2017 届博士学位论文

分类号：_____ 学校代码：10269

密　级：_____ 学　　号：52141500009

華東師範大學

East China Normal University

博 士 学 位 论 文

DOCTORAL DISSERTATION

论文题目：基于矩阵分解的个性化推荐系统

院　　系：	计算机科学与软件工程学院
专 业 名 称：	软件工程
研 究 方 向：	推荐系统
指 导 教 师：	王晓玲 教授
学位申请人：	王科强

2017 年 03 月

Dissertation for doctor degree in 2017 University Code: 10269

Student ID: 52141500009

EAST CHINA NORMAL UNIVERSITY

Personalized Recommender Systems based on Matrix Factorization

Department:	School of Computer Science and Software Engineering
Major:	Software Engineering
Research direction:	Recommender Systems
Supervisor:	Prof. Xiaoling Wang
Candidate:	Keqiang Wang

2017.03

摘 要

随着信息技术和互联网的发展，网民用户和网络产品数量成爆炸式增长。个性化推荐系统对用户行为和企业商品特性数据建模，为用户提供满足他们兴趣和需求的信息，同时为企业推广提供目标客户。现代互联网服务提供商，例如淘宝等在线购物网站、爱奇艺等在线视频网站、大众点评等生活信息服务网站，提供大量商品给用户消费，让用户评分商品以及使用标签描述商品。针对以上用户行为数据，本文以矩阵分解相关理论为基础，在隐式反馈数据上对用户进行商品推荐，然后利用显式评分数据进行用户对目标商品的评分预测，最后利用显式标签数据建立标签推荐模型方便用户输入，提高商品属性表示，帮助推荐系统良性循环。本文的研究问题和技术贡献总结如下:

1. 基于加权局部矩阵分解的商品推荐：现有的针对隐式反馈数据的矩阵分解模型往往只从数据的全局信息出发，忽略了数据之中的局部信息。为了利用隐式反馈数据的局部信息，本文提出了一种加权局部矩阵分解模型（LWMF）进行商品推荐，同时为该模型设计了高效的子矩阵选择算法和改进的交替最小二乘优化算法（ALS），对用户和商品的局部特征建模，同时缓解了数据稀疏性问题。真实数据上的实验结果表明 LWMF 有较优的推荐效果，并验证了考虑隐式反馈数据的局部信息有助于商品推荐。

2. 基于多主题矩阵分解的评分预测：为克服现有工作中针对显式评分数据局部信息建模的不可解释性和目标函数的不一致性，本文提出了多主题矩阵分解模型（PMTMF）。它结合主题模型和概率矩阵分解模型，在利用主题模型建模数据局部信息的同时利用矩阵分解用来建模用户和商品的局部内在特征。并且，本文还扩展了多主题矩阵分解模型的贝叶斯估计版本（BPMTMF），使得模型需要更少的经验设置参数的同时得到更高的推荐准确率。实验结果说

明该模型优于其他考虑局部信息的模型，并且具有对局部建模信息的可解释性。

3. 时间和频率感知的标签推荐：为了利用标签数据中用户标注标签的时间和频率信息，本文提出了时间与频率感知的张量分解模型。该模型是对标签的时间和频率信息建模，将其以权重的方式加入到逐对排序张量分解模型，对所有用户和商品未使用的标签以相同权重对待。实验结果表明该模型能够有效的利用时间和频率信息，提高标签准确度的同时在新用户冷启动问题上也有较好的，并且具有可接受的推荐新颖性。

关键词: 矩阵分解，局部信息，主题模型，评分预测，商品推荐，标签推荐

ABSTRACT

With the development of information technology and the Internet, the world has transformed from the time of lack of information to the information overload era, where the numbers of Internet users and network products show explosive growth. The personalized recommendation system builds models utilizing users'behavior data and characterization data of enterprise product. This kind of system can provide users with information that satisfies their interests and needs and offer target customers for business promotion. Modern Internet service providers, such as online shopping site Taobao, online video site iQIYI, and life information service website Dianping, provide consumers with a large number of goods, allowing users to score goods and describe goods using labels. Focusing on the above user behavior data, this thesis aims at tackling three typical recommendation tasks based on the theory of matrix factorization. First, we recommend items to users on implicit feedback datasets. Second, we do rating prediction that predicts the rating of a user on a given item based on explicit feedback datasets. Finally, we present a tag recommendation model based on explicit tag data, which can make tagging input convenient for users and improve the representation of item properties to help the positive cycle of recommendation system. The research questions and technical contributions in this thesis can be summarized as follows,

1. Local Weighted matrix factorization based item recommendation model: the existing matrix factorization models on implicit feedback datasets only consider the global property of data and ignore the local property in data. To utilize the local property in implicit feedback datasets, we propose Local Weighted Matrix Factorization (LWMF) to recommend items. To solve the model, we design the efficient sub-matrix selection algorithm and improved Alternating Least Square (ALS) optimization algorithm. We model the local property of users and items and relieve the problem of data sparsity. The experimental results on real datasets show that LWMF

has relatively good recommendation performance and verify that considering local property of implicit feedback data is helpful for item recommendation.

2. Multi-Topic Matrix Factorization : in existing work, there are two main weak points. One is the non-interpretability of the models built on the local information in explicit rating data. Another is the inconsistency of the objective function. To overcome these problems, we present a Probabilistic Multi-Topic Matrix Factorization (PMTMF) model. This model combines topic model with probabilistic matrix factorization model. Topic model is used to capture local information of data and matrix factorization models the local inner property of users and items. Furthermore, we extend the multi-topic matrix factorization model into the Bayesian Probabilistic Multi-Topic Matrix Factorization (BPMTMF) version. This version requires fewer efforts in parameter selection and can achieve higher recommendation accuracy. Extensive experiments demonstrate the effectiveness of the proposed model compared with several competitive baselines and the interpretability to local modelling information of the proposed model.

3. Time and Frequency aware tag recommendation model: to utilize the time when users add tags and frequency information in tag data, this thesis presents a time and frequency weighted tensor factorization model. This model incorporates time and frequency information of tags into pairwise tensor factorization model by modelling them as weights, where the unused tags for all users and goods are treated with the same weight. The experimental results show that this model can utilize time and frequency information effectively and can improve the accuracy of tag recommendation. In addition, this model performs well in the cold start situation of users and can achieve good quality to recommend new tags.

Keywords: *Matrix Factorization; Local Information; Topic Model; Rating Prediction; Item Recommendation; Tag Recommendation.*

目录

插图

表格

主要缩写符号对照表

ALS	交替最小二乘法（Area under the ROC Curve）
AUC	ROC 曲线下面积（Area under the ROC Curve）
BPMF	贝叶斯概率矩阵分解（Bayesian Probabilistic Matrix Factorization）
BPR	贝叶斯个性化排序（Bayesian Personalized Ranking）
ERR	期望排序倒数（Expected Reciprocal Rank）
LDA	潜在狄利克雷分布（Latent Dirichlet Allocation）
MF	矩阵分解 (Matrix Factorization)
MRR	平均排序倒数 (Mean Reciprocal Rank)
NDCG	归一化的贴现累计收益 (Normalized Discounted Cumulative Gain)
PLSA	概率潜语义分析 (Probabilistic Latent Semantic Analysis)
PMF	概率矩阵分解 (Probabilistic Matrix Factorization)
PITF	成对相互张量分解（Pairwise Interaction Tensor Factorization）
RMSE	均方根误差（Root Mean Square Error）
ROC	受试者工作特征曲线 (Receiver Operating Characteristic Curve)
SGD	随机梯度下降 (Stochastic Gradient Descent)
SVD	奇异值分解 (Singular Value Decomposition)
TF	张量分解 (Tensor Factorization)
WMF	加权矩阵分解（Weighted Matrix Factorization）

表 1: 本文符号说明

符号	符号描述
N	用户数量
M	商品数量
T	标签数量
K	局部隐藏特征向量的维度 ($\ll \min(N, M)$)
H	子矩阵的个数或者是主题模型中主题个数
$\mathbf{R}$	原始数据矩阵（< 用户，商品，次数/评分 >）($\in \mathbb{R}^{N\times M}$)
$\mathcal{N}$	数据矩阵中 $\mathbf{R}$ 用户索引集合
$\mathcal{M}$	数据矩阵中 $\mathbf{R}$ 商品索引集合
$\mathcal{T}$	数据矩阵中 $\mathbf{R}$ 标签索引集合
$\mathbf{C}$	原始次数数据矩阵 $\mathbf{R}$ 的 01（二值化）化数据矩阵 ($\in \mathbb{R}^{N\times M}$)
$\mathbf{W}$	数据矩阵的置信权重矩阵
$\mathbf{P}$	用户的隐藏特征向量
$\mathbf{Q}$	商品的隐藏特征向量
$\mathbf{T}^{\mathcal{N}}$	与用户对应的标签隐藏特征向量
$\mathbf{T}^{\mathcal{M}}$	与商品对应的标签隐藏特征向量
$\mathbf{R}^h$	第 h 个子数据矩阵
$\mathbf{C}^h$	第 h 个 01（二值化）化子数据矩阵
$\mathbf{W}^h$	01（二值化）化子数据矩阵 $\mathbf{C}^h$ 的置信权重矩阵
$\mathbf{V}^h$	01（二值化）化子数据矩阵 $\mathbf{C}^h$ 的子矩阵权重矩阵 $\mathbf{C}^h$
$\mathcal{N}^h, \mathcal{M}^h$	数据子矩阵中用户索引集合，商品索引集合
$\mathbf{P}^h_u, \mathbf{Q}^h_m$	子数据矩阵 $\mathcal{R}^h$ 中第 u 个用户局部隐藏向量和 m 个商品局部隐藏向量 ($\in \mathbb{R}^K$)
$\mathcal{A}$	数据点集合（用户-商品记录对集合，不包含用户未发现商品的情况）
$a_i = \langle u_i, m_i\rangle$	数据点 $\langle u_i, m_i\rangle$ (用户-商品记录对, $\in \mathcal{A}$)
$\hat{\mathcal{A}}$	锚点集合 ($\subset \mathcal{A}$)
$\hat{a}_h = \langle \hat{u}_h, \hat{m}_h\rangle$	锚点 ($\in \hat{\mathcal{A}}$)
$E(a_i, a_j)$	两个数据点之间的核函数值
z_i ($z_{u,m}$)	数据点 $i = \langle u, m\rangle$ 上分配的主题
θ_u	第 u 个用户的主题分布 ($\in \mathbb{R}^K$)
ϕ_k	第 k 个主题上的商品分布 ($\in \mathbb{R}^M$)
α	主题模型上主题分布的狄利克雷先验参数
β	主题模型上商品分布的狄利克雷先验参数
Ψ_0	概率矩阵分解上高斯先验的 Gaussian-Wishart 先验参数

第一章　绪论

推荐系统是当代互联网信息爆炸的产物，旨在从互联网上大量的商品中帮助用户发现感兴趣的产品，同时帮助商家精准把握商品的目标用户，减少用户搜索和商家推广商品的时间。本文重点研究根据用户对商品的历史访问行为进行个性化推荐的相关问题。本章中，章节 1.1阐述了推荐系统的发展历史和研究价值；章节 1.2详述了本文的具体研究内容以及所遇到的挑战；章节 1.3简述了本文的主要研究方法和研究贡献。

1.1　研究背景

20 世纪 60 年代末期，美国高校实现计算机之间的信息交换，标志着互联网的诞生。1985 年之后，互联网的快速发展，使得用户越来越方便地获取信息。随之带来的是信息超载问题，网络信息的信息爆炸，用户需要在大量无用信息中找到感兴趣的信息。

现代解决互联网信息超载问题主要通过信息过滤的方法，主要包括四类方法：类目导航、搜索引擎、社交网络和推荐系统。类目导航网站充当着引领用户浏览互联网资源的角色。一个目录导航网站通常需要跟踪和收集不同来源的互联网数据和资源，并将其整理为统一的形式存储起来，呈现给用户的通常是一个分类目录，包括首页、频道页、专题页等层次，用户只需在该类目体系中层层点击便可定位到自己所需的网站。该类网站最著名的当属雅虎，它开创了内容免费、广告收费的门户导航网站的商业模式。然后随着互联网泡沫的发生以及搜索引擎公司谷歌的崛起，雅虎等类目导航网站迅速衰败。不同于类目导航网站依赖人工管理维护目录索引的方式，搜索引擎利用爬虫程序自动爬取网页链接，若网页上存在超链接，理论上互联网中的所有网页都会被爬取到。得到这个庞大的互联网网页链接数据库后，为了快速响应用户的搜索请求，搜索引擎还需要对这些信息创建索引，将

它们按照一定的规则进行组织。用户与搜索引擎交互时，会使用关键词发出查询，无需考虑网站的分类问题。搜索引擎在数据库中查找到相关文档，并根据相关程度的高低对结果进行排序后返回给用户。谷歌目前为用户提供了丰富的搜索服务，包括网页、图片、视频、地图、新闻以及问答等。在搜索引擎之后，一类以人类社交为核心构建的网络服务迅速崛起并发展成熟，2012 年全球最大的社交网络公司 Facebook 上市，社交网络的发展潜力被国内外投资公司看好。社交网站模拟了人类社会中真实的社交场景，为一群具有相似爱好或学习工作经历的人创建在线社区，并提供多种基于互联网的交互方式，比如即时通信、文件分享等来促进信息的交流与分享。一方面社交网站可以帮助用户与朋友保持联系并扩大自己的交际圈，另一方面社交网络中的群组由相互熟悉或兴趣相似的用户构成，针对用户群组打广告更加具有针对性，因此社交网站成为了商家在线营销推广的窗口。至今，社交网络已成为覆盖用户最大、传播影响最大、商业价值巨大的互联网业务，为人们获取信息、获取广告提供了巨大的便利。与前面三个方法不同，尽管没有一家互联网巨头公司是以推荐系统技术起家，但是这些公司或多或少需要推荐系统技术，包括广告推荐，音乐、电影推荐，社交网络好友推荐，新闻推荐，电子商务网站商品推荐以及标签推荐等等。推荐系统根据用户的兴趣、爱好，结合商品信息特征，向用户提供感兴趣的商品和信息，为用户提供决策支持。

从信息过滤的方法中，可以看出类目导航和搜索引擎主要是用户主动的寻找需要的信息；社交网络则是通过用户自身主动关注其他用户，然后被动接收关注用户的状态信息；但是，上述三种方法不能完全满足用户对信息的需求，在有明确目的并且能用明确的关键词或者网站或者人物查询可以找到自己需要的信息。而推荐系统则是用户完全被动接收推荐的信息，主要是通过用户消费商品的历史记录，用户个人信息以及商品信息来匹配进行商品推荐。这样使得在用户不明确自己的需求或者难以使用关键字搜索或者其他用户那得到有用的信息等情况下，用户依旧可以通过推荐系统得到自居需求的商品。

个性化推荐系统在现代互联网服务中体现越来越重要的作用。作为一种信息

过滤系统，它通过挖掘用户行为数据来预测用户未来对物品的偏好，为用户提供个性化的推荐列表。自从 1992 年，Goldberg 第一次使用协同过滤的思想，利用用户的标注邮件信息，实现了一个个性化邮件系统 Tapestry，到 1995 年麻省理工学院的 Pattie Maes 实验室成功使得推荐系统技术商业化，创立了 Agent 是公司，再到 2006 年的 Netflix 的百万美元大奖，都体现了个性化推荐系统发展的火热程度。目前推荐系统已在很多领域得到成功应用，比如谷歌新闻中个性化推荐系统提高了 38% 的新闻点击率；而电影服务网站 Netflix 有高达三分之二的电影是因为被推荐而观看的；美国著名电子商务网站 Amazon 声称 35% 的收入是由推荐引擎产生的。同时学术界对推荐系统也一直保持着很高的研究热度，学术界推荐系统顶级会议 ACM 推荐系统大会（RecSys）自从 2007 年以来已成功举办了 10 届，推荐系统也逐步发展成为了一门独立的热门学科。相较于搜索引擎，推荐系统能够发现一些用户自己并不清楚的兴趣点，帮助他们找到一些自己没有办法找到的物品，进一步地让用户对推荐系统产生依赖。这些都说明推荐系统蕴含着巨大的商业价值及研究价值，值得我们科研人员深入研究与发展。

1.2 研究内容与挑战

在做商品推荐任务时，主要可以利用两类用户对商品的历史行为数据 [1]：（1）显式反馈数据；（2）隐式反馈数据。用户访问商品之后，对商品做出了显式反馈，显式地表达了用户对商品的喜好程度，我们称这类数据为显式反馈数据。显式反馈数据包括表达用户对商品喜好的评分数据，用户描述商品特点的标签数据以及表达用户体验商品感受的评论数据等。显式反馈数据能够准确地表达用户的观点，既可以推荐模型更好地刻画用户画像，也能使得商家认识到自家商品的优缺点，即时地进行相应的改进，抓住目标用户的痛点和痒点。现代互联网服务提供商一般都提供显式反馈功能，包括淘宝、京东等著名在线购物网站，顺丰等物流服务提供商，优酷、爱奇艺等多媒体提供商，大众点评等生活信息服务提供商，等等。因此，在推荐系统利用好显式反馈数据，特别是结构化评分数据和标签数据，能够极大

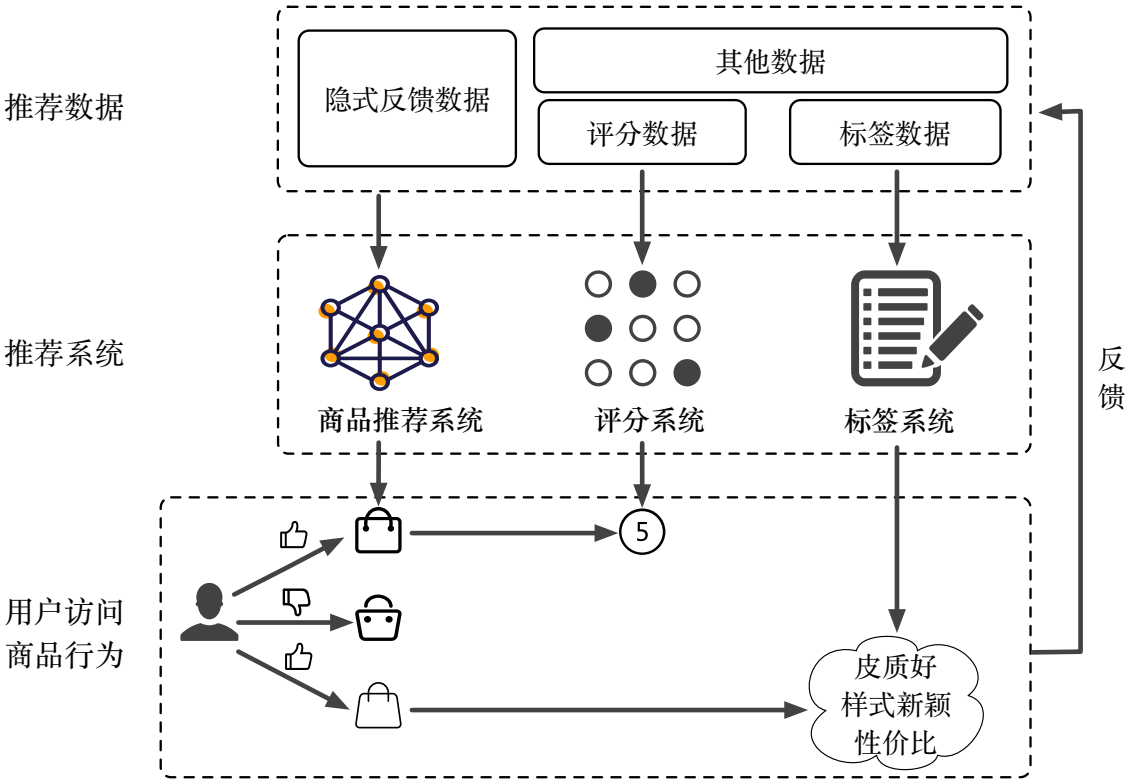


图 1.1: 本文的研究内容

提高推荐系统性能，使得推荐系统良性发展。尽管显式反馈数据能够很好地反应用户对商品的真实喜好，但是需要用户付出一定的时间和劳力代价对商品进行显式打分、标注标签和评论，导致数据相对不容易获取而不够全面。因此，用户对商品的行为数据只要是没有显式评价的访问数据，例如仅仅浏览商品相关网页信息。这类数据虽然没有显式表达用户对商品的喜好值，但隐式地表明用户对这类商品是有兴趣，存在潜在的可能消费该类商品，因此称这类数据为隐式反馈数据。隐式反馈数据，相对于显式反馈数据，不需要用户额外的花费获得用户的个人偏好，数据多而全，但是数据噪声也较多。因此如何有效利用隐式反馈数据是推荐系统的重要课题。

本文主要围绕隐式反馈数据，显式反馈数据中的评分数据和标签数据这三类数据，研究推荐系统中的三类问题[1]，如图 1.1所示：（1）商品推荐[2]是根据用户隐

[1]http://recsyswiki.com/wiki/Category:Task
[2]http://recsyswiki.com/wiki/Item_prediction

图 1.2: 在线购物网站–京东的推荐示例

式访问商品的历史记录信息，推荐用户可能感兴趣的商品列表；（2）评分预测[3]则主要根据历史数据中用户对商品的显式评分数据，预测给定用户对商品的评分，得到用户对感兴趣商品列表的喜好值；（3）而标签推荐[4]则根据历史数据中用户的显式标注标签行为，对用户推荐标签用以描述商品，方便用户输入标签，更加准确表达商品的优缺点，促进推荐系统的良性循环。图 1.2展示了中国著名在线购物网站京东[5]上的这三类推荐功能：用户购买商品之后，可以对商品进行显式的评分，表达自己对商品各方面的满意水平，以及使用标签描述商品的特点，并且京东推荐引擎能够根据用户已往访问商品记录信息推荐相关的商品。下面分别详细介绍上述三个推荐任务以及遇到的挑战。

[3]http://recsyswiki.com/wiki/Rating_prediction
[4]http://recsyswiki.com/wiki/Tag_recommendation
[5]https://www.jd.com/

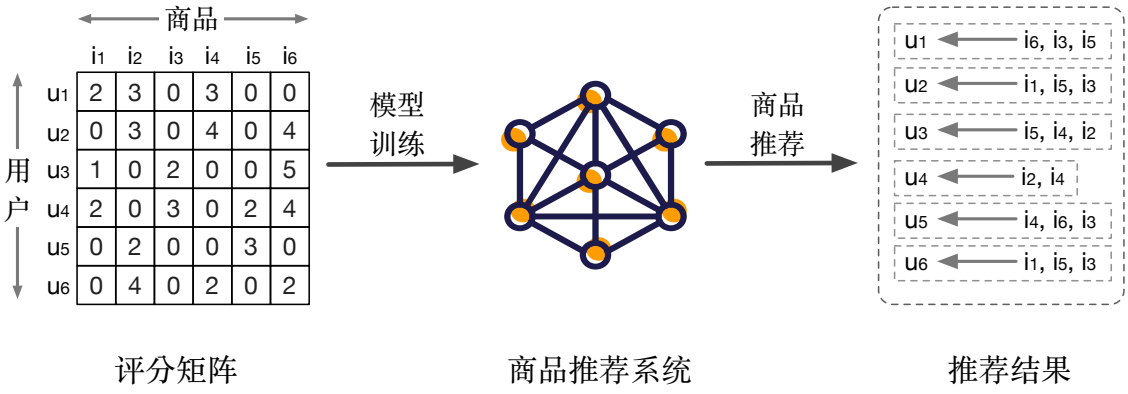


图 1.3: 商品推荐

1.2.1 商品推荐

互联网时代，用户大量使用各大网站或者互联网应用浏览各类商品，这类用户历史行为数据能够客观反应用户对各类商品的感兴趣偏好。而好的推荐引擎需要能够根据用户对商品的历史行为推荐用户感兴趣的商品，帮助用户减少时间，同时增加公司或者商家收益。例如音乐网站根据用户收听歌曲的历史数据推荐用户感兴趣的音乐，电子商务网站根据用户浏览和搜索商品的行为推荐用户感兴趣的产品，甚至连社交网站也可以根据用户的关注信息推荐用户感兴趣的其他用户供其关注。具体地，如图 1.2所示的在线购物网站京东的例子，用户因为购买了电脑显示屏，并且在此之后还经常浏览电脑配件产品，京东推荐引擎推荐了多件相关的商品，例如机械硬盘，电脑内存，电脑主板等电脑配件，极大减少用户寻找相关产品的时间，并且能够有效商品的精准商品推广营销。

因为显式反馈数据需要用户花费时间和精力，许多用户不会对商品进行显式评价，所以用户访问商品的大多数数据属于隐式反馈数据，因此利用隐式反馈数据进行商品推荐显得异常重要。隐式反馈数据中用户访问商品的一个重要数据是访问次数。用户访问商品的次数在一定程度上说明用户对商品的感兴趣程度，次数越多，感兴趣的概率越大。本文主要研究此类数据上以利用矩阵分解模型解决商品推荐问题。如图 1.3所示，本文商品推荐根据用户隐式访问商品行为得到的次数数据矩阵，构建模型来刻画用户和商品的特征偏好，建立推荐模型。推荐模型在

执行推荐任务时，需要针对每个用户在剩下未访问过的商品集合中，计算出用户对各个商品的喜好值，然后根据喜好值大小降序排序商品，最后排序前几个的商品列表推荐给用户。

传统的在隐式反馈数据上针对商品推荐的矩阵分解模型，例如加权矩阵分解（Weighted Matrix Factorization，简称 WMF） [1]，假设隐式反馈数据的次数矩阵在全局上低秩的，能够使用低秩分解，所以可以使用低秩矩阵分解近似原始矩阵。但是，次数矩阵有可能是全局上不是低秩的，而其中的局部矩阵是低秩的。例如，在签到数据集中，整个国家的签到数据集因为城市的不同，其次数矩阵不是低秩的，但是每个城市中数据矩阵甚至是一个城市中每个商区的数据矩阵是低秩的，可以使用低秩矩阵分解。另一个例子是电影推荐中，小孩用户喜欢各类不同的卡通片，该部分数据矩阵可能是低秩的，但是整个电影数据矩阵则是非低秩的。次数矩阵在全局上不是低秩的，而其中的局部矩阵是低秩的。例如，在签到数据集中，整个国家的签到数据集因为城市的不同，其次数矩阵不是低秩的，但是每个城市中数据矩阵甚至是一个城市中每个商区的数据矩阵是低秩的，可以使用低秩矩阵分解。另一个例子是电影推荐中，小孩用户喜欢各类不同的卡通片，该部分数据矩阵可能是低秩的，但是整个电影数据矩阵则是非低秩的。基于上述研究问题，本文面向隐式反馈数据，重点研究如何根据传统的 WMF 模型进行扩展，建模次数数据矩阵的局部信息并进行有效地快速优化，进而可以更好地刻画用户行为的局部信息，以便取得更好的推荐结果。

1.2.2 评分预测

尽管显式评分数据没有隐式反馈数据多，但是评分数据能够精确反映用户对商品的满意程度。因此，如果推荐系统能够有效利用评分数据，能够精确刻画用户对商品的关注点，帮助商品提高相关的服务，对企业和推荐系统本身后续发展都提供很好的帮助。一般总共有五种评分 1-5 分（力荐，推荐，还行，较差，很差）供用户选择反馈。如图 1.2所展示在线购物网站京东的评分系统，总共需要评分商

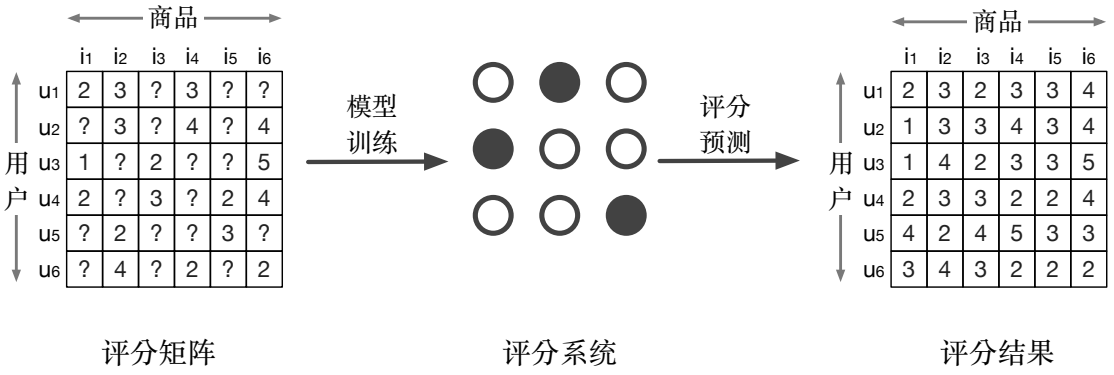


图 1.4: 评分预测

品的四个方面，包括商品的满意度，商品包装，送货速度和配送员服务态度。用户在以上四个方面进行显式评分，推荐系统就能知道用户具体在哪一方面比较看重，可以推荐这方面比较好的商家产品，同时也可以帮助商品改善这项服务。评分推荐任务中有一个很著名的竞赛——Netflix 竞赛[6] [2]，公开了大约 1 亿个 1 – 5 的匿名影片显式评分，数据集仅包含了电影名称、评分等级以及相应用户评分的时间，没有任何文本类信息评价的内容。竞赛要求参赛者预测用户喜好电影的程度，即用户对电影评分。比赛使用预测的评分与真实评分之间的均方根误差进行评价，第一个将其预测性能比 Netflix 官方评分系统提高 10% 的参赛选手将获得 100 万美元奖励。Netflix 竞赛使得评分预测成为推荐系统中的一个著名基本问题，大量的科研人员和公司人员对此进行研究。

评分预测系统将用户历史评分信息构建评分矩阵。与次数矩阵中使用 0 值代替进行拟合不同，如果用户没有对商品进行显式评分，评分矩阵中则是作为缺失值出现，一般最终模型对用户和商品的特征刻画不造成影响。如图 1.4所示，本文将利用评分矩阵构建评分预测系统，然后对缺失值预测，得到用户对商品的喜好值。传统的在显式评分数据上针对评分预测的矩阵分解模型，例如概率矩阵分解（Probabilisitic Matrix Factorization，简称 PMF）[3]，如同隐式反馈数据中的 WMF 模型，假设评分矩阵是全局低秩的，利用 PMF 进行全局分解来刻画用户和商品的特性。前人基于基础矩阵分解的基础上，对显式评分数据的局部信息进行建模 [4–6]，

[6] http://www.netflixprize.com/

它们一般分成两个步骤，首先使用子矩阵选择算法将原始评分矩阵分成一系列子矩阵，子矩阵内部的用户和商品是高度相似的，然后再利用奇异值分解（Singular Value Decomposition，简称 SVD） [7, 8] 进行矩阵分解。因此，整个过程需要优化两个目标，同时需要考虑每个子矩阵所占到原始矩阵的权重。另外，当前的工作子矩阵选择算法只是简单的随机选择 [4, 5] 或者是进行暴力的硬聚类 [6]，导致子矩阵解释性不足，对子矩阵的含义没有直观的感受。基于上述研究问题，本文面向显式评分数据，重点研究如何同时进行子矩阵选择和子矩阵分解，从而优化同一个目标函数，使得更有效地预测评分，同时如何建立更有意义，可解释性更强的局部矩阵分解模型。

1.2.3 标签推荐

除了评分数据，显式反馈数据还包括一类典型的数据，即标签数据。标签数据不仅可以描述商品具体地优缺点，也能反应用户关注商品的哪个方面。如图 1.2所展示在线购物网站京东的标签推荐模块，标签推荐引擎根据用户所购买的显示器以及用户对商品的关注要点，进行个性化推荐标签，主要用来描述显示器显示效果的属性。国内标签系统较为成功的是豆瓣电影[7]标签系统，用户可以对电影各个方面标注标签，例如导演，主演，类型，国家，年份，主题等等，不仅成功用户在电影方面的关注点，同时也众包标签得到电影的各类属性。这类标签数据不仅可以准确刻画用户和电影的属性，提高推荐准确率，促进商品推荐的良性循环，而且能方便用户根据标签主动寻找感兴趣的电影。因此，标签推荐系统就是如图 1.5所示，利用这类标签数据对用户，商品以及标签建立模型，然后在用户评价商品的时候及时的推荐个性化标签，减少用户手动输入标签的时间。

传统的标签推荐系统使用协同过滤模型进行推荐，例如主题模型 [9] 和矩阵分解类模型 [10]。协同过滤模型能够能够很好的建模用户和商品难以表达的信息，增加推荐商品的新颖性。然后传统的基于协同过滤的标签推荐系统无法处理新用户冷启动的问题，同时也无法处理下述用户标记标签的情况。标签推荐中，用户一般

[7]https://movie.douban.com/

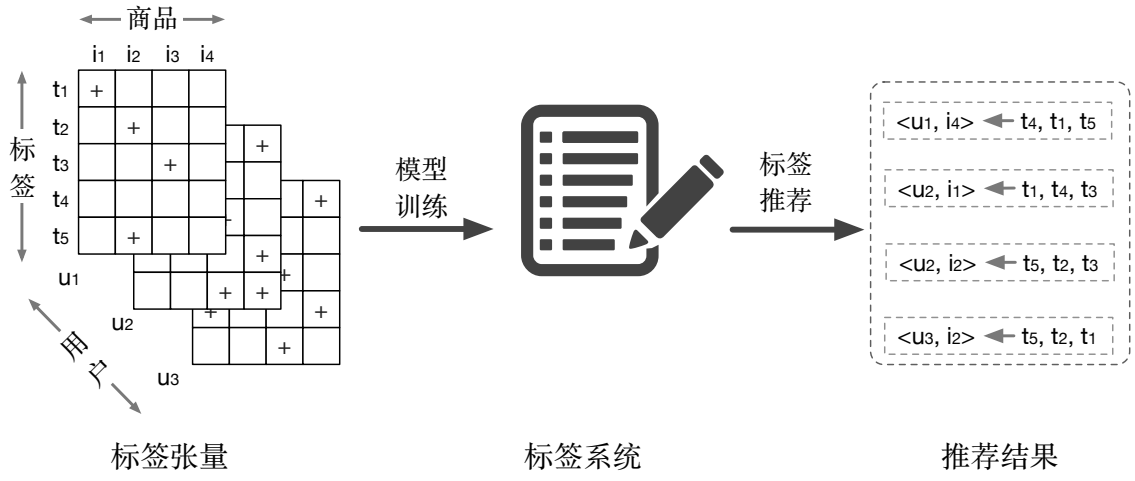


图 1.5: 标签推荐

会在连续一段时间内访问某类相似商品。譬如，用户观看由同一个明星担任角色的电影，或者是在搜索引擎中搜索某类形似的问题等等。这些商品的某些属性是相似的，导致用户标记的标签也相似。而且，用户在某段时间内打的某个标签越多，在之后的一段时间内越有可能再次使用这个标签。另外，因为用户的个人语言习惯，有多个词表示同一个意思（例如“风景”和“景色”）时，用户偏向使用自己熟悉的词。上述的情况表明，标签数据天然地有着时间和频率感知的局部特性。前人 [11, 12] 针对上述情况，根据用户历史标签和商品流行标签，进行启发式推荐。尽管这类模型相对于传统的协同过滤模型提高了推荐准确率，然后同时也极大地减少了推荐标签的新颖性，只会推荐用户曾经使用的和商品被标注的标签。针对上述研究问题，本文着重考虑如何结合上述两类模型的优点，对时间和频率感知的局部特性建模，将其结合到协同过滤模型之中。

1.3 研究贡献

本文围绕推荐系统这一主题和矩阵分解模型这一技术，对推荐数据中的隐式反馈数据、显式评分数据和显式标签数据建模，解决推荐系统中涉及的三个任务：*商品推荐*，*评分预测*和*标签推荐*。具体地，本文的主要分为以下三部分研究内容：（1）基于局部加权矩阵分解的商品推荐；（2）基于多主题矩阵分解的评分预测；（3）时

间和频率感知的标签推荐。下面分别介绍上述任务的研究内容:

1.3.1 基于局部加权矩阵分解的商品推荐

本文遵从加权矩阵分解模型的权重假设，对每个数据点加入次数权重，用来区分用户访问过的商品和没有访问过的商品。针对传统加权矩阵分解模型 [1] 全局低秩的假设，我们提出*隐式反馈数据*中次数矩阵在全局上不是低秩的，而其中的局部矩阵是低秩的。因此，我们针对上面的假设，对隐式反馈数据上的*商品推荐*任务提出一种加权局部矩阵分解模型。该模型不仅能刻画次数数据的局部信息，同时该模型缓解了数据稀疏性问题以及更好地进行分布式矩阵分解。具体地，本文将整个训练数据矩阵分成一系列子矩阵，其中每个子矩阵内的数据点是高度相似的，所有子矩阵组合能够近似成原始矩阵，然后对每个子矩阵利用加权矩阵分解得到用户和商品隐藏特征向量，最后利用隐藏向量对每个预测子矩阵进行加权平均近似原始矩阵。为此，本文设计了一个高效的子矩阵选择算法快速选择子矩阵集合。在模型优化下，本文采用改进的交替最小二乘优化算法（ALS）以适用于加权子矩阵分解模型。基于真实的 Gowalla 和 Foursquare 公开数据集的实验结果表明，本文提出的加权局部矩阵分解模型相对于传统的 WMF 模型，能够有效的提高在隐式反馈数据数据上的商品推荐效果。

1.3.2 基于多主题矩阵分解的评分预测

本文首次在显式评分数据上结合主题模型和矩阵分解模型进行评分预测。与前人只是简单的随机选择或者是进行暴力的硬聚类不同，本文利用主题模型建模评分矩阵的局部信息，增加子矩阵的可解释性。另外，前人对评分数据的局部信息建模，需要分为两个步骤:（1）子矩阵选择;（2）子矩阵分解。因此此类工作需要分两部优化目标函数。针对这类问题，本文结合主题模型和概率矩阵分解，称为多主题矩阵分解模型，在将整个数据矩阵分解成一个个小的子矩阵的同时进行概率矩阵分解，形成唯一的优化目标函数，能够更加准确地建模数据局部信息以及用户和商品的内在特征。此外，我们还对多主题矩阵分解模型模型进行贝叶斯扩展，

提出了全贝叶斯的多主题概率矩阵分解，使得我们需要更少的经验设置参数同时得到更好的推荐效果。基于真实的 Movielens 和 Netflix 公开数据集的实验结果表明，本文提出的贝叶斯多主题矩阵分解模型模型在显式评分数据上进行评分预测要优于前人的局部矩阵分解模型。

1.3.3 时间和频率感知的标签推荐

章节 1.2.3表明标签数据天然地有着时间和频率感知的局部特性。传统的基于协同过滤的标签推荐模型无法解决上述问题，并且对新用户冷启动问题不友好。而根据用户历史标签和商品流行标签进行启发式推荐模型虽然解决上述问题，提高了推荐准确率，但减少了标签推荐的新颖性，不利于推荐的良性循环。针对上述问题，本文提出了时间与频率感知的的张量分解模型。模型的主要思想是建立对用户-标签-时间关系和商品-标签关系相应权重，将时间和频率信息以权重的方式加入到逐对排序张量分解模型，对所有用户和商品未使用的标签以相同权重对待，使用随机梯度下降算法（SGD）优化模型参数。基于真实世界的公开标签数据的实验对比表明，本文提出的模型有效的利用时间和频率信息，提高标签准确度的同时有不错的推荐新颖性，并且在冷启动数据上也有较好的表现。

1.4 章节安排

本文一共分为六章，章节安排如图 1.6所示：

- 第二章较为系统地总结、分析了在基本数据 < 用户，商品，评分/次数 > 上的几种主流推荐系统评分预测模型和商品推荐模型。根据模型分类的不同，分别介绍了基于 K 近邻、基于矩阵分解和基于概率图模型的三类协同过滤推荐算法。最后，本章还介绍了本文需要使用的三个基本矩阵分解类模型，分别用于评分预测、商品推荐和标签推荐三个推荐任务。

- 第三章介绍本文的第一个工作，主要介绍对隐式反馈数据的局部信息建模以便更好地进行商品推荐。具体地，对于隐式反馈数据矩阵，我们认为它不是

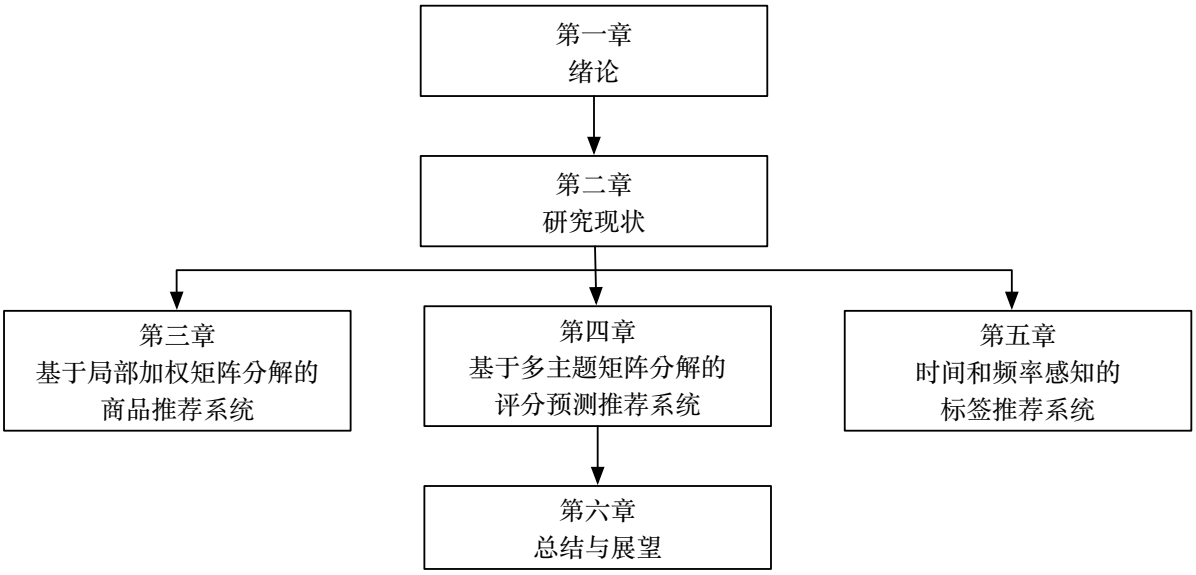


图 1.6: 本文的组织结构

全局低秩的，而其中某些局部子矩阵是低秩的，因此我们提出一个局部加权矩阵分解模型来对局部子矩阵建模。同时，我们提出一个启发式算法进行子矩阵选择去更好地拟合原始数据矩阵。

- 第四章介绍本文的第二个工作，主要介绍对显式反馈数据的局部信息建模以便更好地进行商品评分预测。本章提出了概率图模型和矩阵分解模型的结合算法，巧妙地利用了概率图模型对局部信息建模，然后利用矩阵分解学习每个用户和商品在各个局部子矩阵上的隐藏特征向量。

- 第五章介绍本文的第三个工作，主要针对标签数据的标签推荐。本章在张量分解的基础上增加了时间因素和使用标签频率信息，对用户的使用标签的记忆信息建模，考虑了用户使用标签的行为会随时间变化而变化，并且缓解了标签数据新用户冷启动问题。

- 第六章对上述已有工作进行了总结，并展望了未来的研究方向和内容。

第二章　研究现状

本章首先介绍推荐系统中协同过滤算法，主要分为三类，基于 K 近邻的协同过滤模型，基于矩阵分解的协同过滤模型和基于概率图的协同过滤模型。其中基于矩阵分解模型又可分为逐点矩阵分解模型、逐对矩阵分解模型和逐列矩阵分解模型。然后，我们简单介绍矩阵分解类模型中的一些基本算法，包括概率矩阵分解模型（Probabilisitic Matrix Factorization, 简称 PMF）[3]，加权矩阵分解模型（Weighted Matrix Factorization，简称 WMF） [1, 13, 14] 和成对交互张量分解模型（Pairwise Interaction Tensor Factorization，简称 PITF） [10]。

2.1　协同过滤模型

协同过滤主要思想是，找到相似的用户或商品，预测目标用户对自己没有访问过的商品的感兴趣程度。相对于基于内容的推荐系统，协同过滤能够对一些难以用关键字表达的概念进行建模，且根据相似用户或商品进行过滤，推荐的商品具有一定的新颖性。本节主要讲述协同过滤中的三类常用模型，包括 K 近邻算法、矩阵分解算法和概率图模型。

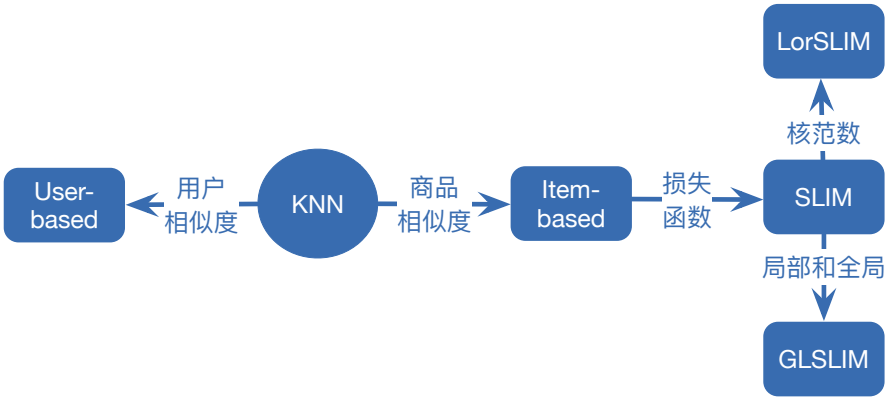


图 2.1: K 近邻模型

2.1.1 基于 K 近邻的协同过滤模型

K 近邻算法（k-Nearest Neighbor，KNN）是机器学习领域经典的通用模型，可以有分类模型和回归模型，主要思想是通过寻找输入样本在训练数据集中的最相似（邻近）的 K 个样本，然后根据这 K 个相似样本的多数类别或者属性平均值作为最终结果。

如图 2.1所示，K 近邻算法在推荐系统中的应用主要由以下几种：userKNN [15–17]，itemKNN [18, 19]，SLIM [20]，LorSLIM [21] 和 GLSLIM [22]。其中主要可以分为两类，基于用户的 K 近邻算法和基于商品的 K 近邻算法。基于用户的 K 近邻算法，顾名思义，例如 userKNN [15–17]，是在训练数据集中计算用户之间的相似度寻找 K 个兴趣相似的用户，根据这 K 个相似用户对某个商品的喜好程度，预测输入用户对这个商品的喜好值。同理，基于商品的 K 近邻算法，例如 itemKNN [18, 19]，是根据商品之间的相似度寻找 K 个相似的商品，根据用户的商品 “访问” 历史记录来进行推荐。这类工作研究的基本问题主要有两个：（1）如何计算用户或者商品之间的相似度；（2）利用得到的用户或者商品之间相似度相似度如何计算得到预测结果。基本的 KNN 算法的相似度计算，如 ItemKNN，是根据商品的被 “访问” 的历史记录形成的数据向量直接计算相似度。这里需要相似度度量或者距离度量来衡量相似程度，常用的度量包括 Pearson 相关系数 [23]，余弦相似度 [24, 25]，Jaccard 相似度 [26] 以及条件概率相似度 [27] 等。对于预测喜好值一般有简单平均，加权平均 [18] 和加权多数预测 [28] 等等。从推荐结果上看，两个基本的 KNN 算法 UserKNN 和 ItemKNN 都可以适用于*评分预测*和*商品推荐*。对于*评分预测*，KNN 算法只需要在计算预测结果值时加入相似用户评分商品或者相似商品被评分的评分值进行预测即可。从推荐领域上看，UserKNN 比较适合新闻的个性化推荐 [23]，而 ItemKNN 适合在电商网站上推荐商品，如亚马逊 [24]。从技术原理角度上看 [29]，因为 UserKNN 需要计算用户相似度矩阵，ItemKNN 需要计算商品相似度矩阵，所以 UserKNN 适合于用户较少的场景，而 ItemKNN 适用于商品的较少的场景。并且，相对于 UserKNN，ItemKNN 是根据用户 “访问” 商品的历史记录推荐，所以只

要用户对新的商品进行 “访问”，就能立即进行个性化推荐，满足实时个性化需求，并给出合理的推荐解释。

与上述两种基本的 KNN 算法不同的是，SLIM [20] 通过拟合目标损失函数来学习商品相似度矩阵，然后根据商品相似度矩阵进行预测推荐结果。具体地，SLIM 主要去最小化损失函数 $||\mathbf{R} - \mathbf{RS}||_F^2$，其中 **R** 是用户商品评分/访问次数矩阵，**S** 是商品相似度矩阵。因为大多数商品之间相似度为 0（也就是相似度矩阵 **S** 是稀疏的）以及防止过拟合，SLIM 还为这个损失函数上还加上了正则化因子 **S** 的 Frobenius 范数 $||\mathbf{S}||_F^2$ 和 L1 范数 $||\mathbf{S}||_1$。并且，相似度需要大于等于 0，SLIM 还加上了相似度矩阵 **S** 所有数值大于等于 0 的约束。可以看出，SLIM 通过通过坐标下降和软阈值算法 [30] 最小化上述损失函数得到商品相似度矩阵 **S**。模型 LorSLIM [21] 认为商品相似度矩阵 **S** 是稀疏的并且是低秩的，因此在在 SLIM 的基础上加入的核范数正则化因子来近似保证相似度矩阵 **S** 是低秩的 [31, 32]，并利用交替方向乘子算法进行优化（Alternating Direction Method of Multipliers，简称 ADMM）[33]。因为核范数的引入，导致需要求解矩阵的奇异值学习参数，使得 LorSLIM 的复杂度高于 SLIM，比较适用于小数据集。SLIM 的另外一个变种是 GLSLIM [22]。GLSLIM 认为不是所有用户都有相同的 “访问” 商品行为，但是其中用户子集存在这种现象，而原本 SLIM 模型只对全局用户进行建模。因此 GLSLIM 考虑了用户的局部信息，建立局部的商品之间的相似度矩阵，与全局商品相似度矩阵结合进行最后的商品推荐。以上三种 SLIM 类原始算法都是需要根据训练数据及学习商品相似度矩阵，都属于基于商品的 K 近邻算法。理论上，经过变换利用损失函数求解用户之间的相似度矩阵成为基于用户的 K 近邻算法也是可能成立的。

2.1.2 基于矩阵分解的协同过滤模型

矩阵分解模型在推荐系统领域中被广泛使用和扩展，适用于*评分预测*和*商品推荐*。它具有较好的正确率和可伸缩性，在各大推荐系统竞赛中扮演着重要的角

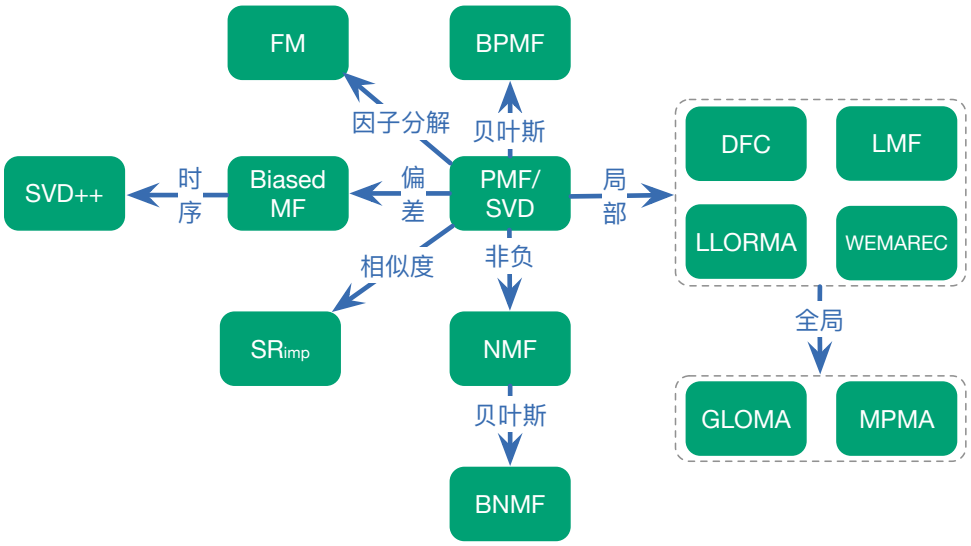


图 2.2: 矩阵分解-评分预测模型

色，例如 *Netflix 百万美元大赛*[1]，*KDD 数据挖掘竞赛 2011 年音乐推荐*[2]，*阿里巴巴大数据竞赛*[3]等等。准确地说，矩阵分解是一种典型的降维技术，将原始评分矩阵（或者次数矩阵）$\mathbf{R}$ 分解上用户隐藏特征矩阵 $\mathbf{P}$ 和商品隐藏特征矩阵 $\mathbf{Q}$，即每个用户 u（或者商品 m）有一个实数向量 $\mathbf{P}_u$（或者 $\mathbf{Q}_m$）表示，而这个向量维度远小于用户或者商品的个数。因此，我们可以使用两个远小于原始数据矩阵的隐藏特征矩阵近似表示原始特征矩阵的内在含义。一般地，用户隐藏特征向量和商品隐藏特征向量的内积表示了这个用户对这个商品的喜好程度。根据模型的用途不同，我们可以将矩阵分解模型分为两大类：（1）评分预测矩阵分解模型；（2）商品推荐矩阵分解模型。因为应用目的不同，两类模型一般不能混用，否则效果可能会比较差，例如在很多文献中，使用 PMF 模型作为商品推荐的基线方法。

2.1.2.1 评分预测模型

首先，我们来简单介绍下评分预测的矩阵分解模型。图 2.2展示了一些应用在评分预测上经典的矩阵分解模型及其扩展。可以看出，其中最基本的模型是 SVD [7, 8] 或者 PMF [3]。推荐系统的 SVD 其实是奇异值分解中的 UV 分解 [34]，以平方误差

[1] http://www.netflixprize.com/
[2] http://www.kdd.org/kdd2011/kddcup.shtml
[3] https://102.alibaba.com/competition/addDiscovery/index.htm

$||\mathbf{I} \odot (\mathbf{R} - \mathbf{P}\mathbf{Q}^\top)||^2$ 作为目标损失函数，其中 $\mathbf{I}$ 是指示函数（当 $\mathbf{R}$ 的项为 0 时，对应 $\mathbf{I}$ 的项也为 0，否则为 1）。一般防止过拟合，SVD 会加入用户和商品隐藏特征向量的 Frobenius 范数正则化因子。对于评分矩阵，SVD 一般使用随机梯度下降学习隐藏特征向量参数。而 PMF 其实是正则化的 SVD 概率版本，它使用高斯分布作为预测评分的分布，相当于 SVD 中的平方误差，然后利用零均值的球形高斯分布作为用户和商品隐藏特征向量的先验，这部分对应 SVD 的正则化因子，从而得到似然函数。我们经过一系列变化后可以发现 PMF 的极大化似然函数相当于最小化 SVD 中带正则化的平方误差损失函数。因此，之后在推荐系统中除非特别说明，我们对 SVD 和 PMF 同等对待。因为多数评分值会受到各自用户和商品的单独影响，例如有些用户对商品的评分都会比其他用户的评分高，有些商品被用户的评分也会比其他商品高，这种现象称之为偏置。因此，BiasedMF [35] 模型在 SVD 的基础上加入了用户偏置 b_u，商品偏置 b_m 和全局平均 u。由于偏置能够捕获更多观察信息，这样使得 BiasedMF 模型在评分预测上一般要好于基本的 SVD。SVD++ [35] 则是考虑了用户对商品 “访问” 历史信息，包括用户的隐式反馈数据和显示反馈数据。SVD++ 将用户历史 “访问” 的商品的隐藏特征向量和用户本身特征向量结合形成用户总的隐藏特征向量，然后类似 BiasedMF 进行评分预测。除了 SVD++ 模型外，有一个基于社交的矩阵分解模型 SR_{imp} [36] 也考虑了用户对商品的历史评价记录。一般地，考虑社交信息的矩阵分解模型，如 [37–42] 这些模型，在基本数据上增加了用户之间的朋友信息或者关注信息等社交信息，它们或者是将社交信息约束在目标函数的正则化中，或者是将其放在矩阵分解中的平方损失函数中增加社交因素对用户评价商品的影响。与这些考虑显式的社交信息的模型不同，SR_{imp} 仅仅是考虑用户对商品的历史评价信息来计算用户之间的相似度，将这个将相似对作为一部分正则化因子放入目标函数之中。另外，基本的推荐数据中除了用户，商品，评分外，一般还包括用户评分商品的时间。而用户对商品的偏好会随时间的变化而变化，商品的受欢迎程度也是会随商品的变化而变化，所以模型 TimeSVD [43] 加入了时间信息，用户和商品在不同的时间段有不同的偏好信息，也就是说有不

同的隐藏特征变量来表示这种不同时间的偏好。一般矩阵分解模型得到的隐藏特征矩阵的数值是实数范围内，可正可负。然而研究工作 [44] 则提出了非负矩阵分解（Non-negative Matrix Factorization, NMF），认为在现实世界中，比如图像，评分等的形成的矩阵中负数的存在是没有意义的，因此在矩阵分解完后的两个特征矩阵也是非负的。研究工作 [45] 进一步考虑 NMF 模型超参数选择的问题，使用了指数分布作为隐藏特征矩阵的先验分布，构造了一个完全贝叶斯模型 BNMF。另外一个全贝叶斯矩阵分解模型是 BPMF [46]，是在 PMF 的零均值球形高斯分布的先验基础上又加入了高斯-维希特分布。贝叶斯模型一般会增加参数学习时间，但在减少超参数设置的同时，有效防止了模型的过拟合现象以及提高了推荐准确率。另外，有一个特殊的模型因子分解机（Factorization Machine，FM) [47] 认为所有的因素（这里指用户和商品）都可以由特征向量表示，可以灵活建模传统线性模型和互异特征之间关系模型（例如用户特征和商品特征的乘积关系），在一定程度上泛化了矩阵分解模型，能够解决评分预测问题和商品推荐问题。基本的矩阵分解模型，如 SVD，可以看成是 FM 模型的一种特殊情况。还有一些工作主要集中在矩阵分解的加速层面，例如分布式矩阵分解 DSGD [48] 和 DNMF [49]，模块化矩阵分解 LMF [50]。

上述的矩阵分解模型都是对训练数据集的全局信息建模，背后的基本假设是评分矩阵是全局低秩的。近年来，有一些工作如 DFC [51]，LLORMA [4, 5] 和 WEMAREC [6]，认为数据评分矩阵不是全局低秩的，而其局部子矩阵是低秩的，并且需要考虑评分矩阵的局部信息。这类方法一般需要从原始评分矩阵选择子矩阵，然后利用基本的矩阵分解模型对子矩阵进行参数学习。其中 LLORMA [4, 5] 利用随机选择数据锚点，利用核范数来寻找与该锚点相似的其他训练数据点组成子矩阵，而 WEMAREC [6] 则利用联合聚类（Co-clustering，简称 CoC) [52] 得到 C_u 个用户聚类和 C_m 个商品聚类，然后交叉获得 $C_u \times C_m$ 个聚类作为子矩阵。进一步的，研究工作 MPMA [53] 和 GLOMA [54] 则认为仅仅考虑局部信息也是不够的，需要结合全局信息和局部信息在矩阵分解中。

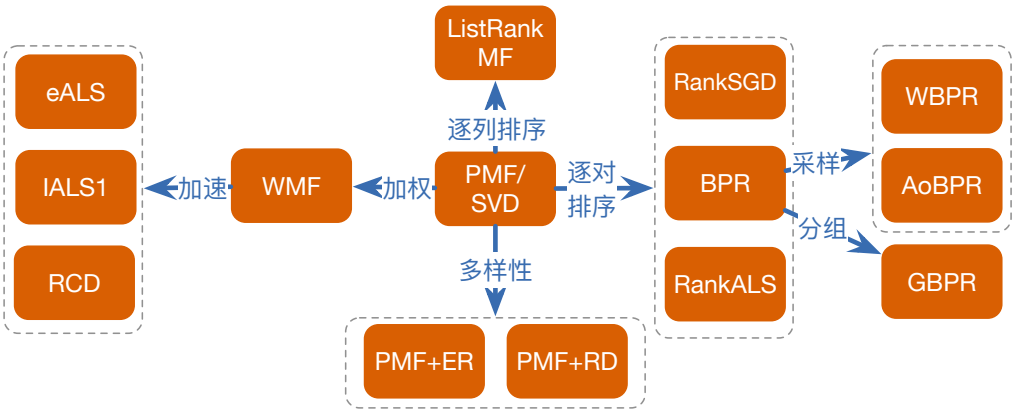


图 2.3: 矩阵分解-商品推荐模型

2.1.2.2 商品推荐模型

接下来我们介绍商品推荐的矩阵分解模型。图 2.3展示了一些商品推荐的矩阵分解模型及其变种。同样的，其中最基本的模型也是是 SVD/PMF [3, 7, 8]。但是，原始的 SVD/PMF 一般只用于评分预测，在商品推荐中效果一般。商品推荐实际上是一个排序问题，因此根据排序学习理论 [55]，我们将商品推荐的矩阵分解模型分为三类：（1）逐点矩阵分解模型；（2）逐对排序矩阵分解模型；（3）逐列排序矩阵分解模型。

逐点矩阵分解模型： 逐点矩阵分解模型是每个数据点 < 用户，商品，评分/次数 > 作为输入空间，直接对数据点进行拟合得到的用户和商品隐藏特征向量。由此可以看出，上一小节中的评分预测的矩阵分解都属于逐点矩阵分解模型。

加权矩阵分解（Weighted Matrix Factorization, 简称 WMF） [1, 13, 14] 是一类典型的逐点矩阵分解模型。商品推荐模型主要是区分用户喜欢的商品与不喜欢的商品。不用于评分矩阵的评分值，评分代表了用户对商品的喜欢程度，没有评分则为缺失值，在次数矩阵中，用户 “访问” 商品的次数代表了用户喜欢该商品的概率。如图 2.4所示，WMF 将次数数据矩阵分为目标矩阵和权重矩阵。目标矩阵表示需要拟合的对象，在次数矩阵数值大于 0 时在目标矩阵中为 1，表示用户喜欢该商品，次数为 0 时在目标矩阵中也为 0。对应的，权重矩阵根据次数的多少表示用户喜欢

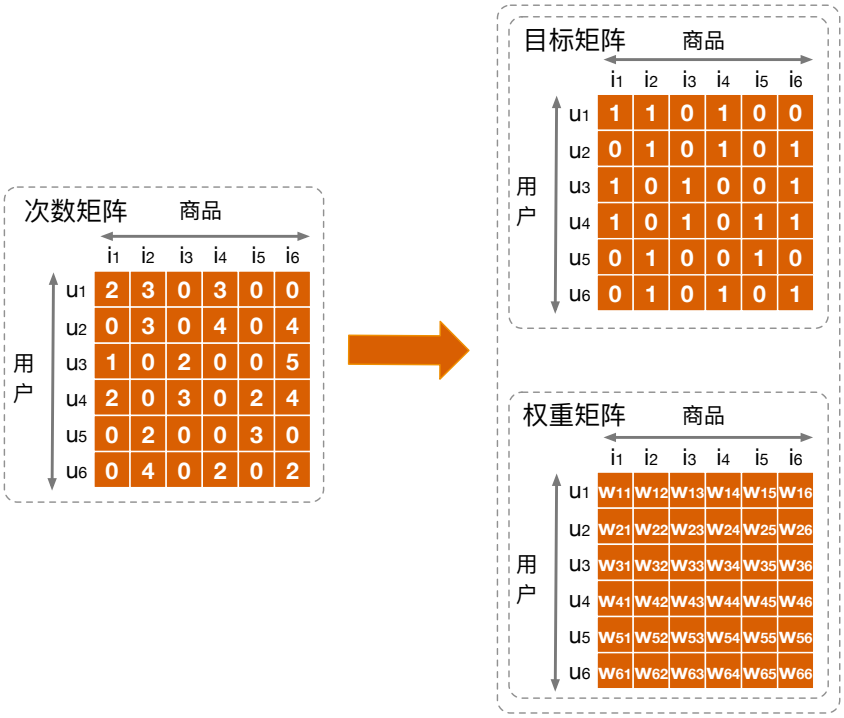


图 2.4: 逐点矩阵分解模型-WMF 模型

商品的概率程度。因此，WMF 在 SVD 的基础上加入权重矩阵，基本的损失函数变为 $||\mathbf{W} \odot (\mathbf{C} - \mathbf{P}^{\top}\mathbf{Q}||^2$。与 SVD 在稀疏矩阵上拟合不同，WMF 需要拟合目标矩阵中的数值 0，因此研究者利用次数为 0 的权重都相同，提出了改进的交替最小二乘法（Alternating Least Square，简称 ALS）[56] 优化算法，提高了训练效率。之后有一些工作 IALS1 [57]，RCD [58]，eALS [59] 等对 WMF 的优化算法进行效率优化，时间复杂度降低 K 倍，K 为隐藏特征向量的维度。IALS1 [57] 利用坐标下降方法提出了一种近似 ALS 的优化算法，在大幅降低学习时间的同时推荐效果没有明显降低。RCD [58] 使用随机区组坐标下降算法，减少时间复杂度的同时适应动态场景，但在每次迭代需要线性搜索选择一个学习速率。研究工作 [59] 提出了元素级交替最小二乘算法 eALS，对学习参数的中间过程变量进行缓存加速优化算法，同时相对于原始 WMF 在缺失值上对不同的商品可以有不同的权重。除了对矩阵分解进行加权方法外，LogDetMC [60] 对特征值做了处理，认为矩阵最大的特征值对矩阵奇异值分解影响过大，所以利用 log 函数降低了最大特征值的权重。LogDetMC 模型需要对矩阵进行真正的奇异值分解，而不是 UV 分解，所以其效率较慢，复杂

度跟矩阵大小成正比。还有一些其他逐点矩阵分解模型，例如 FISMrmse 模型 [61] 考虑了用户 “访问” 商品的历史记录的算法，认为一个用户的隐藏特征向量有由该用户 “访问” 过的商品的隐藏特征向量组成。FISMrmse 的另一个好处是，每当用户新访问了一个商品后能够立刻反应到有该商品组成的用户隐藏特征向量，导致推荐结果的实时变化。另外，近年来一些非矩阵分解的多样性推荐算法 [62, 63] 表明多样性推荐增加了用户的兴趣的二义性和防止推荐列表的冗余性，能够有效地提高推荐有效性。因此，有部分研究工作 PMF+ER [64] 和 PMF+RD [65] 在矩阵分解模型加入了多样性信息（例如利用信息熵来表示多样性），然后利用目标函数的次模性质和单调性来进行推荐。

逐对排序矩阵分解模型： 逐点矩阵分解完全从单一数据点 < 用户，商品，评分/次数 > 的拟合角度出发，没有考虑用户对商品喜好程度顺序出发，与排序无关。而逐对排序矩阵分解是每个用户喜好商品相对排序作为输入空间，对两两商品的喜好排序建模学习用户和商品的隐藏特征向量。

贝叶斯个性化排序模型（Bayesian Personalized Ranking，简称 BPR） [66] 是一个典型的逐对排序矩阵分解模型。如图 2.5所示，BPR 对 “访问” 过的商品作为正样本集合，没有 “访问” 过的商品作为负样本集合，然后每个用户的商品正样本集合和负样本集合两两组成成对排序集合作为输入空间，也就是说 BPR 将次数矩阵转化成了逐对排序矩阵。之后 BPR 以逻辑斯特 sigmoid 函数定义成对排序的概率，这样形成 BPR 的最优化排序准则，结合矩阵分解的思想，使用隐藏特征向量表示用户和商品特征，用户与商品的特征向量内积作为用户对商品的喜好值。因为 BPR 中的输入空间太大，它采用了负采样的随机梯度下降进行参数学习。BPR 对负样本进行均匀采样，但商品的流行度是不一样的，因此有些工作如 WBPR [67] 和 AoBPR [68] 对 BPR 的目标函数或者负采样算法进行优化。WBPR [67] 对负样本中商品被 “访问” 的用户个数和用户 “访问” 商品的个数建立权重，将其加入到 BPR 的排序目标函数中进行参数学习。AoBPR [68] 则是对负样本采样算法进行改进，根据商品的流行度不同建立不均匀分布，以及每隔一段时间动态自适应调整

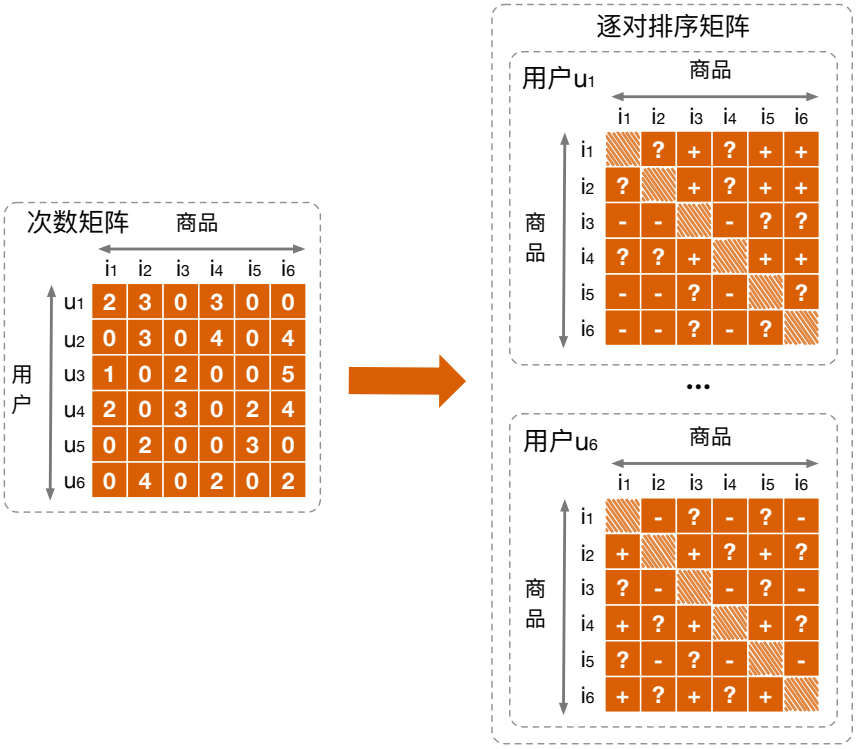


图 2.5: 逐对排序矩阵分解模型-BPR 模型

采样因素，以此解决商品采样的长尾问题。还有一种变种模型 GBPR [69] 在 BPR 基础上增加了一个假设，认为用户 “访问” 商品是因为组内的其他用户 “访问” 了该商品，所以正反馈样本就变成用户组对商品的喜好值代替个人用户对商品的喜好值。BPR 类算法需要最大化逻辑斯特 sigmoid 函数，也就是说最大化每个用户的正反馈商品和负反馈商品之间的差距即可。实际上，BPR 可以说是一个排序优化框架，不止使用在矩阵分解模型性，也有工作将 BPR 使用在张量分解 [10, 70] 中来进行*标签推荐*。具体的*标签推荐*相关技术将在章节 5.2介绍。与 BPR 类算法的不同，RankSGD [71] 和 RankALS [72] 则需要预测值拟合训练数据集中正反馈和负反馈之间的差距，而不是越大越好。其中 RankSGD 使用负采样算法和随机梯度下降算法优化参数，而 RankALS 则利用了 WMF [1] 的优化手段，使用交替最小二乘算法来学习参数。另外，FISMauc [61] 不同于论文中另外一个模型 FISMremse，它结合了逐对排序模型和用户 “访问” 商品的历史记录去最大化曲线下面积（Area Under roc Curve, AUC） [73] 值，同时也有上面提到的 FISMremse 的优点。

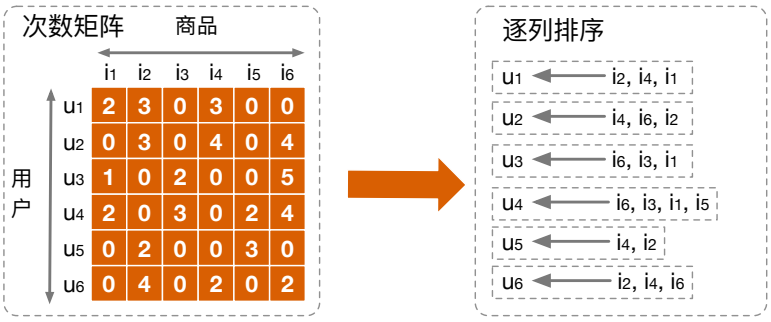


图 2.6: 逐列排序矩阵分解模型

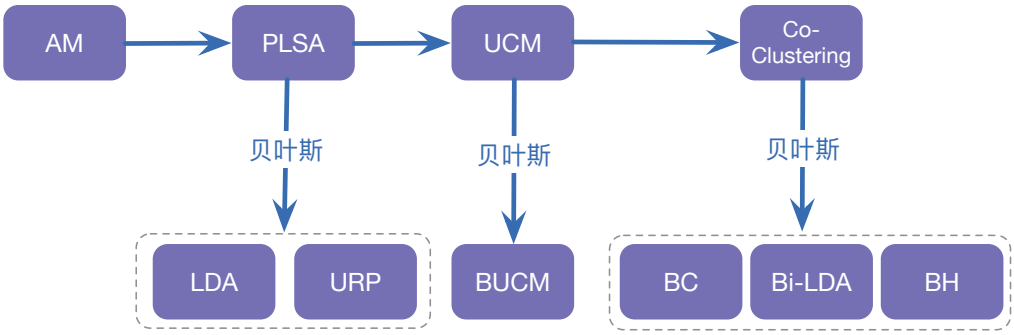


图 2.7: 概率图模型

逐列排序矩阵分解模型： 如图 2.6所示，逐列排序矩阵分解模型是每个用户喜好所有商品排序作为输入空间，也就是次数矩阵转化成每个用户的商品全局排序列表作为训练列表。逐对排序模型中商品相对成对排序不同，一般是最大化 AUC 目标函数。而逐列排序一般是最大化列表排序评价函数。例如 CliMF 模型 [74] 利用了信息检索中国际通用的评价标准平均倒序排名（Mean Reciprocal Rank, 简称 MRR）[75]，提出一个改进的平滑 MRR 作为目标函数。另外一种典型的逐对排序矩阵分解模型是 ListRankMF 算法 [76]，它使用排序学习中第一概率 [77] 的计算方式得到每个用户所 “访问” 过商品的概率，然后利用交叉熵损失函数作为目标函数，体现了训练列表和推荐列表之间的不确定性。

2.1.3 基于概率图的协同过滤模型

概率图模型（Probabilistic Graphical Model, PGM）广泛运用在推荐系统中，对推荐结果具有可解释性。从参数估计角度看，推荐系统中的概率图模型主要分成

两类：（1）极大似然估计模型；（2）贝叶斯估计模型。一般地，极大似然估计模型都会有与之对应的贝叶斯估计模型或者进行先验分布扩展可以得到贝叶斯估计模型。例如，用户和商品的分布一般是多项式分布，因此一般由狄利克雷分布 [78] 作为用户和商品的先验参数分布，从而将极大似然估计模型扩展为贝叶斯估计模型。相对于极大似然估计，贝叶斯估计模型可以减少参数选择对最终结果的影响，并且可以防止模型过拟合。

在隐式反馈数据中，给定第 u 个用户 “访问” 过第 m 个商品，可以有两种预测模型：$p(m|u)$ 和 $p(u,m)$，其中 P 代表概率。第一种模型表示的是用户和商品的联合概率分布。根据概率链式法则，可得 $p(u,m)=p(m|u)p(u)$，是第一种模型和生成第 u 个用户的概率结合。可以看出此类模型，用户 “访问” 的商品数量越多，那么该用户对全局概率的影响越大。AM 模型 [79] 是典型的此类模型，先有一个隐藏因子分布生成一个主题，然后由这个隐藏因子得到隐藏因子-用户分布生成用户和隐藏因子-商品分布生成商品，建立用户和商品之间的联合概率。第二种模型关注点在于给定一个用户，预测该用户选择某个商品的概率。PLSA 模型（Probabilistic Latent Semantic Analysis）[80, 81] 是典型的此类模型，该模型普遍用于文本分析，是文档文本聚类的主题模型。在推荐系统中，一个商品被认为是文档中的一个词，那么一个用户所 “访问” 过的商品集合作为一个文档。因此 PLSA 利用隐藏因子将用户和商品联系起来，首先从用户-隐藏因子多项式分布生成一个隐藏因子，然后根据隐藏因子-商品多项式分布生成一个商品。LDA 模型（Latent Dirichlet Allocation）[82] 则是 PLSA 的贝叶斯扩展，将 PLSA 的多项式分布看作随机变量，引入了狄利克雷分布进行控制。PLSA 模型与 AM 模型是非常类似，不同点是 AM 模型中每个用户有一个隐藏因子与之匹配，而 PLSA 模型中对每个数据点（用户-商品对）分配一个隐藏变量。

在显式反馈数据中，我们假设符号 $\mathbf{R}_{um}$ 代表了第 u 个用户在第 k 个商品的评分。那么给定评分 $\mathbf{R}_{um}$，可以有三种预测模型 [81, 83]：$p(\mathbf{R}_{um},u,m)$，$p(\mathbf{R}_{um}|u,m)$ 和 $p(\mathbf{R}_{um},m|u)$。第一种模型类似于隐式反馈数据中的第一种模型，是对数据点 $<$

用户，商品，评分 > 的联合概率建模。因此，用户 “访问” 的商品数量越多，那么该用户对全局概率的影响也是越大。AM 模型经过简单的扩展生成此类模型 [83]，例如加入隐藏因子-评分分布，与隐藏因子-用户分布和隐藏因子-商品分布相互独立，生成评分。相对于 AM 模型只有一个隐藏因子生成用户和商品，FMM 模型（Flexible Mixture Model） [84] 则有两个不同的隐藏因子分别关联用户和商品，由隐藏因子-用户分布生成用户和隐藏因子-商品分布生成商品，以及根据这两个隐藏因子得到评分分布建模评分。用户和商品分别关联一个隐藏因子，我们一般称此类模型为 CoC 模型（Co-Clustering）。ACCAMS 模型 [85] 结合了 CoC 模型和提升技术，对 CoC 模型线性组合，能够更好地考虑用户和商品各个维度的特征。第二种模型称为强制预测模型（Forced Prediction），是对给定一个用户和一个商品，预测该用户对该商品的评分。也就是给定第 u 个用户在第 m 个商品的评分 $\mathbf{R}_{um}$，强制预测模型关心的是产生此样本的概率 $p(\mathbf{R}_{um}|u,m)$。可以看出，强制预测模型适用于显式反馈数据中的*评分预测*任务，也就是对用户的显式偏好建模。上述的许多工作通过扩展得到强制预测模型。例如 PLSA 模型，研究工作 [86] 在此基础上利用多项式分布对评分建模，另外一种选择是使用高斯分布对评分建模的 GPLSA [87]。研究工作 [88] 提出的 URP 模型（User Rating Profile）则是在 LDA 模型的基础上加入关联主题和商品的多项式分布来生成评分。BMM 模型（Block Mixture Model） [89, 90] 是一种属于强制预测模型的 CoC 模型，由用户-隐藏因子分布（商品-隐藏因子分布）生成用户（商品）相关的隐藏因子，根据这两个隐藏因子得到评分的分布生成评分。除了 BMM 模型外，还有许多贝叶斯估计版本的 CoC 模型属于强制预测模型，例如 Bi-LDA 模型 [91]，BCoC 模型（Bayesian Co-Clustering） [92]，RBC 模型（Residual Bayesian Co-Clustering） [93] 和 BH-Forced 模型 [94] 等。其中 Bi-LDA 模型 [91] 和 BCoC 模型 [92] 是结合了 URP 模型和 CoC 思想，强调了用户和商品的联合聚类性质。RBC 模型 [93] 则是扩展了高斯版本的 URP 模型，并对最终评分加入了偏置参数。BH-Forced 模型 [94] 认为不同的用户对商品的关注点是不同的，因此商品的隐藏因子与用户的隐藏因子和商品本身相关，给定这两者生

成商品的隐藏因子。第三种模型称为自由预测模型（Free Prediction），是对给定第 u 个用户，预测该用户在第 m 商品上的评分为 $\mathbf{R}_{um}$，也就是 $p(\mathbf{R}_{um}, m|u)$。通过链式法则，我们可以有 $p(\mathbf{R}_{um}, m|u) = p(\mathbf{R}_{um}|u, m)p(m|u)$，除了上述强制预测模型部分的对显式偏好建模外还加入了用户的隐式偏好部分。因此，自由预测模型主要是在显式反馈数据上做商品推荐任务。UCM 模型（User Communities Model）[95] 是一个单隐藏因子的自由预测模型，利用一个全局的用户-隐藏因子分布生成每个用户的隐藏因子，然后根据这个隐藏因子生成商品，最终给定这个隐藏因子和商品利用多项式分布或者高斯分布生成评分。进一步地，研究工作 [96] 提出 UCM 的贝叶斯扩展 BUCM 模型。研究工作 [94] 不仅提出了强制预测模型 BH-Forced，同时也提出了自由预测模型 BH-free。BH-free 模型也考虑了 BH-Forced 模型的关注点，同时类似于 UCM 模型考加入了商品显式生成过程，与用户的隐藏因子和商品的隐藏因子相关。

协同过滤中这三类模型各有优缺点。K 近邻模型可以利用用户的历史行为对推荐结果做出解释，并且基于商品的 K 近邻算法还能因为用户新 “访问” 商品的行为对推荐结果进行实时变化，但在线计算复杂度较高，并且需要全局数据导致空间复杂度也较高。矩阵分解类模型推荐性能较好，模型能够离线计算，时间和空间复杂度较低，并且具有非常好的复杂性，但是推荐结果不具有较好的解释性，较难解释矩阵分解后得到的隐藏特征向量的物理意义。概率图模型强于推荐结果可解释性和模型灵活性。它对数据进行软聚类，能够很好解释用户和商品所在类别，说明用户在哪一维度上喜欢某个商品。并且，概率图模型更够灵活的结合其他类别数据，例如主题模型在文本数据上的医用，进一步增加推荐的准确率和可解释性。本文主要从矩阵分解模型角度出发，利用数据的局部信息建模，提高推荐准确率。部分工作（章节 四）还结合了概率图模型提高推荐准确率的同时，增加矩阵分解模型的可解释性。

2.2 基本矩阵分解模型介绍

本节将介绍本文将会使用的基础矩阵分解模型。章节2.2.1介绍适用于推荐系统中的评分预测的概率矩阵分解模型（PMF）。章节2.2.2介绍用于商品推荐的加权矩阵分解模型（WMF）。最后，章节2.2.3介绍成对交互张量分解模型（PITF），是将逐对排序矩阵分解模型中的 BPR 模型思想扩展到三维张量分解，适用于标签推荐。

首先，我们先描述下本文所使用的基本符号表示，如表2.1所示。具体地，除了特别说明，我们统一使用正常字体大小写字母表示标量（一般大写字母表示数量，小写字母表示索引），粗体大写字母表示矩阵，以及手写体大写字母表示集合。矩阵和集合的字符的小写上标，例如 $\mathbf{R}^h$ 和 $\mathcal{N}^h$，表示不同的子矩阵和不同的子集合。矩阵的下标表示该矩阵的索引。例如，$\mathbf{R}^h_{um}$ 表示原始数据矩阵 $\mathbf{R}$ 的第 h 个子矩阵 $\mathbf{R}^h$ 中第 u 个用户，第 m 个商品的值，如果是单个下标，如 $\mathbf{R}_u$，表示第 u 个用户“访问” 商品的评分/次数向量。此外，使用上标 $^\top$ 表示横向量，否则表示为竖向量。

2.2.1 概率矩阵分解模型

概率矩阵分解（PMF）[3] 是带有正则化的奇异值分解（SVD）的概率版本，在推荐系统中尤其在评分预测中被广泛使用。假设评分数据矩阵 $\mathbf{R} \in \mathbb{R}^{N \times M}$，权重矩阵 $\mathbf{W}$ 和隐藏特征向量 $\mathbf{P}$ 和 $\mathbf{Q}$。因为评分预测中一般在测试集上使用均方根误差（Root Mean Square Error，简称 RMSE）作为评测标准，因此 PMF 采用高斯分布作为可观察到评分的条件分布，公式如下表示：

$$p(\mathbf{R}|\mathbf{P},\mathbf{Q},\sigma^2) = \prod_{u=1}^{N}\prod_{m=1}^{M}[\mathcal{N}(\mathbf{R}_{um}|\mathbf{P}_u^\top \mathbf{Q}_m, \sigma^2)]^{\mathbf{O}_{um}} \tag{2.1}$$

其中 $\mathcal{N}(x|\mu,\sigma^2)$ 是均值为 μ，方差为 σ^2 的高斯概率密度函数，以及标识矩阵 $\mathbf{O}_{um}$ 在第 u 个用户对第 m 个商品打过分则为 1，否则为 0，来标识用户是否 “访问” 商品。同时为了避免学习参数过拟合，PMF 加入了零均值球面高斯先验 [97, 98] 作为用户和商品隐藏特征向量的先验分布：

表 2.1: 本文基本符号说明

符号	符号描述
u	用户索引
m	商品索引
t	标签索引
k	隐藏特征向量维度索引 ($\ll \min(N, M)$)
h	子矩阵索引或者主题模型中的主题索引
N	用户数量
M	商品数量
T	标签数量
K	隐藏特征向量的维度 ($\ll \min(N, M)$)
H	子矩阵的个数或者是主题模型中主题个数
$\mathbf{R}$	原始数据矩阵（< 用户，商品，次数/评分 >）($\in \mathbb{R}^{N\times M}$)
$\hat{\mathbf{Y}}$	三维张量标签数据矩阵 ($\in \mathbb{R}^{N\times M\times T}$)
$\mathcal{P}$	数据矩阵中 $\mathbf{R}$ 用户索引集合
$\mathcal{Q}$	数据矩阵中 $\mathbf{R}$ 商品索引集合
$\mathcal{T}$	数据矩阵中 $\mathbf{R}$ 标签索引集合
$\mathbf{I}$	单位矩阵
$\mathbf{C}$	原始次数数据矩阵 $\mathbf{R}$ 的 01（二值化）化数据矩阵 ($\in \mathbb{R}^{N\times M}$)
$\mathbf{W}$	数据矩阵的置信权重矩阵 ($\in \mathbb{R}^{N\times M}$)
$\mathbf{O}$	数据矩阵的标识矩阵 ($\in \mathbb{R}^{N\times M}$)，当 $\mathbf{R}_{um} > 0$ 时，$\mathbf{O}_{um} = 1$，否则 $\mathbf{O}_{um} = 0$
$\mathbf{P}$	用户的隐藏特征向量 ($\in \mathbb{R}^{N\times K}$)
$\mathbf{Q}$	商品的隐藏特征向量 ($\in \mathbb{R}^{M\times K}$)
$\mathbf{T}^{\mathcal{P}}$	与用户对应的标签隐藏特征向量 ($\in \mathbb{R}^{T\times K}$)
$\mathbf{T}^{\mathcal{Q}}$	与商品对应的标签隐藏特征向量 ($\in \mathbb{R}^{T\times K}$)
$\mathcal{A}$	数据点集合（用户-商品记录对集合，不包含用户未 “访问” 商品的情况）
$a_i = \langle u_i, m_i\rangle$	数据点 $\langle u_i, m_i\rangle$ (用户-商品记录对, $\in \mathcal{A}$)
λ	正则化参数

$$p(\mathbf{P}|\sigma_{\mathbf{P}}^2) = \prod_{u=1}^{N} \mathcal{N}(\mathbf{P}_u|0, \sigma_{\mathbf{P}}^2) \tag{2.2}$$

$$p(\mathbf{Q}|\sigma_{\mathbf{Q}}^2) = \prod_{m=1}^{M} \mathbf{N}(\mathbf{Q}_m|0, \sigma_{\mathbf{Q}}^2) \tag{2.3}$$

我们需要学习用户和商品隐藏特征向量，因此需要最大化用户和商品隐藏特征矩阵的后验概率，如下表示：

$$\begin{aligned} & p(\mathbf{P}, \mathbf{Q}|\mathbf{R}, \sigma^2, \sigma_{\mathbf{P}}^2, \sigma_{\mathbf{Q}}^2) \\ & = \prod_{u=1}^{N} \prod_{m=1}^{M} [\mathcal{N}(\mathbf{R}_{um}|\mathbf{P}_u^\top \mathbf{Q}_m, \sigma^2)]^{\mathbf{O}_{um}} \prod_{u=1}^{N} \mathcal{N}(\mathbf{P}_u|0, \sigma_{\mathbf{P}}^2) \prod_{m=1}^{M} \mathbf{N}(\mathbf{Q}_m|0, \sigma_{\mathbf{Q}}^2) \end{aligned} \tag{2.4}$$

将高斯概率密度函数公式代入公式 2.4，log 函数后化简可得：

$$\ln p(\mathbf{P}, \mathbf{Q}|\mathbf{R}, \sigma^2, \sigma_{\mathbf{P}}^2, \sigma_{\mathbf{Q}}^2) = \tag{2.5}$$

可以看出，公式 2.5中令，其实最小化带有正则化因子的平方误差目标函数，也就是正则化的 SVD：

$$\sum_{u=1}^{N} \sum_{m=1}^{M} \mathbf{O}_{um}(\mathbf{R}_{um} - \mathbf{P}_u^\top \mathbf{Q}_m)^2 + \lambda_{\mathbf{P}} \|\mathbf{P}\|_F^2 + \lambda_{\mathbf{Q}} \|\mathbf{Q}\|_F^2 \tag{2.6}$$

其中超参数 $\lambda_{\mathbf{P}}$ 和 $\lambda_{\mathbf{Q}}$ 是正则化参数，用来控制隐藏特征矩阵大小的重要程度。一般地，随机梯度下降 [8] 常被用来学习稀疏矩阵分解的参数，对非零值样本 $\mathbf{R}_{um}$ 的迭代优化公式如下：

$$\mathbf{P}_{uk} \leftarrow \mathbf{P}_{uk} + \iota((\mathbf{R}_{um} - \mathbf{P}_u^\top \mathbf{Q}_m)\mathbf{Q}_{mk} - \lambda_{\mathbf{P}} \mathbf{P}_{uk}) \tag{2.7}$$

$$\mathbf{Q}_{mk} \leftarrow \mathbf{Q}_{mk} + \iota((\mathbf{R}_{um} - \mathbf{P}_u^\top \mathbf{Q}_m)\mathbf{P}_{uk} - \lambda_{\mathbf{Q}} \mathbf{Q}_{mk}) \tag{2.8}$$

其中 ι 是学习速率，$\mathbf{P}_{uk}$ 表示第 u 个用户的隐藏特征向量第 k 个数值，对应 $\mathbf{Q}_{mk}$ 则表示第 m 个商品的隐藏特征向量第 k 个数值。

因此，评分数据的矩阵分解（例如本节的 PMF）就是将评分矩阵分解成用户和商品隐藏特征矩阵，使得而这两个隐藏特征矩阵的相乘可以近似还原原始评分矩阵，如图 2.8所示。

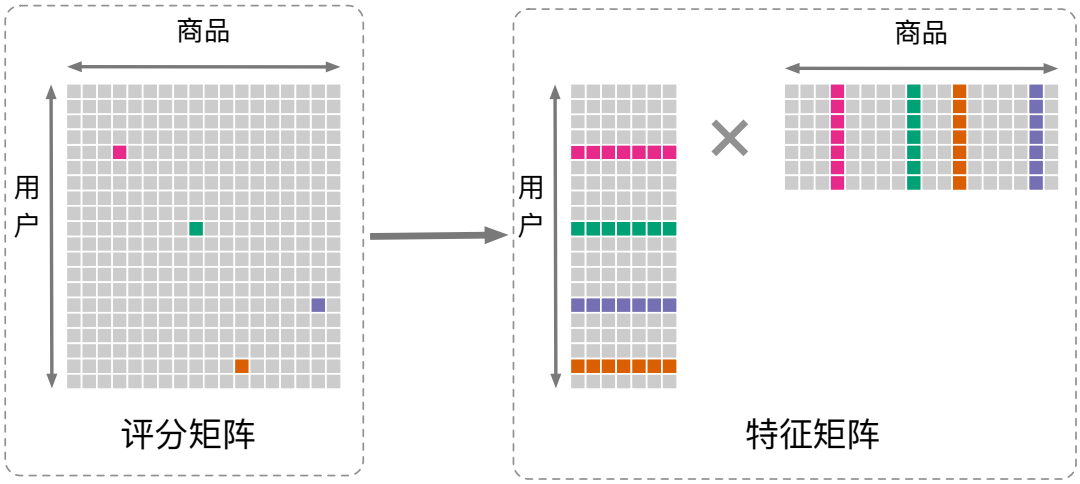


图 2.8: 基本矩阵分解

2.2.2 加权矩阵分解模型

上一小节简述了基本矩阵分解在显式反馈数据上做*评分预测*。本小节简单介绍下矩阵分解在隐式反馈数据上做*商品推荐*任务。正如章节 1.2所述，*显示反馈数据*主要是用户对已消费的商品进行显式的反馈，例如评分，评论等数据，能够显式地说明用户是否喜欢该商品。而*隐式反馈数据*则是表示用户只有 “访问” 商品的记录，例如购买，观看，收听等记录次数，但是没有显式地说明用户喜不喜欢改商品，消费次数越多说明的是用户喜欢该商品的可能性越多。而在*商品推荐*任务中，推荐系统的首要目的是区分用户会 “访问” 的商品和不会 “访问” 的商品，然后在用户 “访问” 过的商品中区分用户商品的喜欢程度。所以在*商品推荐*中，推荐系统中不仅需要考虑用户 “访问” 商品的数据，也要考虑用户未 “访问” 商品的数据。Hu[1] 和 Pan[13, 14] 等人认为基础的矩阵分解（例如 PMF）比较适用于*显式反馈数*

据，特别是评分数据，但是不能直接适用于隐式反馈数据。因此他们同时提出了加权矩阵分解（WMF）对隐式反馈数据建模进行商品推荐。近段时间，WMF 被广泛用于新闻 [99]，电视节目 [100]，音乐 [101] 和兴趣点（Point Of Interest，简称 POI）[102, 103] 的推荐。具体地，为了区分用户 “访问” 过的商品和未 “访问” 过的商品，权重矩阵 $\mathbf{W}$ 被加入到矩阵分解中：

$$\mathbf{W}_{um} = 1 + log(1 + \mathbf{R}_{um} \times 10^{\varepsilon}) \tag{2.9}$$

其中常数 ε 是用来控制权重增量速率。考虑到隐式反馈数据的权重，优化目标函数重写为如下表示：

$$\min_{\mathbf{P},\mathbf{Q}} \sum_{u=1}^{N} \sum_{m=1}^{M} \mathbf{W}_{um}(\mathbf{C}_{um} - \mathbf{P}_u^{\top}\mathbf{Q}_m)^2 + \lambda_{\mathbf{P}}\|\mathbf{P}\|_F^2 + \lambda_{\mathbf{Q}}\|\mathbf{Q}\|_F^2 \tag{2.10}$$

其中 $\mathbf{C}_{um}$ 是 0/1 值，表示第 u 个用户是否消费过第 m 个商品，定义如下：

$$\mathbf{C}_{um} = \begin{cases} 1 & \mathbf{R}_{um} > 0 \\ 0 & \mathbf{R}_{um} = 0 \end{cases}. \tag{2.11}$$

因为加权矩阵分解是对矩阵 $\mathbf{C}$（包括 0 值数据）进行拟合，使用随机梯度下降容易过拟合以及学习速率不容易选择，因此 Hu 等人 [1] 利用交替最小二乘（Alternating Least Square，简称 ALS）进行学习模型参数。因为将所有 0 值数据的权重都设置为 1，即当 $\mathbf{R}_{um} = 0$ 时，$\mathbf{W}_{um} = 1$，他们对 ALS 进行了改进，缓存了部分中间数据，加速了算法的优化，具体迭代优化算法如下：

$$\mathbf{P}_u \leftarrow (\mathbf{Q}^{\top}\mathbf{Q} + \mathbf{Q}^{\top}(\mathbf{W}^u - \mathbf{I})\mathbf{Q} + \lambda_{\mathbf{P}}\mathbf{I})^{-1}\mathbf{Q}^{\top}\mathbf{W}^u\mathbf{C}_u \tag{2.12}$$

$$\mathbf{Q}_m \leftarrow (\mathbf{P}^{\top}\mathbf{P} + \mathbf{P}^{\top}(\mathbf{W}^m - \mathbf{I})\mathbf{P} + \lambda_{\mathbf{Q}}\mathbf{I})^{-1}\mathbf{P}^{\top}\mathbf{W}^m\mathbf{C}_m \tag{2.13}$$

其中 $\mathbf{I}$ 是单位矩阵，对角矩阵 $\mathbf{W}^u \in \mathbb{R}^{n\times n}$ 表示对角上第 u 个用户对每个商品的权重（即 $\mathbf{W}^u_{mm} = \mathbf{W}_{um}$），同理对角矩阵 $\mathbf{W}^m \in \mathbb{R}^{m\times m}$。可以看出，对每个用户（或者商品），公式项 $\mathbf{Q}^\top\mathbf{Q}$（或者 $\mathbf{P}^\top\mathbf{P}$）都是相同的，在每轮迭代开始，可以提前计算缓存，从而加速优化。

2.2.3 成对交互张量分解模型

*标签推荐*是在用户对商品进行描述时进行推荐标签，方便用户输入标签，同时可以提高标签质量，利于后期的商品推荐。与*商品推荐*的不同，*标签推荐*多了对标签的建模，因此传统两维的矩阵分解模型已经不适用，一般使用三维的张量分解，例如 Tucker 分解 [70] 和成对交互张量分解模型（PITF） [10]，进行标签推荐。本小节将介绍成对交互张量分解模型（PITF）。

给定用户-商品对 $< u, m >$，标签推荐就是对所有标签 $t \in T$ 预测第 u 个用户使用第 t 个标签来标注第 m 个商品的可能性大小 $\hat{y}_{umt}$，然后进行降序排序取 Top-n 个标签作为推荐列表，如下形式化表示：

$$Top(u, m, n) := \arg\max^{n}_{t\in T} \hat{y}_{umt} \tag{2.14}$$

因此，*标签推荐*类似于*商品推荐*，实际上是一个排序问题。假定第 u 个用户在第 m 个商品上的喜好标签排序列表为 $>_{u,m} \subset \mathcal{T} \times \mathcal{T}$，*标签推荐*的目标就是最大化如下概率公式：

$$p(\Theta| >_{u,m}) \propto p(>_{u,m} |\Theta)p(\Theta) \tag{2.15}$$

其中 Θ 指代模型的参数。为了达到上述目标，PITF 模型使用 BPR 优化准则，最大化后验概率（MAP）：

$$
\begin{aligned}
BPR-Opt: &= \ln \prod_{(u,m,t_A,t_B)\in\mathcal{D}_{\hat{\mathbf{Y}}}} \rho(\hat{y}_{umt_A} - \hat{y}_{umt_B}) p(\Theta) \\
&= \sum_{(u,m,t_A,t_B)\in\mathcal{D}_{\hat{\mathbf{Y}}}} \ln \rho(\hat{y}_{umt_A} - \hat{y}_{umt_B}) - \lambda ||\Theta||_F^2
\end{aligned} \tag{2.16}
$$

其中 $\rho(x)$ 为 sigmod 函数 $\frac{1}{1+e^{-x}}$，$p(\Theta)$ 是表示参数服从的先验分布，λ_Θ 为正则化参数，$\mathcal{D}_{\hat{\mathbf{Y}}}$ 是使用逐对排序从原始标签张量矩阵构建训练数据的集合，以及 (u, m, t_A, t_B) 表示第 u 个用户给第 m 个商品标注第 t_A 个标签的概率大于第 t_B 个标签。因为训练数据集 $\mathcal{D}_{\hat{\mathbf{Y}}}$ 太过于庞大，PITF 一般使用负采样随机梯度下降进行参数学习。

PITF 是一种过特殊的张量分解模型，用隐藏特征向量表示用户、商品、标签，显式地对用户、商品、标签三者两两之间的关系进行建模：

$$
\hat{y}_{umt} = \mathbf{P}_u^\top \mathbf{T}_t^{\mathcal{P}} + \mathbf{Q}_m^\top \mathbf{T}_t^{\mathcal{Q}} \tag{2.17}
$$

其中 $\mathbf{P}_u$ 是第 u 个用户的隐藏特征向量，$\mathbf{T}_t^{\mathcal{P}}$ 是用户-标签关系中第 t 个标签的隐藏特征向量，$\mathbf{T}_t^{\mathcal{Q}}$ 是商品-标签关系中第 t 个标签的隐藏特征向量，$\mathbf{Q}_m^\top$ 是第 m 个商品的隐藏特征向量。值得注意的是因为给定用户-商品对 $< u, m >$ 时，用户-商品之间的交互关系对最终个性化标签推荐列表排序没有影响，所以公式 2.17中没有显式建模用户-商品的关系。

第三章　基于局部加权矩阵分解的商品推荐系统

本章主要介绍针对隐式反馈数据上商品推荐的研究工作。首先，章节 3.1阐述本章的研究背景、研究问题、研究主要贡献等。其次，章节 3.3重点介绍提出的模型，包括模型框架、优化算法等。随后，章节 3.4分析基于公开真实世界数据集上的实验结果。最后小结本章商品推荐的研究内容。

3.1　引言

商品推荐是推荐系统中的主要任务，是根据用户的历史行为数据，刻画用户画像，帮助人们在海量商品中发现他们感兴趣的商品。而用户的历史数据中按照用户的行为分为显式反馈数据和隐式反馈数据。其中显式反馈数据指的是用户在访问商品（例如购买商品，观看电影，餐馆用餐等行为）之后对商品进行评分表达用户的满意程度，标注标签表达商品的特性以及文本评价说明用户的体验感受。而隐式反馈数据指的是用户访问商品之后没有对商品进行显式评价的行为。因为显式反馈数据需要用户额外花一些时间进行评价，所以用户产生的大部分数据都是隐式反馈的。本章主要研究在隐式反馈数据集上的商品推荐。

隐式反馈数据中，对于用户访问过的商品，次数代表了用户对商品的感兴趣程度，可以作为正反馈样例。大多数用户未访问过商品的数据对学习用户特征也是有帮助的，一般可以作为负反馈样例，但是次数为 0 代表了两种情况：（1）用户对这类商品不感兴趣；（2）用户对这类商品感兴趣，但还未发现此类商品。因此 Hu 等人 [1] 提出了加权矩阵分解（Weighted Matrix Fatorization，简称 WMF），访问次数大于 0 的数据为正反馈样本，次数等于 0 的样本为负样本，而次数则以权重的形式加入矩阵分解中（访问次数越多，权重越大，说明访问次数大于 0 的数据为正反馈样本的概率越大）。具体地，WMF 已经在章节 2.2.2详细阐述。因为 WMF 模型是将用户和商品投射到潜在的低维空间，所以它的一个前提假设隐式反馈数据中

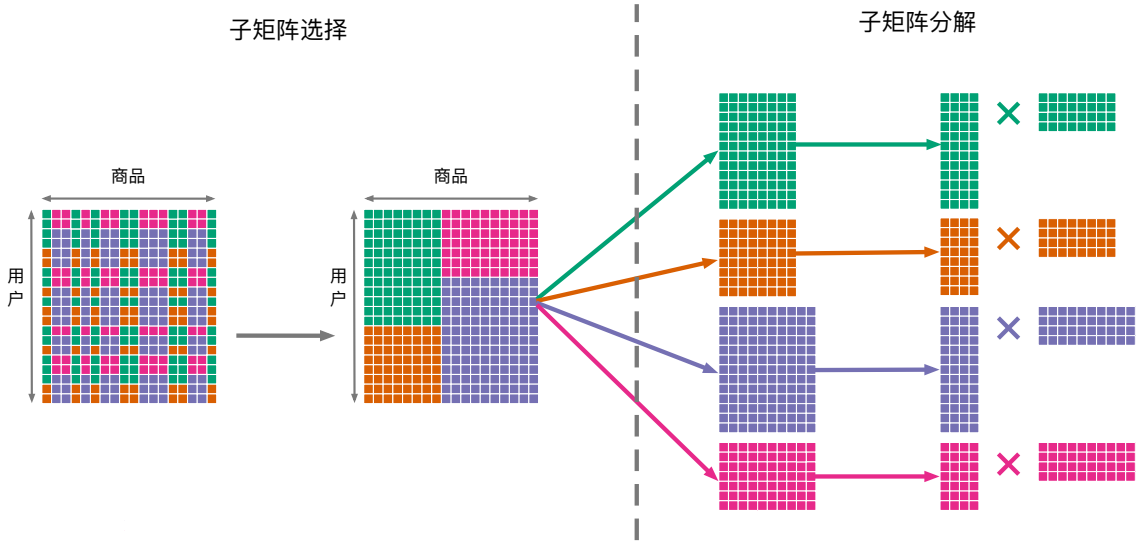


图 3.1: 局部矩阵分解

次数数据矩阵是全局低秩的。最近，Lee 等人 [4, 5] 认为显式评分数据不是全局低秩的，而局部评分矩阵是低秩，拥有局部信息，通过对评分矩阵中的局部子矩阵进行建模能够得到更好的评分预测效果。类似的工作还有 WEMAREC [6]。我们统称这类工作为局部矩阵分解。如图 3.1所示，局部矩阵分解模型首先将原始矩阵划分为若干更小的子矩阵，每个子矩阵内的用户（商品）之间是相互关联的来表示矩阵内的局部结构，然后利用标准的矩阵分解模型将每个子矩阵的用户和商品投射到潜在向量，最后利用加权平均求和来预测最终的评分值。这样，我们就可以利用局部结构特征来获得比原始矩阵分解更好的低秩近似。

因此，考虑到隐式反馈数据和显示反馈数据的相似性，本章节也认为隐式反馈数据也不是全局低秩的，而是局部低秩的，考虑在隐式反馈数据集上对数据的局部性质建模，进行个性化商品推荐。例如，根据用户的签到数据集进行兴趣点（Point Of Interests，简称 POI）推荐，因为天然的兴趣点地理信息的特征的关系，用户更容易访问自身相近的地点，所以在同一个区域的用户访问的兴趣点比不在同一个地点的用户所访问的兴趣点更加类似。

因此，本章节设计了加权局部矩阵分解（Local Weighted Matrix Factorization，简称 LWMF)，将 LLORMA [4, 5] 和 WMF [1] 相结合，使用核函数来寻找子矩阵

和建立权重函数来建模用户局部偏好和商品的局部特性，提高商品推荐的准确率。这种方法也带来两个好处，(1) 因为子矩阵的密度比原始矩阵要高很多，隐式数据中正反馈数据的稀疏性问题也得到一定程度地缓解；(2) 子矩阵之间的分解可以并行进行，使得模型更加容易并行化。

具体地，本章主要贡献总结如下:

- 本章节工作结合了 LLORMA 和 WMF,提出了加权局部矩阵分解模型(LWMF),在隐式反馈数据上进行商品推荐。LWMF 通过将原始矩阵划分为子矩阵来对矩阵进行局部建模，并且减轻次数数据的稀疏性问题以及使得矩阵分解模型更加容易并行化、分布式进行模型训练。

- 为了更好地近似原始矩阵，本章节工作在基于核函数的方法上，提出 DCGASC（折扣累积增益锚点集合覆盖）来选择子矩阵。同时，本章节工作还在在理论上进行 DCGASC 目标函数的次模性质和单调性进行分析，证明利用贪心算法能够以 $1-\frac{1}{e}$ 得到近似最优解。

- 基于商品推荐问题，本章节工作进一步提出了一种基于用户的 LWMF 的变体方法，基于用户的 LWMF 和基于商品的 LWMF，其中基于用户的 LWMF 对商品推荐更为合理，并获得更好的性能。

- 本章节工作在真实数据集上进行广泛的实验，在各个维度上比较了 LWMF 与 WMF 算法，实验结果表明本章节模型的有效性。

3.2 模型基础知识

在表3.1中列出了纸张中使用的符号的词汇表。具体地，我们使用大写粗体字母表示矩阵，手写体表示集合。不同字符的上标，例如 $\mathbf{R}^h$ 和 $\mathcal{U}^h$，表示不同的子矩阵和不同的子集合。矩阵的小标表示该矩阵的索引。例如，$\mathbf{R}^h_{um}$ 表示原始数据矩阵 $\mathbf{R}$ 的第 h 个子矩阵 $\mathbf{R}^h$ 中第 u 个用户，第 m 个商品的值。此外，使用上标 $^\top$ 表示横向量，否则则为竖向量。

表 3.1: 本章主要符号说明

符号	符号描述
$\mathbf{R}^h$	第 h 个子数据矩阵
$\mathbf{C}^h$	第 h 个 01（二值化）化子数据矩阵
$\mathbf{W}^h$	01（二值化）化子数据矩阵 $\mathbf{C}^h$ 的置信权重矩阵
$\mathbf{T}^h$	01（二值化）化子数据矩阵 $\mathbf{C}^h$ 的子矩阵权重矩阵
$\mathcal{P}^h$	数据子矩阵中用户索引集合
$\mathcal{Q}^h$	数据子矩阵中商品索引集合
$\mathbf{P}^h$	子数据矩阵 $\mathcal{R}^h$ 中用户局部隐藏矩阵
$\mathbf{Q}^h$	子数据矩阵 $\mathcal{R}^h$ 商品局部隐藏特征矩阵
$\hat{\mathcal{A}}$	锚点集合 ($\subset \mathcal{A}$)
$\hat{a}_h = \langle \hat{u}_h, \hat{m}_h \rangle$	锚点 ($\in \hat{\mathcal{A}}$)
$E(a_i, a_j)$	两个数据点之间的核函数值

3.3 局部加权矩阵分解

在本章节，我们将介绍我们提出的模型局部加权矩阵分解（Local Weighted Matrix Factorization, LWMF），以及提出了一个启发式的方法去选择子矩阵。最后，我们采用了基于元素的快速交替最小二成优化算法去学习局部隐藏变量 $\mathbf{P}^h$ 和 $\mathbf{Q}^h$。

3.3.1 模型概览

类似于 LLORMA，我们首先从原始数据矩阵选择子矩阵，然后利用 WMF 对子矩阵进行矩阵分解。图3.1展示了这个过程。我们结合了 LLORMA 和 WMF 在隐式反馈数据集上进行推荐商品推荐。我们提出加权矩阵分解，利用 WMF 对 0/1 数据子矩阵进行如下分解：

$$\min_{\mathbf{P}_h, \mathbf{Q}_h} \sum_{u=1}^{N} \sum_{m=1}^{M} \mathbf{T}^h_{um} \mathbf{W}^h_{um} (\mathbf{C}^h_{um} - \mathbf{P}^{h\top}_u \mathbf{Q}^h_m)^2 + \lambda^h_{\mathbf{P}} \|\mathbf{P}^h\|^2_F + \lambda^h_{\mathbf{Q}} \|\mathbf{Q}^h\|^2_F \tag{3.1}$$

其中 $\lambda_{\mathbf{P}}^{h}$ 和 $\lambda_{\mathbf{Q}}^{h}$ 是子矩阵用户和商品的正则化因子。因此原始二值化数据矩阵 $\mathbf{C}$ 可以被一系列子矩阵集合 $\mathcal{C}=\{\mathbf{C}^1,\mathbf{C}^2,...,\mathbf{C}^H\}$ 所组成:

$$\mathbf{C}_{um} \approx \frac{1}{\mathbf{Z}_{um}} \sum_{h=1}^{H} \mathbf{T}_{um}^{h} \mathbf{P}_{u}^{h\top} \mathbf{Q}_{m}^{h} \tag{3.2}$$

其中 $\mathbf{Z}_{um}=\sum_{h=1}^{H}\mathbf{T}_{um}^{h}$ 是归一化因子，$\mathbf{T}_{um}^{h}$ 代表在子矩阵 $\mathbf{C}^h$ 项 $\mathbf{C}_{um}^{h}$ 的权重。从上述公式可以看出，此类子矩阵集成分解的两个关键问题是 (1) 如何选择和生成子矩阵？(2) 如何设置子矩阵的权重和根据子矩阵近似原始矩阵?

类似于LLORMA的选择子矩阵的方法,我们首先在数据集合 $\mathcal{A}=\{a_1,a_2,...,a_{|\mathbf{R}|}\}$ 中找一个数据点 $a_i=\langle u_i,m_i\rangle$ 作为锚点 $\hat{a}_h$。然后我们利用相似度计算方法或者核函数计算锚点和其他数据点之间的相关程度。最后，我们选择那些相关程度大于一定常数的数据点组成子矩阵。可以看出，子矩阵中的数据点是相似的。而且，我们可以用相似的方法选择更多的锚点，然后得到更多的子矩阵。

实际上，本文中我们使用 Epanechnikov 核函数去计算两个数据点 $a_i=(u_i,m_i)$ 和 $a_j=(u_j,m_j)$ 之间的相关程度。利用用户之间的 Epanechnikov 核函数 ($E_b(u_i,u_j)$) 和商品之间的 Epanechnikov 核函数 ($E_b(m_i,m_j)$) 的乘积作为数据点 $a_i=(u_i,m_i)$ 和 $a_j=(u_j,m_j)$ 之间的相关程度，如下表示:

$$E(a_i,a_j)=E_b(u_i,u_j)\times E_b(m_i,m_j) \tag{3.3}$$

其中 $E_b(u_i,u_j)$ 和 $E_b(m_i,m_j)$ 是 Epanechnikov 核函数，

$$\begin{aligned} E_b(u_i,u_j) &\propto (1-d(u_i,u_j)^2)\,\mathbf{1}_{\{d(u_i,u_j)\le b\}} \\ E_b(m_i,m_j) &\propto (1-d(m_i,m_j)^2)\,\mathbf{1}_{\{d(m_i,m_j)\le b\}} \end{aligned}$$

其中 b 是核函数的宽度参数。两个用户（或者商品）的距离利用两个用户（或者商

品）的隐藏特征向量计算。初始的用户隐藏特征向量和商品隐藏特征向量从 WMF 中学习得到。对应地，我们利用 $d(u_i,u_j)=arccos(\frac{\mathbf{P}_{u_i}\cdot\mathbf{P}_{u_j}}{\|\mathbf{P}_{u_i}\|\cdot\|\mathbf{P}_{u_j}\|})$ 计算的用户 u_i 和 u_j 之间的距离，其中 $\mathbf{P}_{u_i}$, $\mathbf{P}_{u_j}$ 就是第 u_i 个用户和第 u_j 个用户的局部隐藏特征向量。使用同样的方法计算商品之间的距离。因此在选定锚点 $\hat{a}_h$ 的前提下，在子矩阵 $\mathbf{R}^h$ 中我们设置用户商品对 $\langle u_j,m_j\rangle$ 的权重为 $\mathbf{T}^h_{u_jm_j}=E(\hat{a}_h,a_j)$ ，设置子矩阵的正则化因子为 $\lambda^h_{\mathbf{P}}=\lambda_{\mathbf{P}}E_b(\hat{u}_h,u_j)$ 和 $\lambda^h_{\mathbf{Q}}=\lambda_{\mathbf{Q}}E_b(\hat{m}_h,m_j)$。

从上述选择子矩阵可以看出，每选出的一个锚点就代表一个子矩阵。选择子矩阵集合 $\mathcal{C}$ 实际上就是选择一个锚点集合 $\hat{\mathcal{A}}=\{\hat{a}_1,\hat{a}_2,...,\hat{a}_H\}$。具体地，选择锚点集合将在下一小节讨论。

3.3.2 锚点集合选择

直观地，子矩阵集合 $\mathcal{C}=\{\mathbf{C}^1,\mathbf{C}^2,...,\mathbf{C}^H\}$ 应该需要覆盖整个原始矩阵 $\mathbf{C}$，也就是说 $\mathbf{C}=\cup_{\mathbf{C}^h\in\mathcal{C}}\mathbf{C}^h$。因此那些能够覆盖原始矩阵的子矩阵集合 $\mathcal{C}$ 一般能够比不能覆盖原始矩阵的子矩阵集合更好地近似原始矩阵 $\mathbf{C}$ 。因此，锚点集合选择问题就变成锚点集合覆盖问题。

3.3.2.1 简单锚点集合覆盖

我们将所有非零用户-商品对，也就是所有数据点集合 $\mathcal{A}=\{a_1,a_2,...,a_{|\mathbf{R}|}\}$ 作为锚点候选点集合。每个候选点 a_i 能够覆盖自己和一些其他候选点，表示为 $\mathcal{A}^i=\{a_i,a_{i1},a_{i2},...,a_{iD}\}\subset\mathcal{A}$。然后我们提出一个简单锚点集合覆盖方法，简称为锚点集合覆盖（Anchor point Set Cover problem, ASC），返回如下锚点集合 $\hat{\mathcal{A}}\subset\mathcal{A}$:

$$\begin{aligned}\max J(\hat{\mathcal{A}})&=|\cup_{i\in\hat{\mathcal{A}}}\mathcal{A}^i|\\ s.t.&|\hat{\mathcal{A}}|=H\end{aligned}\tag{3.4}$$

显然，ASC 问题是符合次模性质和单调性 [104]。因此，简单的贪心算法即可得到 $1-\frac{1}{e}$ 的近似最优解。

3.3.2.2 折扣累计收益锚点集合覆盖

然而，集合覆盖问题仅仅需要覆盖数据点一次。某个数据点被覆盖之后被覆盖就没有收益。但是在本文覆盖训练数据点，仅仅覆盖一次是不够的，并且之后的覆盖也是对最终覆盖和推荐也有帮助的。然而，随着数据点被覆盖的次数越多，性能提高越少，收益被打折扣。这种情况类似于在信息检索中排序质量评测方法，归一化折扣累计收益 (Normalized Discounted Cumulative Gain，简称 NDCG) [105, 106] 和期望排序倒数 (Expected Reciprocal Rank，简称 ERR) [107]。NDCG 和 ERR 的主要思想是相关程度比较高的结果却排到了后面，那么在统计收益时，对收益大小根据位置的前后进行打折扣。从这个折扣思路学习，我们提出了一种启发式方法去建模这种锚点选择情形，称之为折扣累计收益锚点集合覆盖（Discounted Cumulative Gain Anchor Point Set Cover，简称 DCGASC）。DCGASC 在每次数据点被覆盖时都获得覆盖收益，但是收益一次递减。具体地，通过最大化如下目标函数方式返回一个锚点排序列表 $\hat{\mathcal{A}}=\{\hat{a}_1,\hat{a}_2,...,\hat{a}_H\}\subset\mathcal{A}$:

$$\max J(\hat{\mathcal{A}})\sum_{h=1}^{H}\sum_{a_l\in\hat{\mathcal{A}}^h}\alpha^{o_{lh}-1}(1-max_{h'\in\{1,...,h-1\}}E_b(\hat{a}_h,\hat{a}_{h'}))$$
$$s.t.|\hat{\mathcal{A}}|=H \tag{3.5}$$

其中 o_{lh} 代表数据点 a_l 被已选择的锚点 $\{\hat{a}_1,\hat{a}_2,...,\hat{a}_h\}$ 的覆盖次数。$\alpha\in(0,1)$ 是折扣系数。当数据点 a_l 之前已经被一个锚点覆盖，那么下次该数据点被覆盖时覆盖收益将会减少。当折扣系数 $\alpha=0$ 时，这个问题简化为简单锚点集合覆盖问题，详见小节 3.3.2.1。当 $\alpha=1$ 时，该问题解就变成选择每次选择覆盖其他数据点最多的锚点。项 $(1-max_{h'\in\{1,...,h-1\}}E_b(\hat{a}_h,\hat{a}_{h'})$ 则表示 DCGASC 倾向于选择那些远离已经选择过的锚点。接下来我们可以证明目标函数 $J(\cdot)$ 是符合次模性质和单调性。

定理 3.3.1. *DCGASC 目标函数 3.5是次模的，同时也是单调非减的。*

证明. 假设 $\mathcal{S} = \{\hat{a}_1, \hat{a}_2, ..., \hat{a}_{H-1}\}$ ，$\mathcal{V} = \{\hat{a}_1, \hat{a}_2, ..., \hat{a}_{H-1}, ..., \hat{a}_{X-1}\}$ 都是锚点集合，同时 $X \geq H$ ，以及 $a_i = \hat{a}_X \in \mathcal{A} \backslash \mathcal{V}$ 是下一个选择的锚点。我们有:

$$
\begin{aligned}
& J(\mathcal{V} \cup \{\hat{a}_X\}) - J(\mathcal{V}) \\
& = \sum_{h=1}^{X} \sum_{a_l \in \hat{\mathcal{A}}^h} \alpha^{o_{lh}-1}(1 - max_{h' \in \{1,...,h-1\}} E_b(\hat{a}_h, \hat{a}_{h'})) \\
& - \sum_{h=1}^{X-1} \sum_{a_l \in \hat{\mathcal{A}}^h} \alpha^{o_{lh}-1}(1 - max_{h' \in \{1,...,h-1\}} E_b(\hat{a}_h, \hat{a}_{h'})) \\
& = \sum_{a_l \in \hat{\mathcal{A}}^X} \alpha^{o_{lX}-1}(1 - max_{h' \in \{1,...,X-1\}} E_b(\hat{a}_X, \hat{a}_{h'})) \geq 0
\end{aligned} \tag{3.6}
$$

所以 DCGASC 目标函数 3.5是单调非减的。

$$
\begin{aligned}
& J(\mathcal{S} \cup \{\hat{a}_X\}) - J(S) - (J(\mathcal{V} \cup \{\hat{a}_X\}) - J(\mathcal{V})) \\
& = \sum_{a_l \in \hat{\mathcal{A}}^X} \alpha^{o_{lX'}-1}(1 - max_{h' \in \{1,...,H-1\}} E_b(\hat{a}_X, \hat{a}_{h'})) \\
& - \sum_{a_l \in \hat{\mathcal{A}}^X} \alpha^{o_{lX}-1}(1 - max_{h' \in \{1,...,X-1\}} E_b(\hat{a}_X, \hat{a}_{h'}))
\end{aligned} \tag{3.7}
$$

其中 $o_{lX'}$ 代表数据点 a_l 被锚点集合 $S \cup \{\hat{a}_X\}$ 覆盖的次数。因为锚点结合覆盖的数目满足 $o_{lX'} \leqslant o_{lX}$，折扣系数 $\alpha \in [0, 1]$ 以及 $max_{h' \in \{1,...,H-1\}} E_b(\hat{a}_X, \hat{a}_{h'}) \leq \max_{h' \in \{1,...,X-1\}} E_b(\hat{a}_X, \hat{a}_{h'})$，我们可以知道 $J(\mathcal{S} \cup \{\hat{a}_X\}) - J(\mathcal{S}) - (J(\mathcal{V} \cup \{\hat{a}_X\}) - J(\mathcal{V})) \geq 0$。所以 DCGASC 目标函数 3.5是次模的。

综上所述，DCGASC 目标函数 3.5是次模的，同时也是单调非减的。 □

因为 DCGASC 目标函数的单调性和次模性，我们能够简单地使用使用贪心算法能够得到 $1 - \frac{1}{e}$ 的近似最优解保证 [108]。算法 1展示了这个贪心算法：第 1 和

2 行首先将得到能够覆盖其他数据点最大的数据点作为锚点，然后 3-5 行使用公式 3.7依次得到接下来的 $(H-1)$ 个锚点。

算法 1 DCGASC 贪心算法

输入: 数据点集合 $\mathcal{A}$，锚点数量 H, DCGASC 函数 J 和被数据点 a_i 覆盖的数据点集合 A^i;

输出: 锚点列表 $\hat{\mathcal{A}} \subseteq \mathcal{A}$，其中 $|\hat{\mathcal{A}}| = H$;

1: $\hat{a}_1 \leftarrow \arg\max_{a_i \in \mathcal{A}} |\mathcal{A}^i|$;
2: $\hat{\mathcal{A}} \leftarrow \{\hat{a}_1\}$;
3: **for** $h = 2 : H$ **do**
4: $\quad \hat{a}_h \leftarrow \arg\max_{a'_i \in \mathcal{A} \setminus \hat{\mathcal{A}}} f(\hat{\mathcal{A}} \cup \{a'_i\}) - f(\hat{\mathcal{A}})$;
5: $\quad \hat{\mathcal{A}} \leftarrow \hat{\mathcal{A}} \cup \{\hat{a}_h\}$;
6: **return** $\hat{\mathcal{A}}$;

3.3.3 优化算法

交替最小二乘（Alternating Least Square，ALS）是一种优化加权矩阵分解的流行算法 [1]。不同于原始的 ALS，He *et al.* [59] 提出快速元素级交替最小二乘学习算法。该方法通过固定隐藏特征向量中其他维度数值，优化每维坐标数值，同时通过基于商品消费次数对缺失值引入对应权重，避免大量的重复计算，从而加速计算、学习参数，参数学习效率提高 K 倍且有相似的推荐效果。在本文中，我们使用元素级交替最小二乘算法学习子矩阵隐藏特征向量，并且利用子矩阵数据点权重的计算方法，对元素级交替最小二乘做了类似于文章 [59] 的加速优化，使得能够较快地学习子矩阵隐藏特征向量。具体地，子矩阵 $\mathbf{R}^h$ 中第 u 个用户隐藏特征向量的基本迭代公式如下:

$$\mathbf{P}^h_{uk} = \frac{\sum_{m \in \mathcal{M}^h} (\mathbf{C}_{um} - \hat{\mathbf{C}}^h_{um,k}) \mathbf{T}^h_{um} \mathbf{W}_{um} \mathbf{Q}^h_{mk}}{\sum_{m \in \mathcal{M}^h} \mathbf{T}^h_{um} \mathbf{W}_{um} \mathbf{Q}^h_{mk} \mathbf{Q}^h_{mk} + \lambda^h_{\mathbf{P}}} \tag{3.8}$$

其中，$\mathcal{M}^h$ 表示子矩阵 $\mathbf{R}^h$ 中存在于原始矩阵 $\mathbf{R}$ 的商品索引集合，$\hat{\mathbf{C}}^h_{um,k}$ 表示出了需要更新坐标的隐藏特征向量的预测值，也就是 $\hat{\mathbf{C}}^h_{um,k} = \hat{\mathbf{C}}^h_{um} - \mathbf{P}^h_{uk} \mathbf{Q}^h_{mk}$（$\hat{\mathbf{C}}^h_{um}$ 表示预测值）。我们可以注意到 $\mathbf{C}_{um}$ 和 $\mathbf{W}_{um}$ 在不同的子矩阵中都是一样的。同时相应的子矩阵权重 $\mathbf{T}^h_{um}$ 在公式 3.8中是和原始 WMF 比较唯一不同的项，导致不能对

公式 3.8进行类似于文章 [1] 和 [59] 的加速计算。幸运地是，因为 $\mathbf{T}^h_{um}$ 的计算方式是由用户和商品组成，即 $\mathbf{T}^h_{um} = E_b(\hat{u}_h, u) \times E_b(\hat{m}_h, m)$ ，而且正则化系数也是有用户和商品权重成分（$\lambda^h_{\mathbf{P}} = \lambda_{\mathbf{P}} E_b(\hat{u}_h, u)$），我们仍然能够通过计算中间结果，加速优化效率。首先，项 $E_b(\hat{u}_h, u)$ 在公式 3.8中分子和分母是同时存在的，我们能够消去它。而且，我们可以发现如果 $E_b(\hat{u}_h, u) = 0$，那么在子矩阵权重为 0，我们就不需要进行计算它的局部隐藏特征向量。这里我们首先聚焦于分子部分的计算：

$$\begin{aligned}
&\sum_{m \in \mathcal{M}^h} (\mathbf{C}_{um} - \hat{\mathbf{C}}^h_{um,k}) E_b(\hat{m}_h, m) \mathbf{W}_{um} \mathbf{Q}^h_{mk} \\
&\quad = \sum_{m \in \mathcal{M}^h_u} [\mathbf{W}_{um} \mathbf{C}_{um} - (\mathbf{W}_{um} - 1) \hat{\mathbf{C}}^h_{um,k}] E_b(\hat{m}_h, m) \mathbf{Q}^h_{mk} \\
&\quad - \sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \hat{\mathbf{C}}^h_{um,k} \mathbf{Q}^h_{mk}
\end{aligned} \tag{3.9}$$

其中 $\mathcal{M}^h_u$ 代表的是第 u 个用户在子矩阵 $\mathbf{R}^h$ 中消费的商品集合。因为对于任何一个用户项 $E_b(\hat{m}_h, m)$ 都是一样的，因此在这里可以使用缓存的方法。项 $\sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \hat{\mathbf{C}}^h_{um,k} \mathbf{Q}^h_{mk}$ 可以加速运算：

$$\sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \hat{\mathbf{C}}^h_{um,k} \mathbf{Q}^h_{mk} = \sum_{f \neq k} \mathbf{P}_{uf} \sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \mathbf{Q}^h_{mk} \mathbf{Q}^h_{mf} \tag{3.10}$$

因此，在进行学习局部用户隐藏特征向量时，项 $\sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \mathbf{Q}^h_{mk} \mathbf{Q}^h_{mf}$ 是可以提前计算好的，然后用于学习所有用户的局部隐藏特征向量。相似地，我们也可以使用相同的方法运用在公式 3.8中分母的计算。我们定义局部商品缓存矩阵变量 $\mathbf{S}^{\mathbf{Q}^h}$，令它为 $\mathbf{S}^{\mathbf{Q}^h} = \sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \mathbf{Q}^h_m \mathbf{Q}^{h\top}_m$，那么公式 3.8就可以变为如下计算：

$$\mathbf{P}_{uk}^{h} = \{ \sum_{m \in \mathcal{M}_u^h} [\mathbf{W}_{um}\mathbf{C}_{um} - (\mathbf{W}_{um} - 1)\hat{\mathbf{C}}_{um,k}^{h}] E_b(\hat{m}_h, m)\mathbf{Q}_{mk}^{h} - \sum_{f \neq k} \mathbf{P}_{uf}^{h}\mathbf{S}_{fk}^{\mathbf{Q}^h} \}$$
$$/\{ \sum_{m \in \mathcal{M}_u^h} E_b(\hat{m}_h, m)(\mathbf{W}_{um} - 1)\mathbf{Q}_{mk}^{h}\mathbf{Q}_{mk}^{h} + \mathbf{S}_{kk}^{\mathbf{Q}^h} + \lambda_{\mathbf{P}} \} \tag{3.11}$$

其中项 $\mathbf{S}_{fk}^{\mathbf{Q}^h}$ 是缓存矩阵 $\mathbf{S}^{\mathbf{Q}^h}$ 的第 f 行第 k 列元素。

相似地，我们可以定义局部用户缓存矩阵变量 $\mathbf{S}^{\mathbf{P}^h}$，令其为 $\mathbf{S}^{\mathbf{P}^h} = \sum_{u \in \mathcal{U}^h} E_b(\hat{u}_h, u)\mathbf{P}_u^h\mathbf{P}_u^{h\top}$，那么局部商品隐藏特征变量迭代公式如下表示：

$$\mathbf{Q}_{mk}^{h} = \{ \sum_{u \in \mathcal{U}_m^h} [\mathbf{W}_{um}\mathbf{C}_{um} - (\mathbf{W}_{um} - 1)\hat{\mathbf{C}}_{um,k}^{h}] E_b(\hat{u}_h, u)\mathbf{P}_{uk}^{h} - \sum_{f \neq k} \mathbf{Q}_{mf}^{h}\mathbf{S}_{fk}^{\mathbf{P}^h} \}$$
$$/\{ \sum_{u \in \mathcal{U}_m^h} E_b(\hat{u}_h, u)(\mathbf{W}_{um} - 1)\mathbf{P}_{uk}^{h}\mathbf{P}_{uk}^{h} + \mathbf{S}_{kk}^{\mathbf{P}^h} + \lambda_{\mathbf{Q}} \} \tag{3.12}$$

在子矩阵 $\mathbf{R}^h$ 每轮用户局部隐藏特征向量学习迭代中，计算缓存矩阵变量 $\mathbf{S}^{\mathbf{Q}^h}$ 的复杂度为 $O(|\mathcal{M}^h|K^2)$，学习每个用户的局部隐藏特征向量的复杂度为 $O(|\mathbf{R}^h|K)$，其中 $|\mathcal{M}^h|$ 代表集合 $\mathcal{M}^h$ 的大小，$|\mathbf{R}^h|$ 代表的是子矩阵 $\mathbf{R}^h$ 非零元素的个数，那。同理，子矩阵 $\mathbf{R}^h$ 每轮商品局部隐藏特征向量学习迭代的复杂度为 $O(|\mathcal{U}^h|K^2 + |\mathbf{R}^h|K)$。假设原始矩阵 $\mathbf{R}$ 被 H 个子矩阵覆盖，每个非零元素覆盖的次数为 $\hat{H}$，那么有 $\hat{H}|\mathbf{R}| = \sum_{h=1}^{H} |\mathbf{R}^h|$，因此整个 LWMF 模型每一轮迭代的复杂度为 $O(\hat{H}(NK^2 + MK^2 + |\mathbf{R}|K))$，是全局加权矩阵分解复杂度的 $\hat{H}$ 倍。

算法 2 概括了学习局部隐藏特征向量的过程。首先，我们使用快速元素级最小二乘法 [59] 学习全局隐藏特征向量。然后我们通过算法 1获得 H 个锚点集合。最后，我们使用改进的快速元素级最小二乘法学习每一个子矩阵的局部隐藏特征向量。

算法 2 LWMF 优化算法

输入: < 用户，商品，次数 > 数据矩阵 $\mathbf{R}$，锚点数量 H，DCGASC 函数 J and 被数据点 a_i 覆盖的集合 $\mathcal{A}^i$，权重矩阵 $\mathbf{W}$，正则化系数 $\lambda_{\mathbf{P}}$ 和 $\lambda_{\mathbf{Q}}$，隐藏特征向量维度 K;

输出: 用户隐藏特征矩阵集合 $\mathcal{P} = \{\mathbf{P}^1, \mathbf{P}^2, ..., \mathbf{P}^H\}$，商品隐藏特征矩阵集合 $\mathcal{Q} = \{\mathbf{Q}^1, \mathbf{Q}^2, ..., \mathbf{Q}^H\}$ 和子矩阵权重集合 $\mathcal{T} = \mathbf{T}^1, \mathbf{T}^2, ..., \mathbf{T}^H$;

利用公式 2.11从原始数据矩阵 $\mathbf{R}$ 计算二值化矩阵 $\mathbf{C}$;
使用元素级最小二乘算法 [59] 全局隐藏特征向量 $\mathbf{P}$ 和 $\mathbf{Q}$;
使用 DCGASC 算法 1得到锚点集合 $\hat{\mathcal{A}} = \{\hat{a}_1, \hat{a}_2, ..., \hat{a}_H\}$;
for $h \leftarrow 1$ to H **do**
  for $u \leftarrow 1$ to N **do**
    if $E_b(\hat{u}_h, u) > 0$ **then**
      $\mathcal{U}^h \leftarrow \mathcal{U}^h \cup \{u\}$;
  for $m \leftarrow 1$ to M **do**
    if $E_b(\hat{m}_h, m) > 0$ **then**
      $\mathcal{M}^h \leftarrow \mathcal{M}^h \cup \{m\}$;
  //更新局部用户隐藏特征向量
  $\mathbf{S}^{\mathbf{Q}^h} = \sum_{m \in \mathcal{M}^h} E_b(\hat{m}_h, m) \mathbf{Q}_m^h \mathbf{Q}_m^{h\top}$;
  for all $u \in \mathcal{U}^h$ **do**
    for all $m \in \mathcal{M}_u^h$ **do**
      $\hat{\mathbf{C}}_{um}^h \leftarrow \mathbf{P}_u^{h\top} \mathbf{Q}_m^h$;
    for $k \leftarrow 1$ to K **do**
      for all $m \in \mathcal{M}_u^h$ **do**
        $\hat{\mathbf{C}}_{um,k}^h \leftarrow \hat{\mathbf{C}}_{um}^h - \mathbf{P}_{uk}^h \mathbf{Q}_{mk}^h$;
      利用公式 3.11计算 $\mathbf{P}_{uk}^h$;
      for all $m \in \mathcal{M}_u^h$ **do**
        $\hat{\mathbf{C}}_{um,k}^h \leftarrow \hat{\mathbf{C}}_{um}^h + \mathbf{P}_{uk}^h \mathbf{Q}_{mk}^h$;
  //更新局部商品隐藏特征向量
  $\mathbf{S}^{\mathbf{P}^h} = \sum_{u \in \mathcal{U}^h} E_b(\hat{u}_h, u) \mathbf{P}_u^h \mathbf{P}_u^{h\top}$;
  for all $m \in \mathcal{M}^h$ **do**
    for all $u \in \mathcal{U}_m^h$ **do**
      $\hat{\mathbf{C}}_{um}^h \leftarrow \mathbf{P}_u^{h\top} \mathbf{Q}_m^h$;
    for $k \leftarrow 1$ to K **do**
      for all $u \in \mathcal{U}_m^h$ **do**
        $\hat{\mathbf{C}}_{um,k}^h \leftarrow \hat{\mathbf{C}}_{um}^h - \mathbf{P}_{uk}^h \mathbf{Q}_{mk}^h$;
      利用公式 3.12计算 $\mathbf{Q}_{mk}$;
      for all $u \in \mathcal{U}_m^h$ **do**
        $\hat{\mathbf{C}}_{um,k}^h \leftarrow \hat{\mathbf{C}}_{um}^h + \mathbf{P}_{uk}^h \mathbf{Q}_{mk}^h$;
return $\mathcal{P}, \mathcal{Q}$;

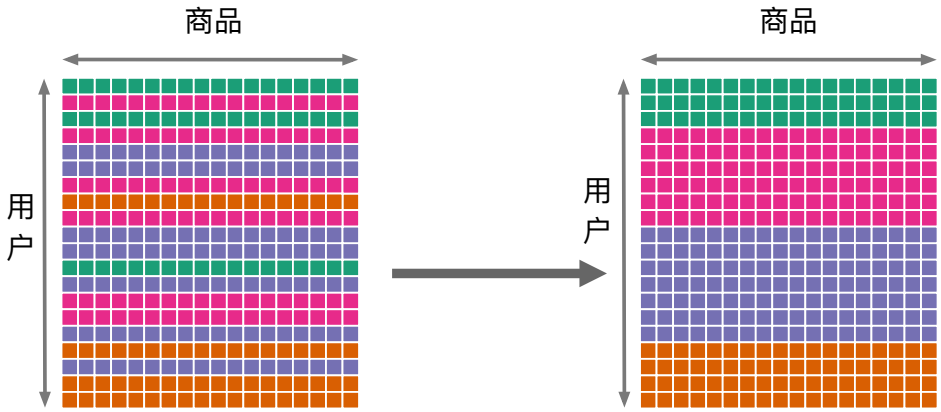


图 3.2: 基于用户的局部矩阵分解

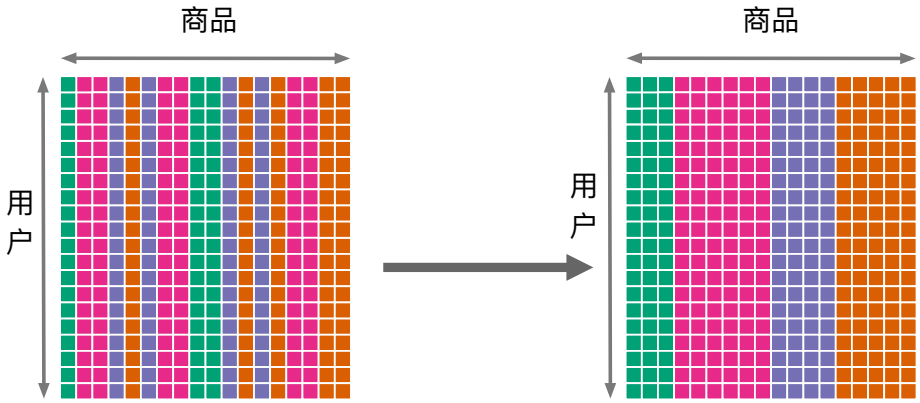


图 3.3: 基于商品的局部矩阵分解

3.3.4 基于用户的局部加权矩阵分解

上述的 LWMF 方法使用被选中的子矩阵去对局部性质进行建模，而忽视了全局信息。特别对于商品推荐问题，我们应该考虑从所有商品中推荐用户感兴趣的商品。因此，我们提出一种变种的 LWMF 方法-基于用户的局部加权矩阵分解，它仅仅只考虑从用户角度选择锚点，而将所有商品放入子矩阵中。给定用户集合 $\mathcal{U} = \{u_1, u_2, ..., u_N\}$ ，以及每个用户 u_i 能覆盖起本身和其他一些用户，表示为 $\mathcal{U}^i = \{u_i, u_{i1}, u_{i2}, ..., u_{iD}\}$，我们需要找出一个用户锚点集合 $\hat{\mathcal{U}} = \{\hat{u}_1, \hat{u}_2, ..., \hat{u}_H\}$ 去最大化覆盖用户集合，目标函数如下：

$$\max J(\hat{\mathcal{U}}) \sum_{h=1}^{H} \sum_{u_l \in \mathcal{U}^h} \alpha^{o_{lh}-1}(1 - max_{h' \in \{1,...,h-1\}} E_b(\hat{u}_h, \hat{u}_{h'}))$$

$$s.t. |\hat{\mathcal{U}}| = H \tag{3.13}$$

显然地，这个基于用户的 DCGASC 目标函数也是符合次模性质和非减函数。图 3.2展示了基于用户的 LWMF 模型去选择子矩阵局的过程。因为我们不需要考虑商品，基于用户的 LWMF 会较快地选择用户锚点集合。而且，对于商品推荐来说，基于用户的 LWMF 模型也相对比较合适的。作为基于用户的 LWMF 模型的比较，我们也加入了基于商品的 LWMF 的比较，它仅仅考虑商品去选择锚点且把所有用户放入子矩阵中。

3.4 实验及分析

在本节，我们将在真实隐式数据上评估本文提出的模型。我们首先介绍所使用的真实世界数据集和实验设置。然后我们在特定参数下和其他一些著名方法比较，特别是 WMF。我们也进行了不同锚点数量（也就是不同数量的子矩阵）和不同锚点选择方法下结果的比较。

3.4.1 实验设置

3.4.1.1 数据集

我们选择论文 [109] 中的两个真实世界数据集。一个是新加坡 2010 年 8 月到 2011 年 7 月的 Foursquare 签到数据，另外一个是加利福利亚州和内华达州 2009 年 2 月到 2010 年 10 的签到数据。这两个数据都是非常流行的在线移动位置服务数据，只有签到数据，但没有用户的喜好数据，是非常典型的用户隐式反馈数据。

Foursquare 签到数据包含由 2,312 个用户，5,596 个 POIs 组成的 194,108 个签到数据，稠密度是 1.50×10^{-2}。Gowalla 签到数据包含由 10,162 个用户，24,238 个 POIs 组成的 456,967 个签到数据，稠密度是 1.86×10^{-3}。两个数据集都是比较稀疏。表3.2展示两个数据集更详细的信息。另外，我们根据用户-地点-签到次数随机将 80% 数据集分割为训练集，剩下的 20% 为测试集。

表 3.2: Gowalla 和 Foursquare 数据集的详细信息

	Foursquare	Gowalla
#users	2,321	10,162
#locations	5,596	24,238
#check-ins	194,108	456,967
avg. #users per loc.	34.69	18.85
avg. #loc. per user	83.63	44.97
max #users per loc.	695	2,195
max #loc. per user	311	1,113

3.4.1.2 参数设置

接下来，说明下参数设置。正则化因子 λ 设置为 10，并且我们发现推进效果对该参数的设置不是非常敏感，10 是一个相对较好的参数。权重参数 ε 在数据集 Foursquare 设置为 2，另一个数据集 Gowalla 上设置为 3。我们设置 Epanechanikov 核函数带宽参数 $b = 0.8$。另外我们设置锚点选择方法 DCGASC 的折扣参数 α 为 0.4。折扣参数对推荐效果的影响后面有具体实现进行说明。对两个数据集，我们

都选择 100 个锚点进行最后矩阵分解。在实验中，我们观察到随着锚点数量增多，也就是子矩阵数量增多，推荐效果越好，但是训练时间随之增加并且推荐效果增强收益却越少。

3.4.1.3 评价标准

我们使用正确率 Precision@n 和召回率 Recall@n 衡量模型的推荐性能。对第 u 个用户，我们假设符号 $\mathcal{I}_u^P$ 为他的商品推荐列表，符号 $\mathcal{I}_u^T$ 为该用户在测试数据集中的真实商品列表。因此正确率 Precision@n 和召回率 Recall@n 分别表示为

$$Precision@n = \frac{1}{N}\sum_{u=1}^{N}\frac{|\mathcal{I}_u^P \bigcap \mathcal{I}_u^T|}{n} \tag{3.14}$$

$$Recall@n = \frac{1}{N}\sum_{u=1}^{N}\frac{|\mathcal{I}_u^P \bigcap \mathcal{I}_u^T|}{|\mathcal{I}_u^T|} \tag{3.15}$$

其中 $|\mathcal{I}_u^P|$ 表示列表 $\mathcal{I}_u^P$ 的大小，等于 n。在我们的基础实验中，我们选择 $n = 10$ 来评价实验结果。

3.4.1.4 对比方法

我们总共比较 7 个隐式反馈数据推荐的模型方法：

- MP: 这是最基本方法，对目标用户推荐最流行的商品。
- KNN$_u$: 这个是基于用户的协同过滤方法，利用训练集中计算用户跟用户之间的相似度，求出 K[1]个最相似的用户对商品的评分之和作为最终预测值。
- KNN$_m$: 这个方法跟 KNN$_u$ 类似，是基于商品的协同过滤方法，利用训练集中计算商品跟商品之间的相似度，根据目标用户对其他商品的评测值和跟目标商品之间的相似度计算预测值。特别地，我们设置两个 K 近邻方法的近邻数量为 100。

[1]这里的 K 指的是近邻数量，不是指局部隐藏特征向量的维度。

- WMF: 该方法为现在最流行隐式反馈数据 top-n 商品推荐方法 [**? ?**]，本文的方法也是基于该方法上提出的，它对缺失数据设置统一的权重在整个数据集上进行隐藏特征向量参数学习优化。其他基本参数设置和上面参数设置说明一致。

- LWMF$_{both}$: 这个方法是本文提出的，利用核函数选择锚点，进而得到子矩阵，来对数据集的局部性质进行建模。

- LWMF$_{u}$: LWMF$_{both}$ 的一个变种，该方法仅仅考虑用户去选择锚点，并把所有商品放入子矩阵中。

- LWMF$_{m}$: LWMF$_{both}$ 的另一个变种，该方法仅仅考虑商品去选择锚点，并把所有用户放入子矩阵中。

另外，我们也比较了两类不同的锚点选择方法来研究锚点选择的不同对最终 LWMF 推荐效果的影响:

- 随机选择: 利用均匀分布从训练集中随机选择锚点，和论文 [4, 5] 中锚点选择方法类似。

- 最大化折扣累计收益锚点集合覆盖锚点选择（DCGASC）: 利用最大化折扣累计收益锚点集合覆盖函数进行锚点选择。

因此 LWMF 根据锚点选择方法的不同也将分为两个子方法: LWMF_random 和 LWMF_DCGASC。默认地，不作特别说明，LWMF 代表的模型是 LWMF_DCGASC 方法。

以上所有方法，我们分别独立进行 5 次实验，因此最终 5 次实验的平均值作为最终推荐方法的结果。

3.4.2 实验结果及分析

在本小节，我们将具体介绍在 Foursquare 和 Gowalla 两个数据集上的实验结果，主要从以下四个方面进行讨论:

- 各类不同推荐方法的对比；
- 不同锚点数量（即不同子矩阵个数）对本文模型 LWMF 最终推荐效果的影响；
- 不同锚点选择方法对本文模型 LWMF 最终推荐效果的影响；
- 折扣参数的不同对本文模型 LWMF 最终推荐效果的影响。

3.4.2.1 不同推荐方法的对比

表3.3列出了在数据集 Foursquare 和 Gowalla 上上述 7 个方法的正确率和召回率。跟之前文献经验一样，WMF 在维度合适的情况下，性能比 K 近邻算法要好。尽管在 WMF 隐藏特征向量维度 K 较低时，KNN$_u$ 和 KNN$_m$ 效果较好，但是随着维度 K 的增大，WMF 的推荐效果逐渐接近并超过 KNN$_u$ 和 KNN$_m$。其次，LWMF 在多数情况下都要优于 WMF。另外，我们的模型 LWMF 要优于 WMF，这跟论文 [4] 展示的 LLORMA 要比 SVD 好一样。我们可以看出，WMF 和 LWMF 的推荐效果随着隐藏特征向量维度 K 的增大而提高。然后，在数据集 Foursquare 上，但维度 K 达到 40 时，两者的性能反而有所下降，这表明当维度在 40 维时，模型已经过拟合了。另外一方面，数据集 Gowalla 上的实验结果表明维度 K 在 40（或者大于 40）时效果最好。因此在接下来的实验中，为了维度一致，我们都把隐藏特征向量维度 K 设置为 20。很显然地，LWMF$_b oth$ 及其两个变种方法 LWMF$_u$ 和 LWMF$_m$ 在正确率和召回率上在所有维度（$K = 5, 10, 20, or 40$）上都要优于 WMF。特别是 Gowalla 数据集上，LWMF$_u$ 要比 WMF 至少 25% 好以上。在维度 K 设置为 5 时，LWMF$_{both}$ 的正确率甚至高出 WMF 52 个百分点。这些显著的提高，我们认为是因为 LWMF 对数据集的局部性质进行建模。例如，在签到数据集中，每个城市中会有一些商区，而商业 POIs 在每个商区中是天然地地理上的相近，并且用户也喜欢去离本身较近的 POIs，方便用户本身访问。对于我们提出的模型 LWMF 及其两个变种，我们可以发现它们之间的推荐效果比较相近。但是在全局角度看，基于用户的 LWMF$_u$ 要略微优于其他两个方法。LWMF$_u$ 基于用户进行推荐，利用全局商品

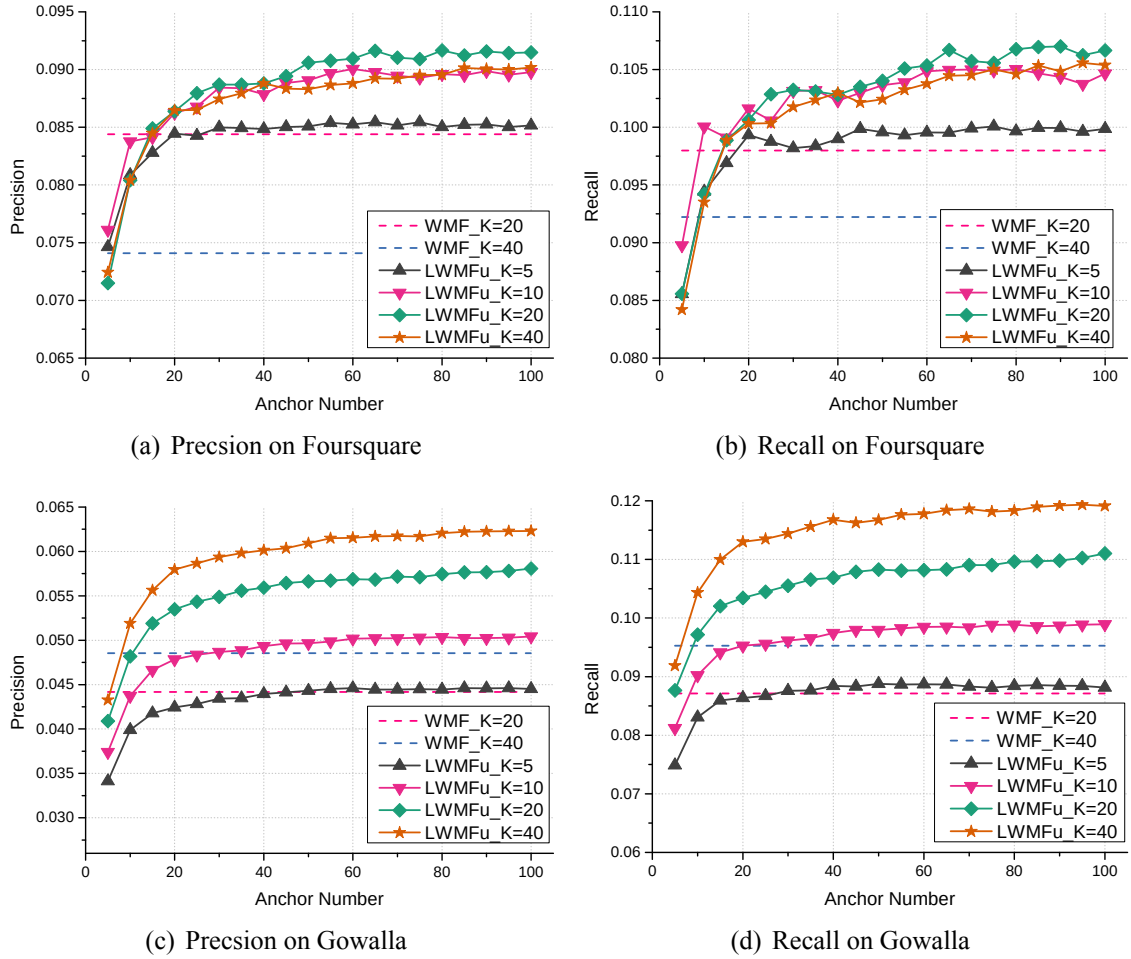
表 3.3: 不同方法准确率和召回率对比，其中行 “Improve” 代表 LWMF 对于基线方法 WMF 推荐效果提高百分比

(a) Foursquare 数据集

	d=5		d=10		d=20		d=40	
Methods	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
MP	0.0615	0.0680	0.0615	0.0680	0.0615	0.0680	0.0615	0.0680
KNN_u	0.0741	0.8212	0.0741	0.8212	0.0741	0.8212	0.0741	0.8212
KNN_i	0.0698	0.7975	0.0698	0.7975	0.0698	0.7975	0.0698	0.7975
WMF	0.0792	0.0905	0.0847	0.0993	0.0844	0.0980	0.0741	0.0922
$LWMF_b$	0.0823	0.0952	0.0847	0.0995	0.0832	0.0982	0.0828	0.0945
$LWMF_i$	**0.0869**	0.0962	0.0878	0.0990	0.0893	0.1021	**0.0907**	0.1028
$LWMF_u$	0.0852	**0.0999**	**0.0898**	**0.1047**	**0.0915**	**0.1067**	0.0902	**0.1054**
Improve	9.8%	10.34%	6.03%	5.44%	8.39%	8.85%	22.45%	14.27%

(b) Gowalla 数据集

Gowalla	d=5		d=10		d=20		d=40	
Methods	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
MP	0.0203	0.0460	0.0203	0.0460	0.0203	0.0460	0.0203	0.0460
KNN_u	0.0552	0.1055	0.0552	0.1055	0.0552	0.1055	0.0552	0.1055
KNN_i	0.0587	0.1014	0.0587	0.1014	0.0587	0.1014	0.0587	0.1014
WMF	0.0321	0.0664	0.0385	0.0779	0.0442	0.0871	0.0485	0.0953
$LWMF_b$	**0.0489**	**0.0923**	**0.0528**	**0.0990**	0.0558	0.1035	0.0578	0.1067
$LWMF_i$	0.0478	0.0884	0.0526	0.0936	0.0565	0.1006	0.0584	0.1034
$LWMF_u$	0.0445	0.0881	0.0504	0.0989	**0.0581**	**0.1110**	**0.0623**	**0.1191**
Improve	52.56%	39.05%	37.10%	27.01%	31.44%	27.41%	28.36%	25.04%



(a) Precsion on Foursquare (b) Recall on Foursquare

(c) Precsion on Gowalla (d) Recall on Gowalla

图 3.4: 不同锚点数量推荐结果对比

作为子矩阵商品，对于基于商品的 LWMF$_m$ 看上去更为合理一些。因此在接下来的对比实验中，我们都以基于用户的 LWMF$_u$ 作为 LWMF 的默认方法。

3.4.2.2 不同锚点数量推荐结果对比

图 3.4展示了不同锚点数量（即不同子矩阵个数）对本文模型 LWMF 最终推荐效果的影响。在两个数据集上，LWMF 的正确率和召回率都随着隐藏特征向量维度 K 提高而提高，并且 LWMF 在维度 K 大于等于 10 时以后，推荐效果就优于 WMF。当锚点数量 H 大于 20，也就是说子矩阵数量大于 20 时，LWMF 的推荐效果开始超过 WMF，并且随着锚点数量的增多，效果越好，但同时效果边际收益却在下降。但是因为锚点的增加，子矩阵变多，训练模型的时间也会随着增加，因此，

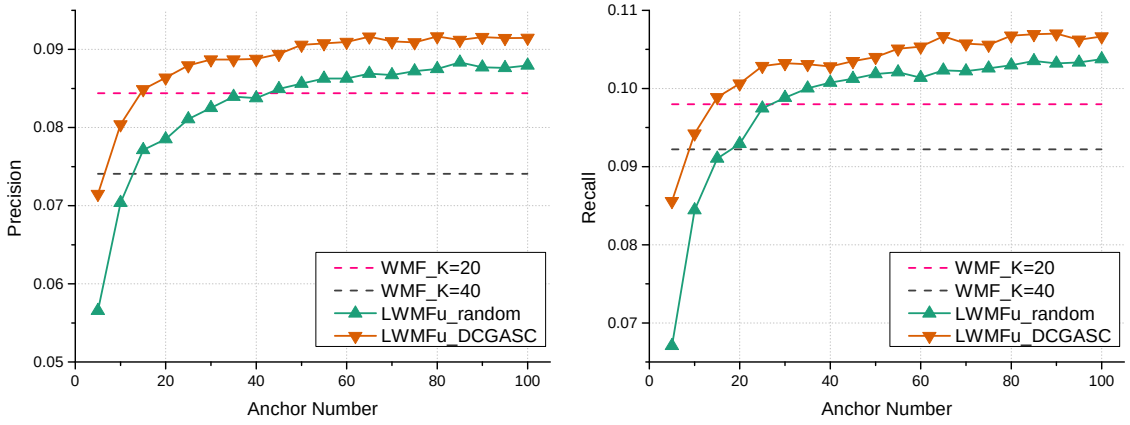
我们可以以及本身时间和推荐精度需求，动态选择锚点数量。一般地，当锚点数量 H 等于 50 的时候，我们将得到相对较好的实验结果。使用基于元素级最小二乘算法的 WMF 每一轮迭代的复杂度为 $O(NK^2 + MK^2 + |\mathbf{R}|K)$，而整个 LWMF 模型每一轮迭代的复杂度为 $O(\hat{H}(NK^2 + MK^2 + |\mathbf{R}|K))$，两者的复杂度相差 $\hat{H}$ 倍。两个数据集中，每个子矩阵大小平均是原始矩阵的 10% 左右。因此每个子矩阵的训练时间是远快于整个矩阵的分解训练。当子矩阵个数为 50 时，训练时间大概是 WMF 的 5 倍左右，基于推荐效果的提高，LWMF 的训练训练时间还是能够接受的，并且 LWMF 经过子矩阵选择后，有着比 WMF 更好的数据和模型并行度，易于并行扩展。

3.4.2.3 不同锚点选择方法推荐结果对比

接下来，我们展示不同锚点选择方法对本文模型 LWMF 最终推荐效果的影响，如图 3.5所示。DCGASC 折扣系数 α 设置为 0.4。隐藏特征向量维度 K 在数据集 Foursquare 设置为 20，数据集 Gowalla 上设置为 40。从图 3.5中可以看出，锚点数量从 0 到 100，基于用户选择锚点的 LWMF$_u$ 的正确率和召回率都要优于基于训练集随机选择锚点的 LWMF。不过随着锚点数量的增加，两者间的差距越来越小。可以肯定地是，随着锚点数量越来越多，两者性能将会差不多。但考虑到锚点数量影响模型的训练时间，因此锚点数量越少越好，总体来看，LWMF$_u$ 在两个数据集上都要优于 LWMF$_u$_Random。

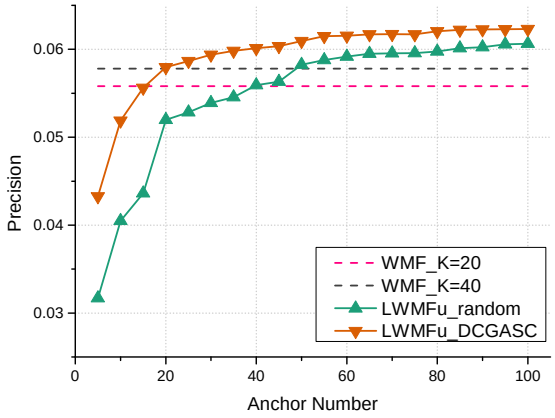
3.4.2.4 不同折扣参数推荐结果对比

最后，我们将研究折扣参数的不同对本文模型 LWMF$_u$ 最终推荐效果的影响。跟上小节一样，隐藏特征向量维度 K 在数据集 Foursquare 设置为 20，数据集 Gowalla 上设置为 40。对于折扣系数 α，我们验证了范围 $[0.1, 0.9]$ 之间以 0.1 为间隔的推荐结果。因为不同的折扣参数，推荐的正确率和召回率较为相似，我们在本文中只画出 $\alpha \in \{0.2, 0.4, 0.6, 0.8\}$ 的结果曲线，如图 3.6所示。可以看出，四个不同折扣参数之间的推荐效果差距是非常小的。相对来说，折扣参数 $\alpha = 0.4$ 是

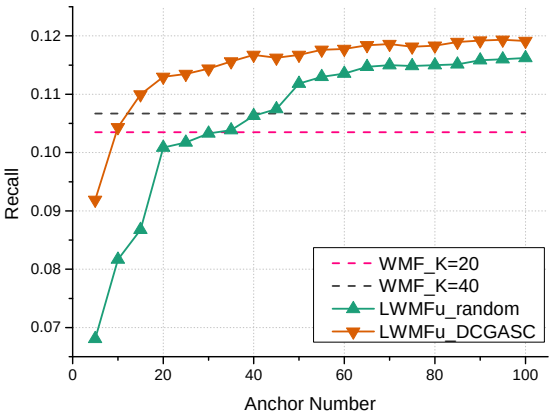


(a) Precsion on Foursquare

(b) Recall on Foursquare

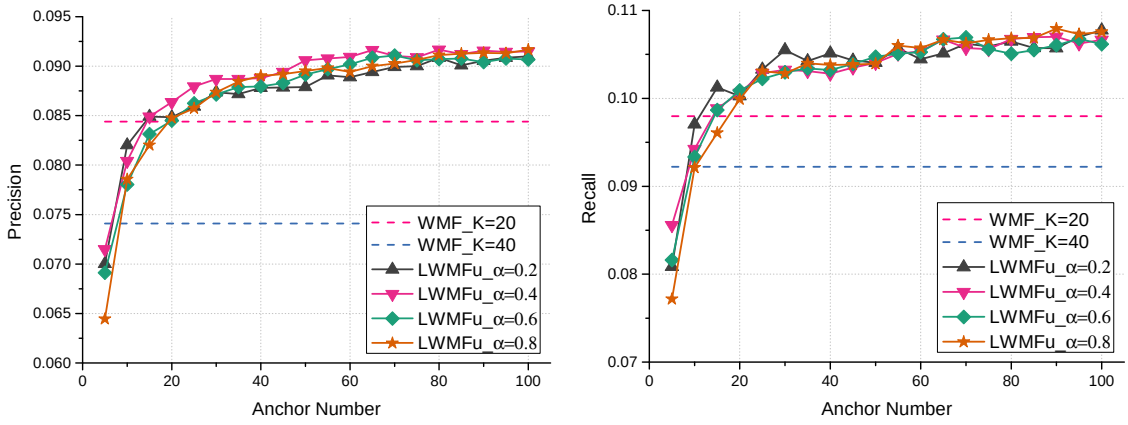


(c) Precsion on Gowalla



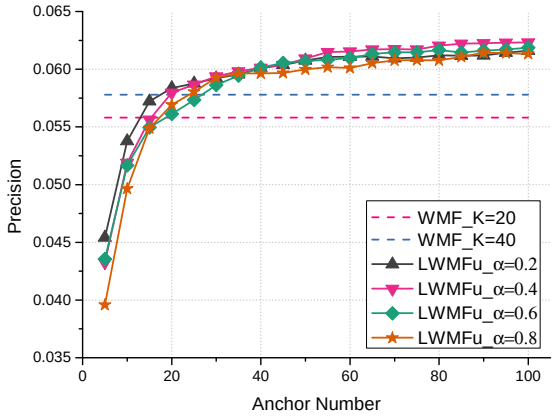
(d) Recall on Gowalla

图 3.5: 不同锚点选择方法推荐结果对比

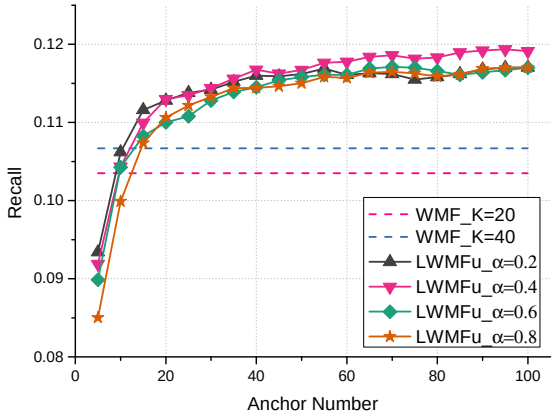


(a) Precsion on Foursquare

(b) Recall on Foursquare



(c) Precsion on Gowalla



(d) Recall on Gowalla

图 3.6: 不同折扣参数推荐结果对比

略微优于其他参数设置。总得来说，LWMF 的推荐性能对折扣参数的设置不敏感，主要依靠于锚点数量，锚点的选择方法以及隐藏特征向量维度三个方面。

3.5 本章小结

在本文中，我们提出 LWMF，其选择子矩阵以更好地建模用户行为。LWMF 通过子矩阵分解来解决稀疏问题。此外，我们提出 DCGASC 选择提高 LWMF 性能的子矩阵集。对两个真实数据集的大量实验证明了我们的方法与最先进的 WMF 方法相比的有效性。

我们想研究另外三个方向:（1）加快选择子矩阵;（2）在本文中，我们首先通过选择锚点选择子矩阵集合，然后对每个子矩阵进行加权矩阵分解。因此，我们需要两个步骤来优化目标函数。我们可以尝试找到仅在一个目标函数中优化局部矩阵分解的方法;（3）在一些特殊情况下（例如，POI 推荐器中的地理信息），我们可以进一步利用其他特殊附加信息到 LWMF。

第四章　基于多主题矩阵分解的评分预测推荐系统

本章重点介绍显式评分数据评分预测的研究内容。首先，章节 4.1描述本章的研究内容和大概的解决思路。接着，章节

4.1　引言

如今，推荐系统在电子商务服务中发挥越来越重要的作用。用于个性化推荐的典型任务是评级预测，其基于她的历史数据来预测用户对给定项目的评级。在等级预测，特别是矩阵分解技术（MF）[Koren 等，2009] 的文献中已经提出了各种方法。MF 将用户和项目投入潜在的低维空间。此外，可以使用用户和项目潜在向量之间的点积来恢复原始矩阵中的丢失条目。MF 已被证明在许多真实系统和竞争中表现良好，例如 Netflix 奖和 KDD Cup 2011 推荐音乐作品。

最近，局部矩阵分解 [Lee 等，2013] 已被证明比传统的 MF 更有效。原始矩阵被分成几个较小的子矩阵，其中我们可以利用局部结构来获得更好的低秩近似。在每个子矩阵中，应用标准 MF 技术来为用户和项目生成子矩阵特定的潜在向量。通常，使用聚类技术获得这些子矩阵。通过组合多个局部 MF 的结果，原始矩阵 R 由 a 重构设置

$$\hat{\mathbf{R}}_{um} = \frac{1}{\mathbf{Z}_{um}} \sum_{k=1}^{K} \mathbf{L}_{um}^{(k)} \mathbf{R}_{um}^{(k)} \tag{4.1}$$

表示子矩阵 R（k）中的项 Rum 的权重。这种子矩阵集合方法的两个关键问题是（1）如何产生子矩阵和（2）如何设置子矩阵的集合权重。使用随机抽样 [Mackey et al。, 2011]，用最近的邻居扩展锚点，尝试解决这两个问题 [Lee et al。, 2013; 2014] 或基于共聚类的矩阵分割 [Chen et al。, 2015]。虽然这些研究在某种程度上比传统 MF 有所改进，但缺乏一种更有原则的方法来表征局部矩阵分解。通过回顾以前的研究 [Mackey 等人，2011; Lee 等人，2013; 2014; Chen 等人，2015]，我们有两个重

要的意见:（1）每个子矩阵可以被认为是用户和项目的本地群集;（2）用户或项目在不同子矩阵中具有多个潜在表示。受这两个观察的启发，我们提出了一种新颖的贝叶斯概率多主题矩阵因子分解模型（BPMTMF）来进行评级预测。我们的模型包括两个部分，即建模的额定项目和建模的评级。对于第一部分，我们将用户的一组额定项目视为文档，并且使用潜在主题模型来将项目“聚类”为主题，其是项目集合上的多项分布。随后，用户具有在主题集合上的分布（即，主题分布）。基于这样的主题，我们进一步为用户和项目设置主题特定的潜在向量。我们以一个完整的贝叶斯方法整合上述两个部分。我们的模型的最终预测是结果的整体

从每个主题中的主题专用潜在向量生成。我们的模型描述了公式中的核心思想。1 以贝叶斯概率的方式：每个主题可以被认为是一个集群。使用多主题潜在表示，我们的模型更强大以反映用户和项目在评级预测中的复杂特性。使用主题的另一个重要优点是我们的方法具有更好的模型可互操作性。由于主题模型有效地发现共同的主题语义 [Blei 等人，2003]，我们模型中的派生主题也将高度相关的项目分组在一起。这样，一个主题将比在以前的研究中获得的更加一致 [Mackey et al。，2011; Lee 等人，2013; 2014; Chen 等人，2015]。将主题作为上下文信息，我们可以分析用户的评级偏好在不同的主题上下文中是如何变化的。我们的工作第一次提出了一种局部矩阵分解的贝叶斯公式，它将主题模型与概率矩阵分解模型结合起来。通过使用主题作为集群，我们的方法有更好的模型可解释性。对大型真实世界数据集的广泛实验证明了所提出的模型与多个竞争性基线相比的有效性。

4.2 模型预备知识

本文中使用的符号术语表列在表 1 中。在下面，在表4.1中列出了纸张中使用的符号的词汇表。具体地，我们使用大写粗体字母表示矩阵，手写体表示集合。不同字符的上标，例如 $\mathbf{R}^h$ 和 $\mathcal{U}^h$，表示不同的子矩阵和不同的子集合。矩阵的小标表示该矩阵的索引。例如，$\mathbf{R}^h_{um}$ 表示原始数据矩阵 $\mathbf{R}$ 的第 h 个子矩阵 $\mathbf{R}^h$ 中第 u 个用户，第 m 个商品的值。此外，使用上标 $^\top$ 表示横向量，否则则为竖向量。

表 4.1: 本章主要符号说明

符号	符号描述
N, M	用户和商品数量（数据矩阵的行数和列数）
$\mathbf{R}$	数据评分矩阵 ($\in \mathbb{R}^{N\times M}$)
K	主题个数 ($\ll \min(N, M)$) （子矩阵个数）
D	隐藏特征向量维度 ($\ll \min(N, M)$)
$i = \langle u, m\rangle$	数据评分矩阵 $\mathbf{R}$ 中第 i 个数据点用户-商品对
$\mathbf{P}_u^{(k)}$	第 u 个用户在第 k 个主题上的隐藏特征向量 ($\in \mathbb{R}^D$)
$\mathbf{Q}_m^{(k)}$	第 m 个商品在第 k 个主题上的隐藏特征向量 ($\in \mathbb{R}^D$)
z_i ($z_{u,m}$)	第 i 个观察数据点 $i = \langle u, m\rangle$ 上分配的主题
θ_u	第 u 个用户的主题分布 ($\in \mathbb{R}^K$)
ϕ_k	第 k 个主题上的商品分布 ($\in \mathbb{R}^M$)
α	主题模型上主题分布的狄利克雷先验参数
β	主题模型上商品分布的狄利克雷先验参数
Ψ_0	概率矩阵分解上高斯先验的 Gaussian-Wishart 先验参数

4.3 贝叶斯多主题矩阵分解

在本节中，我们提出我们的 BPMTMF 模型用于评级预测。

4.3.1 模型概览

本章节的主要思想是利用主题模型构造子矩阵聚类，然后利用多个特定主题的概率矩阵分解集成预测评分值。因此，我们的模型主要由两部分组成：

- 对用户消费商品这一行为建模；
- 用户消费商品后，会对商品进行评分，对这一分数进行建模。

4.3.1.1 对用户消费商品行为建模

为了对用户消费商品行为建模，我们采用一种类似用于文本文档聚类的标准主题模型的方法（例如，潜在狄利克雷分布 [82]）做以下的类比：一个商品被认为是文档中的一个词，那么一个用户所评价过的商品集合作为一个文档。使用这种方法，一个主题（*i.e.,* 商品主题）被定义为一个在商品集合上的多项式分布。我们让

ϕ_k 表示为第 k 个主题，以及让 $\phi_{k,m}$ 表示第 k 个主题中生成第 m 个商品的概率。给定 K 个主题的集合，用户偏好则是这 K 个主题上的多项式分布。我们让 θ_u 表示第 u 个用户的主题分布，以及让 $\theta_{u,k}$ 表示第 u 个用户的主题分布上第 k 个主题概率。我们引入对象狄利克雷先验 $Dir(\alpha)$ 和 $Dir(\beta)$ 来分别生成多项式分布 θ 和 ϕ，超参数分别为 α 和 β。主题建模一般用于产生用户消费过的商品集合，如下表示：

$$P(\{\langle u,m\rangle\}) \propto \prod_{\langle u,m\rangle} \left(\sum_k \theta_{u,k}\phi_{k,m} \right), \tag{4.2}$$

其中 $\langle u,m\rangle$ 数据点对表示第 u 个用户已经消费过第 m 个商品。公式 4.2列举了训练数据集中所有的 $\langle u,m\rangle$ 数据点对。

4.3.1.2 对评分建模

为了对分数建模，主题被认为是上下文信息，更进一步说，一个用户或者商品在每个主题上会有一个主题相关的隐藏特征向量与之相对应。我们让 $\mathbf{P}_u^{(k)} \in \mathbb{R}^D$（或者 $\mathbf{Q}_m^{(k)} \in \mathbb{R}^D$）表示第 u 个用户（或者 m 第个商品）第 k 个主题对应的特定主题隐藏特征向量。我们假设，每个用户将在不同主题上下文信息下表现出不同评分偏好，以及每个商品也将会在不同主题上下文信息下显式出不同的被评分模式。举个例子，如果一个用户是星球大战粉丝，那么相对于其他动作电影来说，这个用户可能会对 “星球大战” 系列电影打出较高的分数。这里需要注意，简单地融合目录偏差，如论文 [110, 111] 所示，是不适用于上述例子的，因为用户在 “动作” 类影片中既可能打高分也有可能大低分。融合目录偏差，比较适用于以下情形，例如，儿童用户更喜欢 “动漫” 类电影，而理解不了 “人性” 类电影从而打低分。而我们的模型尝试去建模用户评分行为时主题上下文对个人因素的影响。这样就能保证我们的模型能够适用于以上两种，甚至更多的情形。为了获得特定主题的隐藏特征向量，我们采用在隐藏特征向量 $\mathbf{P}^{(k)}$ 和 $\mathbf{Q}^{(k)}$ 上使用 Gaussian-Wishart 先验参数，对

应的超参数有 $\Psi_{\mathbf{P}}^{(k)} = \{\mu_{\mathbf{P}}^{(k)}, \Lambda_{\mathbf{P}}^{(k)}\}$ 和 $\Psi_{\mathbf{Q}}^{(k)} = \{\mu_{\mathbf{Q}}^{(k)}, \Lambda_{\mathbf{Q}}^{(k)}\}$，具体生成如下表示：

$$P(\Psi^{(k)}|\Psi_0^{(k)}) = \mathcal{N}(\mu^{(k)}|\mu_0^{(k)}, (\xi_0^{(k)}\Lambda^{(k)})^{-1})\mathcal{W}(\Lambda^{(k)}|\mathbf{W}_0^{(k)}, \nu_0^{(k)}) \tag{4.3}$$

其中 $\nu_0^{(k)}$ 是 Wishart 分布 $\mathcal{W}^{(k)}$ 自由度参数，$\mathbf{W}_0^{(k)}$ 是规模矩阵（对于用户，$\mathbf{W}_0^{(k)} \in \mathbb{R}^{N\times N}$；对于商品，$\mathbf{W}_0^{(k)} \in \mathbb{R}^{M\times M}$），$\Psi_0^{(k)} = \{\mu_0^{(k)}, \nu_0^{(k)}, \mathbf{W}_0^{(k)}\}$ 则是所有第 k 个主题对应的参数。类似于论文 [46] 所示，如上先验参数，则是由如下公式生成：

$$\begin{aligned}
&\mu_0^{(k)^*} = \frac{\beta_0\mu_0 + N^{(k)}\bar{\mathbf{P}^{(k)}}}{\beta_0 + N^{(k)}},\ \ \beta_0^{(k)^*} = \beta_0 + N^{(k)},\ \ \nu_0^{(k)^*} = \nu_0 + N^{(k)} \\
&[\mathbf{W}_0^{(k)^*}]^{-1} = \mathbf{W}_0^{-1} + N^{(k)}\bar{S}^{(k)} + \frac{\beta_0 N^{(k)}}{\beta_0 + N^{(k)}}(\mu_0 - \bar{\mathbf{P}}^{(k)})(\mu_0 - \bar{\mathbf{P}}^{(k)})^{\mathrm{T}} \\
&\bar{\mathbf{P}}^{(k)} = \frac{1}{N^{(k)}}\sum_{u\in U^{(k)}}\mathbf{P}_u^{(k)},\ \ \bar{S} = \frac{1}{N^{(k)}}(\mathbf{P}_u^{(k)} - \bar{\mathbf{P}}^{(k)})(\mathbf{P}_u^{(k)} - \bar{\mathbf{P}}^{(k)})^{\mathrm{T}}
\end{aligned}$$

因此，给定第 k 个主题，利用 Gaussian-Wishart 公式 4.3得到先验参数进而计算特定主题的用户隐藏特征向量 $\mathbf{P}^{(k)}$ 和商品隐藏特征向量 $\mathbf{Q}^{(k)}$，我们就可以根据高斯分布计算出该主题下的第 u 个用户对第 m 个商品的评分值：

$$P(\mathbf{R}_{um}|\mathbf{P}_u^{(k)}, \mathbf{Q}_m^{(k)}, \sigma_k^2) = \mathcal{N}(\mathbf{R}_{um}|{\mathbf{P}_u^{(k)}}^{\top}\mathbf{Q}_m^{(k)}, \sigma_k^2), \tag{4.4}$$

其中 ${\mathbf{P}_u^{(k)}}^{\top}\mathbf{Q}_m^{(k)}$ 为高斯分布均值，σ_k^2 是高斯分布的方差。

4.3.1.3 最终模型

我们提出的模型，称为贝叶斯概率多主体矩阵分解模型（Bayesian Probabilistic Multi-Topic Matrix Factorization, BPMTMF），融合了两名两个组件，*i.e.,* 对用户消费商品行为建模（公式 4.2）和对评分建模（公式 4.4），使用了全贝叶斯的方法。图 4.1展示了 BPMTMF 的生成过程。我们使用商品主题连接两个组件，也就是使用特定主题的隐藏特征向量。生成过程可以如下描述。当第 u 个用户想要对第 m 个商品进行评分时，她首先需要根据她的主题分布 θ_u 选择第 k 个主题 z，然后第

m 个商品有该主题 z 生成。最后，评分是基于第 k 个主题 z 的用户隐藏特征向量 $\mathbf{P}_u^{(k)}$ 和商品隐藏特征向量 $\mathbf{Q}_m^{(k)}$ 生成，该评分分布符合高斯分布。因此，给定参数，对于所有的评分的似然函数如下表示：

$$
\begin{aligned}
P(\mathbf{R}|\alpha,\beta,\Psi_0,\sigma) = & \\
& \int \Big(\prod_u P(\theta_u|\alpha)\Big)\Big(\prod_k P(\phi_k|\beta)P(\Psi_{\mathbf{P}}^{(k)}|\Psi_0^{(k)})P(\Psi_{\mathbf{Q}}^{(k)}|\Psi_0^{(k)})\Big) \\
& \Big(\prod_k\prod_m P(\mathbf{Q}_m^{(k)}|\Psi_{\mathbf{Q}}^{(k)})\Big)\Big(\prod_k\prod_u P(\mathbf{P}_u^{(k)}|\Psi_{\mathbf{P}}^{(k)})\Big) \\
& \Big(\prod_{\langle u,m\rangle}\sum_k \theta_{u,k}\cdot\phi_{k,m}\cdot P(\mathbf{R}_{um}|\mathbf{P}_u^{(k)},\mathbf{Q}_m^{(k)},\sigma_k^2)\Big) \\
& \mathrm{d}\mathbf{P}_u^{(k)}\mathrm{d}\mathbf{Q}_m^{(k)}\mathrm{d}\Psi_{\mathbf{P}}^{(k)}\mathrm{d}\Psi_{\mathbf{Q}}^{(k)}\mathrm{d}\theta_u\mathrm{d}\phi_k.
\end{aligned}
\tag{4.5}
$$

值得注意地是，设置特定主题的隐藏特征向量将会随着优化而造成过拟合，然而我们的贝叶斯方法通过先验参数可以有效地防止模型的复杂度。尽管我们融合了更多的超参数，在论文 BPMF [46] 以及我们实验结果表明，模型性能相对来说对参数的取值不敏感。

4.3.2 吉布斯采样参数学习

在我们的模型中，需要学习的参数（或者变量）列出如下：

(1) $\{\theta_u\}$，用户主题分布；

(2) $\{\phi_k\}$，主题商品分布；

(3) $\{\mathbf{P}_u^{(k)}\}$，第 k 个主题的用户隐藏特征向量，$k = 1, 2, ..., K$；

(4) $\{\mathbf{Q}_m^{(k)}\}$，第 k 个主题的商品隐藏特征向量，$k = 1, 2, ..., K$；

我们的任务就是去学习这些参数 $\{\theta, \phi, \mathbf{P}, \mathbf{Q}\}$，以至于可以最大化观察评分矩阵 $\mathbf{R}$ 的似然函数。因为参数的复杂性和隐藏变量等因素的存在，去直接优化该目

1. 对每一个主题 $k = 1, ..., K$,

   (1) 抽样一个多项式主题分布 $\phi_k \sim Dir(\beta)$

   (2) 抽样特定主题的用户隐藏特征向量和商品隐藏特征向量的超参数 $P(\Psi_{\mathbf{P}}^{(k)}|\Psi_0^{(k)})$ 和 $P(\Psi_{\mathbf{Q}}^{(k)}|\Psi_0^{(k)})$

2. 对每一个商品 $m = 1, ..., M$,

   i. 对每个主题 $k = 1,...,K$，抽样特定主题的商品隐藏特征向量 $\mathbf{Q}_m^{(k)} \sim P(\mathbf{Q}_m^{(k)}|\Psi_{\mathbf{Q}}^{(k)})$

3. 对每一个用户 $u = 1, ..., N$,

   i. 抽样 $\theta_u \sim Dir(\alpha)$

   ii. 对每一个主题 $k = 1, ..., K$，抽样特定主题的用户隐藏特征向量 $\mathbf{P}_u^{(k)} \sim P(\mathbf{P}_u^{(k)}|\Psi_{\mathbf{P}}^{(k)})$

   iii. 对每个被第 u 个用评分的商品（第 m 个商品）

   (1) 抽样一个主题 $z \sim Disc(\theta_u)$（第 k 个主题）

   (2) 从第 k 个主题中抽样第 m 个商品，$m \sim Disc(\phi_z)$

   (3) 根据高斯分布抽样评分，$\mathbf{R}_{um} \sim \mathcal{N}(\mathbf{R}_{um}|{\mathbf{P}_u^{(k)}}^\top \mathbf{Q}_m^{(k)}, \sigma_k^2)$

图 4.1: BPMTMF 模型生成过程

标函数显得非常困难。因此，在推理和参数学习上我们将采用广泛使用的 Gibbs 采样算法进行优化。在每轮迭代中，我们相互迭代学习主题分布，更新特定主题的用户隐藏特征向量 $\{\mathbf{P}_u^{(k)}\}$ 和商品隐藏特征向量 $\{\mathbf{Q}_m^{(k)}\}$。当算法达到收敛时，我们将使用各个数据点的主题选择来估计用户主题分布变量 $\{\theta_u\}$ 和主题商品分布变量 $\{\phi_k\}$。

4.3.2.1 推断主题分布

固定所有其他特定主题的隐藏特征向量和超参数，我们对数据点 $i = \langle u, m\rangle$ 能够得到以下条件概率分布：

$$
\begin{aligned}
&P(z_i = k|\mathbf{Z}_{\neg i}, \mathbf{R}, \mathbf{P}, \mathbf{Q}, \alpha, \beta, \sigma) \\
&\propto \frac{n_u^k + \alpha - 1}{\sum_{j=1}^{K}(n_u^j + \alpha) - 1} \times \frac{n_m^k + \beta - 1}{\sum_{h=1}^{M}(n_h^k + \beta) - 1} \times \mathcal{N}(\mathbf{R}_{um}|\mathbf{P}_u^{(k)\top}\mathbf{Q}_m^{(k)}, \sigma_k^2),
\end{aligned} \tag{4.6}
$$

其中 n_u^k 表示第 u 个用户评分商品属于第 k 个主题的个数，n_m^k 代表是那些在第 m 个商品评过分且属于第 k 个主题的用户个数，以及评分值 $\mathbf{R}_{um}$ 是通过在第 k 个主题上利用该主题的隐藏特征向量，使用高斯分布 $\mathcal{N}(\mathbf{R}_{um}|\mathbf{P}_u^{(k)\top}\mathbf{Q}_m^{(k)}, \sigma_k^2)$ 产生的。实际上，该采样公式类似于原始 LDA 模型的 Gibbs 优化公式 [112]，只是在该公式的基础上我们加入了评分产生的公式项。

4.3.2.2 更新特定主题的隐藏特征向量。

更新特定主题的用户和商品隐藏特征向量跟原始的贝叶斯概率矩阵分解（BPMF）[46] 非常相似。两者不同是，我们假设每个被评分的商品的主题是给定的，因此每个特定主题的用户商品隐藏特征向量的更新跟特定主题的用户和商品有关。这里，特定主题的用户隐藏特征向量 $\mathbf{P}_u^{(k)}$ 的条件概率是高斯分布：

$$P(\mathbf{P}_u^{(k)}|\mathbf{R},\mathbf{Q}^{(k)},\Psi_{\mathbf{P}}^{(k)},\sigma_k^2)=\mathcal{N}(\mathbf{P}_u^{(k)}|\mu_{\mathbf{P}}^{(k)^*},[\Lambda_{\mathbf{P}}^{(k)^*}]^{-1}) \tag{4.7}$$
$$\propto P(\mathbf{P}_u^{(k)}|\mu_{\mathbf{P}}^{(k)},\Lambda_{\mathbf{P}}^{(k)})\prod_{m=1}^{M}\mathcal{N}(\mathbf{R}_{um}|{\mathbf{P}_u^{(k)}}^{\top}\mathbf{Q}_m^{(k)},\sigma_k^2)^{\mathbf{I}_{um}^{(k)}},$$

这里原始高斯分布的均值和方差如下公式计算:

$$\Lambda_u^{(k)^*}=\Lambda_{\mathbf{P}^{(k)}}+\frac{1}{\sigma_k^2}\sum_{m=1}^{M}(\mathbf{Q}_m^{(k)}{\mathbf{Q}_m^{(k)}}^{\top})^{\mathbf{I}_{um}^{(k)}} \tag{4.8}$$

$$\mu_u^{(k)^*}=[\Lambda_u^{(k)^*}]^{-1}(\Lambda_{\mathbf{P}^{(k)}}\mu_{\mathbf{P}^{(k)}}+\frac{1}{\sigma_k^2}\sum_{m=1}^{M}(\mathbf{Q}_m^{(k)}\mathbf{R}_{um}))^{\mathbf{I}_{um}^{(k)}} \tag{4.9}$$

其中 $\mathbf{I}_{um}^{(k)}$ 是指示标识值，当主题分布符合 $z_{u,m}=k$ 时，值为 1，其他情况则为 0，即不考虑在内。同理，我们使用相似的公式计算特定主题的商品隐藏特征向量 $\{\mathbf{Q}_m^{(k)}\}$，这里省略描述。

4.3.2.3 整体学习算法。

In Alg. **??**，我们描述了一个为本文 BPMTMF 模型进行优化的整体 Gibbs 采样学习算法 [113]。在最开始，我们使用 BPMF 得到全局的用户隐藏特征向量 $\mathbf{P}$ 和商品隐藏特征向量 $\mathbf{Q}$，去初始化特定主题的用户隐藏特征向量 $\mathbf{P}^{(k)}$ 和商品隐藏特征向量 $\mathbf{Q}^{(k)}$。对于数据点主题分布，我们使用标准的 LDA（不考虑评分的情况下）进行初始化训练集中数据点主题分布。在每轮迭代，我们首先对所有可观察到的训练集数据点进行抽样，分配主题，然后固定数据点的主题分布更新特定主题的用户隐藏特征向量 $\mathbf{P}^{(k)}$ 和商品隐藏特征向量 $\mathbf{Q}^{(k)}$。过了初始时间阶段，我们将通过简单的计算方法估计每一轮后的用户主题分布参数 $\{\theta_u\}$ 和主题商品分布参数 $\{\phi_k\}$，公式如下:

算法 3 BPMTMF 学习算法

输入: 评分矩阵 $\mathbf{R}$，主题个数 K，用户和商品隐藏特征向量维度 D;

使用 BPMF 得到的全局用户和商品隐藏特征向量初始化各个局部隐藏特征向量 $\mathbf{P}^{(k)}$ and $\mathbf{Q}^{(k)}$;

使用 LDA 的主题分布初始化 BPMTMF 中的主题分布;

repeat

  for 每一个数据点 $i = \langle u, m \rangle$ **do**

    利用公式 4.6对数据点 i 分配主题 z_i:

$$z_i = P(z_i = k | \mathbf{Z}_{\neg i}, \mathbf{R}, \mathbf{P}, \mathbf{Q}, \alpha, \beta, \sigma)$$

  for 每个主题 $k = 1, 2, ..., K$ **do**

    根据公式 4.3采样高斯-维斯特先验参数:

$$\Psi_{\mathbf{P}}^{(k)} = P(\Psi_{\mathbf{P}}^{(k)} | \Psi_0^{(k)}),$$
$$\Psi_{\mathbf{Q}}^{(k)} = P(\Psi_{\mathbf{Q}}^{(k)} | \Psi_0^{(k)})$$

    for 每个用户 $u = 1, 2, ..., N$ **do**

      根据公式 4.7采样用户主题特定的隐藏特征向量:

$$\mathbf{P}_u^{(k)} = P(\mathbf{P}_u^{(k)} | \mathbf{R}, \mathbf{Q}^{(k)}, \Psi_{\mathbf{P}}^{(k)}, \sigma_k)$$

    for 每个商品 $m = 1, 2, ..., M$ **do**

      采用商品主题特定的隐藏特征向量:

$$\mathbf{Q}_m^{(k)} = P(\mathbf{Q}_u^{(k)} | \mathbf{R}, \mathbf{P}^{(k)}, \Psi_{\mathbf{Q}}^{(k)}, \sigma_k)$$

until 收敛

$$\theta_{u,k} = \frac{n_u^k + \alpha}{\sum_{j=1}^{K}(n_u^j + \alpha)}, \tag{4.10}$$
$$\phi_{k,m} = \frac{n_m^k + \beta}{\sum_{h=1}^{M}(n_h^k + \beta)},$$

其中 n_u^k，n_u^k，n_m^k 和 n_h^k 是在公式 4.6各类计数。

4.3.2.4 计算复杂度分析。

我们让符号 $|\mathbf{R}|$ 表示数据矩阵 $\mathbf{R}$ 的非零观察值的个数，以及 $n_{u,\cdot}^k = \sum_m \mathbf{I}_{um}^{(k)}$ 代表第 u 个用户第 k 个主题中评过分的商品个数。在每一轮迭代中，更新主题分配任务中，总共需要采样 $|\mathbf{R}|$ 次，每次样本点根据公式 4.6需要计算 K 次概率，且每次概率计算（公式 4.6）因为加入了评分项计算，复杂度为 $O(D)$，所以更新整个主题分配的复杂度为 $\mathcal{O}(KDS)$。对于更新用户的隐藏特征向量，主要的计算花费在公式 Eq. 4.8上 $\sum_{m=1}^{M}(\mathbf{Q}_m^{(k)}\mathbf{Q}_m^{(k)^\top})^{\mathbf{I}_{um}^{(k)}}$ 计算均值部分以及公式 4.9计算方差的矩阵逆计算项 $\Lambda_u^{(k)^*}$，对应的复杂度分别为 $\mathcal{O}(D^2 n_{u,\cdot}^{(k)})$ 和 $\mathcal{O}(D^3)$。这里类似于文章 [1]，尽管有更高效的算法存在，我们假设对于矩阵大小为 $\mathbb{R}^{D\times D}$ 的逆所需要的时间复杂度为 $\mathcal{O}(D^3)$。因此，更新均值 $\Lambda_u^{(k)^*}$ 和方差 $\mu_u^{(k)^*}$ 的时间复杂度为 $\mathcal{O}(D^3 + D^2 n_{u,\cdot}^{(k)})$。并且，我们还需要遍历 N 个用户和 K 个主题，这将会导致整个更新隐藏特征向量迭代复杂度变为 $\mathcal{O}(D^3KN + D^2|\mathbf{R}|)$，其中 $|\mathbf{R}| = \sum_{u,k} n_{u,\cdot}^k$。同理，更新全部商品的各个主题的隐藏特征向量的时间复杂度为 $\mathcal{O}(D^3KM + D^2|\mathbf{R}|)$。综合以上所有部分，每一轮迭代本文模型 BPMTMF 的时间复杂度为 $\mathcal{O}(DK|\mathbf{R}| + D^2|\mathbf{R}| + D^3KN + D^3KM)$。可以看出，总的算法复杂度跟数据集大小成正比，跟用户个数和商品个数成正比，跟主题个数成正比，跟隐藏特征向量维度的三次成正比。因为隐藏特征向量维度和主题个数一般不会太大，模型的时间复杂度跟数据集大小成正比是比较容易接受的。

4.3.3 评分预测

当所有 BPMTMF 的参数都学习优化完之后，我们将使用以下公式进行最终的用户商品评分预测：

$$\hat{\mathbf{R}}_{um} \approx \frac{1}{\sum_{k'=1}^{K} \theta_{u,k'} \cdot \phi_{m,k'}} \sum_{k=1}^{K} \left\{ (\theta_{u,k} \cdot \phi_{m,k}) (\mathbf{P}_u^{(k)\top} \mathbf{Q}_m^{(k)}) \right\},$$

其中 $\hat{\mathbf{R}}_{um}$ 是 $\mathbf{R}_{um}$ 的预测评分，其实就是用户商品在各个主题的加权平均预测值。需要注意的是，上述预测评分公式仅仅只用了一轮的 Gibbs 采样值，在多轮采样上则需要进行简单平均每轮预测值。

4.3.3.1 与相关工作的联系

公式 4.1展示了之前在非概率局部矩阵分解上相关工作的一般数学形式 [4, 6, 51]。感兴趣地是，我们的预测公式 4.11和公式 4.1有一个非常紧密的联系。给定一组用户-商品对数据点 $\langle u, m \rangle$，我们可以有相似的对应关系：$\mathbf{P}_u^{(k)\top} \mathbf{Q}_m^{(k)} \rightarrow \hat{\mathbf{R}}_{um}^{(k)}$ 代表的是在第 k 个 “聚类” 中数据点 $\mathbf{R}_{um}$ 的预测评分，$\theta_{u,k} \cdot \phi_{m,k} \rightarrow \mathbf{L}_{um}^{(k)}$ 则是代表第 k 个 “聚类” 该商品-用户对数据点的预测值 $\hat{\mathbf{R}}_{um}^{(k)}$，以及 $\sum_{k'=1}^{K} \theta_{u,k'} \cdot \phi_{m,k'} \rightarrow \mathbf{Z}_{um}$ 则扮演了归一化系数的角色。诸如此类的类比，我们的模型形式就可以被看作是之前非概率局部矩阵分解的概率版本：在公式 4.1一个聚类在本文 BPMTMF 模型中则代表的是一个主题。如此一个紧密的连接使得之前的非概率局部矩阵分解相关工作能够被概率的形式所解释，进而可以有更深入的理论分析和扩展。

4.4 实验及分析

4.4.1 实验设置

4.4.1.1 数据集说明

我们将在两个公共的、广泛作为标准数据集使用的电影数据上验证我们的BPMTMF 模型。一个是 Movielens 数据集[1], 一个是 Netflix 数据集[2]（2007 年, Netflix 百万美元奖金竞赛的数据集）。两个数据集的具体描述如表 4.2所示。我们随机将整个数据集以 9 : 1 的比例切分为训练集和测试集。最后取 5 次相同的切分数据集的模型预测作为最终推荐结果。

表 4.2: 两个数据集详细描述

数据集	# 用户	# 商品	# 评分	稠密度
Movielens	69,878	10,677	10,000,054	1.31%
Netflix	480,189	17,770	100,000,000	1.17%

4.4.1.2 评价指标

我们采用两个常用的评价指标去评价预测正确率，均方根误差（Root Mean Square Error, RMSE）和平均绝对误差（Mean Absolute Error, MAE），如下公式定义：

$$RMSE = \sqrt{\frac{\sum_{\langle u,m\rangle}(\mathbf{R}_{um} - \hat{\mathbf{R}}_{um})^2}{N_{test}}} \tag{4.11}$$

$$MAE = \frac{|\mathbf{R}_{um} - \hat{\mathbf{R}}_{um}|}{N_{test}} \tag{4.12}$$

其中 N_{test} 指的是测试集中有评分值的数据点数量。更小的 RMSE 和 MAE 代表了更好的性能。

[1] http://www.grouplens.org/

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

4.4.1.3 对比方法

我们将本文提出的 BPMTMF 与以下基线方法进行对比：

- DFC [51]: 该模型将大规模矩阵分解任务随机分解成更小的子问题，其中每个子矩阵中的数据没有一定的相关性，然后用均值将组合子矩阵预测值来近似原始矩阵。
- LLORMA [4][3]: 该方法从训练集中随机选择数个数据点作为锚点集合，然后根据每个锚点利用非参核函数选择于该锚点相关的训练数据点组成子矩阵，从而保证每个子矩阵具有局部相关性质，最后对每个子矩阵进行奇异值分解后利用公式 4.1进行加权平均聚合成原始近似矩阵。
- WEMAREC [6][4]: 该方法利用分段联合聚类对训练数据集聚类，每个聚类作为一个子矩阵，然后提出一种基于子矩阵的权重策略来预测最后的用户商品评分。
- PMTMF: 作为贝叶斯版局部矩阵分解的直接比较，我们也实现了非贝叶斯版的概率多主题矩阵分解模型，没有先验超参数，使用极大似然估计和期望最大化算法（Expectation-Maximization，EM 算法）进行优化主题分布部分，使用随机梯度下降优化特定主题的隐藏特征向量学习部分。

因为之前的研究 [4, 6, 51] 已经展示了上述基于局部的矩阵分解模型推荐性能优于传统方法，我们本文就不使用传统的矩阵分解 [8, 114] 作为基线方法进行比较。我们在 Java 推荐系统开源工具 librec [115] 上实现了本文的两个局部矩阵分解模型 PMTMF 和 BPMTMF。

[3]http://prea.gatech.edu/download.html#ver20，Ver2.0 版本

[4]https://github.com/ldscc/StableMA

4.4.1.4 参数设置

跟随论文 [4, 6] 参数的设置，对于所有局部矩阵分解模型，隐藏特征向量的维度 D 设置为 20。对于本文模型 PMTMF 和 BPMTMF，我们根据经验设置模型主题个数 K 为 20；跟随论文 [116]，用户主题分布狄利克雷超参数 α 设置为 $\frac{50}{K}$，主题商品分布狄利克雷超参数 β 设置为 0.01；对于特定主题的隐藏特征向量部分的超参数设置，我们跟随论文 BPMF [46] 的设置，初始化 $\mu_0^{(k)} = 0, \nu_0^{(k)} = D, \mathbf{W}_0^{(k)}$ 为单位矩阵以及方差 $\sigma_k^2 = 2$。由论文 [46] 的实验结果可以发现，当 BPMF 的迭代轮数超过 150 轮时，该模型的推荐准确率已经达到一个相对平衡的状态。因此，对于本文贝叶斯版本的概率主题模型，我们首先有 200 轮的主题分布模型和 100 轮的隐藏特征向量学习来初始化，之后又进行了 150 轮的相互迭代进行最终模型参数的学习。因为我们有相似的实验设置和相同的数据集，基线方法的其他参数设置将根据原始论文设置了其报告的最优参数数值。

4.4.2 实验结果及分析

4.4.2.1 评分预测性能比较

在表 4.3中，我们展示了各类评分预测模型的推荐性能。在所有基线方法中，最近提出的局部矩阵分解模型 WEAREC 性能最好。WEMAREC 采用了分段联合聚类方法来生成子矩阵，代表了当前局部矩阵分解模型的推荐性能水平。同时，我们也检验了本文提出的模型 PMTMF 和 BPMTMF 的性能。我们能够发现非贝叶斯版的概率多主题矩阵分解的推荐准确率仅仅比最好的基线方法 WEMAREC 差一点点。并且，可以看出贝叶斯版 BPMTMF 比 PMTMF 和 WEMAREC 预测准确率都要好上不少，同时也证明了全贝叶斯方法的有效性。因此，BPMTMF 比 PMTMF 推荐准确率要高很重要的一个原因是因为贝叶斯模型能够更加有效的控制模型的复杂度。对比这些基线方法，BPMTMF 提供了一种更有原则性的解决方案，去融合子矩阵生成和权重选择。得益于贝叶斯方法，有了先验超参数的存在，我们需要更少的经验去设置参数的选择同时得到较好的推荐效果。作为比较，WEMAREC

需要首先去设置联合聚类的一系列参数，并且在融合权重的时候需要更多相关参数设置的注意。

表 4.3: 不同方法 RMSE 结果对比

方法	MovieLens	Netflix
DFC	0.8064	0.8451
LLORMA	0.7834	0.8243
WEMAREC	0.7769	0.8142
PMTMF	0.7792	0.8198
BPMTMF	**0.7679**	**0.8081**

4.4.2.2 聚类分析

除了性能的提高外，本文提出的模型另外一个重要的优点是我们将聚类描绘为主题，使得每个子矩阵内的数据有了更多的内在含义。为了说明这一点，我们构造了定量的聚类结果分析。在这些基线方法中，DFC 利用随机的方法生成子矩阵，而 WEMAREC 虽然利用联合硬聚类来生成子矩阵，但却尝试产生更少的子矩阵（原文实验结果显示生成 4 个子矩阵 WEMAREC 推荐性能最好），因此这两中方法都适合用户聚类分析。我们选择了 LLORMA 作为 BPMTMF 的聚类分析比较对象。在 Movielens 数据集中，每部电影拥有一系列电影类型标签。我们首先利用 LLORMA 和 BPMTMF 去产生一个聚类（或者主题）集合。给定学习好的聚类（或者主题），我们将这个聚类中前十个代表电影出现最多的标签作为这个聚类的电影类型标签。表 4.4展示了两个有 LLORMA 和 BPMTMF 生成的聚类例子。我们能够发现 BPMTMF 能够生成更加清晰的聚类电影，也就是主题。有了跟家一致的主题属性，BPMTMF 也就能够为用户和商品提供更加可解释性的内在含义。

4.4.2.3 主题个数的影响

由于我们需要根据每个主题设置主题特定的用户隐藏特征向量和商品隐藏特征向量，因此本文提出的 BPMTMF 一个重要的参数是主题个数（*i.e.,* K）。我们以

表 4.4: Movielens 数据集上采样的两个主题或聚类的前 10 个流行电影

(a) 动作片

序号	LLORMA	BPMTMF
1	星球大战 4 之新希望	夺宝奇兵 3 之圣战骑兵
2	星球大战 5 之帝国反击战	夺宝奇兵
3	美国丽人	虎胆龙威
4	莎翁情史	星球大战 4 之新希望
5	拯救大兵瑞恩	终结者
6	外星人 E.T.	星球大战 5 之帝国反击战
7	傀儡人生	蝙蝠侠
8	第六感	星球大战 6 之绝地归来
9	黑衣人	夺宝奇兵 2 之魔域奇兵
10	星球大战 6 之绝地归来	猎杀红色十月号

(b) 剧情片

序号	LLORMA	BPMTMF
1	美国丽人	沉默的羔羊
2	勇敢的心	拯救大兵瑞恩
3	拯救大兵瑞恩	肖申克的救赎
4	洛城机密	美国丽人
5	星球大战 4 之新希望	低俗小说
6	沉默的羔羊	心灵捕手
7	亡命天涯	冰血暴
8	星球大战 6 之绝地归来	第六感
9	辛德勒的名单	阿甘正传
10	玩具总动员	勇敢的心

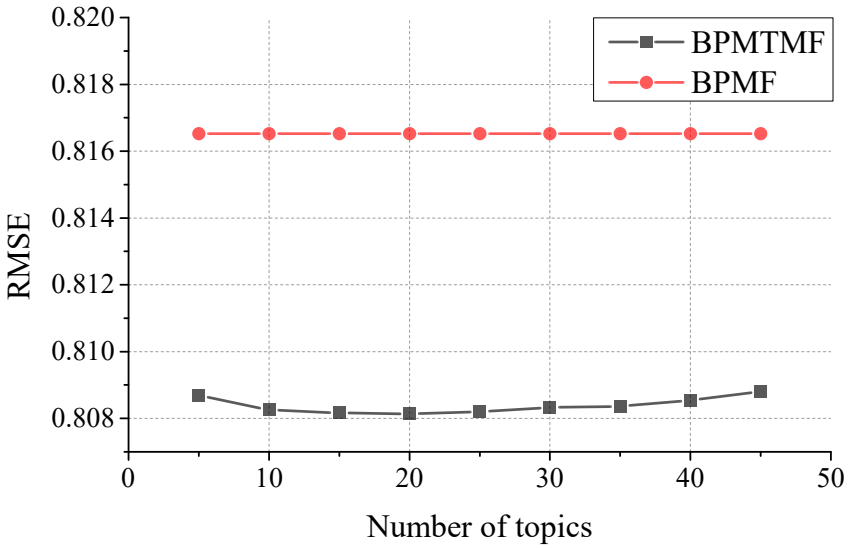


图 4.2: 不同主题个数对推荐准确率的影响

间隔为 5 在区间 $[5, 45]$ 上设置不同的主题个数 K。图 4.2报告了 BPMTMF 的 RMSE 性能。我们能观察到在主题个数符合 $15 \leq K \leq 25$ 时，BPMTMF 取得的推荐准确率最好。作为比较，我们也比较了 BPMF 的性能，BPMF 其实是只有一个主题的 BPMTMF，是 BPMTMF 的一种特殊情形。可以看出，当主题个数 $K = 5$ 时，BPMTMF 的推荐准确率已经远好于主题个数为 1 时。通过结合表 4.4聚类分析，我们能够看出利用多主题的矩阵分解模型能够有效地对用户和商品复杂的特性进行建模，从而提高预测评分的准确性。

我们同时也报告了 BPMTMF 每轮迭代的算法运行时间，如图 4.3所示。该运行时间实验我们使用电脑配置是 Ubuntu 系统，CPU 为 4-core Intel Xeon E5-1603 2.86GHz 以及 16G 内存。我们使用 Java 编程语言实现 BPMTMF。在之前的时间复杂度分析中，每一轮迭代本文模型 BPMTMF 的时间复杂度为 $\mathcal{O}(DK|\mathbf{R}| + D^2|\mathbf{R}| + D^3KN + D^3KM)$。在数据集大小和隐藏特征向量维度不变的情况下，BPMTMF 的时间复杂度与主题个数 K 成正比。从图 4.3可以看出，实验结果也符合我们的时间复杂度理论分析。为了提高算法运行效率，我们在以后工作中引入并行优化算法去加速参数学习过程。

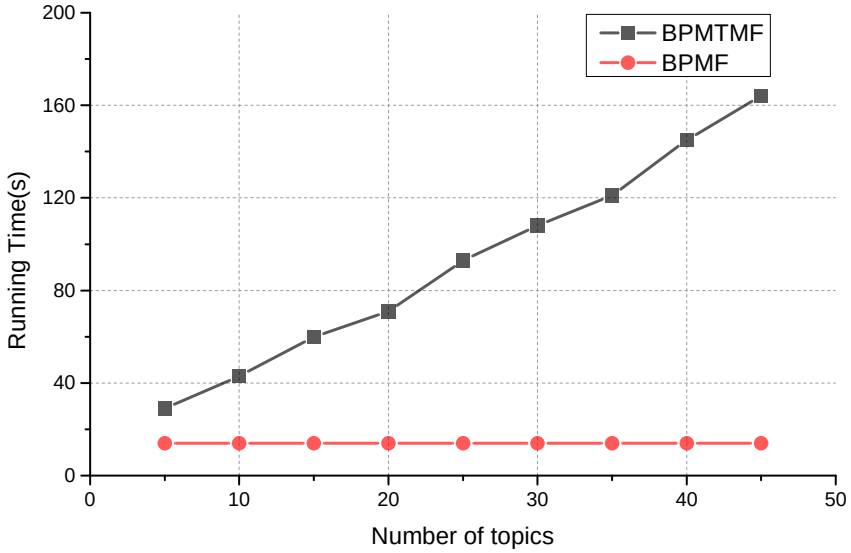


图 4.3: 不同主题个数对模型训练时间的影响

4.5 本章小结

在本章节，我们提出了局部矩阵分解的概率形式。我们的方法巧妙地融合了概率矩阵分解和主题模型到同一个模型当中，进而提出全贝叶斯版本的概率多主体矩阵分解 BPMTMF。我们同时也说明了之前一些非概率版本的矩阵矩阵分解模型和本文模型的紧密联系特征。我们的模型在评分预测上能够更加有效地反应用户和商品复杂的内在特性，以及有更好的模型解释性。另外，在大规模真实数据集上实验也表明了本文模型的有效性。

第五章　时间和频率感知的标签推荐系统

本章重点介绍时间和频率感知的标签推荐研究内容。首先，章节 5.1阐述标签推荐的研究背景、意义、研究贡献。其次，章节 5.2讨论与标签推荐以及模型相关的研究工作。然后章节 5.3介绍本章提出模型的相关基础模型。随后，章节 5.4重点介绍本章提出的标签推荐模型，以及相关参数学习等。章节 5.5通过实验验证模型的有效性。最后总结本章标签推荐的研究内容。

5.1　引言

随着信息技术和互联网的发展，从大量信息中找到自己感兴趣的信息是一件非常困难的事，推荐系统就是解决信息过载的重要途径之一，它通过分析用户的历史行为对用户的兴趣建模，从而能够主动给用户推荐能够满足他们兴趣和需求的信息。分众分类标签系统如 Delicious、Last.fm 和 Movielens 等网站允许用户使用关键字标记书签、音乐、电影等资源，这种行为被称为打标签。标签是 Web 2.0 的一个重要的特征，它可以帮助用户管理自己收藏的资源，方便浏览和检索。标签可以看作是一个隐式的打分，不仅可以用来表明资源的特征，也可以反映用户的个性特征。标签推荐就是当用户想要标注某个资源时，给这个用户推荐一系列标签供他选择。个性化的标签推荐系统通过分析用户以往的打标签行为来预测用户将来会使用的标签，推荐结果由用户和资源共同决定。例如两个用户曾经对同一件资源打过相同的标签，那么将来他们也有可能对另一件资源使用相同的标签。也有可能一个用户近期大量使用某个标签，那么之后的一段时间也很可能还使用该标签对网络资源进行标记。进行个性化的推荐之所以有意义是因为对同样的资源，用户倾向于使用不同的标签，比如 Last.fm 提供非个性化的标签推荐，但用户仍然会使用不同的标签来标记音乐资源。此外 [70] 也证明了个性化的标签推荐方法要优于非个性化的推荐方法的理论上限。

一些标签推荐系统使用张量分解技术来为候选标签排序。基于张量分解的模型将用户-资源-标签张量分解成三个维度的特征矩阵，来表示与用户、资源、标签相关的隐含特征。基于 Canonical 分解的 PITF[10] 张量分解方法，改进了基于 Tucker 分解的 RTF[70] 模型复杂度与分解维度成立方比的缺点，使得模型运行时间与数据集的大小及特征维度均成线性关系，可以更好地处理高维分解，同时保持张量分解类算法较好的推荐新信息的能力以及发现用户潜在的但自己尚未发现的兴趣偏好标签的优点。经典的张量分解类方法虽然有众多协同过滤的优点，但它们没有考虑到用户打标签的行为会随时间变化这一特征，以及较难处理好稀疏性问题。因此最近出现了一些基于时间和频次的标签推荐系统 BLL(Base-Level Learning）类方法，GIRP[117]，GIRPTM[117]，BLL[118]，BLLac[119] 等，这些方法仿照人脑存取长期记忆的方式，认为用户会倾向于再次使用自己最近最多使用过的标签。其中 BLL 类方法利用用户以往打标签行为与当前时间的间隔 (Recency)、标签的使用频次 (Frequency) 来估计用户将来重复使用某一标签的概率，并结合该网络资源中最流行标签进行推荐。但此类方法只能推荐历史记录中有的标签，没有推荐新标签的能力。

综合分析和考虑上述两类方法的优缺点，本文提出时间和频次加权的 PITF 模型 TFWPITF(Time and Frequency Weighted PITF)，在 PITF 模型的基础上增加对用户-标签-时间关系的权重以及资源-标签关系的权重，使得 TFWPITF 模型既能考虑用户打标签行为随时间变化而变化的现象，也能有效地利用相似用户，相似资源，相似标签的信息，从而更好地对用户对资源打标签的行为进行建模，提高标签推荐的准确度和新颖性。最后，我们在 Movielens、LastFM 和 Delicious 三个数据集上进行了对比实验，实验结果表明本文提出的方法在准确性上优于当前最新的标签推荐算法，同时还有较好的推荐新标签的能力。

5.2 相关工作

近年来标签成为社交网络中的重要特征，它方便用户以一种简单的机制来协同管理或搜索网络中的资源 [120]。尽管研究表明使用个人标签可以极大的提高网络资源的搜索效率 [121]，但仍然有很多人懒于给自己收藏的内容加上标签，因此需要个性化的标签推荐系统自动地给用户推荐标签，从而帮助用户更好的管理资源。目前的标签推荐方法分为两类：非个性化标签推荐方法和个性化标签推荐方法。非个性化的标签推荐系统针对某一资源，会给所有的用户推荐相同的标签，比如 MP_i，针对某一资源会给所有用户推荐目标资源上最热门的标签，这是一种非个性化的推荐。而 Rendle 等 [70] 证明了个性化推荐方法比理论上最优的非个性化的推荐方法的效果要好，因此本文只关注第二种方法。

最简单的个性化标签推荐方法包括 MP_u，给用户推荐他自身使用最多的标签；$\text{MP}_{u,i}$ ，将 MP_u 和 MP_i 进行线性组合；以及 [122] 中使用协同过滤算法 (CF) 进行标签推荐。Hotho 等人提出自适应的 PageRank(APR) 算法 [123]，其主要思想是如果一个资源被重要的用户用重要的标签标注过，那么它也是个重要的资源。之后 Robert 等 [124] 扩展了 APR 算法，提出 FolkRank(FR) 算法，推荐效果优于基于频率的方法和协同过滤算法。基于 Tucker 分解的分解模型也被广泛用于标签推荐，例如，文章 [125] 使用了 HOSVD 模型，该方法将所有未观察到的数据看作 0，以此来优化平方损失函数。除此之外，隐语义模型也被用来标签推荐，代表性方法有基于 LDA 主题模型的标签推荐系统 [9]，该方法使用拥有大量标签的资源来抽取隐含的主题，然后把待标注的资源映射到相应的主题上，这样就能把属于这些主题的标签推荐给用户；还有基于张量分解的 PITF 模型 [10]，显式地对用户、资源、标签两两之间的关系进行建模，有效地利用相似用户，相似资源，相似标签等信息来进行标签推荐；NLTF[126]，以非线性方式扩展了 Canonical 分解，并且使用了高斯径向基核函数，从而能更好地利用特征等。

除了最简单的基于标签热门程度的方法，其他方法完成推荐任务所需的计算量相对较大，而且这些方法忽略了用户打标签的行为会随时间变化这一现象 [127]。

因此最近的研究尝试在进行标签推荐时考虑时间因素，Zhang 等提出了 GIRP 方法 [117]，该方法基于用户打标签行为的频次和时间实现，使用指数分布对标签第一次被使用及最后一次被使用的时间进行建模；而 GIRPTM 方法 [117] 扩展了 GIRP 模型，推荐时考虑了目标资源上最热门的标签。另外还有基于认知科学的 BLL 类模型，比如 BLL$_{ac}$[119]、BLL+MP$_i$[11]、BLL$_{ac}$+MP$_i$[12]、3LT+MP[119][128] 等，这类方法认为用户会倾向于再次使用自己最近最多使用过的标签。BLL$_ac$ 扩展了 BLL，除了使用长尾分布对用户历史行为的频次和时间间隔进行建模外，还增加了关联模块 (Association Component) 描述目标资源的特征对用户的影响，即用户不一定总是选择自己最近最多使用过的标签，会随着目标资源的不同做出调整，但只会给用户推荐他之前使用过的标签。BLL+MP$_i$ 和 BLL$_{ac}$+MP$_i$ 分别扩展了 BLL 和 BLL$_{ac}$，都考虑给用户推荐目标资源上最热门的标签。3LT+MP 方法 [119, 128] 除了利用用户对标签的遗忘规律，还使用 LDA 来模仿用户存取记忆时访问的语义场，以获得对用户和资源更好的画像刻画，提高推荐的准确度。

5.3 模型基础知识

5.3.1 问题描述

标签推荐的任务是当用户要标注或描述某一资源时提供标签列表供用户选择。比如某个用户在浏览音乐网站时想要给某首歌打上标签，系统就会为其推荐可能会使用到的关键词标签列表。个性化标签推荐时会根据用户以往打标签的行为以及其他用户打标签的历史记录，推荐候选标签。比如系统会使用目标用户以前给其他资源打的标签或者其他用户给目标资源打过的标签等进行推荐。因此个性化标签推荐，就是在给定所有用户对资源带标签的历史记录，对用户在资源的推荐个性化标签列表。表 1 是本文可能用到的基本符号描述。

表 5.1: 基本符号描述

符号	符号描述
U, I, T	用户，资源，标签集合
u, i, t	某个用户，资源，标签
R	$R \subseteq U \times I \times T$, 用户打标签的历史记录
P_R	$\{(u,i)\|\exists t \in T : (u,i,t) \in R\}$ 观察到的用户-资源对
s	每个打标签记录对的时间戳

5.3.2 BLL+MP$_i$ 模型

BLL+MP$_i$ [11] 使用用户以往打标签行为的频次 (Frequency) 及历史打标签行为时间与当前时间的间隔 (Recency) 来建模，同时考虑了目标资源上最热门的标签对用户的影响。

首先根据某个标签被目标用户使用的记录来计算标签的活跃程度 BLA(Base-Level Activation):

$$BLA(t,u) = \ln(\sum_{i=1}^{n}(s_{ref} - s_j)^{-d}) \tag{5.1}$$

其中 n 是标签 t 被用户 u 使用的总次数，第 j 次的时间戳为 s_j，s_{ref} 表示用户当前要打标签的时间，时间间隔是两者的差值[1]，d 的取值为 0.5[118]。经过归一化处理后:

$$||BLA(t,u)|| = \frac{\exp(BLA(t,u))}{\sum_{t'=1}^{m}\exp(BLA(t',u))} \tag{5.2}$$

其中 m 表示用户 u 使用过的标签的总数。

除此之外该模型还考虑了用户当前所处上下文中的语义提示信息，即 MP$_i$，表示的是目标资源 i 上最热门的标签。给定用户-资源对 (u,i), 推荐标签预测分数可以下公式计算:

$$\hat{y}_{u,i,t} = \beta||BLA(u,t)|| + (1-\beta)|||Y_{t,i}||| \tag{5.3}$$

β 为调节用户-标签和资源-标签偏好的权重，当 $\beta = 0.0$ 时模型退化为 MP$_i$ 模型，

[1]原论文 BLL 代码: `https://github.com/learning-layers/TagRec/` 使用的是两者时间戳的差值 +1.0，且时间戳以秒为单位。

当 $\beta = 1.0$ 时模型变为基本的 BLL 模型，$|Y_{t,i}|$ 表示目标资源 i 在标签 t 上的数量。

5.4 局部时间和频率感知的排序模型

5.4.1 模型概览

从第5.3节可以看出，BLL 类方法（包括 BLL+MP$_i$ 和 BLL$_{ac}$+MP$_i$）考虑了用户打标签的历史记录和时间戳，能够模拟用户会倾向于再次使用自己最近最多使用过的标签的行为进行推荐，同时考虑了目标资源最热门的标签，部分解决了稀疏性问题，也有较强的个性化推荐，在标签推荐情境下能有较高的准确率。然而 BLL 类方法只能够推荐用户本来就熟悉的标签或者是目标资源最热门的标签，完全没有推荐新标签的能力。而 PITF 使用的基本模型是张量分解，本质上是协同过滤的推荐方法，具有协同过滤方法的优点，譬如有推荐新信息的能力，发现用户潜在的但自己尚未发现的兴趣偏好标签等。然而 PITF 缺少了时间和内容信息，没有对用户标签记忆模式建模，以及在冷启动情况下推荐效果不佳等缺点。

因此，本文综合考虑 BLL+MP$_i$ 和 PITF 模型的优缺点，提出基于时间和频次加权的 PITF 模型 (TFWPITF)，其对用户-资源 (u, i) 的预测值如下：

$$\hat{y}_{u,i,t,s} = w^s_{u,t} \sum_{k=1}^{K} \hat{u}_{u,k} \cdot \hat{t}^U_{t,k} + w_{i,t} \sum_{k=1}^{K} \hat{i}_{i,k} \cdot \hat{t}^I_{t,f} \tag{5.4}$$

其中 $w^s_{u,t}$ 是用户-标签-时间 (u, t, s) 权重，$w_{i,t}$ 是资源-标签对 (i, t) 权重。当用户 u 在时间 s 之前打过越多标签 t，且时间越近，那么用户-标签-时间 (u, i, t) 权重 $w^s_{u,t}$ 越大。类似地，所有用户在资源 i 上打过标签 t 越多，那么资源-标签对 (i, t) 权重越大。同时我们需要保证当用户 u 在时间 s 前未打过标签 t 时（或者没有用户在资源打过标签 t），权重 $w^s_{u,t}(w_{i,t})$ 大于零以保证在历史记录中未出现过但相关的标签也能有机会推荐给目标用户和目标资源。因此我们使用类似于 Hu 等 [1] 提出

的加权矩阵分解 (WMF) 中权重的设置方法：

$$w^s_{u,t} = 1 + \log_{10}(10^{\alpha^U} \cdot ||BLA(t,u)||) \tag{5.5}$$

$$w_{i,t} = 1 + \log_{10}(10^{\alpha^I} \cdot |||Y_{t,i}|||) \tag{5.6}$$

其中，常数 α^U 和 α^I 是控制权重的增长速率。当 $\|BLA(t,u)\|$ 越大时，$w^s_{u,t}$ 大，且 $\|BLA(t,u)\| = 0$ 时，$w^s_{u,t} = 1$，也就是说用户 u 在时间 s 前未打过标签 t 时，用户 u 对标签 t 都为 1。类似地，$w_{i,t}$ 也是如此。另外目标函数 Eq. 5.4中 $\sum_f \hat{u}^T_{u,f} \cdot \hat{t}^U_{t,f}$（或者 $\sum_f \hat{i}^T_{i,f} \cdot \hat{t}^I_{t,f}$）继承了 PITF 的方法，能够较好地刻画用户，资源，标签三者的潜在向量特征，能够发现用户喜好（或者资源相关）但未发现的标签，提高推荐的新颖性。

算法 4 TFWPITF 优化算法

输入: 用户对资源打标签历史记录 R

输出: 用户潜在特征向量 $\hat{U} \in \mathbb{R}^{|U|\times K}$，资源潜在特征向量 $\hat{I} \in \mathbb{R}^{|I|\times K}$，以及对应的标签潜在特征向量 $\hat{T}^U \in \mathbb{R}^{|T|\times K}, \hat{T}^I \in \mathbb{R}^{|U|\times K}$;

1: 统计时间，频次，计算 R 中每条历史记录 $r=(u,i,t,s)$ 的用户-标签-时间权重 $w^s_{u,t}$ 以及资源-标签权重 $w_{i,t}$;

2: 使用高斯分布 $N(\mu,\sigma^2)$ 初始化 $\hat{U},\hat{I},\hat{T}^U,\hat{T}^I$;

3: **repeat**

4: 从训练集 R 中均匀采样 $r=(u,i,t_A,s_A)$ 和对应的负样本标签 t_B;

5: 根据 (u,t,t_B,s_A) 计算负采样本的用户-标签权重 $w^{s_A}_{u,t_B}$;

6: $\hat{y}_{u,i,t_A,t_B} \leftarrow \hat{y}_{u,i,t_A} - \hat{y}_{u,i,t_B}$

7: $\delta \leftarrow (1-\sigma(\hat{y}_{u,i,t_a,t_B}))$

8: **for** k from 1 to K **do**

9: $\hat{u}_{u,k} \leftarrow \hat{u}_{u,k} + \alpha\cdot(\delta\cdot(\hat{w}^{s_A}_{u,t_A}\cdot\hat{t}^U_{t_A,k} - \hat{w}^{s_A}_{u,t_B}\cdot\hat{t}^I_{t_B,k}) - \lambda\cdot\hat{u}_{u,k})$

10: $\hat{i}_{i,k} \leftarrow \hat{i}_{i,k} + \alpha\cdot(\delta\cdot(\hat{w}^{s_A}_{u,t_A}\cdot\hat{t}^U_{t_A,k} - \hat{w}^{s_A}_{u,t_B}\cdot\hat{t}^I_{t_B,k}) - \lambda\cdot\hat{i}_{i,k})$

11: $\hat{t}^U_{t_A,k} \leftarrow \hat{t}^U_{t_A,k} + \alpha\cdot(\delta\cdot\hat{u}_{u,k}\cdot w^{s_A}_{u,t_A} - \lambda\cdot\hat{t}^U_{t_A,k})$

12: $\hat{t}^U_{t_B,k} \leftarrow \hat{t}^U_{t_B,k} + \alpha\cdot(\delta\cdot(-\hat{u}_{u,k})\cdot w^{s_A}_{u,t_B} - \lambda\cdot\hat{t}^U_{t_B,k})$

13: $\hat{t}^I_{t_A,k} \leftarrow \hat{t}^I_{t_A,k} + \alpha\cdot(\delta\cdot\hat{i}_{i,k}\cdot w_{i,t_A} - \lambda\cdot\hat{t}^I_{t_A,k})$

14: $\hat{t}^I_{t_B,k} \leftarrow \hat{t}^I_{t_B,k} + \alpha\cdot(\delta\cdot(-\hat{i}_{i,k})\cdot w_{i,t_B} - \lambda\cdot\hat{t}^I_{t_B,k})$

15: **until** 收敛

16: **return** $\hat{U},\hat{I},\hat{T}^U,\hat{T}^I$

5.4.2 优化算法

TFWPITF 使用 BPR 框架进行最大化成对排序目标函数，本文依照论文 **BPR** 采用负采样随机梯度下降算法进行迭代优化，具体算法如**算法 1** 所示。首先，在循环迭代之前，根据训练集大标签的时间和频次统计，计算每条历史记录 $r=(u,i,t,s)$ 中的用户-标签-时间 (u,t,s) 权重 $w^s_{u,t}$ 以及所有物品-标签对 (i,t) 的权重 $w_{i,t}$(第 1 行)。然后使用高斯分布分别初始化潜在用户 $\hat{U}$，资源 $\hat{I}$，标签 $\hat{T}^U,\hat{T}^I$ 隐藏特征向量（第 2 行）。每次迭代中，需要证样本和相应的负样本标签采样。因为正样本用户-标签-时间 (u,t,s) 权重 $w^s_{u,t}$ 和所有物品-标签对 (i,t) 的权重 $w_{i,t}$ 预先计算，所以只需计算负样本标签的用户-标签-时间 (u,t_B,s_A) 权重 $w^{s_A}_{u,t_B}$。最后根据随机梯度下

降公式进行迭代优化潜在向量（第 3-15 行）。

5.4.3 时间复杂度分析

因为每条历史记录 $r=(u,i,t,s)$ 中的用户-标签-时间 (u,t,s) 权重 $w^{s}_{u,t}$ 以及所有资源-标签对 (i,t) 的权重 $w_{i,t}$ 在迭代优化之前已经预先计算好，所以每次迭代只要计算负样本标签的用户-标签-时间 (u,i,t_B,s_A) 权重 $w^{S}_{u,t}$。因此每次迭代，复杂度相对于 PITF（PITF 每次迭代的复杂度为 $O(K)$）来说仅仅多了计算 $w^{s_A}_{u,t_B}$ 的时间，复杂度变为 $O(K+V)$，其中 V 是跟计算 w_{u,t_B} 有关的值。由公式 E.q. 5.5 和负样本采样方法中可以看出，$w^{s_A}_{u,t_B}$ 的计算跟用户 u，打正样本标签 t_A 的时间 s_A，用户 u 之前打过的标签，用户 u 在资源 i 打过的标签集合以及负样本标签 t_B 这几个方面有关。所以每次计算负样本用户-标签-时间 (u,i,t_B,t_A) 权重分两种情况：1）负样本标签 t_B 不在用户 u 打过的标签集合中 T_u 出现过，那么不需要计算，权重 $w^{s_A}_{u,t_B}=1$；2）如果负样本标签 t_B 在用户 u 打过的标签集合中 T_u 出现过，那么复杂度为 $N_{u,<s_A}$（$N_{u,<s_A}$ 表示用户在时间 s_A 之前打过标签的数量）。这里假设每个历史记录循环迭代一遍，那么每条记录迭代计算 $w^{s_A}_{u,t_B}$ 的复杂度期望跟标签数据的分布有关，为 $\frac{1}{|R|}\sum_{r=(u,i,t)}\frac{|T_u|-|T_{u,i}|}{T}$，其中 T_u 表示用户 u 打过的标签集合，$T_{u,i}$ 表示用户 u 在资源 i 上打过的标签集合。可以看出，如果数据集比较稠密的情况下，TFWPITF 每次迭代的运行时间会比 PITF 高出很多。考虑到真实世界网络海量数据的稀疏性，TFWPITF 每次迭代的运行时间对比于 PITF 的运行时间增加的时间是有限的，在可接受的范围之内。

5.5 实验及分析

在这节中，我们将详细描述我们实验中用到的数据集、实验设置、度量指标、评估方法以及实验结果。

5.5.1 实验设置

5.5.1.1 数据集

我们采用了三种网上可免费获取的分众分类数据集: Movielens、LastFM、Delicious[2]。三个数据集来自不同的领域，其中 Movielens 是一个电影推荐系统，LastFM 是一个网络电台和音乐社区，Delicious 则是一个书签网站。三个数据集的统计信息如表5.2所示, 其中 $|U|$ 表示用户数量，$|R|$ 表示资源数量，$|T|$ 表示标签数量，$|P|$ 表示用户-资源对，$|P|/|R|$ 体现数据集的稀疏程度。core 定义了对数据集的过滤程度。p-core 表示数据集中的用户、资源以及标签在所有用户-资源对中都至少出现 p 次。我们这里选取了 no core(完整数据集) 和 core 3 两种情况。

表 5.2: 数据集统计信息

DataSet	core	$\|U\|$	$\|R\|$	$\|T\|$	$\|P\|$	$\|P\|/\|R\|$
Movielens	-	2,113	5,908	9,079	27,712	4.69
	3	656	2,376	2,061	18,427	7.76
LastFM	-	1,892	12,523	9,749	71,064	5.68
	3	1,277	5,940	2,761	59,692	10.05
Delicious	-	1,867	69,223	40,897	104,799	1.51
	3	1,458	5,074	4,233	20,543	4.05

5.5.1.2 参数设置

为了评价我们的标签推荐方法，我们采用“留一法”对数据集切分为训练数据集和测试数据集。当用户打过标签的资源数唯一的时候，则默认将这个用户-资源对放入训练集中。

在参数设置上，BLL+MP$_i$ 与 BLL$_{ac}$+MP$_i$ 与 [11] 设置一样，权重因子 $\beta = 0.5, d = 0.5$; PITF 与 [10] 中设置一样，隐含维度 $K = 64$，正则化因子 $\lambda = 0.00005$，学习速率为 0.05。为了保证公平，TFWPITF 中 d=0.5, 隐含维度和正则化因子与 PITF 设置一样。PITF 和 TFWPITF 迭代轮数均为 100 轮。

[2]http://files.grouplens.org/datasets/hetrec2011

5.5.1.3 度量标准

为了模拟真实世界标签推荐的环境，我们从两个维度去评估算法的性能:

- **准确度**：准确度是推荐系统离线评测指标中最重要的一个。常见的指标有精准率 (Precision)、召回率 (Recall) 以及 F1 值，公式如下:

$$Prec(S_{test}, N) = \text{arg}_{(u,i)\in P_{S_{test}}} \frac{|Top(u,i,N) \bigcap \{t|(u,i,t) \in S_{test}\}|}{N} \quad (5.7)$$

$$Rec(S_{test}, N) = \text{arg}_{(u,i)\in P_{S_{test}}} \frac{|Top(u,i,N) \bigcap \{t|(u,i,t) \in S_{test}\}|}{|\{t|(u,i,t) \in S_{test}\}|} \quad (5.8)$$

$$F1(S_{test}, N) = \frac{2 \cdot Prec(S_{test}, N) \cdot Rec(S_{test}, N)}{Prec(S_{test}, N) + Rec(S_{test}, N)} \quad (5.9)$$

  如果没有特别说明，本文使用 F1@5 衡量准确度。

- **新颖性**： 我们使用 [129] 中的 AIP@10 来定义推荐的标签列表的新颖性。如果推荐的标签从没标注过目标资源，则认为这个推荐是新颖的，所以对于目标资源，如果推荐的标签的流行程度越低，则新颖性越高。

5.5.1.4 对比方法

- **MP**$_i$: 方法对给定目标资源会推荐该资源下最受欢迎的标签。

- **PITF**: 由 Rendle 和 Schmidt-Thieme 提出的一种改进的张量分解模型，显示地为用户、资源以及标签之间的两两相互作用建模。

- **BLL+MP**$_i$: 将 BLL 与 MP$_i$ 结合，BLL 部分利用用户以往打标签行为与当前时间的间隔、标签的使用频次来估计用户将来重复使用某一标签的概率。

- **BLL**$_{ac}$**+MP**$_i$: 与 **BLL+MP**$_i$ 类似，但是在 BLL 上加入关联模块，描述目标资源的特征对用户的影响。根据 [130] 中标签推荐标准测试，该方法是当前准确度最好的标签推荐方法。

5.5.2 实验结果及分析

我们首先研究权重因子 α 对 TFWPITF 准确度的影响, 然后详细的对比 TFW-PITF 与其他标签推荐方法的性能差异，最后从迭代的收敛速度以及算法运行时间上对比 TFWPITF 与 PITF 的差异。

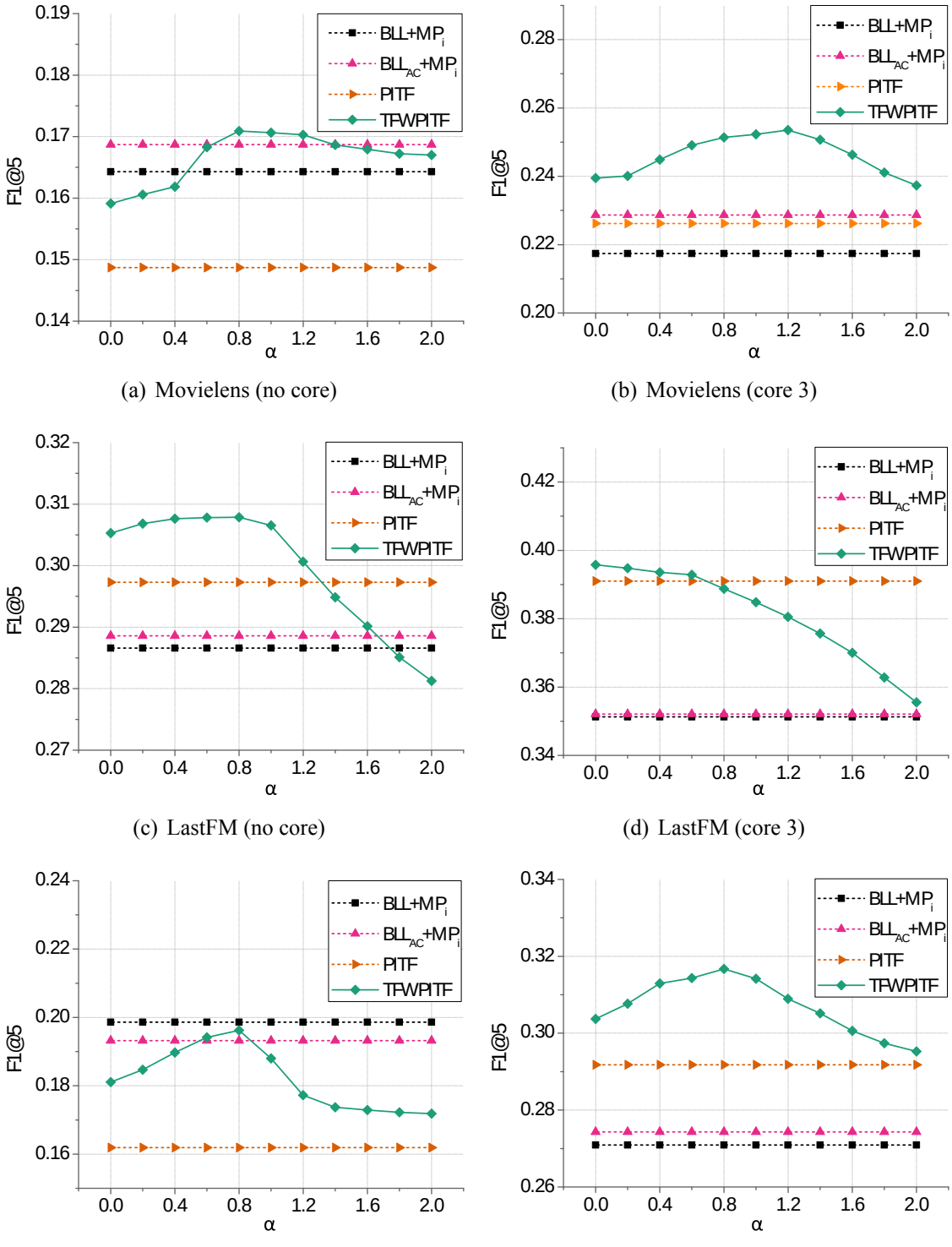
5.5.2.1 权重因子对 TFWPITF 推荐结果的影响

在这里，我们设置用户-标签-时间与资源-标签的权重因子相同，通过改变 α 来观察 TFWPITF 的性能变化 (F1@5)。从图5.1可以看出来，对所有的数据集，随着 α 值的增大，TFWPITF 的性能不断提高，直到达到最优值，继续增大 α，TFWPITF 的性能会逐渐下降 (对数据集 LastFM core 3 的情况，如果将 α 值调成负数，那么 TFWPITF 性能会明显下降，所以其最优值就是在 $\alpha = 0$ 时取得)。当权重值 α 调到最优时，可看到我们的方法在所有数据集上基本都是优于 PITF 和 BLL+MP$_i$(最差也能与这两种方法持平)，这说明通过对 PITF 的用户-标签-时间与资源-标签上增加合理的权重是可以有效改善推荐性能。并且，在对数据集过滤之后 (core 3)，可以看到 TFWPITF 的性能明显优于其他对比方法。这有可能因为 PITF 对较稠密数据有着更好的推荐效果，所以明显优于 BLL+MP$_i$，而我们的 TFWPITF 方法在 PITF 的基础上考虑了时间对用户行为的影响，性能得到进一步的提升。

在接下来的实验中，我们都是选择 TFWPITF 最好性能时的 α 值，对数据集 Movielens core 3 时取 $\alpha = 1.2$，对数据集 LastFM core 3 时取 $\alpha = 0$，其他数据集均取 $\alpha = 0.8$。

5.5.2.2 与其他标签推荐方法对比分析

这一节，我们将我们的 TFWPITF 方法与当前最新的一些标签推荐方法进行对比。图5.2显示在数据集 Movielens、LastFM 和 Delicious 上，TFWPITF 与其他标签推荐方法在 TOP 1 到 TOP 10 上的召回率/精准率曲线。从总体上来说，对于所有数据集，TFWPITF 基本上都是性能最优的，这进一步说明了对 PITF 加入时间对用户



(a) Movielens (no core)

(b) Movielens (core 3)

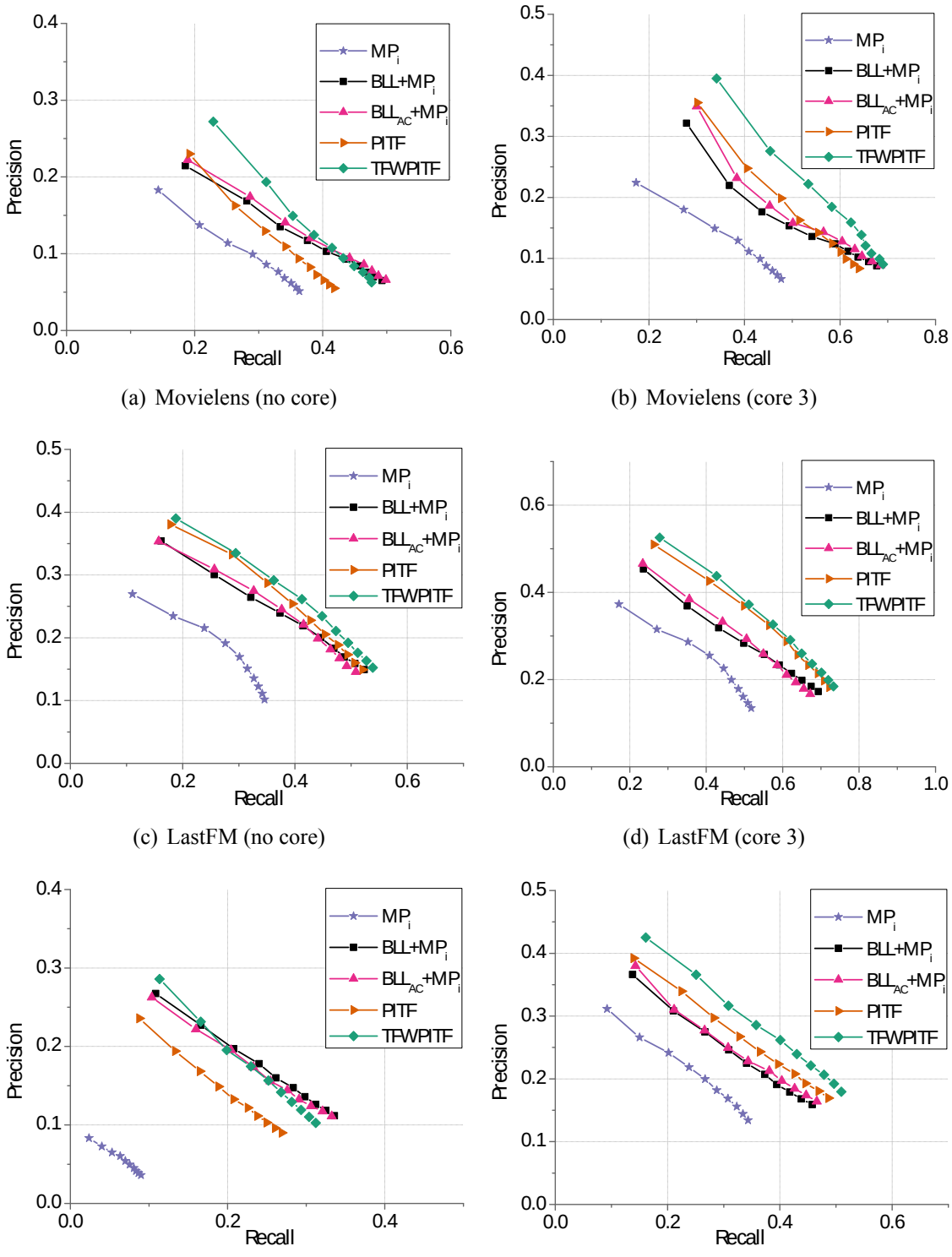
(c) LastFM (no core)

(d) LastFM (core 3)

(e) Delicious (no core)

(f) Delicious (core 3)

图 5.1: 不同 α 对推荐效果影响



(a) Movielens (no core)　(b) Movielens (core 3)

(c) LastFM (no core)　(d) LastFM (core 3)

(e) Delicious (no core)　(f) Delicious (core 3)

图 5.2: 召回率/精准率曲线

行为的影响是十分有效的。在未过滤的数据集 Movielens 和 Delicous 上，尽管当推荐的 TOP N 列表长度较大时，TFWPITF 性能略差于 BLL+MP$_i$ 和 BLL$_{ac}$+MP$_i$，但是在实际应用中，对于一个给定的资源，用户关注的标签往往不会太多，有时甚至只会关心排名最前的那个标签。所以在这种情况下，依然可以认为我们的 TFWPITF 方法是有效的。

在新颖性上，从表5.3可以看出，PITF 在所有数据集上的新颖性都是最好的，而 TFWPITF 的新颖性略次于 PITF，但是优于 MP$_i$，BLL+MP$_i$ 和 BLL$_{ac}$+MP$_i$。这说明我们的 TFWPITF 方法在获得最优准确度的前下，还能得到较好的新颖度。

表 5.3: 不同方法 AIP@10 的对比

DataSet	Core	MP$_i$	BLL+MP$_i$	BLL$_{ac}$+MP$_i$	PITF	TFWPITF
Movielens	-	0.806	0.788	0.786	**0.854**	0.812
	3	0.761	0.762	0.757	**0.803**	0.777
LastFM	-	0.732	0.724	0.709	**0.779**	0.764
	3	0.668	0.660	0.639	**0.697**	**0.697**
Delicious	-	0.881	0.873	0.878	**0.949**	0.912
	3	0.787	0.767	0.767	**0.827**	0.778

5.5.2.3 TFWPITF 对比 PITF

图5.3显示了 TFWPITF 和 PITF 在不同迭代次数下的准确度对比，为了确保算法达到收敛，我们设置迭代轮数为 100 轮。从图中可以明显的看出，我们的 TFW-PITF 方法在数据集 Movielens、LastFM 以及 Delicious no core 和 core 3 上的准确度和收敛速度都要优于 PITF。TFWPITF 在所有数据集上都能在 40 轮迭代前收敛，而 PITF 基本上要达到 60 轮迭代之后才逐渐收敛，而且 TFWPITF 在前 20 轮的收敛速度要远快于 PITF。

而在运行时间上，相比 PITF，TFWPITF 由于在每轮迭代过程中要额外计算用户-标签-时间的权重因子，所以每轮迭代的时间要比 PITF 慢。表 5.4显示了 TFW-PITF 与 PITF 在不同数据集上运行时间对比，我们选取了 100 轮迭代时间的平均值作为最终运行时间 (其中每轮迭代为训练集样本数目的 100 倍)。可以看出 TFW-

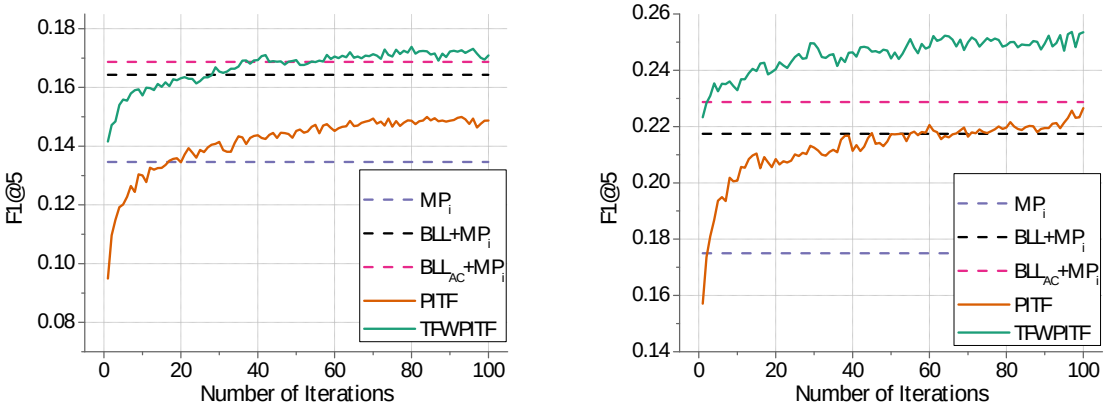
PITF 的在数据集 LastFM 和 Delicious 上的运行时间比 PITF 略长，而在数据集上 Movielens 上时间要比 PITF 长很多。但是如章节5.4.3所示，经过计算统计，因为在数据集 Movielens 上的 V 的期望值要比另外两个数据集大 10 倍以上，也就是说 Movielens 的数据十分稠密，所以 TPWPITF 的运行时间要比 PITF 长很多，而在当前网络时代，数据往往都是非常稀疏的，所以在大多数情况下，我们的算法仍然是十分有效的。

表 5.4: TFWPITF 与 PITF 运行时间对比

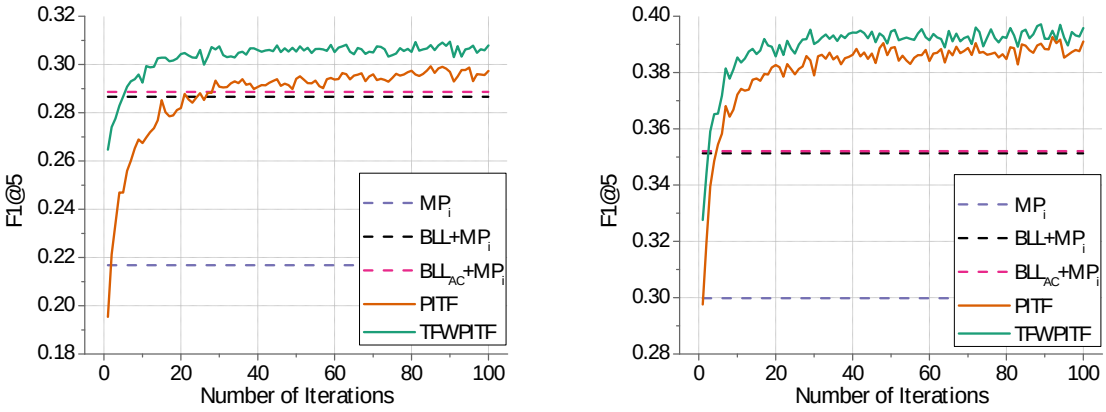
DataSet	Core	PITF	TFWPITF
Movielens	-	10.3	31.3
	3	5.8	23.2
LastFM	-	41.4	51.7
	3	32.6	44.6
Delicious	-	110.7	145.0
	3	16.8	24.0

5.6 本章小结

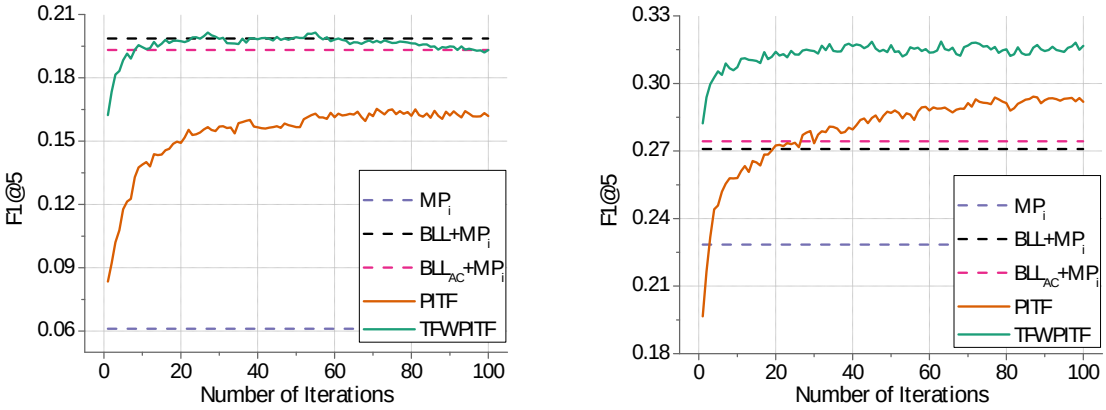
本文综合分析现阶段个性化标签推荐系统的优缺点，提出了基于时间和频次加权张量分解模型 TFWPITF。PITF 模型的基础上增加对用户-标签-时间关系的权重以及资源-标签关系的权重，使得 TFWPITF 模型既能考虑用户打标签行为随时间变化而变化的现象，也能有效地利用相似用户，相似资源，相似标签的信息，从而更好地对用户对资源打标签的行为进行建模，提高标签推荐的准确度和新颖性。且通过不同场景的真实数据实验，表明 TFWPITF 在准确性上优于当前流行的个性化标签推荐算法，同时有较好的推荐新颖性。



(a) Movielens (no core)　　(b) Movielens (core 3)



(c) LastFM (no core)　　(d) LastFM (core 3)



(e) Delicious (no core)　　(f) Delicious (core 3)

图 5.3: TFWPITF 和 PITF 准确率与收敛速度对比

第六章　总结与展望

本文重点研究了基于矩阵分解相关理论的个性化推荐系统。本章对全文研究内容和技术模型进行总结归纳，并展望未来的重点研究工作。

6.1　研究总结

互联网发展以来，大量用户和企业产生了海量数据，面临着信息过载的问题。用户需要在大量无用信息中寻找少量自身感兴趣的信息，而企业也需要寻找与企业产品相关的目标用户，进行产品精准营销和推广。针对上述问题，个性化推荐系统应用而生，能够根据用户访问商品的历史行为数据建模，刻画用户和商品之间的交互关系，将可能感兴趣的商品推荐给目标用户。在使用互联网服务时，用户网上寻找目标商品，遇到满意商品进行消费，然后进行显式评分，并利用标签描述商品的特点，且使用文本评价自己使用商品的感受。本文重点研究了上述的推荐场景，意在解决三个相关问题：（1）*商品推荐*；（2）*评分预测*；（3）*标签推荐*。*商品推荐*。*商品推荐*主要是根据用户访问商品的历史行为，根据用户的个人喜好，推荐用户感兴趣的商品。*评分预测*则是根据用户的历史评分行为，预测对特定商品的满意程度。*标签推荐*则是利用用户的历史使用标签的信息和特定商品的属性信息，个性化推荐给用户标签来描述商品，方便用户输入。*商品推荐*是推荐系统的主要目标，而好的*评分预测*系统和*标签推荐*系统则是提高用户体验，并且能够促进推荐系统的良性循环。具体地，本文的研究问题和技术贡献总结如下。

首先，针对*商品推荐*，本文在隐式反馈数据提出一种加权局部矩阵分解模型（LWMF）。传统加权矩阵分解模型是全局低秩的假设，不能对隐式反馈数据的局部信息建模。而本文模型设计了高效的子矩阵选择算法建模数据的局部信息，并改进了交替最小二乘算法进行子矩阵加权分解，刻画用户和商品的局部内在特征。并且，LWMF 带来两个额外好处，缓解了数据稀疏性问题和更好地进行分布式矩

阵分解。基于公开的真实数据集的实验验证了 LWMF 算法的有效性。

其次，针对评分预测，本文强调了局部矩阵分解模型在显式评分数据上的局部可解释性和目标一致性。近期的工作不能对数据的局部信息进行直观解释，并且分为两个步骤进行局部矩阵分解。基于此，本文结合主题模型和概率矩阵分解，称为多主题矩阵分解模型（PMTMF）在同一个目标函数里分别对局部信息和用户、商品局部特征建模。此外，本文还是用狄利克雷分布和高斯-韦斯特分布作为 PMTMF 参数的先验分布，将其扩展为全贝叶斯版本（BPMTMF），使得模型对参数设置不敏感，并且能获得更加准确的评分预测。基于公开的真实数据集的实验验证了 BPMTMF 在显式评分数据上进行的评分预测有效性，并对数据局部信息做出一定的可解释性。

最后，针对标签推荐，本文强调了标签的时间和频率信息对标签推荐的帮助。现有的基于协同过滤模型的工作没有考虑用户使用标签的时间和频率因素（TFW-PITF），并且协同过滤模型对新用户冷启动的标签推荐不友好。针对上述问题，本文提出了时间与频率感知的的张量分解模型，建立标签关于时间和频率的模型，并和逐对排序张量分解模型结合，提高模型的推荐新颖性。基于公开数据集实验表明 TFWPITF 在准确性优于当前流行的个性化标签推荐算法，同时具有可接受的推荐新颖性。

6.2 研究展望

本文重点研究了在用户的隐式反馈数据，显式评分数据和显式标签数据上的三个推荐系统相关问题。作者还认为可以从下面三个方面进行扩展。

首先，本文针对隐式反馈数据和显式评分数据进行了局部矩阵分解，需要对局部信息建模，增加了模型的复杂度，尽管本文提出了一些改进的优化算法，但局部矩阵分解模型时间高于之前的单个全局模型。基于此，未来需要研究更高效的优化算法，加快模型参数学习效率。

其次，本文使用了较为规整的用户访问次数数据，评分数据以及标签数据。推

荐数据还包括许多其他数据，类似用户的评论文本数据，描述商品的图片数据以及领域数据（例如签到数据中地点位置信息，音乐中的音频数据等）。这些数据对提高个性化推荐系统的准确率都有极大的帮助。因此，如何融合多源数据也是未来研究的重点方向。

最后，本文提出的模型都是基于矩阵分解或者张量分解模型，都属于线性模型。但用户访问、评分、描述商品的行为是非常复杂的，受很多因素影响。因此，使用非线性模型，例如深度学习，来捕捉这类复杂关系，更好地刻画用户画像和商品特性。

参考文献

[1] HU Y, KOREN Y, VOLINSKY C. Collaborative filtering for implicit feedback datasets[C] // 2008 Eighth IEEE International Conference on Data Mining. 2008 : 263 – 272.

[2] BENNETT J, LANNING S, OTHERS. The netflix prize[C] // Proceedings of KDD cup and workshop : Vol 2007. 2007 : 35.

[3] SALAKHUTDINOV R, MNIH A. Probabilistic Matrix Factorization.[C] // Nips : Vol 1. 2007 : 2 – 1.

[4] LEE J, KIM S, LEBANON G, et al. Local Low-Rank Matrix Approximation.[J]. ICML (2), 2013, 28 : 82 – 90.

[5] LEE J, BENGIO S, KIM S, et al. Local collaborative ranking[C] // Proceedings of the 23rd international conference on World wide web. 2014 : 85 – 96.

[6] CHEN C, LI D, ZHAO Y, et al. WEMAREC: Accurate and Scalable Recommendation through Weighted and Ensemble Matrix Approximation[C] // Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2015 : 303 – 312.

[7] PATEREK A. Improving regularized singular value decomposition for collaborative filtering[C] // Proceedings of KDD cup and workshop : Vol 2007. 2007 : 5 – 8.

[8] KOREN Y, BELL R, VOLINSKY C, et al. Matrix factorization techniques for recommender systems[J]. Computer, 2009, 42(8) : 30 – 37.

[9] KRESTEL R, FANKHAUSER P, NEJDL W. Latent dirichlet allocation for tag recommendation[C] // Proceedings of the third ACM conference on Recommender systems. 2009 : 61 – 68.

[10] RENDLE S, SCHMIDT-THIEME L. Pairwise interaction tensor factorization for personalized tag recommendation[C] // Proceedings of the third ACM international conference on Web search and data mining. 2010 : 81 – 90.

[11] KOWALD D, SEITLINGER P, TRATTNER C, et al. Long time no see: The probability of reusing tags as a function of frequency and recency[C] // Proceedings of the 23rd International Conference on World Wide Web. 2014 : 463 – 468.

[12] KOWALD D, KOPEINIK S, SEITLINGER P, et al. Refining frequency-based tag reuse predictions by means of time and semantic context[G] // Mining, Modeling, and Recommending'Things' in Social Media. [S.l.] : Springer, 2015 : 55 – 74.

[13] PAN R, ZHOU Y, CAO B, et al. One-class collaborative filtering[C] // Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. 2008 : 502 – 511.

[14] PAN R, SCHOLZ M. Mind the gaps: weighting the unknown in large-scale one-class collaborative filtering[C] // Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. 2009 : 667 – 676.

[15] SHARDANAND U, MAES P. Social information filtering: algorithms for automating "word of mouth"[C] // Proceedings of the SIGCHI conference on Human factors in computing systems. 1995 : 210 – 217.

[16] KONSTAN J A, MILLER B N, MALTZ D, et al. GroupLens: applying collaborative filtering to Usenet news[J]. Communications of the ACM, 1997, 40(3) : 77 – 87.

[17] SARWAR B, KARYPIS G, KONSTAN J, et al. Analysis of recommendation algorithms for e-commerce[C] // Proceedings of the 2nd ACM conference on Electronic commerce. 2000 : 158 – 167.

[18] SARWAR B, KARYPIS G, KONSTAN J, et al. Item-based collaborative filtering recommendation algorithms[C] // Proceedings of the 10th international conference on World Wide Web. 2001 : 285 – 295.

[19] DESHPANDE M, KARYPIS G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1) : 143 – 177.

[20] NING X, KARYPIS G. Slim: Sparse linear methods for top-n recommender systems[C] // Data Mining (ICDM), 2011 IEEE 11th International Conference on. 2011 : 497 – 506.

[21] CHENG Y, YIN L, YU Y. LorSLIM: low rank sparse linear methods for top-n recommendations[C] // Data Mining (ICDM), 2014 IEEE International Conference on. 2014 : 90 – 99.

[22] CHRISTAKOPOULOU E, KARYPIS G. Local item-item models for top-n recommendation[C] // Proceedings of the 10th ACM Conference on Recommender Systems. 2016 : 67 – 74.

[23] RESNICK P, IACOVOU N, SUCHAK M, et al. GroupLens: an open architecture for collaborative filtering of netnews[C] // Proceedings of the 1994 ACM conference on Computer supported cooperative work. 1994 : 175 – 186.

[24] LINDEN G, SMITH B, YORK J. Amazon. com recommendations: Item-to-item collaborative filtering[J]. IEEE Internet computing, 2003, 7(1) : 76 – 80.

[25] CHOWDHURY G. Introduction to modern information retrieval[M]. [S.l.] : Facet publishing, 2010.

[26] TAN P-N, OTHERS. Introduction to data mining[M]. [S.l.] : Pearson Education India, 2006.

[27] KARYPIS G. Evaluation of item-based top-n recommendation algorithms[C] // Proceedings of the tenth international conference on Information and knowledge management. 2001 : 247 – 254.

[28] NAKAMURA A, ABE N. Collaborative Filtering Using Weighted Majority Prediction Algorithms.[C] // ICML : Vol 98. 1998 : 395 – 403.

[29] 项亮. 推荐系统实践 [J]. 北京: 人民邮电出版社, 2012.

[30] FRIEDMAN J, HASTIE T, TIBSHIRANI R. Regularization paths for generalized linear models via coordinate descent[J]. Journal of statistical software, 2010, 33(1) : 1.

[31] FAZEL M. Matrix rank minimization with applications[D]. [S.l.] : PhD thesis, Stanford University, 2002.

[32] CANDÈS E J, TAO T. The power of convex relaxation: Near-optimal matrix completion[J]. IEEE Transactions on Information Theory, 2010, 56(5) : 2053 – 2080.

[33] BOYD S, PARIKH N, CHU E, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers[J]. Foundations and Trends® in Machine Learning, 2011, 3(1) : 1 – 122.

[34] LESKOVEC J, RAJARAMAN A, ULLMAN J D. Mining of massive datasets[M]. [S.l.] : Cambridge University Press, 2014.

[35] KOREN Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C] // Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. 2008 : 426 – 434.

[36] MA H. An experimental study on implicit social recommendation[C] // Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval. 2013 : 73 – 82.

[37] MA H, YANG H, LYU M R, et al. Sorec: social recommendation using probabilistic matrix factorization[C] // Proceedings of the 17th ACM conference on Information and knowledge management. 2008 : 931 – 940.

[38] MA H, KING I, LYU M R. Learning to recommend with social trust ensemble[C] // Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval. 2009 : 203 – 210.

[39] JAMALI M, ESTER M. A matrix factorization technique with trust propagation for recommendation in social networks[C] // Proceedings of the fourth ACM conference on Recommender systems. 2010 : 135 – 142.

[40] MA H, ZHOU D, LIU C, et al. Recommender systems with social regularization[C] // Proceedings of the fourth ACM international conference on Web search and data mining. 2011 : 287 – 296.

[41] GUO G, ZHANG J, YORKE-SMITH N. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings.[C] // Aaai. 2015 : 123 – 129.

[42] YANG B, LEI Y, LIU J, et al. Social collaborative filtering by trust[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016.

[43] KOREN Y. Collaborative filtering with temporal dynamics[J]. Communications of the ACM, 2010, 53(4) : 89 – 97.

[44] LEE D D, SEUNG H S. Algorithms for non-negative matrix factorization[C] // Advances in neural information processing systems. 2001 : 556 – 562.

[45] SCHMIDT M N, WINTHER O, HANSEN L K. Bayesian non-negative matrix factorization[C] // International Conference on Independent Component Analysis and Signal Separation. 2009 : 540 – 547.

[46] SALAKHUTDINOV R, MNIH A. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo[C] // Proceedings of the 25th international conference on Machine learning. 2008 : 880 – 887.

[47] RENDLE S. Factorization machines[C] // Data Mining (ICDM), 2010 IEEE 10th International Conference on. 2010 : 995 – 1000.

[48] GEMULLA R, NIJKAMP E, HAAS P J, et al. Large-scale matrix factorization with distributed stochastic gradient descent[C] // Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. 2011 : 69 – 77.

[49] LIU C, YANG H-C, FAN J, et al. Distributed nonnegative matrix factorization for web-scale dyadic data analysis on mapreduce[C] // Proceedings of the 19th international conference on World wide web. 2010 : 681 – 690.

[50] ZHANG Y, ZHANG M, LIU Y, et al. Localized matrix factorization for recommendation based on matrix block diagonal forms[C] // Proceedings of the 22nd international conference on World Wide Web. 2013 : 1511 – 1520.

[51] MACKEY L W, JORDAN M I, TALWALKAR A. Divide-and-conquer matrix factorization[C] // Advances in Neural Information Processing Systems. 2011 : 1134 – 1142.

[52] DHILLON I S, MALLELA S, MODHA D S. Information-theoretic co-clustering[C] // Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. 2003 : 89 – 98.

[53] CHEN C, LI D, LV Q, et al. MPMA: mixture probabilistic matrix approximation for collaborative filtering[C] // Proceedings of the 25th International Joint Conference on Artificial Intelligence. 2016 : 1382 – 1388.

[54] CHEN C, LI D, LV Q, et al. GLOMA: Embedding Global Information in Local Matrix Approximation Models for Collaborative Filtering[J], 2017.

[55] LIU T-Y, OTHERS. Learning to rank for information retrieval[J]. Foundations and Trends® in Information Retrieval, 2009, 3(3) : 225 – 331.

[56] BELL R M, KOREN Y. Scalable collaborative filtering with jointly derived neighborhood interpolation weights[C] // Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on. 2007 : 43 – 52.

[57] PILÁSZY I, ZIBRICZKY D, TIKK D. Fast als-based matrix factorization for explicit and implicit feedback datasets[C] // Proceedings of the fourth ACM conference on Recommender systems. 2010 : 71 – 78.

[58] DEVOOGHT R, KOURTELLIS N, MANTRACH A. Dynamic matrix factorization with priors on unknown values[C] // Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2015 : 189 – 198.

[59] HE X, ZHANG H, KAN M-Y, et al. Fast matrix factorization for online recommendation with implicit feedback[C] // Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. 2016 : 549 – 558.

[60] CHENG Z K C P Q. Top-N Recommender System via Matrix Completion[J], 2016.

[61] KABBUR S, NING X, KARYPIS G. FISM: Factored Item Similarity Models for Top-N Recommender Systems[J], 2013.

[62] GE M, DELGADO-BATTENFELD C, JANNACH D. Beyond accuracy: evaluating recommender systems by coverage and serendipity[C] // Proceedings of the fourth ACM conference on Recommender systems. 2010 : 257 – 260.

[63] ZHOU T, KUSCSIK Z, LIU J-G, et al. Solving the apparent diversity-accuracy dilemma of recommender systems[J]. Proceedings of the National Academy of Sciences, 2010, 107(10) : 4511 – 4515.

[64] QIN L, ZHU X. Promoting Diversity in Recommendation by Entropy Regularizer.[C] // IJCAI. 2013.

[65] SHA C, WU X, NIU J. A Framework for Recommending Relevant and Diverse Items[J], .

[66] RENDLE S, FREUDENTHALER C, GANTNER Z, et al. BPR: Bayesian personalized ranking from implicit feedback[C] // Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence. 2009 : 452 – 461.

[67] GANTNER Z, DRUMOND L, FREUDENTHALER C, et al. Personalized Ranking for Non-Uniformly Sampled Items.[C] // KDD Cup. 2012 : 231 – 247.

[68] RENDLE S, FREUDENTHALER C. Improving pairwise learning for item recommendation from implicit feedback[C] // Proceedings of the 7th ACM international conference on Web search and data mining. 2014 : 273 – 282.

[69] PAN W, CHEN L. GBPR: Group Preference Based Bayesian Personalized Ranking for One-Class Collaborative Filtering.[C] // IJCAI : Vol 13. 2013 : 2691 – 2697.

[70] RENDLE S, BALBY MARINHO L, NANOPOULOS A, et al. Learning optimal ranking with tensor factorization for tag recommendation[C] // Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. 2009 : 727 – 736.

[71] JAHRER M, TÖSCHER A. Collaborative Filtering Ensemble.[C] // KDD Cup. 2012 : 61 – 74.

[72] TAKÁCS G, TIKK D. Alternating least squares for personalized ranking[C] // Proceedings of the sixth ACM conference on Recommender systems. 2012 : 83 – 90.

[73] HANLEY J A, MCNEIL B J. The meaning and use of the area under a receiver operating characteristic (ROC) curve.[J]. Radiology, 1982, 143(1) : 29 – 36.

[74] SHI Y, KARATZOGLOU A, BALTRUNAS L, et al. CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering[C] // Proceedings of the sixth ACM conference on Recommender systems. 2012 : 139 – 146.

[75] VOORHEES E M, OTHERS. The TREC-8 Question Answering Track Report.[C] // Trec : Vol 99. 1999 : 77 – 82.

[76] SHI Y, LARSON M, HANJALIC A. List-wise learning to rank with matrix factorization for collaborative filtering[C] // Proceedings of the fourth ACM conference on Recommender systems. 2010 : 269 – 272.

[77] CAO Z, QIN T, LIU T-Y, et al. Learning to rank: from pairwise approach to listwise approach[C] // Proceedings of the 24th international conference on Machine learning. 2007 : 129 – 136.

[78] WONG T-T. Generalized Dirichlet distribution in Bayesian analysis[J]. Applied Mathematics and Computation, 1998, 97(2-3) : 165 – 181.

[79] HOFMANN T, PUZICHA J. Latent class models for collaborative filtering[C] // IJCAI : Vol 99. 1999.

[80] HOFMANN T. Probabilistic latent semantic indexing[C] // Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. 1999 : 50 – 57.

[81] HOFMANN T. Latent semantic models for collaborative filtering[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1) : 89 – 115.

[82] BLEI D M, NG A Y, JORDAN M I. Latent dirichlet allocation[J]. the Journal of machine Learning research, 2003, 3 : 993 – 1022.

[83] BARBIERI N, MANCO G, RITACCO E. Probabilistic approaches to recommendations[J]. Synthesis Lectures on Data Mining and Knowledge Discovery, 2014, 5(2) : 1 – 197.

[84] JIN R, SI L, ZHAI C. A study of mixture models for collaborative filtering[J]. Information Retrieval, 2006, 9(3) : 357 – 382.

[85] BEUTEL A, AHMED A, SMOLA A J. Accams: Additive co-clustering to approximate matrices succinctly[C] // Proceedings of the 24th International Conference on World Wide Web. 2015 : 119 – 129.

[86] HOFMANN T. Learning what people (don't) want[C] // European Conference on Machine Learning. 2001 : 214 – 225.

[87] HOFMANN T. Collaborative filtering via gaussian probabilistic latent semantic analysis[C] // Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. 2003 : 259 – 266.

[88] MARLIN B M. Modeling User Rating Profiles For Collaborative Filtering.[C] // NIPS. 2003 : 627 – 634.

[89] GOVAERT G, NADIF M. Clustering with block mixture models[J]. Pattern Recognition, 2003, 36(2) : 463 – 473.

[90] GOVAERT G, NADIF M. An EM algorithm for the block mixture model[J]. IEEE Transactions on Pattern Analysis and machine intelligence, 2005, 27(4) : 643 – 647.

[91] PORTEOUS I, BART E, WELLING M. Multi-HDP: A Non Parametric Bayesian Model for Tensor Factorization.[C] // Aaai : Vol 8. 2008 : 1487 – 1490.

[92] SHAN H, BANERJEE A. Bayesian co-clustering[C] // Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. 2008 : 530 – 539.

[93] SHAN H, BANERJEE A. Residual bayesian co-clustering for matrix approximation[C] // Proceedings of the 2010 SIAM International Conference on Data Mining. 2010 : 223 – 234.

[94] BARBIERI N, MANCO G, ORTALE R, et al. Balancing prediction and recommendation accuracy: hierarchical latent factors for preference data[C] // Proceedings of the 2012 SIAM International Conference on Data Mining. 2012 : 1035 – 1046.

[95] BARBIERI N, MANCO G, RITACCO E. A probabilistic hierarchical approach for pattern discovery in collaborative filtering data[C] // Proceedings of the 2011 SIAM International Conference on Data Mining. 2011 : 630 – 641.

[96] BARBIERI N, COSTA G, MANCO G, et al. Modeling item selection and relevance for accurate recommendations: a bayesian approach[C] // Proceedings of the fifth ACM conference on Recommender systems. 2011 : 21 – 28.

[97] DUECK D, FREY B, DUECK D, et al. Probabilistic sparse matrix factorization[J]. University of Toronto technical report PSI-2004-23, 2004.

[98] TIPPING M E, BISHOP C M. Probabilistic principal component analysis[J]. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 1999, 61(3): 611–622.

[99] ILIEVSKI I, ROY S. Personalized news recommendation based on implicit feedback[C] // Proceedings of the 2013 International News Recommender Systems Workshop and Challenge. 2013: 10–15.

[100] ZIBRICZKY D, HIDASI B, PETRES Z, et al. Personalized recommendation of linear content on interactive TV platforms: beating the cold start and noisy implicit user feedback.[C] // UMAP Workshops. 2012.

[101] YANG D, CHEN T, ZHANG W, et al. Local implicit feedback mining for music recommendation[C] // Proceedings of the sixth ACM conference on Recommender systems. 2012: 91–98.

[102] LIAN D, ZHAO C, XIE X, et al. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation[C] // Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014: 831–840.

[103] LI H, GE Y, ZHU H. Point-of-Interest Recommendations: Learning Potential Check-ins from Friends[C] // Proceedings of the 22th ACM SIGKDD international conference on on Knowledge discovery and data mining. ACM. 2016.

[104] GUILLORY A, BILMES J. Interactive submodular set cover[J]. arXiv preprint arXiv:1002.3345, 2010.

[105] JÄRVELIN K, KEKÄLÄINEN J. Cumulated gain-based evaluation of IR techniques[J]. ACM Transactions on Information Systems (TOIS), 2002, 20(4): 422–446.

[106] CLARKE C L, KOLLA M, CORMACK G V, et al. Novelty and diversity in information retrieval evaluation[C] // Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. 2008: 659–666.

[107] CHAPELLE O, METLZER D, ZHANG Y, et al. Expected reciprocal rank for graded relevance[C] // Proceedings of the 18th ACM conference on Information and knowledge management. 2009: 621–630.

[108] NEMHAUSER G L, WOLSEY L A, FISHER M L. An analysis of approximations for maximizing submodular set functions—I[J]. Mathematical Programming, 1978, 14(1) : 265 – 294.

[109] YUAN Q, CONG G, MA Z, et al. Time-aware point-of-interest recommendation[C] // Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval. 2013 : 363 – 372.

[110] MIRBAKHSH N, LING C X. Clustering-based factorized collaborative filtering[C] // Proceedings of the 7th ACM conference on Recommender systems. 2013 : 315 – 318.

[111] HU L, SUN A, LIU Y. Your neighbors affect your ratings: on geographical neighborhood influence to rating prediction[C] // Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. 2014 : 345 – 354.

[112] HEINRICH G. Parameter estimation for text analysis[R]. [S.l.] : Technical report, 2005.

[113] ANDRIEU C, DE FREITAS N, DOUCET A, et al. An introduction to MCMC for machine learning[J]. Machine learning, 2003, 50(1-2) : 5 – 43.

[114] MNIH A, SALAKHUTDINOV R. Probabilistic matrix factorization[C] // Advances in neural information processing systems. 2007 : 1257 – 1264.

[115] GUO G, ZHANG J, SUN Z, et al. LibRec: A Java Library for Recommender Systems.[C] // UMAP Workshops. 2015.

[116] GRIFFITHS T L, STEYVERS M. Finding scientific topics[J]. Proceedings of the National Academy of Sciences, 2004, 101(suppl 1) : 5228 – 5235.

[117] ZHANG L, TANG J, ZHANG M. Integrating temporal usage pattern into personalized tag prediction[C] // Asia-Pacific Web Conference. 2012 : 354 – 365.

[118] ANDERSON J R, BOTHELL D, BYRNE M D, et al. An integrated theory of the mind.[J]. Psychological review, 2004, 111(4) : 1036.

[119] KOWALD D, SEITLINGER P, KOPEINIK S, et al. Forgetting the words but remembering the meaning: Modeling forgetting in a verbal and semantic tag recommender[G] // Mining, Modeling, and Recommending'Things' in Social Media. [S.l.] : Springer, 2015 : 75 – 95.

[120] DELLSCHAFT K, STAAB S. Measuring the influence of tag recommenders on the indexing quality in tagging systems[C] // Proceedings of the 23rd ACM conference on Hypertext and social media. 2012 : 73 – 82.

[121] JÄSCHKE R, MARINHO L, HOTHO A, et al. Tag recommendations in social bookmarking systems[J]. Ai Communications, 2008, 21(4) : 231 – 247.

[122] MARINHO L B, SCHMIDT-THIEME L. Collaborative tag recommendations[G] // Data Analysis, Machine Learning and Applications. [S.l.] : Springer, 2008 : 533 – 540.

[123] HOTHO A, JÄSCHKE R, SCHMITZ C, et al. Information retrieval in folksonomies: Search and ranking[C] // European Semantic Web conference. 2006 : 411 – 426.

[124] JÄSCHKE R, MARINHO L, HOTHO A, et al. Tag recommendations in folksonomies[C] // European Conference on Principles of Data Mining and Knowledge Discovery. 2007 : 506 – 514.

[125] SYMEONIDIS P, NANOPOULOS A, MANOLOPOULOS Y. Tag recommendations based on tensor dimensionality reduction[C] // Proceedings of the 2008 ACM conference on Recommender systems. 2008 : 43 – 50.

[126] FANG X, PAN R, CAO G, et al. Personalized Tag Recommendation through Nonlinear Tensor Factorization Using Gaussian Kernel.[C] // AAAI. 2015 : 439 – 445.

[127] YIN D, HONG L, XUE Z, et al. Temporal dynamics of user interests in tagging systems[C] // Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence. 2011 : 1279 – 1285.

[128] SEITLINGER P, KOWALD D, TRATTNER C, et al. Recommending tags with a model of human categorization[C] // Proceedings of the 22nd ACM international conference on Conference on information & knowledge management. 2013 : 2381 – 2386.

[129] BELÉM F, MARTINS E, ALMEIDA J, et al. Exploiting novelty and diversity in tag recommendation[C] // European Conference on Information Retrieval. 2013 : 380 – 391.

[130] KOWALD D, LEX E. Evaluating tag recommender algorithms in real-world folksonomies: A comparative study[C] // Proceedings of the 9th ACM Conference on Recommender Systems. 2015 : 265 – 268.