacks by adversaries. For the examples to be attacked, we first select the examples that hit the top-10 list in predictions which are about 3900 for MovieLens-1M and 675 for MovieLens-100K. Then we randomly select 1000 examples from MovieLens-1M and all 675 examples from MovieLens-100K as the examples to be changed. We implement C&W attack on "user" and "user+item" which is discussed in Section 6.3.2. For FNCF-Single and FNCF-Multi,

the noise level is set to $\sigma = 0.05$. We report the success rate of C&W attack on different models under different const c specified in Equation (6.4). From Figure 6.6, we observe that FNCF-Distill does not show effectiveness on resisting adversarial attacks. FNCF-Single and FNCF-Multi report lower attack success rates compared to baselines. This verifies the effectiveness of the proposed hints training algorithm.

6.6.5 Model's robustness under different noise levels

In this battery of experiments, we evaluate the model's robustness against malicious perturbations. The robustness measurement is defined in Equation (6.10). To exhaustively search over the entire input space is impossible. We average the distortions on examples that are successfully attacked. As shown in Table 6.4, with the increase of noise levels, the success rate of attack on both FNCF-Single and FNCF-Multi decreases, which validate the effectiveness of our robust training algorithm. The resulting L_2 distortion is also reported in Table 6.4. Notice that when no noise is introduced in $\sigma = 0$, FNCF-Single is more robust than FNCF-Distill. We credit this improvement to the hints training step that regularizes the student network.

6.7 Conclusions

We have proposed a robust training strategy for neural collaborative filtering systems to improve its robustness to adversarial perturbations. We first use multiple hints from different layers of the teacher model to guide the training of different layers of the student network. Then we inject noises in the intermediate layers of student networks in the training time to increase the model's robustness to feature perturbations. By aligning the output of teacher and student networks, we obtain the benefits that recommendation performance is not degraded, and the model's robustness is improved. Our experiments on benchmark datasets emphasize that defensive method with stage-wise hints training and noise injection can still work even under current strong attackers.

Chapter 7

Conclusions

In the final chapter, we reflect on the collective findings and review the main topics of this dissertation. We have proposed new recommendation models to exploit major features, including review comments, constrained user behaviours and user/item demographics. This has allowed us to progress the understanding of the vulnerability of recommendation systems and present a practical strategy for enhancing their robustness.

In Chapter 2, we studied a privileged matrix factorization method for collaborative filtering. We used review texts that are in companion with rating values to assist the learning of user and item factors. In contrast to existing approaches that use review texts to describe either user or item factors, we noticed that a review comment is related to a specific pair of user and item factors concurrently and it is used in the learning of both. We took review texts as privileged information for matrix factorization and express them here as feature vectors through a Paragraph Vector transformation. The discrepancies between predictions and ground truth ratings are modeled by a privileged function dependent on the review vector. Testing our model on several challenging Amazon review datasets produced a better performance. While the remarkable strength of our method is discussed above, the weakness shared with the other collaborative filtering methods is an implicit assumption on complete review comments. In practice, not every rating comes with a piece of review comment.

We next examined – in Chapter 3 – the collaborative budget allocation problem in a sharing economy system. To measure the distance between two histogram data points, we

proposed using either Riemannian metric on simplex or Aitchison mappings to preserve the geometric relationship between histogram data points. To intensify the model's robustness in real applications, relaxed recovery constraints are studied. The proposed methods are optimized by Riemannian conjugate gradients on the simplex. We verify our models' superiority over baseline methods and compare the difference between distance metrics introduced in this chapter. We concluded that (i) to learn the metric based on the training data tends to deliver better performance; (ii) to relax the exact recovery constraint can improve the performance of budget allocation when preference value observations are corrupted by noise.

Chapter 4 is devoted to a study of the robust inductive matrix completion method for recommendation problems in this work. These completion methods explicitly explore user side information such as "age" and "gender", along with items such as "genre" and "release date" to boost recommendation performance. Considering that noise may be present in some attributes, how to eliminate its negative influence in recommendation algorithms is a key problem. We revisited the common low-rank principle in inductive matrix completion modeling and introduced a group sparse part to identify the noisy user and item attributes. We theorized that, with perfect side features, our RIMC method could recover the ground truth latent matrices and, with informative but not perfect side features, RIMC requires fewer observations to achieve ε-recovery than methods that do not use side information. We validated our model on the MovieLens and NCI Challenge datasets and achieved a promising performance.

Chapter 5 introduces an initial study on evaluating the robustness of deep learning-based recommendation systems. Our research showed that state-of-the-art recommendation models are vulnerable to adversarial attacks. To perform efficient attacks on sparse input features, we proposed a two-stage method which begins by optimizing over the continuous embedding space before projecting the altered embedding vectors back to ground-truth to shrink the set of attacked categorical variables. Our experiments demonstrated that the proposed algorithm was highly successful in breaking the state-of-the-art recommendation systems; by altering one feature in the MovieLens data, our attack entirely defeats Wide&Deep model from Google with a 100% success rate.

This thesis concludes, in Chapter 6, with our proposing a robust training strategy for neural collaborative filtering systems to improve their robustness to adversarial perturbations. We first used multiple hints from different layers of the teacher model to guide the training of different layers of the student network. Then we injected noises in the intermediate layers of student networks within the training time to increase the model's robustness to feature perturbations. By aligning the output of teacher and student networks, we reached a balance where the recommendation performance was not degraded and the model's robustness is improved. Our experiments on benchmark datasets demonstrate that the proposed stage-wise hints training strategy improves recommendation model's resistance to current strong attackers.

References

[1] Julian Mcauley. Hidden factors and hidden topics : Understanding rating dimensions with review text. In *RecSys*, 2013. ISBN 9781450324090.

[2] Amjad Almahairi, Kyle Kastner, Kyunghyun Cho, and Aaron Courville. Learning Distributed Representations from Reviews for Collaborative Filtering. *RecSys*, pages 147–154, 2015. doi: 10.1145/2792838.2800192. URL http://dl.acm.org/citation.cfm?doid=2792838.2800192.

[3] Guang Ling, Michael R Lyu, and Irwin King. Ratings Meet Reviews , a Combined Approach to Recommend. *RecSys*, pages 105–112, 2014.

[4] Yali Du, Chang Xu, and Dacheng Tao. Privileged matrix factorization for collaborative filtering. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1610–1616. AAAI Press, 2017. doi: 10.24963/ijcai.2017/223.

[5] Yanfeng Sun, Junbin Gao, Xia Hong, Bamdev Mishra, and Baocai Yin. Heterogeneous tensor decomposition for clustering via manifold optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):476–489, 2016.

[6] Yali Du, Chang Xu, and Dacheng Tao. Collaborative rating allocation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 1617–1623, 2017. doi: 10.24963/ijcai.2017/224.

[7] Yali Du, Chang Xu, and Dacheng Tao. Matrix factorization for collaborative budget allocation. *IEEE Transactions on Automation Science and Engineering*, (99):1–12, 2018.

[8] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. Svdfeature: a toolkit for feature-based collaborative filtering. *Journal of Machine Learning Research*, 13 (Dec):3619–3622, 2012.

[9] Nagarajan Natarajan and Inderjit S Dhillon. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics*, 30(12):i60–i68, 2014.

[10] Prateek Jain and IS Dhillon. Provable inductive matrix completion. *arXiv preprint arXiv:1306.0626*, pages 1–22, 2013. URL http://arxiv.org/abs/1306.0626.

[11] Miao Xu, Rong Jin, and Zhi-Hua Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in Neural Information Processing Systems*, pages 2301–2309, 2013.

[12] Si Si, Kai-Yang Chiang, Cho-Jui Hsieh, Nikhil Rao, and Inderjit S Dhillon. Goal-directed inductive matrix completion. In *ACM International Conference on Knowledge Discovery and Data Mining*, 2016.

[13] Kai-Yang Chiang, Cho-Jui Hsieh, and Inderjit S Dhillon. Matrix completion with noisy side information. In *Advances in Neural Information Processing Systems*, pages 3447–3455, 2015.

[14] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[15] Michael P O'Mahony, Neil J Hurley, and Guenole CM Silvestre. Promoting recommendations: An attack on collaborative filtering. In *International Conference on Database and Expert Systems Applications*, pages 494–503. Springer, 2002.

[16] Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. Effective attack models for shilling item-based collaborative filtering systems. In *Proceedings of the 2005 WebKDD Workshop*, volume 2005, pages 13–21, 2005.

[17] Shyong K Lam and John Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th International Conference on World Wide Web*, pages 393–402. ACM, 2004.

[18] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in Neural Information Processing Systems*, pages 1885–1893, 2016.

[19] Yudong Chen, Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust matrix completion and corrupted columns. In *Proceedings of the 28th International Conference on Machine Learning*, pages 873–880, 2011.

[20] Yudong Chen, Ali Jalali, Sujay Sanghavi, and Constantine Caramanis. Low-rank matrix recovery from errors and erasures. *IEEE Transactions on Information Theory*, 59(7):4324–4337, 2013.

[21] Feiping Nie, Hua Wang, Xiao Cai, Heng Huang, and Chris Ding. Robust matrix completion via joint schatten p-norm and lp-norm minimization. In *IEEE 12th International Conference on Data Mining (ICDM)*, pages 566–574. IEEE, 2012.

[22] Yali Du, Meng Fang, Jinfeng Yi, Chang Xu, Jun Cheng, and Dacheng Tao. Enhancing the robustness of neural collaborative filtering systems under malicious attacks. *IEEE Transactions on Multimedia*, 2018.

[23] Yali Du, Meng Fang, Jinfeng Yi, Jun Cheng, and Dacheng Tao. Towards query efficient black-box attacks: An input-free perspective. In *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*, pages 13–24. ACM, 2018.

[24] Lei Han, Peng Sun, Yali Du, Jiechao Xiong, Qing Wang, Xinghai Sun, Han Liu, and Tong Zhang. Grid-wise control for multi-agent reinforcement learning in video game ai. In *International Conference on Machine Learning (ICML)*, pages 2576–2585, 2019.

[25] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The Adaptive Web*, pages 325–341. Springer, 2007.

[26] Joonseok Lee, Mingxuan Sun, and Guy Lebanon. A comparative study of collaborative filtering algorithms. *arXiv preprint arXiv:1205.3193*, 2012.

[27] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295. ACM, 2001.

[28] Xiaoyuan Su and Taghi M Khoshgoftaar. Collaborative filtering for multi-class data using belief nets algorithms. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, pages 497–504. IEEE, 2006.

[29] Koji Miyahara and Michael J Pazzani. Improvement of collaborative filtering with the simple bayesian classifier. *Information Processing Society of Japan*, 43(11), 2002.

[30] Xiangyu Wang, Dayu He, Danyang Chen, and Jinhui Xu. Clustering-based collaborative filtering for link prediction. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 332–338, 2015.

[31] Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of 22nd International Conference on Machine Learning*, pages 713–719. ACM, 2005.

[32] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, volume 1, pages 2–1, 2007.

[33] Tongliang Liu and Dacheng Tao. On the performance of manhattan nonnegative matrix factorization. *IEEE Transactions on Neural Networks and Learning Systems*, 27(9):1851–1863, September 2016. doi: 10.1109/TNNLS.2015.2458986.

[34] Yue Shi, Martha Larson, and Alan Hanjalic. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, 47(1): 3, 2014.

[35] Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5):544–557, 2009.

[36] Vladimir Vapnik and Rauf Izmailov. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research*, 16:2023–2049, 2015.

[37] William W Hager and Hongchao Zhang. A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization*, 2(1):35–58, 2006.

[38] Jorge Nocedal and Stephen J Wright. Numerical optimization, second edition. *Numerical optimization*, pages 497–528, 2006.

[39] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of International Conference on Machine Learning*, volume 14, pages 1188–1196, 2014.

[40] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[41] Spyros Reveliotis and Zhennan Fei. Robust deadlock avoidance for sequential resource allocation systems with resource outages. In *IEEE International Conference on Automation Science and Engineering*, pages 864–871. IEEE, 2016.

[42] Siyang Gao, Loo Hay Lee, Chun-Hung Chen, and Leyuan Shi. A sequential budget allocation framework for simulation optimization. *IEEE Transactions on Automation Science and Engineering*, 14(2):1185–1194, 2017.

[43] Parul Pandey, Dario Pompili, and Jingang Yi. Dynamic collaboration between networked robots and clouds in resource-constrained environments. *IEEE Transactions on Automation Science and Engineering*, 12(2):471–480, 2015.

[44] Anil Kumar Chorppath, Srikrishna Bhashyam, and Rajesh Sundaresan. A convex optimization framework for almost budget balanced allocation of a divisible good. *IEEE Transactions on Automation Science and Engineering*, 8(3):520–531, 2011.

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

[46] Yehuda Koren. The bellkor solution to the netflix grand prize. *Netflix Prize Documentation*, 81:1–10, 2009.

[47] Kai Yu, Shenghuo Zhu, John Lafferty, and Yihong Gong. Fast nonparametric matrix factorization for large-scale collaborative filtering. In *Proceedings of the 32nd international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 211–218. ACM, 2009.

[48] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[49] Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, pages 1329–1336, 2004.

[50] Dor Kedem, Stephen Tyree, Fei Sha, Gert R Lanckriet, and Kilian Q Weinberger. Non-linear metric learning. In *Advances in Neural Information Processing Systems*, pages 2573–2581, 2012.

[51] Marco Cuturi and David Avis. Ground metric learning. *Journal of Machine Learning Research*, 15(1):533–564, 2014.

[52] Tam Le and Marco Cuturi. Unsupervised riemannian metric learning for histograms using aitchison transformations. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2002–2011, 2015.

[53] Tam Le and Marco Cuturi. Adaptive euclidean maps for histograms: generalized Aitchison embeddings,. *Machine Learning*, 99(2):169–187, 2015.

[54] Kai-Yang Chiang, Cho-Jui Hsieh, and Inderjit Dhillon. Robust principal component analysis with side information. In *International Conference on Machine Learning*, pages 2291–2299, 2016.

[55] John Aitchison. *The statistical analysis of compositional data*. London: Chapman and Hall Ltd, 1986.

[56] J. Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society*, 44(2):139–177, 1982. ISSN 00359246. URL http://www.jstor.org/stable/2345821.

[57] John D Lafferty and David M Blei. Correlated topic models. In *Advances in Neural Information Processing Systems*, pages 147–154, 2006.

[58] J. Aitchison and S. M. Shen. Logistic-normal distributions: some properties and uses. *Biometrika*, 67(2):261–272, 1980. ISSN 00063444.

[59] John Aitchison. A concise guide to compositional data analysis. In *CDA workshop*, 2003.

[60] Juan José Egozcue, Vera Pawlowsky-Glahn, Glòria Mateu-Figueras, and Carles Barcelo-Vidal. Isometric logratio transformations for compositional data analysis. *Mathematical Geology*, 35 (3):279–300, 2003.

[61] Neal Parikh and Stephen P Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[62] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[63] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15:1455–1459, 2014.

[64] Bart Vandereycken. Low-rank matrix completion by riemannian optimization. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

[65] Zaiwen Wen, Wotao Yin, and Yin Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4(4):333–361, 2012.

[66] Benjamin Marlin. *Collaborative filtering: a machine learning perspective*. PhD thesis, University of Toronto, 2004.

[67] Andrea Vedaldi and Brian Fulkerson. Vlfeat: an open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM International Conference on Multimedia*, pages 1469–1472. ACM, 2010.

[68] F Maxwell Harper and Joseph A Konstan. The movielens datasets: history and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19, 2016.

[69] Paul McJones. Eachmovie collaborative filtering data set. *DEC Systems Research Center*, 249, 1997.

[70] Benjamin Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12(Dec):3413–3430, 2011.

[71] Emmanuel Candes and Benjamin Recht. Exact matrix completion via convex optimization. *Communications of the ACM*, 55(6):111–119, 2012.

[72] Ruoyu Sun and Zhi-Quan Luo. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, 2016.

[73] Ashis Biswas, Mingon Kang, Dongchul Kim, and Jean Gao. Robust inductive matrix completion strategy to explore associations between lincrnas and human disease phenotypes. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2018.

[74] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[75] S Derin Babacan, Martin Luessi, Rafael Molina, and Aggelos K Katsaggelos. Sparse bayesian methods for low-rank matrix estimation. *IEEE Transactions on Signal Processing*, 60(8): 3964–3977, 2012.

[76] Ohad Shamir and Shai Shalev-Shwartz. Matrix completion with the trace norm: learning, bounding, and transducing. *Journal of Machine Learning Research*, 15(1):3401–3423, 2014.

[77] Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[78] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

[79] Rong Ge, Chi Jin, and Yi Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1233–1242. JMLR. org, 2017.

[80] Tuo Zhao, Zhaoran Wang, and Han Liu. A nonconvex optimization framework for low rank matrix estimation. In *Advances in Neural Information Processing Systems*, pages 559–567, 2015.

[81] Jin Lu, Guannan Liang, Jiangwen Sun, and Jinbo Bi. A sparse interactive model for matrix completion with side information. In *Advances In Neural Information Processing Systems*, pages 4071–4079, 2016.

[82] Bo Xin, Yizhou Wang, Wen Gao, and David Wipf. Exploring algorithmic limits of matrix rank minimization under affine constraints. *IEEE Transactions on Signal Processing*, 64(19): 4960–4974, 2016.

[83] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[84] Junfeng Yang and Xiaoming Yuan. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of Computation*, 82(281):301–329, 2013.

[85] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in Neural Information Processing Systems*, pages 1813–1821, 2010.

[86] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[87] Nathan Srebro and Adi Shraibman. Rank, trace-norm and max-norm. In *Proceedings of the 18th Annual Conference on Learning Theory*, volume 5, pages 545–560. Springer, 2005.

[88] Anneleen Daemen, Obi L Griffith, Laura M Heiser, Nicholas J Wang, Oana M Enache, Zachary Sanborn, Francois Pepin, Steffen Durinck, James E Korkola, Malachi Griffith, et al. Modeling precision treatment of breast cancer. *Genome Biology*, 14(10):R110, 2013.

[89] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 355–364. ACM, 2017.

[90] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

[91] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pages 7–10. ACM, 2016.

[92] Ying Shan, T Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and JC Mao. Deep crossing: Web-scale modeling without manually crafted combinatorial features. In *ACM International Conference on Knowledge Discovery and Data Mining*, pages 255–262. ACM, 2016.

[93] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[94] Carlos E Seminario and David C Wilson. Robustness and accuracy tradeoffs for recommender systems under attack. In *FLAIRS Conference*, volume 314, 2012.

[95] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. *Proceedings of the 35th International Conference on Machine Learning,{ICML}*, 2018.

[96] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*. Citeseer, 2014.

[97] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. In *UAI*, pages 550–559, 2018.

[98] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *KDD Deep Learning Day*, 2018.

[99] Yin Zheng, Bangsheng Tang, Wenkui Ding, and Hanning Zhou. A neural autoregressive approach to collaborative filtering. In *International Conference on Machine Learning*, pages 764–773, 2016.

[100] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, pages 111–112. ACM, 2015.

[101] Minhao Cheng, Jinfeng Yi, Huan Zhang, Pin-Yu Chen, and Cho-Jui Hsieh. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. *arXiv preprint arXiv:1803.01128*, 2018.

[102] Steffen Rendle. Factorization machines. In *IEEE 10th International Conference on Data Mining (ICDM)*, pages 995–1000. IEEE, 2010.

[103] Linas Baltrunas, Karen Church, Alexandros Karatzoglou, and Nuria Oliver. Frappe: Understanding the usage and perception of mobile app recommendations in-the-wild. *arXiv preprint arXiv:1505.03014*, 2015.

[104] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL http://arxiv.org/abs/1503.02531.

[105] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *International Conference on Learning Representations*, 2015.

[106] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. *arXiv preprint arXiv:1712.00673*, 2017.

[107] 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.

[108] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[109] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[110] Chen Gong, Xiaojun Chang, Meng Fang, and Jian Yang. Teaching semi-supervised classifier via generalized distillation. In *IJCAI*, pages 2156–2162, 2018.

[111] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

[112] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283, 2018.

[113] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[114] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[115] Caglar Gülçehre and Yoshua Bengio. Knowledge matters: Importance of prior information for optimization. *Journal of Machine Learning Research*, 17(1):226–257, 2016.

[116] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, 1989.

[117] François Chollet et al. Keras. https://keras.io, 2015.

[118] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, pages 265–283. USENIX Association, 2016.

[119] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[120] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1661–1670. ACM, 2015.