学校代号 10532　　学　　号 S141000882

分 类 号 TP301　　密　　级 普通

湖南大學
HUNAN UNIVERSITY

硕士学位论文

多源数据融合的协同过滤算法研究

学位申请人姓名 刘四平

培 养 单 位 信息科学与工程学院

导师姓名及职称 曾庆光 教授 李仁发 教授

学 科 专 业 计算机科学与技术

研 究 方 向 分布式计算系统

论文提交日 期 2017 年 5 月 10 日

学校代号：10532

学　　号：S141000882

密　　级：普通

湖南大学硕士学位论文

多源数据融合的协同过滤算法研究

学位申请人姓名：　刘四平

导师姓名及职称：　曾庆光 教授　李仁发 教授

培　养　单　位：　信息科学与工程学院

专　业　名　称：　计算机科学与技术

论 文 提 交日 期：　2017 年 5 月 10 日

论 文 答 辩日 期：　2017 年 5 月 21 日

答辩委员会主席：　邝继顺 教授

Research on collaborative filtering algorithm for multi-source data fusion

by

Siping Liu

B.E. (Lanzhou Jiaotong University) 2014

A thesis submitted in partial satisfaction of the

Requirements for the degree of

Master of Engineering

in

Computer Science and Technology

in the

Graduate School

Of

Hunan University

Supervisor

Professor Qingguang Zeng

Professor Renfa Li

May, 2017

湖 南 大 学

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律后果由本人承担。

作者签名：　　日期：　　年　月　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权湖南大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于

1、保密□，在_____年解密后适用本授权书。

2、不保密□。

（请在以上相应方框内打“√”）

作者签名：　　日期：　　年　月　日

导师签名：　　日期：　　年　月　日

摘　要

推荐系统面临数据稀疏性和冷启动问题，在大数据时代的背景下，提高用户获取信息的效率特别重要。本文以大数据环境下推荐系统的多源数据融合分析问题为背景，设计基于容器技术 Docker、大数据平台 Spark 和 Hadoop 的多源数据融合模型，并实现协同过滤推荐算法的并行化设计。将自动编码器和协同过滤推荐算法结合起来，提高隐式反馈中 Top-N 推荐的准确率。具体工作内容可以概括为：

1）提出多源数据融合的模型。利用用户在互联网（包括微信，微博，网站等）中访问的数据、数据库数据和日志数据等，提出统一显式与隐式反馈模型（UEIFM），通过对可观察用户选择行为的隐式用户反馈数据和评分等显式反馈数据，将项目推荐问题转化为优化问题，以提高推荐的准确率。

2）优化基于模型的协同过滤算法。在潜在因子模型的基础上优化基于分布式和迭代计算的并行模型，有效地实现矩阵分解算法的并行化。并提供基于 Spark 平台的实现，以处理大规模多源数据。

3）提出结合自动编码器的协同过滤框架—协同自编码器（CF-AE）。针对协同过滤算法中的数据稀疏输入的问题，设计自编码网络来学习的用户和项目之间的复杂关系，进而得到协同自动编码器框架。

最后，利用公共数据集 Movielens 验证模型的有效性，并对比其他算法来评估模型的准确性，有效解决推荐系统中数据的稀疏性问题。通过大量的对比实验验证 UEIFM 模型及并行化实现的有效性。使用自动编码器和矩阵分解方法，进一步提高推荐的准确性。在大数据平台下利用分布式并行化方法，解决当前推荐系统面临的可扩展性问题。借助云计算平台，提高系统和算法的鲁棒性。

关键词：推荐系统；协同过滤；矩阵分解；显式和隐式反馈；自动编码器

Abstract

The recommender systems is facing data sparsity and cold start problem. In the background of big data era, it is very important to improve the efficiency of users' information acquisition. Based on the multi-source data fusion analysis problem of the recommender system in the big data environment, this paper designs a multi-source data fusion model based on the container technology Docker and the big data platform Spark and Hadoop, and implements the parallel design of the collaborative filtering recommendation algorithm. Combining auto-encoder with collaborative filtering recommendation algorithms to improve the accuracy of Top-N recommendation in implicit feedback. Specific work can be summarized as:

1) We propose multi-source data fusion model. (UEIFM), through the use of users in the Internet (including WeChat, WeiBo, website, etc.) access to the data, database data and logs data, etc., We propose a unified explicit and implicit feedback model (UEIFM), by observing the user to select the behavior of implicit users feedback data and other explicit feedback data, the recommendation problem convert into optimization issues to improve the recommendation precision.

2) We optimize model-based collaborative filtering algorithm. Based on the latent factor model, the parallel model based on distributed and iterative computation is optimized, and the parallelization of matrix decomposition algorithm is realized effectively. And we provide implementations based on the Spark platform to handle large scale multi-source data.

3) We propose collaborative auto-encoder (CF-AE), a collaborative filtering framework combine with auto-encoder. Aiming at the problem of data sparse input in collaborative filtering algorithm, we design a auto-encoder network to learn the complex relationship between users and items, and then get the cooperative auto-encoder framework.

The validity of the model is verified by the utility dataset, and the precision of the model is evaluated by comparing the other algorithms, and the sparseness of the data in the recommender system is effectively solved. The validity of UEIFM model and parallelization is verified by a number of

contrast experiments. The use of auto-encoder and matrix decomposition methods to further improve the recommendation precision. In the big data platform using distributed parallelisam method to solve the current recommendation system is facing the scalability problem. With the cloud computing platform, improve the robustness of systems and algorithms.

Key Words: Recommender Systems; Collaborative Filtering; Matrix Factorization; Explicit and implicit feedback; AutoEncoder

目 录

插图索引

附表索引

第1章 绪论

1.1 研究背景

互联网“信息过载”日趋加剧的问题，是亟待解决的问题之一，推荐系统作为有效缓解该问题的方法，受到工业界和学术界越来越多的关注。如何充分利用丰富的用户反馈、社会化网络等信息进一步提高推荐系统的性能和用户满意度，成为大数据环境下推荐系统的主要任务。

传统的推荐系统在处理大数据时存在的问题正在限制其性能的发挥。为了充分挖掘数据价值，提高推荐系统的性能和实时性，进一步有效缓解信息过载的问题，系统和算法的扩展和改进、支撑理论的发展成为推荐系统亟待解决的问题。随着信息规模的扩大，在为用户提供信息和服务的同时，也为用户提供了更为丰富的选择。但是，面对如此多样的新闻信息，用户如何才能快速、准确地定位到自己关心的内容，也成了用户最为关心的话题。因此，无论对服务提供方还是用户而言，推荐系统都具有巨大的发展前景和应用价值。

随着移动应用的发展，信息数据量正爆炸式增长，大数据概念受到了工业界和学术界的普遍关注。大数据正在引发一场新的技术革命，科技在丰富人类生活的同时，也给人们带来了选择的困扰，如何快速有效地从繁杂的数据中获取有价值的信息。美国 Netflix 视频网站拥有海量的视频资源，同时运用推荐技术来提高用户的使用体验。Netflix 的推荐系统[1]专注于两类算法的集成，分别是 SVD++[2]和 RBM[3]。Netflix 网站 2/3 的观看电影都由推荐产生。大多数电子商务网站都有推荐引擎来驱动一些用户体验。亚马逊是第一家以推荐系统为核心的大型电子商务公司，亚马逊最初使用一种简单的项目-项目协同过滤方法[4]。亚马逊购物网站 35%的销售额都来自推荐。Google 新闻使用推荐引擎后，增加了 38%的点击率。

互联网中两种主要的获取信息的方式是搜索引擎和推荐系统。搜索是用户的主动行为，有明确的目的性。搜索引擎根据需求将提供结果显示在搜索列表中（如谷歌，百度等）。推荐系统接受用户信息是被动的，是在用户没有明确需求的情况下，需求是模糊的。但是搜索引擎和推荐系统都可以通过用户的浏览和点击来明确判断是否满足用户的需求。很多互联网产品都提供了搜索和推荐的服务。但是二者并不冲突，还是相辅相成的。搜索引擎可以融合推荐系统的结果。例如对提供电影、音乐、新闻或者电商的网站，必然要提供搜索功能，当用户想找某首歌或某样商品的时候，输入名字就能搜到。与此同时，也同时要提供推荐功能，当用户就是想听好听的歌，或者想看看新闻，但并不明确一定要听哪首的

时候，需要提供良好的推荐结果，来提升用户体验。在搜索引擎结果中，也可以显示推荐系统产生的推荐结果。作为大数据应用的两大类信息获取方式，搜索引擎和推荐系统既相互伴随和影响，又满足不同的应用场景需求。推荐系统预测人们可能喜好的项目，并通过探索用户和项目之间的联系来促进这个过程。与搜索引擎不同的是，推荐引擎会试图向用户呈现的相关内容并不一定就是用户搜索的，其返回的结果甚至是用户都没有听说过的。

搜索引擎从用户主观的角度解决了主动获取信息的问题，但是搜索引擎不提供信息的个性化展示，推荐系统可以主动根据推送信息给用户。简而言之，搜索是用户在寻找什么的时候做什么，推荐是不知道信息存在的时候，主动展现的。推荐的价值在于提高人们获取信息的质量。

个性化程度的高低是搜索引擎和推荐系统的重要区别。推荐系统在个性化方面的运作空间要比搜索引擎大得多。推荐系统可以根据用户历史的观看行为、评分记录等信息生成一个对当前用户最有价值的结果。虽然推荐的种类有很多（例如相关推荐、个性化推荐等），但是个性化对于推荐系统是如此独特，所以在很多时候推荐系统就是“个性化推荐”或“智能推荐”。

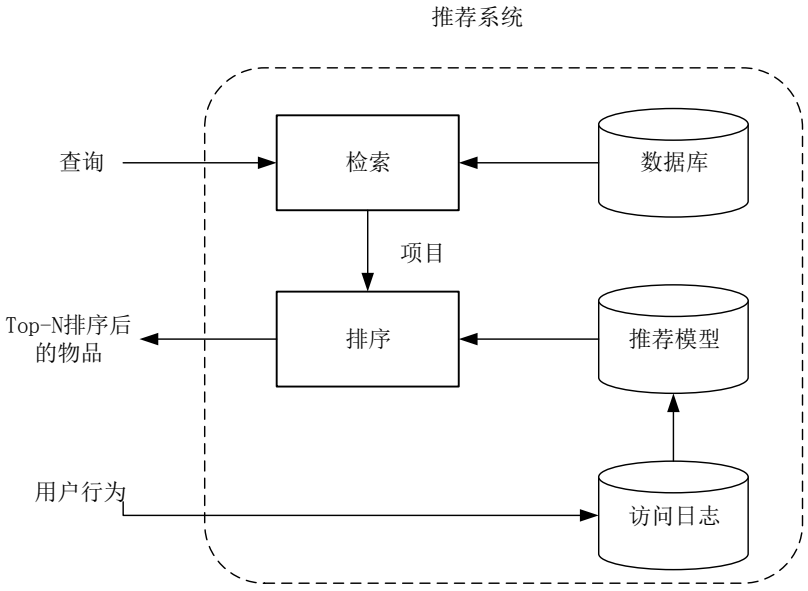


图 1.1 推荐系统概览

现代推荐系统通过学习用户历史反馈信息，根据项目的属性，发现用户的兴趣偏好来提供个性化的推荐。推荐系统帮助人们从海量信息中发掘用户可能感兴趣的东西，如电影，音乐，新闻[5]，书籍，视频[1]，商品[4]等。如图 1.1 所示。推荐系统通过收集用户的行为信息，以日志的形式保存起来，将这些数据预处理之后，用于推荐引擎训练推荐模型，分析用户兴趣偏好。当外部系统需要查询用户的推荐列表时，推荐系统从数据库中检索出项目信息，结合推荐模型给出的用户偏好信息，对用户感兴趣的项目作排序处理，最后提供给外部查询。

信息技术部门中机器学习的主要应用之一是向潜在用户或客户推荐项目。可以分为两种主要的应用：在线广告和项目推荐。两者都依赖于预测用户和项目之间的关联，如果展示了广告或向该用户推荐了该产品，推荐系统要么预测用户购买产品或该行为的一些代替行为的概率或预期增益。目前，互联网的资金主要来自于各种形式的在线广告。经济的主要部分依靠网上购物。包括 Amazon 和 eBay 在内的公司使用机器学习推荐产品。有时，项目不是实际出售的产品。如选择在社交网络新闻信息流上显示的帖子、推荐观看的电影、推荐笑话、推荐专家建议、匹配视频游戏的玩家或匹配约会的人。

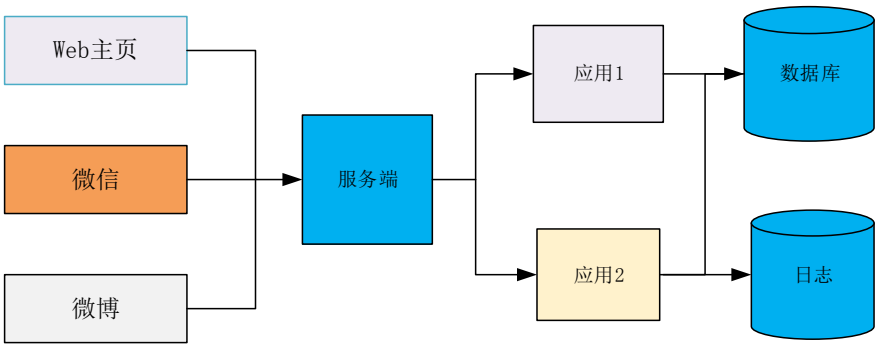


图 1. 2 多数据来源的获取和采集

当前大数据环境下的推荐系统成为主流，为了充分挖掘数据的价值，提高推荐的实时性和准确性，进一步有效地缓解信息过载的问题，系统和算法的可扩展性和性能问题都是面临的重要挑战。同时，数据产生的速度更快，数据高维且稀疏，内容采样渠道更多。多源数据在融合的过程中由于结构和采集方式的不同会引入噪声和冗余，更多的是非结构数据和半结构的数据。因此，有效利用多源数据成为获取有效推荐的必要前提，值得进行研究。

同时，数据的收集和存储，成为大数据不可回避的问题。如图 1.2 所示，数据的是收集来源主要包括 web 页面，移动应用如微信、微博等社交软件等，服务器端响应不同的后台应用来处理请求，将数据存储在数据库中，同时访问记录存储在日志文件中。

1.2 研究内容

推荐系统的任务是联系用户和产品，解决信息过载问题。大数据环境下的推荐系统是传统推荐系统的延伸，由于大数据环境比传统环境面临更加复杂的信息提供环境和数据特征，只有在充分、准确提取和预测用户产生的各种数据中蕴含的用户兴趣偏好后，才能有效生成准确度更高的推荐。因此，尽管大数据环境下推荐系统的基本思想与传统推荐系统是相似的，但着重考虑大数据环境给推荐系统带来的影响：数据产生的速度更快，数据高维稀疏，内容采样渠道更多，包括日志数据、微博数据、电商数据和微信数据。这些数据构成了多个数据来源，不

同的数据类型和格式。但多源数据在融合时由于结构和采集方式的不同会引入更高的噪声和冗余，数据结构比例发生变化，非结构数据、半结构数据成为主要数据，流式数据也成为常见数据类型。数据内容变得丰富，推荐系统可以采集到丰富的用户隐式反馈数据。移动网络的快速发展，促使移动应用变得丰富多彩，用户使用移动设备或登录移动应用产生丰富的移动社会化网络数据，尤其以社交网络为主的微信，微博等提供了数据流量入口。

利用用户在互联网（包括微信，微博，网站等）访问的数据来追踪和勾画用户偏好行为并以此为据，提供用户潜在感兴趣信息的推荐是大数据应用的重要方向。推荐系统作为智能决策的支持手段之一，在大数据应用和发展过程中还面临巨大的挑战，多源数据的融合就成为大数据分析中的瓶颈。

融合多源数据以形成有效的用户行为分析数据集仍是推荐系统面对的一个更具挑战性的瓶颈。其中多源数据的融合是最耗费资源的任务之一。如何组织和利用各个来源的数据进行定量的分析，需要对数据的来源和结构有一定的控制和深层的了解，并给出一个特定的模型来完成。

随着 Web 应用和移动应用规模的不断扩大，数据规模也将变得越来越大。以数据处理为主的诸多大数据问题使推荐系统对数据处理效率的要求更高，丰富的数据使得用户对推荐系统准确性要求更高，同时用户要求具有实时性的要求获得良好的用户体验。这使得传统推荐系统的方法并不能直接应用到大数据环境下的推荐系统中，需要对算法进行改进和扩展，才能达到大数据环境下推荐系统处理海量数据的要求。基于 Hadoop 和 Spark 等大数据处理平台可以在海量数据中充分挖掘用户潜在的兴趣，采用合适的推荐技术为用户提供良好的服务成为当前推荐系统中最核心的问题。

推荐系统的功能就是评估效用函数用来预测用户是否会喜欢项目或者说喜欢项目的概率。一个好的推荐系统一定是根据用户喜好个性化地产生推荐列表。产生的推荐也是多样性的，代表着用户所有可能的兴趣。推荐一些用户不知道的东西，但可能会感兴趣的项目给用户。

近年来，推荐系统已经被广泛商业应用到各个行业中。推荐系统的研究热点包括冷启动和混合推荐、情景敏感推荐、安全和用户隐私、推荐系统评估方法、推荐的多样性和新颖性、推荐理论和方法、排序和 Top-N 推荐、推荐系统的新应用等[6]。

深度学习在图像和语音识别方面具有巨大的成功，神经网络在协同过滤中的研究日趋热门。稀疏输入决定了协同过滤的效果，而深度神经网络在学习有效特征表示方面效果显著。有效地结合深度神经网络和协同过滤算法，成为问题的关键。

对于给定一组用户，项目和用户—项目的历史交互信息，这些系统可以推荐

用户可能喜欢的其他项目。个性化推荐是机器学习在电子商务等领域的关键应用之一。许多推荐系统使用协同过滤方法来设计推荐引擎。在实际生产系统中，推荐系统通常基于 Top-N 推荐的性能来评估，因为通常每次向用户仅显示少数推荐结果。因此，Top-N 推荐方法一直是研究的热点方向。

在基于模型的推荐系统方法中，大多数机器学习模型可以通过两个部分来指定：训练期间的模型定义和目标函数。模型定义阐述了输入（例如，用户 ID，项目 ID，交互信息，其他特征等）和输出（评分或项目的隐式反馈）。目标函数是训练过程优化以找到最佳模型参数。

设计基于模型的推荐系统的两个关键组成部分：1）一种表示输入和输出之间的关系的合适方式；2）适当的目标函数和适当的方法来处理观察到的关系和不可观测的反馈。

1.3 主要贡献

大多数真实世界的推荐服务基于向最终用户显示的 Top-N 推荐结果来测量其性能。因此，Top-N 推荐的研究进展在实际应用中具有深远的影响。本文以大数据环境下推荐系统的多源数据融合分析问题为背景，同时面临数据稀疏性问题和冷启动问题，设计了基于云计算环境和大数据平台的多源数据融合模型和协同过滤推荐算法，并实现了算法的并行化，结合深度神经网络，提高隐式反馈中进行 Top-N 推荐的等精度问题。本文的具体工作内容如下：

第一，介绍协同过滤算法，并说明算法的不足和改进的方向。并分析协同过滤算法对解决推荐问题的有效性，特别是在实际背景下的推荐精度问题。

第二，介绍当前推荐系统面临的可扩展性问题，基于云计算环境和大数据平台，可以有效地解决可扩展性和鲁棒性问题。

第三，利用用户在互联网（包括微信，微博，网站等）访问的数据、数据库数据和日志数据等，统一多种数据资源，提出一种融合显式与隐式反馈融合的推荐模型 UEIFM，通过对可观察用户选择行为数据集（隐式用户反馈数据）和评分（显式反馈数据）的后验估计，将推荐问题转化为数据融合和优化问题，以提高推荐质量。

第四，在 ALS 基础上进一步提出了基于分布式计算和迭代计算的并行优化模型 UEIFM 并提供了 Spark 的实现，以处理大规模多源数据，最后，在实验中验证所提算法模型 UEIFM 及和并行化实现的有效性和可扩展性。

第五，提出结合自动编码器的协同过滤框架 CF-AE。稀疏输入对于协同过滤是至关重要的，指出被学习的用户和项目表示之间的复杂关系。介绍了去噪自动编码器，自动编码器与协同过滤中的低秩矩阵因子分解是密切相关的，进而引出协同自动编码器架构。然后解释如何在其顶部添加额外的约束，以便网络处理

稀疏输入。最后，详细介绍完整的设计和训练过程，完成实验。

1.4 章节组织

本文共由五个章节组成，全篇围绕协同过滤推荐算法进行的 Top-N 推荐问题展开。

第一章，主要介绍推荐系统课题研究的目的和意义，并介绍了研究的主要工作是协同过滤算法和多源数据融合问题，重点研究隐式反馈对推荐精度的影响。简述本文的主要研究工作以及文章的组织结构。

第二章，主要介绍了国内外对推荐系统的研究现状，并介绍了云计算和大数据领域一些最新的技术，并将这些技术应用到推荐系统的架构中，基于云计算环境平台实现推荐算法和并行模型。协同过滤是推荐系统中的重要算法，广泛应用到各种类型的推荐领域中，将协同过滤和深度学习结合也是近年来的研究热点，并介绍了协同过滤推荐中最重要的矩阵分解技术。

第三章，主要研究多源数据融合问题，提出统一显式与隐式反馈融合的推荐模型 UEIFM，在 ALS 基础上进一步提出了基于分布式计算和迭代计算的并行优化模型 UEIFM，并提供了基于 Spark 的实现。

第四章，主要介绍了结合自动编码器的协同过滤框架 CF-AE。并介绍了去噪自动编码器，提出了协同自动编码器架构。然后解释了如何在其顶部添加额外的约束，以使网络处理稀疏输入。

第五章，搭建了基于 Docker 技术的 Spark 分布式测试平台，分析推荐的精度问题。完成了第三章和第四章提出的模型和算法的实验，并和其他算法进行对比，然后得出实验结果。

最后是结论部分。对本文所做的工作进行了总结，主要工作包括多源数据融合模型和结合深度学习的协同自动编码器，以及展望未来的工作方向。

第2章 相关研究和基础知识

2.1 引言

面对“信息过载”问题，个性化推荐系统应运而生。它有效弥补了搜索引擎的不足，使得用户在互联网中获取有效信息的方式更加高效。其中推荐引擎是整个推荐系统中最核心的部分，推荐算法构成了推荐引擎。

推荐系统的稀疏性和冷启动是具有挑战性的问题。如何缓解冷启动问题，以及处理数据的稀疏性问题，国内外研究人员提供了很多不同的研究方法。同时，以协同过滤为重点研究方向。如何在大数据环境下解决推荐的效率和可扩展性问题，也成为重点研究方向。

本章详细阐述了推荐系统的研究现状，按照协同过滤算法研究现状和协同过滤算法结合深度学习研究现状的顺序分别阐述。同时详细地阐述了推荐系统中的各种类型的推荐算法，每个推荐算法都有各自的优点和存在的不足。最后，介绍了云计算和大数据技术中的主流技术，包括容器技术 Docker 和大数据处理框架 Spark。

2.2 协同过滤的研究基础与相关进展

协同过过滤算法是推荐系统中最重要算法之一，下面将协同过滤算法的相关研究进展，并介绍基于大数据平台，对协同过滤算法的并行设计方法以及矩阵分解算法。

2.2.1 推荐系统

推荐系统的关键组成部分包括：1）数据；2）算法；3）系统。

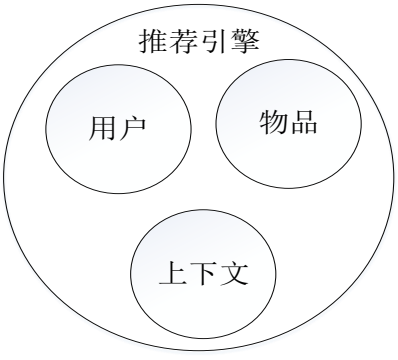


图 2. 1 推荐引擎的重要组成部分

推荐引擎的重要组成部分包括用户、项目和上下文，如图 2.1 所示。有 3 种联系用户和项目的方式用户喜欢项目，项目和其他项目之间有内在相似的关系，用户也可能喜欢相似的项目；用户和用户之间有存在相似的兴趣偏好，相似用户喜欢的项目，可能该用户也会喜欢；用户喜欢项目的某些特征，而这些特征包含在某个项目中，用户可能会喜欢该项目，如图 2.2 所示。

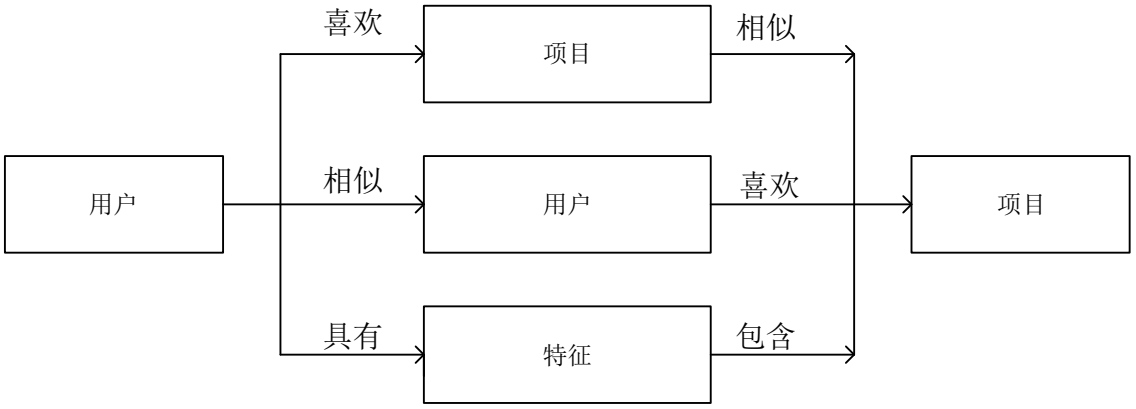


图 2. 2 3 种方式联系用户和项目

2.2.2 推荐算法介绍

在推荐系统中，推荐算法是整个推荐系统最核心、最关键的部分，推荐算法的选择很大程度上决定了推荐系统性能的优劣。下面详细介绍主要的推荐算法及其原理。

基于内容的推荐（Content Based Recommendation）是在项目的内容属性信息上作出推荐的，不需要用户对项目的评价等相关信息，而是依靠机器学习的方法从内容的特征描述中获取用户的可能感兴趣的内容。在基于内容的推荐系统中，通过相关的特征的属性来定义项目，系统根据用户评价项目的特征，学习到用户的兴趣，评估用户资料信息与预测项目信息的匹配程度。用户的资料模型取决于采用的学习算法，常用的有决策树和基于向量的表示法等。基于内容的用户资料信息是需要提供用户的历史数据，用户资料模型可能随着用户的内容和偏好的改变而发生变化。基于内容的推荐不需要其他用户的信息，所以没有冷启动和稀疏性问题，并且能为具有特殊兴趣爱好的用户推荐项目。同时，推荐列表具备很好的可解释性。基于内容的推荐要求内容能容易抽取成有意义的特征，有良好的结构性，并且用户的偏好要用内容特征的形式来表达。

协同过滤（Collaborative Filtering）推荐技术是推荐系统中应用最早和最为成功的技术之一。根据用户的历史行为数据来产生推荐。通过收集来自许多用户（协同）的口味信息来预测（过滤）对新项目的用户偏好。分为两大类：基于邻域的协同过滤推荐和基于模型的协同过滤推荐。

基于邻域的协同过滤推荐分为基于项目的协同过滤[7]和基于用户的协同过

滤。根据用户或者项目的相似度来进行推荐。然后利用用户（项目）的最近邻居用户（项目）对项目（用户）评价的加权评价值来预测目标用户对特定商品的喜好程度，系统从而根据这一喜好程度来对目标用户进行推荐。相似度计算的方式也有很多可供选择，比如皮尔逊相似度、余弦相似度等。基于邻域的协同过滤相似度的计算是瓶颈，时间复杂度很高，当用户和项目数量很多时，计算时间会很长。通常会分成两个处理步骤，第一步是离线相似度的计算，预先计算好相似度并保存。第二步是在线预测过程，根据离线计算的相似度进行预测，产生推荐项目列表。两个步骤是循环过程，需要更新用户和项目的信息，重新计算相似度。相对来说用户的相似度计算是动态的，但是预先计算用户的相似度可能导致推荐效果很差。项目的相似度计算是相对静态的，只要项目不增加的情况下，预先计算项目的近邻。在很多情况下能够产生很好的推荐，需要大量的显式的评分，并且是可靠的。基于的假设是先前的行为决定现在的行为模式，而没有把上下文信息考虑进来。

基于模型的协同过滤推荐利用模型来学习用户和项目之间的交互模式，进而产生推荐。模型是从底层数据学习的，而不是启发式。协同过滤算法是众包的智慧，利用大量已有的用户偏好来估计用户对项目的兴趣偏好。协同过滤推荐为用户找到感兴趣的内容方法是找到与该用户兴趣相似的其他用户，然后将其他用户感兴趣的项目且该用户未知的项目推荐给该用户。基于协同过滤的推荐系统是从用户的角度进行相应推荐，用户获得的推荐结果是模型通过评论结果、购买模式和浏览行为等方式获取的。文献[8]中介绍常见的基于模型的协同过滤推荐算法有聚类协同过滤模型、隐语义模型、概率因素模型、贝叶斯信念网协同过滤模型、隐因子模型也称为矩阵分解模型，采用 SVD、主成分分析、SVD++等矩阵分解方法来推荐。

虽然协同过滤作为一种典型的推荐技术并且得到了广泛应用，但是协同过滤算法仍有许多的问题需要解决。比如许多项目可供选择；不能很好地解释；每个用户可能只有少量数据；没有新用户或新项目的数据，会有冷启动问题。在非常多的用户和项目的时候，数据会很稀疏，数据集也会很大。最典型的问题有稀疏性问题和可扩展性问题。数据稀疏性问题可以通过降维的方法解决，包括矩阵分解，聚类，主从成分分析（PCA）等。

混合推荐（Hybrid Recommendation）中研究和应用最多的是基于内容推荐和协同过滤推荐的组合。通过组合后避免或弥补了各自推荐技术的弱点。在组合方式上，有加权（Weight）、变换（Switch）、混合（Mixed）、特征组合（Feature combination）、层叠（Cascade）、特征扩充（Feature augmentation）、元级别（Meta-level）等。

推荐系统典型的任务有 3 种：

1）评分预测

最早的推荐系统任务（也是目前较为常见的任务）是评分预测。输入归纳起来可以分为用户、物品和评分三个方面，因此可以使用一个二维矩阵来刻画评分预测的输入，分别对应于一个矩阵中的行、列、值。为了解决这一问题，常见的算法如基于相似近邻的协同过滤算法、矩阵分解等。其中矩阵分解算法得到了广泛应用，并且在实践中具有很好的效果[9]。

2）物品推荐

与评分预测相似，输入归纳起来可以分为用户对应的物品二维矩阵来刻画输入，不同的是每个矩阵数值不是一个具体的打分，而是一个用户是否选择了某一物品。大部分评分预测算法都可以（可能需要适当改动）应用到物品推荐中。

3）基于背景或者特征的推荐

推荐系统的不断发展进一步丰富了可供推荐算法使用的信息。如对于新闻推荐，物品的属性则有可能是新闻的文本内容、关键词、时间等，同时包括用户的点击、收藏和浏览行为等等。在电商网站上，还可能包含很多信息评论文本、用户查看的历史记录、用户购买的记录等。还可能获得用户的反馈信息，总体上可以分为两类：一是显式的用户反馈（Explicit Feedback），这是用户对商品或信息给出的显式反馈信息，例如评分、评论等；另一类是隐式的用户反馈（Implicit Feedback），这类一般是用户在使用网站的过程中产生的数据，它们也反映了用户对物品的喜好，比如用户查看了某物品的信息，用户在某一页面上的停留时间等。对于基于背景敏感的推荐，可以使用 SVD++等基于特征的推荐算法。

推荐的结果产生需要经过几个步骤。首先，推荐引擎产生初始的推荐结果，然后经过过滤和排名操作，最后得到最终的推荐结果。整个流程如图 2.3 所示。

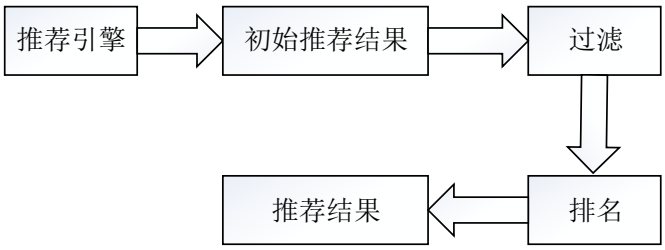


图 2. 3 推荐结果产生流程

推荐问题解决的问题归纳为给用户 u 推荐项目 i，并且满足用户偏好、新奇度和多样性等。协同过滤和矩阵分解是常用的方法。

协同过滤按照用户和项目维度可以分为基于用户的协同过滤和基于项目的协同过滤，主要思想是计算用户与用户、项目与项目以及用户和项目的相关度。以项目协同过滤举例，将项目表示为用户维度的向量，这样就可以计算项目之间的相似度，则用户与项目相似度为公式（2.1）：

$$r_{ui} = \frac{\sum_{j \in S^k_{I:u}} s_{ij} r_{uj}}{\sum_{j \in S^k_{I:u}} s_{ij}} \tag{2.1}$$

其中 $S_{I:u}$ 表示和项目 i 最相似的项目集合。

早期推荐系统的工作依赖于作为这些预测输入的最小信息：用户 ID 和项目 ID。在这种情况下，唯一的推广方式是依赖于不同用户或不同项目的目标变量值之间的模式相似性。假设用户 1 和用户 2 都喜欢项目 A，B 和 C。由此，可以推断出用户 1 和用户 2 具有类似的口味。如果用户 1 喜欢项目 D，那么这可以强烈提示用户 2 也喜欢 D。基于此原理的算法称为协同过滤。非参数方法（例如基于估计偏好模式之间相似性的最近邻方法）和参数方法都可能用来解决这个问题。参数方法通常依赖于为每个用户和每个项目学习分布式表示（也称为嵌入）。目标变量的双线性预测（例如评级）是一种简单的参数方法，这种方法非常成功，通常被认为是最先进系统的组成部分。通过用户嵌入和项目嵌入之间的点积（可能需要通过仅依赖于用户 ID 或项目 ID 的常数来校正）获得预测。

2.2.3 矩阵分解

用户对项目的打分矩阵可以通过矩阵分解技术得到用户和项目在低维空间的表示，这样可以通过向量相似度计算得到用户-用户、项目-项目以及用户到项目之间的相似度。SVD 分解可以解决该问题，由于用户对项目打分矩阵不是完备的，需要对缺少数据进行处理，此时应用 SVD 才能得到较好效果。

进行因子分解时，对 SVD 问题进行补充，考虑已知的打分，如公式（2.2）：

$$\min \sum_{r_{ui} \neq 0} (r_{ui} - x_u^{\mathrm{T}} y_i)^2 + \lambda (\sum_u \| x_u \|^2 + \sum_i \| y_i \|^2) \tag{2.2}$$

基于模型的的协同过滤方法是对“用户-项目”偏好建模，对未知的“用户-项目”组合上应用该模型便可以得到新的偏好。基于矩阵分解的模型在协同过滤中表现出很好的性能。

最小二乘法（Alternating Least Squares，ALS）是一种求解矩阵分解问题的最优化方法。将用户项目矩阵分解为用户因子矩阵和项目因子矩阵。该方法效果较好，且相对容易并行化。ALS 算法的实现原理是迭代式求解一系列最小二乘回归问题。在每一次迭代时，固定用户因子矩阵或是项目因子矩阵中的一个，然后用固定的这个矩阵以及评级数据来更新另一个矩阵。之后，被更新的矩阵被固定住，再更新另外一个矩阵。如此迭代，直到模型收敛或迭代到了预先设置好的次数。并设置每一次迭代中的最小二乘法是否都要非负约束。使用交替最小二乘来求解基于矩阵分解的协同过滤推荐问题。通过观察用户给项目的评分或者浏

览信息，来推断用户的喜好并向用户推荐可能感兴趣的项目。

2.2.4 Spark 分布式并行计算技术

Apache Hadoop 是用于批处理的处理框架。基于谷歌公司有关海量数据处理发表的多篇论文与经验的 Hadoop 重新实现了相关算法和组件堆栈，让大规模批处理技术变得更易用。

Hadoop 的组件包括：

1）HDFS：HDFS 是一种分布式文件系统层，可对集群节点间的存储和复制进行协调。HDFS 确保了无法避免的节点故障发生后数据依然可用，可将其用作数据来源，可用于存储中间态的处理结果，并可存储计算的最终结果。

2）YARN：Yet Another Resource Negotiator 即另一个资源管理器，可充当 Hadoop 堆栈的集群协调组件。该组件负责协调并管理底层资源和调度作业的运行。作为集群资源的接口，YARN 使得用户能在 Hadoop 集群中使用比以往的迭代方式运行更多类型的工作负载。

3）MapReduce：MapReduce 是 Hadoop 的原生批处理引擎。

Hadoop 的处理功能来自 MapReduce 引擎。MapReduce 的处理技术符合使用键值对的 map、shuffle、reduce 算法要求。基本处理过程包括：

1）从 HDFS 文件系统读取数据集；

2）将数据集拆分成小块并分配给所有可用节点；

3）针对每个节点上的数据子集进行计算，并将计算的中间态结果会重新写入 HDFS；

4）重新分配中间态结果并按照键进行分组；

5）通过对每个节点计算的结果进行汇总和组合对每个键的值进行 reduce；

6）将计算而来的最终结果重新写入 HDFS。

由于这种方法依赖持久存储，每个任务需要多次执行读取和写入操作，因此速度相对较慢。但另一方面由于磁盘空间通常是服务器上最丰富的资源，这意味着 MapReduce 可以处理非常海量的数据集。同时也意味着相比其他类似技术，Hadoop 的 MapReduce 通常可以在廉价硬件上运行，因为该技术并不需要将一切都存储在内存中。因此，最适合处理对时间要求不高的非常大规模数据集。通过非常低成本的组件即可搭建完整功能的 Hadoop 集群，使得这一廉价且高效的处理技术可以灵活应用在很多案例中。与其他框架和引擎的兼容与集成能力使得 Hadoop 可以成为使用不同技术的多种工作负载处理平台的底层基础。

Apache Spark 是一种包含流处理能力的下一代大数据处理框架。与 Hadoop 的 MapReduce 引擎基于各种相同原则开发而来的 Spark 主要侧重于通过完善的内存计算和处理优化机制加快批处理工作负载的运行速度。

Spark 可作为独立集群部署，但是需要相应存储层的配合，或与 Hadoop 集成并取代 MapReduce 引擎。与 MapReduce 不同，Spark 的数据处理工作全部在内存中进行，只在一开始将数据读入内存，以及将最终结果持久存储时需要与存储层交互。所有中间态的处理结果均存储在内存中。虽然内存中处理方式可大幅改善性能，Spark 在处理与磁盘有关的任务时速度也有很大提升，因为通过提前对整个任务集进行分析可以实现更完善的整体式优化。为此 Spark 可创建代表所需执行的全部操作，需要操作的数据，以及操作和数据之间关系的有向无环图（Directed Acyclic Graph，DAG），借此处理器对任务进行更高效的调度。

为了实现内存中批计算，Spark 使用弹性分布式数据集（Resilient Distributed Dataset，RDD），即利用 RDD 的编程模型来处理数据。RDD 是一种抽象的数据集，保存在内存中，不可变的结构。针对 RDD 执行的操作可生成新的 RDD。每个 RDD 可通过血统（Lineage）回溯至父级 RDD，并最终回溯至磁盘上的数据。Spark 可通过 RDD 在无需将每个操作的结果写回磁盘的前提下实现容错。

Spark 相比 Hadoop MapReduce 的优势主要是速度。利用基于内存的计算策略和 DAG 调度等机制，Spark 可以用更快的速度处理相同的数据集。Spark 的另一个重要优势在于多样性，可作为独立集群部署，或与现有 Hadoop 集群集成。Spark 集群可以运行批处理和流处理等不同类型的任务。除了引擎自身的能力外，围绕 Spark 还建立了包含各种库的生态系统，可为机器学习、交互式查询等任务提供更好的支持。

Spark 已经取代 Hadoop 成为最活跃的开源大数据项目。Hadoop 和 Spark 均是大数据框架，都提供了一些执行常见大数据任务的工具。但是它们所执行的任务并不相同，彼此也并不排斥。在特定的情况下，Spark 要比 Hadoop 快 100 倍，但 Spark 并没有分布式存储系统。而分布式存储是如今许多大数据项目的基础。它可以将 PB 级的数据集存储在几乎无限数量的普通计算机的硬盘上，并提供了良好的可扩展性，只需要随着数据集的增大增加硬盘。因此，Spark 需要一个第三方的分布式存储。也正是因为这个原因，许多大数据项目都将 Spark 安装在 Hadoop 之上。这样，Spark 的高级分析应用程序就可以使用存储在 HDFS 中的数据。与 Hadoop Mapreduce 相比，Spark 真正的优势在于速度。Spark 的大部分操作都是在内存中，而 Hadoop 的 MapReduce 系统会在每次操作之后将所有数据写回到物理存储介质上。这是为了确保在出现问题时能够完全恢复，但 Spark 的弹性分布式数据存储也能实现这一点。在高级数据处理（如实时流处理和机器学习）方面，Spark 的功能要胜过 Hadoop。

有效结合 Hadoop 和 Spark 可以充分发挥大数据处理系统的功能，Hadoop 提供分布式存储能力的 HDFS，保证海量数据存储的功能和满足数据的可靠性要

求。YARN 可作为集群资源管理平台，Spark 提供分布式并行处理的计算平台，满足一站式计算需求，有效弥补 Hadoop 在计算效率方面的不足。同时 Hadoop 和 Spark 两者结合使用，是大数据处理生态系统提供的多种框架的组合使用的趋势。Spark 和 Hadoop 组合架构，如图 2.4 所示。

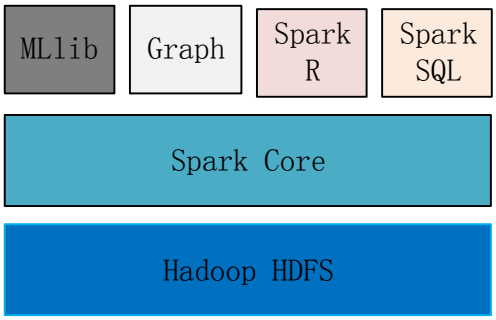


图 2. 4 Spark 和 Hadoop 组合架构图

面对计算的及时性需求，越来越多的应用需要达到实时处理的要求。实时处理意味着可以在数据捕获的瞬间将其提交给分析型应用程序，并立即获得反馈。在各种各样的大数据应用程序中，这种处理的用途越来越多。比如，零售商使用的推荐引擎、制造业中的工业机械性能监控。Spark 平台的速度和流数据处理能力也非常适合机器学习算法。机器学习算法可以自我学习和改进，直到找到问题的理想解决方案。流处理技术是最先进制造系统（如预测零件何时损坏）和无人驾驶汽车的核心技术。Spark 有机器学习库 MLlib，而 Hadoop 需要借助第三方机器学习库 Apache Mahout。

2.2.5 协同过滤的相关研究

协同过滤推荐是推荐算法中最成功的策略之一[10]。传统的协同过滤算法以基于邻域的算法为主，该算法先根据用户对项目的评分数据计算出用户或项目间的相似度，然后找到最近邻的项目，最后产生推荐列表得到项目推荐结果进行推荐。

SVD 在 Netflix 奖的竞争中表现得非常好，该竞赛的目的是仅基于大量匿名用户的之前评级来预测电影的评级。许多机器学习专家参加了 2006 年和 2009 年之间的这场比赛。它提高了使用先进机器学习的推荐系统的研究水平，并改进了推荐系统。即使简单的双线性预测或 SVD 本身并没有赢得比赛，但它是大多数竞争对手提出的整体模型中一个组成部分。

矩阵分解（MF）技术[11]在推荐系统中取得了巨大成功，已经成为研究热点和重点模型，逐渐取代传统的基于邻域（相似用户[12]或相似项目[13]）的协同过滤算法。矩阵分解通过发现最相关的隐式维度，包括显式和隐式反馈信息[14]。基于邻域的协同过滤算法需要计算用户-用户相似性或产品-产品相似性，这需要消耗大量的计算资源。概率矩阵分解[15]将用户（项目）映射到低维空间，并通

过拟合评分信息来估计用户或项目潜在特征向量，即将评分矩阵分解为用户潜在特征矩阵和项目潜在特征矩阵，反过来可以重构评分矩阵。矩阵分解利用降维技术处理高维稀疏数据，来提高预测准确率，同时利用稀疏的评分数据进行模型参数优化，避免模型计算复杂度直接随用户（项目）数增加而快速增长。因此，矩阵分解的预测准确率高和可扩展性好。文献[16]将不同的“置信权重”与正和非观察反馈相关联，然后将所得到的加权矩阵因式分解。而文献[17]将非观察反馈样本作为负实例，并且对相似的加权矩阵进行因子分解，并提出分组的贝叶斯排序模型。文献[18]将所有未选择的项目都作为负例，但设置一个相对小的置信度（权值）这种策略借助于这一权值来控制这些引入的不确定负例的影响。

文献[19]提出贝叶斯个性化排序（Bayesian Personalized Ranking，BPR）模型，该模型是一个通用的优化标准和学习算法的个性化排名。优化准则从问题的贝叶斯分析导出的最大后验估计量。提出了基于具有自举采样的随机梯度下降的通用学习算法。将推荐任务转化为学习排序（Learning to Rank，LTR）问题，该方法可以应用于矩阵分解和自适应 K 近邻的推荐模型中。学习排序方法将排名视为一个简单的二进制分类问题，其中唯一的输入是正面和负面的例子。在这种情况下使用的典型模型包括逻辑回归或梯度提升决策树。成对算法优化在成对偏好上定义的损失函数以最小化反转的数量。

文献[20]利用深度贝叶斯网络分析项目的图片视觉信息，提供了可解释的用户行为隐式反馈分析方法，并且能够缓解冷启动问题，从视觉维度定量的分析用户的偏好，BPR 是该方法的一个很好的例子。列表式方法尝试直接优化整个列表的排名，而不是单个对。例如，RankALS[21]定义了直接包括排名优化，然后使用交替最小二乘优化的目标函数。这些方法使用秩特定的信息检索度量来测量排名模型的性能。理想情况下，直接优化本文的模型。然而，它们不是可区分的，并且不能直接应用标准方法。一些方法如 CLiMF[22]使用目标函数的平滑版本来运行梯度下降。

对矩阵分解的并行优化算法的优化分成两类，一是优化方法本身，另一类是可适用于的分布式计算环境。优化用户潜在特征矩阵以及项目潜在特征矩阵。

在矩阵分解问题中，随机梯度下降（Stochastic Gradient Descent，SGD）优化方法特别有效[6]，该方法实现简单和高效，每一次迭代计算复杂度低。但是 SGD 并行化是一个挑战，因为梯度的更新依赖上一步梯度的值，并且并行化 SGD 会导致重写问题，并行更新时需要更新相同的用户和项目参数。当 R 矩阵很大且很稀疏时，重写问题更加严重，文献[23][24]都是对 SGD 的重写效率等提出的改进方法。文献[24]依赖于共享内存的高效访问，但同时也存在规模限制，如要求共享内存足够完全装载用户和项目特征矩阵；另一类是通用方法，并没有对分布式计算环境有特殊要求，文献[23][25]都基于 MapReduce 框架实现

了其并行优化算法。但 SGD 固有的重写问题，在并行分布式环境下依然不能得到很好地解决。

交替最小二乘法（Alternative Least Sqares，ALS）进行矩阵分解求解时每次迭代计算复杂度较高。ALS 特别适合并行化，当固定 U 后，可以并行计算 V 的每一列，反之亦然。针对 ALS 每次迭代计算复杂度较高的问题，文献[26]提出循环坐标下降法（Cyclic Coordinate Descent，CCD）进行了改进，该方法在固定其他变量的基础上每次只更新一个变量，加速后计算复杂度降低。在此基础上，文献[27]提出 CCD++，该方法通过改变变量更新的序列，进一步降低了计算复杂度。文献[28]实现并行化的 ALS，通过避免耗时的最小二乘计算加速 ALS，每一个维度可以并行地分开处理。但是这些都是考虑在单机环境下的并行加速方法，并没有将分布式计算模型考虑进来，随着数据的规模越来越大，单机的计算已经不能适应对处理数据的时间要求。

多源信息融合的推荐也成为了研究的热点，如文献[29]利用评分数据和评论文本的融合，文献[30]利用评分数据和用户其他信息（社交网络）的融合。文献[31]也利用评分数据和项目类别的融合。随着信息技术的不断发展，推荐系统所面临的推荐场景不在局限于单一用户信息领域和单一物品信息领域。例如，同一个用户可能同时对应着多个社交账号的信息，可能需要对其推荐多种类型的物品，实现信息的跨网站应用。在这种情况下，对于跨平台的信息融合与聚合尤为重要。文献[3]表明神经网络模型在异构信息融合上已经发挥了一定的效果，该方向值得继续深入挖掘。

另外，冷启动问题是推荐系统面临的一大挑战，特别是在基于协同过滤的推荐方法中尤为值得研究。冷启动问题通常可采用混合推荐方法和协同过滤方法、用户人口统计信息如性别、年龄等，以及社会网络关系[30]等来解决。

混合推荐模型始终是研究的热点，不同的混合模型有不同的亮点。单一方法无法解决推荐系统的所有问题，融合推荐模型更有效。混合的形式多种多样，文献[32][33]都是运用内容过滤和协同过滤的混合，文献[34]运用基于项目和基于用户的协同过滤算法的混合，文献[35]使用的是情景敏感和矩阵分解推荐算法的混合，文献[36]是多种情景算法的混合，文献[37]提出了在线和离线方法的混合推荐方法。

利用上下文信息推荐是推荐系统研究的重点之一，推荐系统的上下文信息包括时间，位置，天气等，利用这些信息可以提供推荐系统的推荐精度。间接的信息关联很多上下文信息，有可能有效地改善使用上下文的应用程序。然而，间接和上下文变量表示必须处理经常是噪声的更多数据并且创建更高维度的问题空间。诸如协同竞争过滤（CCF）[38]方法可以通过不仅建模相似用户和项目之间的协同，而且模拟项目之间用于用户关注的竞争来使用上下文信息。文献[39]使

用交互式的方式，根据用户行为动态地适应情景变化，匹配用户最近的兴趣偏好，为用户生成该上下文中最合适的推荐结果。文献[32]则通过设置“情景开关”的方法，解决推荐系统在上下文情景自适应的问题。文献[20]利用可视化的图像信息发现用户的偏好。高斯过程分解机（Gaussian Process Factorization Machines，GPFM）[40]运用非线性的模型解决显式和隐式反馈问题，对于基于上下文的推荐精度有较大的提升。在基于信任的推荐的推荐模型中，根据社会网络构建图分析模型，文献[17]提出分组的关系网，利用用户之间的交互信息。文献[41]使用社交关系来估计用户的产品排名。

训练模型的历史反馈信息主要分为两大类：显式反馈和隐式反馈。显式反馈有评分。隐式反馈有购买记录和评论[42]，订阅，浏览，搜索，鼠标活动等。隐式反馈更容易获得，噪声更少，但是数据来源复杂。已经有许多方法[17][43]可以使用隐式反馈信息。隐式和显式反馈也可以以不同的方式组合，正在进行的结合显式和隐式反馈的研究。例如使用 SVD++和逻辑顺序回归[44]来提供映射，或者采用大规模在线贝叶斯方法[43]。基于隐式反馈进行推荐的难点在于缺乏显式的负例，即明确地知道用户喜欢什么但不清楚用户不喜欢什么。文献[18]将这一类问题定义为 One-Class 协同过滤（OCCF）。文献[16][18]将所有未选择的项目都作为负例，但通过置信度权值策略来控制引入，带来不确定负例的影响。同时，由于无法保证负例中不存在潜在的正例，引入负例会带来噪声。

数据稀疏性问题是推荐系统最常见的的问题。当数据量比较大时，用户涉及的信息量往往很少，且评分矩阵数据极其稀疏，难以找到最近邻用户列表信息，导致推荐效果较差。文献[45]使用分布估计方法来建立用户兴趣模型，计算用户间的相似度，改善了因数据稀疏性而导致的预测精度的问题。

可扩展性问题也是推荐系统面临的常见问题。传统的协同过滤推荐算法研究是单机模式的设计，然而随着用户和项目数目的增加，单机模式的推荐算法逐渐难以满足在线推荐的需求。近年来，分布式协同过滤算法成为推荐算法的研究热点[46]。文献[47]提出基于Hadoop平台的改进聚类协同过滤推荐算法，结合Hadoop分布式计算特点，在离线模式下环境下使用ALS矩阵分解算法对稀疏数据进行预处理，然后对项目特征属性采用项目聚类算法来构建模型，根据聚类模型和相似度计算形成推荐候选空间，最终在线完成推荐。文献[48]研究基于矩阵分解的ALS协同过滤算法的并行化问题，并在Hadoop平台上实现该算法的并行化。文献[49]实现在Spark平台上基于用户、物品的协同过滤和基于ALS模型推荐算法的并行化。文献[50]实现Spark平台的矩阵分解并行化算法，提高了协同过滤推荐算法在大数据规模下的执行效率，并且加速比值达到了线性的结果。文献[51]实现了在Spark上的基于ALS的协同过滤算法并行化运行，并且与Hadoop对比证明了算法在Spark上运行的快速性。这些方法都考虑了海量数据的处理需

求，利用分布式并行编程模型来提高算法的效率。但是这些算法并没有考虑数据的多样性，只考虑数据处理后的计算模型。

2.3 协同过滤结合深度学习的相关研究

近些年来，深度学习发展取得了巨大突破，在多个领域取得了重要进展，包括图像分类和分割、语音识别、自然语言处理等。深度学习为研究人员提供了一种非常有效的技术途径，利用对数据特征进行深层次的抽象挖掘，通过大规模数据来学习有效的特征表示以及复杂映射机制，从而建立起有效的数据模型。深度神经网络成为研究的热点，促进了人工智能领域的快速发展。

深度学习方法结合协同过滤方法成为推荐领域研究的重要方法。深度神经网络可以很大程度上替代特征工程的过程，自动化地完成特征的提取。YouTube 代表了现有最大规模和最复杂的工业推荐系统之一，但依然注重由深度学习带来的性能改进[52]。在基于模型的协同过滤器方法中，将神经网络应用到推荐系统的一些相关工作中。限制波尔兹曼机（RBM）[3]是第一个将神经网络模型应用于推荐系统的工作。然而 RBM 的目标评分预测，而不是 Top-N 推荐，其损失函数仅考虑观察到的评分。将 Top-N 推荐所需的负面抽样纳入 RBM 训练是一项技术上的挑战。矩阵因式分解通过简单的固定函数即作为对于对应的行和列的潜在特征向量的内积来近似矩阵的条目。文献[53]使用一个任意函数替代内积，在学习潜在特征向量的同时从数据中学习，用多层前馈神经网络替换内积，并通过交替优化固定潜在特征的网络和优化固定网络的潜在特征来学习。

虽然稀疏输入受到很少的关注，但仍然是神经网络中一个具有挑战性的问题。文献[54]介绍了一个神经网络架构，从稀疏评级输入计算非线性矩阵分解。文献[55]提出了卷积矩阵分解（ConvMF），一种上下文感知推荐模型，将卷积神经网络（CNN）集成到概率矩阵分解（PMF），以此来捕获文档的上下文信息。基于自动编码器的架构被证明是有前景的一种方法。文献[56]也使用自动编码器的推荐系统，这项工作研究文章推荐的特殊问题，并通过用贝叶斯自动编码器替代其主题模型组件来改进协同主题回归模型[57]，该自动编码器用于学习文章的潜在特征表示。文献[58]提出使用神经自动回归分布估计（Neural Autoregressive Distribution Estimator）来改进上述问题，该方法不需要显式对于二元隐含变量进行推理，减少了模型复杂度，并且使用排序代价函数来进行参数最优化。文献[59]提出协同去噪自动编码器（CDAE）方法，使用去噪自动编码器得到 Top-N 推荐。该模型是一个通用模型，解决了一般的 Top-N 推荐问题，输入是用户行为，而不是项目特征。这两个工作相同的地方都是基于用户的原始评分或者反馈来挖掘深度的数据模式特征。还有一些工作利用深度神经网络

模型当做信息变换模块来引入辅助信息，可以看做是一种同时基于协同过滤和基于内容进行推荐的方法。

文献[60]通过使用适用于输入具有缺失值的数据的损失函数，以及通过并入边信息来增强自动编码器架构。实验表明边信息仅仅略微改善了对所有用户/项目的平均测试误差，但它对冷用户/项目具有更大的影响。文献[61]提出了AutoRec，一种用于协同过滤（CF）的新型自动编码器框架，AutoRec 是紧凑和高效的可训练模型。但只考虑损失函数中观察到的评分，不能保证 Top-N 推荐的性能。在线服务在很大程度上依赖于自动个性化来向大量用户推荐相关内容，这要求系统迅速扩展以适应第一次访问在线服务的新用户流。

文献[62]提出一个基于内容的推荐系统，以解决推荐质量和系统的可扩展性，根据网络浏览历史和搜索查询使用丰富的功能集来表示用户。并使用深度学习方法将用户和项目映射到潜在空间，其中用户及其首选项目之间的相似性被最大化，通过引入多视图深度学习模型来扩展模型，以共同学习来自不同领域和用户特征的项目的特征。

上述工作都是基于用户与物品的点对推荐模式，并没有充分考虑物品的时序关系。文献[63]使用循环神经网络进行基于会话的推荐，该工作是对于 RNN 的一个直接应用。文献[64]使用神经网络模型进行解决出租车位置预测问题，对于多种多层感知器模型以及循环神经网络模型进行对比，发现基于改进后的多层感知器模型取得了最好的效果，比结构化的循环神经网络的效果还要好。文献[65]同时结合 RNN 及其变种 GRU 模型来分别刻画用户运动轨迹的长短期行为模式，通过实验验证，在位置推荐任务中取得了不错的效果。给定一个用户生成的轨迹序列，在预测下一个地点时，直接临近的短期访问背景和较远的长期访问背景都同时被刻画。

目前神经网络模型，特别是深度模型，在推荐系统中的应用还是处于很初步的阶段，比较成功的神经网络模型是基于多层感知器架构进行变型的模型，基于结构化的神经网络模型是重要的研究方向，文献[63][65]是该方向的相关研究基础。这里结构化神经网络主要包括基于序列的循环神经网络或者树结构的递归神经网络。目前推荐系统面临的数据附加信息不断增加，原始的用户物品二维矩阵不能刻画复杂的推荐场景，如基于会话的推荐等。因此，如何在实践中充分挖掘结构化神经网络模型的实战效果将是一个很重要的研究方向。

2.4 Docker 容器技术

虚拟化技术是云计算中的核心技术。传统虚拟化技术是对 OS 的软件和硬件都进行，这也就意味着，虚拟的过程是“重量级”的。近年来，轻量级的虚拟化技术逐渐流行，并成为热门的应用方向，例如 Docker 技术。传统虚拟化是重量

级的虚拟化，对硬件的虚拟，例如 VMware，Openstack 等。容器虚拟化是操作系统内核直接调用硬件，是经量级的。

容器（Container）代表了一个更智能和经济的方式来完成技术转型。容器让企业更快更有效地发展和输送应用。这对于消费者来说这是福音，他们通常会希望应用程序能够更快一些。容器提供了必要的计算资源来运行一个应用，就像它是操作系统中唯一要运行的应用一样。它保证了其他应用容器在同样的机器上运行时不会产生冲突。虽然容器有很多优势，但是首要问题是安全性，它必须被改进以使得容器应用到现实当中。容器这种轻量级的虚拟化技术，对于软件工程领域是重大的利好，极大地提高了软件开发，测试，部署，运维的效率。

如图 2.5 所示，是传统虚拟化技术和容器技术的对比，最大的区别就在于传统的虚拟化技术是虚拟硬件设备，进而在 Hypervisor 层之上使用虚拟的操作系统。而 Docker 容器是借助宿主操作系统的隔离机制，提供运行库的隔离。

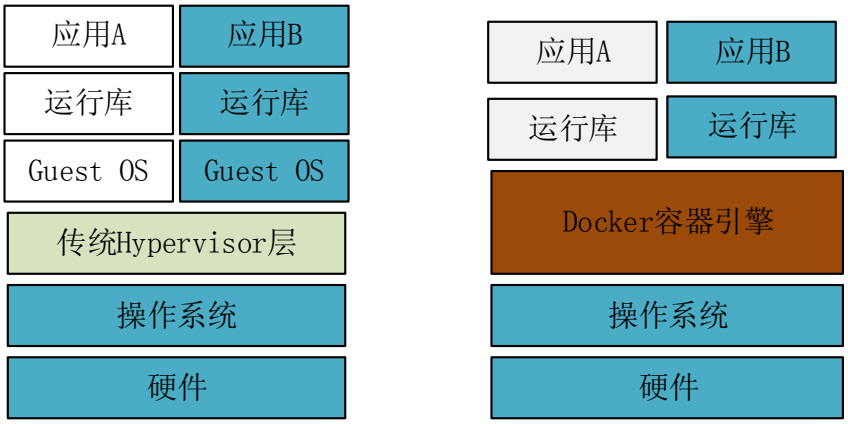


图 2. 5 传统虚拟化技术和容器技术对比

Docker 技术最先在 Linux 操作系统中产生。由 Google 工程师提出，后来被加入到内核中。利用 Linux 内核提供的限制，记录和隔离进程组所使用的资源，通过不同的子系统（blkio，cpu，cpuacct 等）来实现对不同资源使用的控制和记录。进程隔离，每个进程看成是一个操作系统。通过多进程的方式实现分布式的用户空间。

因为操作系统常驻内存，分为用户空间，内核空间和硬件层。资源隔离由 Linux container 提供，利用 namespace 隔离机制，实现容器间的隔离。cgroups 作为资源控制（例如 cpu/memory/网络/文件）。Chroot 提供文件系统的隔离。

2.5 小结

本章重点介绍了国内外对推荐系统的研究现状，并介绍了云计算和大数据领域一些最新的技术，并将这些技术应用到推荐系统的架构中，基于云计算环境平台实现推荐算法和并行模型。协同过滤是推荐系统中的重要算法，广泛应用到各种类型的推荐领域中，并介绍了协同过滤推荐中最重要的矩阵分解技术。

第3章 多源数据融合与矩阵分解算法优化

3.1 引言

在大数据时代，数据源是多样化的并且自然形成。大量数据通常是结构化的，但有时是半结构化或非结构化的。为了实现推荐的准确性，将用户数据梳理后进行挖掘和分析。在这个过程中，多源数据融合是一个不可或缺的步骤，充分利用数据来源的多样性。并结合协同过滤算法来为用户提供推荐服务。

本章将描述数据来源的多样性，并详细阐述数据融合的基本过程。在多种数据来源里面，用户的行为数据分为显式和隐式反馈两类，利用基于矩阵分解的协同过滤算法，可以有效实现对用户的 Top-N 项目推荐。

3.2 数据融合

面向数据融合是以产生决策智能为目标将多种数据源中的相关数据提取、融合、梳理成一个分析数据集。这个分析数据集是个独立的和灵活的实体，可随数据源的变化重组、调整和更新。

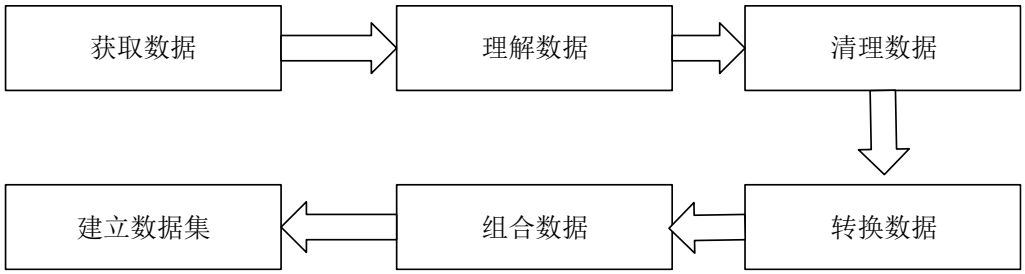


图 3. 1 数据融合有六个基本步骤

如图 3.1 所示，数据融合有六个基本步骤：

1）获取数据；连接所需多源数据库并获取相关数据；

2）理解数据；研究分析和理解所获得的数据；

3）清理数据；梳理数据之间的关系，并清理数据；

4）转换数据；通过数据转换，建立结构关系；

5）组合数据；将不同的数据集合进行组合得到新的数据集；

6）建立数据集；对组合的数据集建立分析数据集。

这个过程的每一步都需要数据工作者认真细致的思考、辨认、测试、清理、最后产生可信赖、有意义的分析数据库。

多源的数据可以归纳为三大类：

1）原始数据（Primary Data），包括企业或组织直接采集掌控的内部运行数

据和营销数据；

2）二级数据（Secondary Data），包括第三者采集、整理、和提供的二手数据；

3）科学数据（Scientific Data），包括科学研究的成果、指数、算法、模型等。

这三类数据为数据为驱动的智能决策提供了不同的观察角度。

3.3 融合显性反馈和隐性反馈分析

利用用户在互联网（包括微信，微博，网站等）访问的数据来追踪和勾画用户偏好行为并以此为据，提供用户潜在感兴趣信息的推荐是大数据应用的重要方向。推荐系统作为智能决策的支持手段之一，在大数据应用和发展过程中还面临巨大的挑战。在这里，多源数据的融合就成为大数据分析中的瓶颈。

融合多源数据以形成有效的用户行为分析数据集仍是推荐系统面对的一个更具挑战性的瓶颈。其中多源数据的融合是最耗费资源的任务之一。如何组织和利用各个来源的数据进行定量的分析，需要对数据的来源和结构有一定的控制和深层的了解，并给出一个特定的模型来完成。

数据融合是以产生决策智能为目标将多种数据源中的相关数据提取、融合、梳理成一个分析数据集。这个分析数据集是个独立的和灵活的实体，可随数据源的变化重组、调整和更新。

基于矩阵分解的协同过滤的方法将用户和项目矩阵中的项视为由用户给予项目的显式偏好，例如，给予电影评分的用户。在许多实际案例中，通常只能访问隐式反馈（例如浏览，点击，评论，购买记录，点赞，喜欢收藏，分享等）。隐反馈数据集中处理隐式反馈信息。本质上讲这种方法并没有直接构建评级的矩阵模型，它只是优先使用的是二进制的数据。相关的评级只是用来观察用户的偏好，而不是明确的评级。然后用模型找到潜在的因素，进而预测用户的偏好。模型中处理这样的数据使用的方法是取自隐式反馈数据集的协同过滤，不是尝试直接对评级矩阵建模，这种方法将数据视为表示用户动作的观察强度的数字（例如点击数，或者观看电影的累积持续时间），这些数字然后与所观察的用户偏好的置信度水平相关，而不是给予项目的明确的评分。然后，模型试图找到可用于预测期望用户对项目的偏好。

3.3.1 显式评分矩阵分解

由用户所提供的自身的偏好数据，包括项目评级，点赞，喜欢等用户对项目的实际评价用户-项目评分矩阵，大部分情况下，单个用户只会和少部分项目接触，矩阵只有少部分的数据是非零的，即矩阵是稀疏的。模型会创建一个用户因

子矩阵和项目因子矩阵，此时用户因子向量和项目因子向量的点积，得到的是预计评分的估值。基于模型的协同过滤，其中用户和项目由一小组潜在因素描述，可用于预测缺失的评分。使用交替最小二乘法（ALS）算法来学习这些潜在因素。由于潜在的因子数目一般较少，所以进行低秩矩阵分解，如图 3.2 所示。

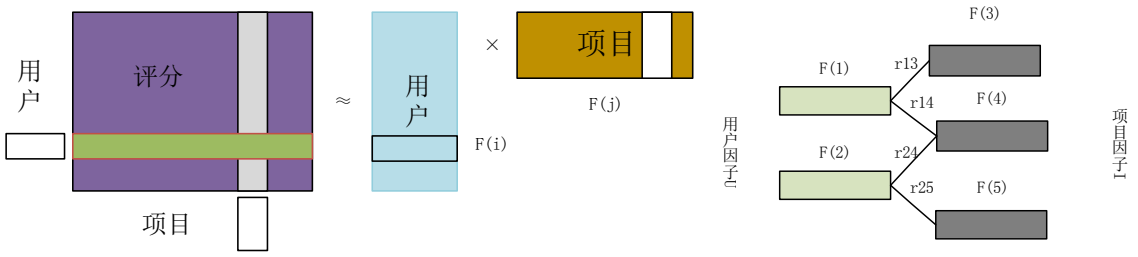


图 3. 2 低秩矩阵分解图

3.3.2 隐式矩阵分解

隐式反馈数据，隐含在用户和项目的交互记录当中，包含二元数据和计数数据等。考虑到多源的情况，将不同来源的隐性反馈数据加权重处理，分包含正反馈而没有负反馈的情况。最终得到评级数据矩阵，并将其视为两个矩阵，一个二元偏好矩阵 P 以及一个信心权重矩阵 C。隐性模型仍然会创建一个用户因子矩阵和项目因子矩阵，但是不同于求解显性评级近似矩阵，模型求解的是偏好矩阵。此时用户因子向量和项目因子向量的点积，所得到的是某个用户对某个项目偏好的估值。

将多种来源的隐式反馈和显式反馈的数据集成在一起，构成多源数据融合的显式和隐式反馈的评级矩阵，并基于显式和隐式反馈矩阵，进行矩阵分解，并构建新的排名模型。本文的模型称为统一显式和隐式反馈模型（UEIFM，Unifying Explicit and Implicit Feedback Model）。为了使推荐模型足够高效和可扩展的大规模数据，在 Spark 分布式并行计算框架中设计和实现 UFIFM。

3.4 统一显式和隐式反馈模型 UEIFM

借鉴矩阵分解应用于隐式反馈应用场景，并将隐式反馈和显式反馈结合在一起，因为在实际系统设计时统一两类反馈信息是必要的。利用矩阵分解模型的优越性，将正例设置为 1，而将负例设置为 0。这样的 0-1 矩阵分解避免了引入噪声，并实现并行优化以处理大规模数据。

根据微博，微信或者网站页面的数据，即点赞，评论，浏览网站，数据库数据和日志数据。建立多源数据融合模型，本文引入数据权重，反映每个数据源的信息质量。假设存在 m 个显式数据源，总体显式权重为 s，第 k 个显式数据权重

为 p_k，如公式（3.2），第 k 个显式分数为 E_k。隐式数据源的数量为 n，隐含权重为 t。第 v 个隐式的数据权重为 q_v，如公式（3.3）。项目的隐含分数是 I_v。得到评级数据矩阵的一个用户 i 对项目 j 的评级值，如公式（3.1）：

$$R_{ij} = s\sum_{k=1}^{m} p_k E_k + t\sum_{v=1}^{n} q_v I_v \tag{3.1}$$

$$p_k = \frac{Number \text{ of } k_{th} \text{ score items of user i}}{Average \text{ number of score items available to the score group}} \tag{3.2}$$

$$q_v = \frac{Number \text{ of data items for the } v_{th} \text{ behavior of user i}}{Average \text{ number of data items available to the behavior group}} \tag{3.3}$$

以上得到的评级矩阵是非常大的而且稀疏的，所以选择坐标矩阵。基于 Spark 平台的弹性分布数据集（Resilient Distributed Dataset，RDD）的编程模型，RDD 的本质是自定义的可并行的数据容器，不同的数据集格式对应不同类型的 RDD。通过 RDD[MatrixEntry] 实例创建分布式矩阵 CoordinateRowMatrix，令每一项为（i: Long, j: Long, value: Double）形式的行列值的元组。其中 i 是行坐标，j 是列坐标，value 是值。把元组变换成 MatrixEntry（long, long, Double）形式，这样方便进行分布式并行计算。

目标函数的优化如公式（3.4）：

$$F(X,Y) = \min_{x_*, y_*} \sum_{u,i} c_{ui}(R_{ui} - x_u^{\mathrm{T}} y_i)^2 + \lambda(\sum_u \| x_u \|^2 + \sum_i \| y_i \|^2) \tag{3.4}$$

c_{ui} 是偏置项，表示可信度，用户评价的权重值项是用户 u 和项目 i 的乘积矩阵 R 的值，R_{ui} 表示用户 u 对项目 i 进行的评价。λ 是正则化参数，其控制正则化的程度，并用于避免过拟合问题。表示用户的特征向量，表示项目的特征向量。由于大量的数据，随机梯度下降法不再适用。而交替最小二乘法（ALS）被设计为找到两个低维矩阵 X（m*k）和矩阵 Y（k*n）。为了逼近近似 R（m*n），本文使用并行 ALS 方法逐步获得值与迭代近似。ALS 矩阵分解优化算法的核心就是通过填充稀疏矩阵方法：

1）假设 m*n 的评分矩阵 R，可以被近似分解成 $U*(V)^T$

2）U 为 m*d 的用户特征向量矩阵

3）V 为 n*d 的项目特征向量矩阵 $(V)^T$ 代表 V 的转置

4）d 为 user/product 的特征值的数量

在该模型中对 r_{ui} 引入喜好变量和置信度变量，分别表示如下喜好变量 p_{ui} 是一个二元变量，表示用户是否具有该无偏偏好，定义如公式（3.5）：

$$f(n) = \begin{cases} 1 \text{ if } r_{ui} \geq 0 \\ 0 \text{ if } r_{ui} = 0 \end{cases} \tag{3.5}$$

置信度变量 c_{ui} 表示用户对项目喜好的置信程度，常用度量如公式（3.6）：

$$c_{ui}=1+\alpha r_{ui} \quad c_{ui}=1+\alpha\log(1+\frac{r}{\varepsilon}) \tag{3.6}$$

最终模型如公式（3.7）：

$$\min\sum_{ui}c_{ui}(R_{ui}-x_u^{T}y_i)^2+\lambda(\sum_u\|x_u\|^2+\sum_i\|y_i\|^2) \tag{3.7}$$

这里需要解决两个问题 1）如何对置信度进行度量 2）如何求解该最优化问题。针对以上两个问题，求解模型：

1）置信度度量方法，可以采用上面提及的两种方法，其中针对第一种方法实验验证当 α 为 40 时效果较好；

2）模型求解，该模型相对于因子分解，需要求解一个稠密矩阵的分解，按照原始 SGD 进行求解时间复杂度较高；

3）利用 ALS 进行求解，对用户维度和项目维度分别进行求解，每一步求解都是一个最优化问题，可以采用正规方程进行求解，如公式（3.8）和公式（3.9）。

$$\frac{\partial F(X,Y)}{\partial X}=0 \tag{3.8}$$

$$X_u=(Y^{T}C_uY+\lambda I)^{-1}Y^{T}C_uR(u) \tag{3.9}$$

得到Y_i的方法与类似X_u，$R(u)$是从步骤（3）得到的评分矩阵的值，如公式（3.10）。

$$R_{ui}=X_u*Y_i^{T} \tag{3.10}$$

算法的步骤如下：

1）计算函数式（3.8）；

2）计算函数式（3.9）；

3）使导数等于 0，然后得到如上所述的值；

4）实施步骤 1），2），3）；

5）交替地确定和向量，并计算另一个，使得损失函数的值减小。重复上述过程，直到值收敛或稳定；

6）函数式（3.10）计算得到R_{ui}。

3.5 基于 Spark 的 UEIFM 并行实现

Spark 平台下的 UEIFM 并行实现是基于 Spark 的编程模型，利用 Spark 的 RDD 编程抽象和矩阵存储方法来实现并行化。

3.5.1 Spark 矩阵存储方法

RDD（Resillient Distributed Dataset）是 Spark 的编程抽象，

CoordinateRowMatrix 是分布式矩阵的一种 matrix。该坐标矩阵也是一种 RDD 存储的分布式矩阵。每一项都是一个（i: Long, j: Long, value: Double）形式的行列值的元组。其中 i 是行坐标，j 是列坐标，value 是值。本文的矩阵是非常大的而且稀疏的，坐标矩阵是最好的选择。坐标矩阵则是通过 RDD[MatrixEntry] 实例创建，MatrixEntry 类的属性是（Long, Long, Double）形式，这样方便进行分布式并行计算。

3.5.2 模型的并行化实现

输入：

F 作为特征数，ITERATIONS 是迭代次数，LAMBDA 正则化系数

隐式和显式用户反馈数据集

输出：

用户特征矩阵和项目特征矩阵

（1）通过 RDD [MatrixEntry] 数据计算 R，表示评分坐标矩阵（DistributedMatrix），因为原始类型是 tuple（long，long，double），转换为 MatrixEntry（Long，Long，Double）。

（2）将 U*F 初始化为 ms，F*M 作为随机矩阵。由于块大小和分区器，可以得到 itemInBlocks 和 userInBlocks。

（3）sparkcontext 对象广播 R 和 ms 和 um 给 Worker 中的 Executor。

（4）逐个更新项目然后用户，固定 M，逐个更新每个用户的每个特征，以使偏导数等于 0；然后逐个固定 U，以更新每个特征 mkj 的每个项，使得偏导数等于 0。然后通过用户特征和项特征来计算 RMSE。

（5）更新用户或项目进程，解决它与 Cholesky 求解器的最小二乘问题与 L2 正则化。

（6）最后，得到 userFacotrs 和 itemFactors 的结果。

如上述代码所示，基于 ALS 的推荐算法主要是模型不断迭代更新的过程，其中固定 U 更新 M，或者固定 M 更新 U，更新过程如下。并且在迭代过程中并行化实现多个用户或者项目隐语义属性同时更新。

3.6 小结

基于矩阵分解的协同过滤推荐方法，能很好地解决推荐问题，相比于基于邻域的协同过滤，矩阵分解模型能从中学习隐含因子，相比于 SVD 算法能更好地运用分布式计算能力，并且最小二乘法能够很好地扩展到大规模的数据集进行训练。将隐式和显式信息综合考虑，得出综合推荐结果。

利用用户在互联网（包括微信，微博，网站等）访问的数据、数据库数据和

日志数据等，统一多种数据资源，提出了统一显式与隐式反馈模型 UEIFM，通过对可观察用户选择行为数据集（隐式反馈数据）和评分（显式数据）的后验估计，将项目推荐问题转化为数据融合和优化问题，以提高推荐质量。并且，在 ALS 基础上进一步提出了基于分布式计算和迭代计算的并行优化模型 UEIFM 并提供了 Spark 的实现，以处理大规模多源数据。实验结果见第五章，以验证提出算法模型 UEIFM 及和并行化实现的有效性和可扩展性。

第4章 结合自动编码器的协同过滤推荐

4.1 引言

虽然深度学习在图像和语音识别方面取得了巨大的成功，但是神经网络在协同过滤中没有得到广泛研究。同时稀疏输入受到较少的关注，并且仍然是神经网络中一个具有挑战性的问题。稀疏输入对于协同过滤是至关重要的，通过协同过滤来处理用户和项目的交互信息。但是传统的协同过滤方法存在冷启动和稀疏性问题，而深度神经网络在学习有效特征表示方面效果显著。有效地结合深度神经网络和协同过滤算法，成为问题研究的关键。

第 3 章介绍了将显式和隐式反馈融合起来建模分析。本章介绍自动编码器结合协同过滤的框架，针对显式的评分和隐式反馈问题，从稀疏的显式评分或隐式反馈输入来计算非线性矩阵分解问题，从而得到项目的 Top-N 推荐。

4.2 自动编码器和稀疏输入

介绍自动编码器，并描述去噪自动编码器和稀疏输入。

4.2.1 去噪自动编码器

自动编码器是一种前馈神经网络，属于无监督的网络，其中网络的输出仅需要重构初始输入。网络被限制使用窄的隐藏层，迫使数据的隐含维数降低。使用重建误差和反向传播上的平方误差损失对网络进行训练。

表 4. 1 符号描述

符号	描述
M	用户的数量
N	项目的数量
d	隐因子的维数
p	用户特征维数
q	项目特征维数
$R \in \mathbb{R}^{m \times n}$	评级矩阵
U	用户隐因子向量
V	项目隐因子向量
W	自编码器 X 的映射函数
P	U 的投影矩阵

使用堆栈自动编码器预训练深层神经网络，该过程使得网络的最低层能够发现低维表示，这样能提高整个网络的质量。但是经典的自动编码器通常退化为恒等网络，且未能学习数据之间的关系。可以通过损坏的输入，推动网络去噪输出来解决这个问题。利用三个过程来损坏数据：

1）高斯噪声：将高斯噪声添加到输入的子集；

2）掩码噪声：小部分的输入随机强制为零；

3）一般噪声：小部分的输入随机强制为输入最大或最小值。

因此，去噪自动编码器（DAE）的损失函数被修改以强调网络的去噪方面。它基于两个主要的超参数α,β。它们平衡了网络是否专注于对输入（α）进行去噪或重建输入（β），如公式（4.1）。

$$L_{2,\alpha,\beta}(x,\tilde{x})=\alpha(\sum_{j\in J(\tilde{x})}[nn(\tilde{\mathrm{x}})_j-x_j]^2)+\beta(\sum_{j\notin J(\tilde{x})}[nn(\tilde{\mathrm{x}})_j-x_j]^2) \tag{4.1}$$

其中$nn(\mathrm{x})_k$是网络的第 k 个输出，x 是 x 的损坏输入，J 是 x 的损坏元素的索引。

4.2.2 稀疏输入

没有使用稀疏向量作为神经网络输入的标准方法。大多数处理稀疏输入方法是通过预计算缺失值的估计来解决。本文通过自动编码器处理这个预测问题。但是缺失值的数量非常低（小于 5%），并且所有缺失值在训练期间是已知的。在协同迭代过程中，输入向量非常稀疏，并且目标向量具有大量的缺失值。

以下小节提供了一个训练框架，来解决稀疏自动编码器训练问题：

1）通过将输入中的值清零来抑制输入图层的边缘；

2）通过将反向传播的值归零来抑制输出层的边缘；

3）使用去噪损失来强调评级预测超过评级重建。

禁止输入边缘的一种方法是将缺失值转为零。为了使自动编码器始终返回零，使用经验损失，忽略未知值的损失变为零。因此，误差反向传播将被传播用于实际的零值，而丢弃它用于丢失值。这样缺失值不会将信息带到网络中。

在当前的问题中使用掩码噪声有两个优点。首先，它作为一个强大的正则化项。其次，它训练自动编码器以预测缺失值。因此，去噪自动编码器（DAE）的损失成为目标函数。为了处理稀疏输入，未知值的错误被丢弃了，如公式（4.2）：

$$L_{2,\alpha,\beta}(x,\tilde{x})=\alpha(\sum_{j\in J(\tilde{x})\cap\kappa(\mathrm{x})}[nn(\tilde{\mathrm{x}})_j-x_j]^2)+\beta(\sum_{j\notin J(\tilde{x})\cap\kappa(\mathrm{x})}[nn(\tilde{\mathrm{x}})_j-x_j]^2) \tag{4.2}$$

其中 K（x）是 x 的已知值的索引。

4.3 自动编码器结合协同过滤矩阵分解

将自动编码器结合基于矩阵分解的协同过滤算法中，考虑稀疏矩阵的学习模型，利用自动编码器框架进行稀疏矩阵分解。

4.3.1 学习模型

用户偏好由评分 R 的稀疏矩阵编码。给定 N 个用户和 M 个项目，评分 r_{ij} 是由第 i 个用户给出的对于第 j 个项目的评级。然后通过稀疏向量 u_i 来描述用户，并且通过稀疏向量 v_j 来表示项目。目标是预测每个缺失评级的估计，即目标是完成用户或项目的稀疏向量。因此，定义两个自动编码器来计算 $\hat{R}$：

1）以稀疏向量 u_i 作为输入并计算稠密向量 u_i' 作为输出的 Uencoder，通过该网络学习用户表示；

2）将稀疏向量 v_j 作为输入并计算密集向量 v_j' 作为输出的 Vencoder，通过该网络学习项目表示。

神经网络实际上计算 R 的低秩近似。如果忽略输出传递函数，则通过最后一层的权重矩阵和第二到最后层的激活的标量积来迭代地构建。例如，给定 Uencoder，V 项表示是最上面的权重矩阵，U 是激活层。Vencoder 的情况相反。关键点是这两个表示之间的连接。用户表示和项表示不在同一空间中。一个表示属于加权矩阵，而另一个表示属于通过先前层的稀疏输入的密集重建。如果比较来自 Uencoder 的激活的 U 表示和来自 Vencoder 权重的 V 表示，则任务变得更复杂，加上非线性使问题求解更加困难。但是自动编码器的输出仍然可以被认为是潜在的非线性因子矩阵分解。

4.3.2 协同自动编码器框架

本节介绍了提出的协同自编码器（CF-AE）框架，其将深度学习模型与基于矩阵分解的协同过滤统一起来。CF-AE 是一种混合模型，它利用评级矩阵和边信息，并将矩阵分解和特征学连接在一起。

给定用户-项目评定矩阵 R，用户侧信息 X 和项目侧面信息 Y，联合分解 R，并且从评级和辅助信息（即 X 和 Y）中学习潜在因子（即 U 和 V）通过公式（4.3）：

$$\arg\min_{U,V} \mathrm{l}(\mathrm{R},\mathrm{U},\mathrm{V})+\beta(\|U\|_F^2+\|V\|_F^2)+\gamma L(X,U)+\delta L(Y,V) \tag{4.3}$$

其中 α,β,δ 是权衡参数。

在 CF-AE 框架中有两个关键组件：

1）用于将评级矩阵 R 分解为两个潜在矩阵的函数 l（R，U，V）；

2）将用户/项目上下文特征与潜在因素连接的函数 L（X，U）和 L（Y，V）。通过矩阵分解导出的第一个分量从评级矩阵中提取潜在知识。使用深度学习模型设计的第二个组件建立了边信息与潜在因素的连接。

4.4 协同自动编码器设计

在设计协同自动编码器的过程中，详细介绍自动编码器的结构、自动编码器的优化方法和自动编码器的可扩展性问题。

4.4.1 自动编码器结构

自动编码器的输入是用户项目评级矩阵 R，用户特征集合 X 和项目特征集合 Y。本文的方法联合分解 R 并从等级和辅助信息（即 X 和 Y）学习潜在因子（即 U 和 V）。特别地，潜在因子从深层网络的隐藏层中提取。自动编码器的特征分解过程，如图 4.1 所示。

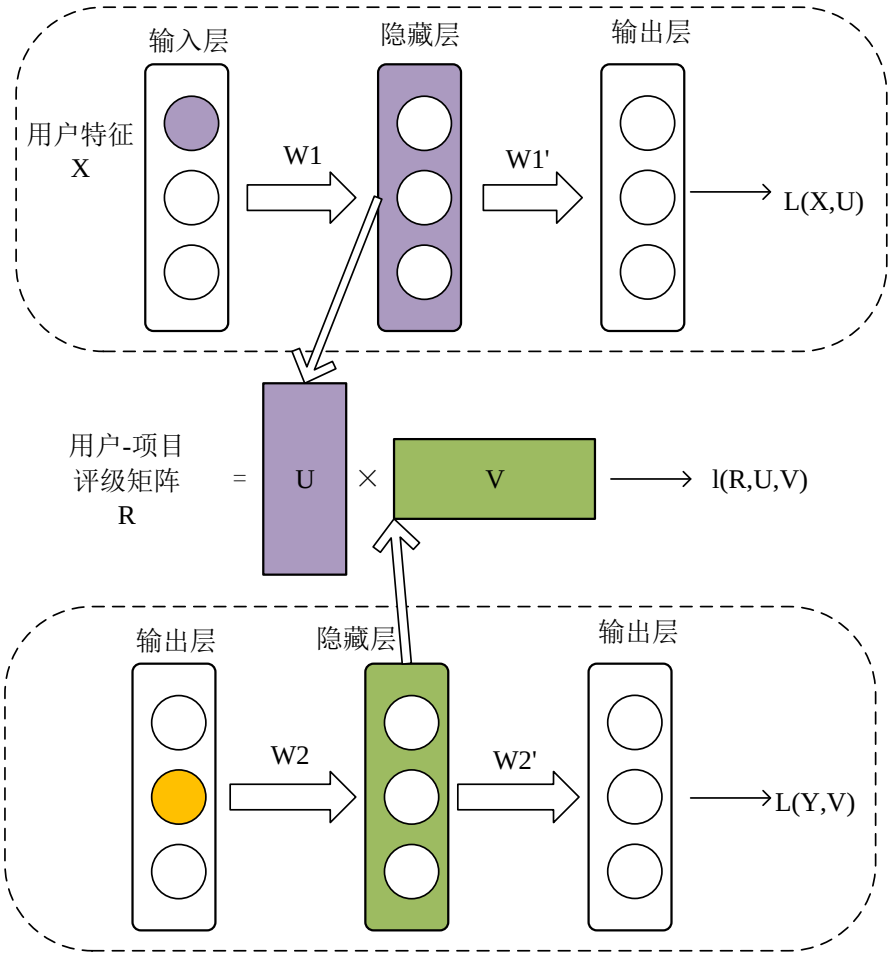


图 4. 1 自动编码器的特征分解

协同自动编码器框架中的目标优化函数可以分解为用户目标函数和项目目标函数，分别如公式（4.4）和公式（4.5）：

$$L(W_1,P_1,U)=\|\bar{X}-W_1\tilde{X}\|_F^2+\lambda\|P_1U^{\mathrm{T}}-W_1X\|_F^2 \tag{4.4}$$

$$L(W_2,P_2,V)=\|\bar{X}-W_2\tilde{X}\|_F^2+\lambda\|P_2V^{\mathrm{T}}-W_2X\|_F^2 \tag{4.5}$$

其中 $W_1\in\mathbb{R}^{p\times p}$ 和 $W_2\in\mathbb{R}^{q\times q}$ 是重建映射，$P_1\in\mathbb{R}^{p\times d}$ 和 $P_2\in\mathbb{R}^{q\times d}$ 是投影矩阵，α,β,δ 是权衡参数。为了简便，设置 α,δ 为 1。

$L(W_1,P_1,U)$ 中的第一项表示去噪自动编码器中的学习过程。它测量输入用户特征 $\bar{X}$ 之间的重建误差和被破坏的输入的映射特征。W_1 是预期将损失最小化的学习映射。第二项连接隐层特征 $W_1\tilde{X}$ 和潜在因子 U。通常，潜在因子具有比原始特征低得多的维度。因此，将潜在因子映射到特征空间的低维投影 P_1。

4.4.2 自动编码器优化

尽管协同自动编码器中的优化问题在所有变量中不是共同凸的，但是当固定其他变量时，它们中的每一个都是凸的。因此，可以交替地优化协同自动编码器框架中的每个变量，详细过程如下。

首先，文献[60][65]推导出一个公式来解决 W_1 和 W_2。通过忽略与 W_1 无关的变量，协同自动编码器框架的目标函数定义为公式（4.6）：

$$\arg\min_{W_1}=\|\bar{X}-W_1\tilde{X}\|_F^2+\lambda\|P_1U^{\mathrm{T}}-W_1X\|_F^2 \tag{4.6}$$

考虑无数的噪声数据副本，并获得最优解公式（4.7）：

$$W_1=E[S_1]E[Q_1]^{-1} \tag{4.7}$$

其中，$S_1=\bar{X}\tilde{X}^{\mathrm{T}}+\lambda P_1U^{\mathrm{T}}X^{\mathrm{T}}$ 和 $Q_1=\bar{X}\tilde{X}^{\mathrm{T}}+\lambda XX^{\mathrm{T}}$ 用于计算期望 $E[S_1]$ 和 $E[Q_1]$ 的有效求解器如文献[65]。

接下来，通过删除不相关变量 w.r.t. P_1，目标函数变为公式（4.8）：

$$\arg\min_{P_1}=\lambda\|P_1U^{\mathrm{T}}-W_1X\|_F^2 \tag{4.8}$$

可得到封闭的解，如公式（4.9）：

$$P_1=W_1XU(U^{\mathrm{T}}U)^{-1} \tag{4.9}$$

类似地，P_2 的最优解如公式（4.10）：

$$P_2=W_2YV(V^{\mathrm{T}}V)^{-1} \tag{4.10}$$

为了解决潜在因子 U 和 V，使用流行的随机梯度下降（SGD）算法。特别地，当与 U 和 V 无关的其他变量是固定的时，使用 $f(U,V)$ 来表示协同自动编码器中的目标。更新规则为公式（4.11）和公式（4.12）：

$$u_i=u_i-\eta\frac{\partial}{\partial u_i}f(U,V) \tag{4.11}$$

$$v_j=v_j-\eta\frac{\partial}{\partial v_j}f(U,V) \tag{4.12}$$

其中η是学习率，详细的导数定义为公式（4.13）和公式（4.14）：

$$\frac{\partial f(U,V)}{\partial u_i} = \lambda(P_1^{\mathrm{T}} u_i - (W_1 X)_i)) + \beta u_i - \alpha \sum_{(i,j)\in} (R_{ij} - u_i v_j^{\mathrm{T}}) v_j \tag{4.13}$$

$$\frac{\partial f(U,V)}{\partial v_j} = \lambda(P_2^{\mathrm{T}} v_j - (W_2 Y)_j)) + \beta v_j - \alpha \sum_{(i,j)\in} (R_{ij} - u_i v_j^{\mathrm{T}}) u_i \tag{4.14}$$

重复上述步骤直到收敛。最后，获得潜在因子用户特征矩阵 U 和项目特征矩阵 V。

4.4.3 可扩展性问题

大多数协同过滤必须解决的一个主要问题是可扩展性，因为数据集通常有成千上万的用户和项目。有效的算法必须在合理的时间内训练，并在评估期间提供快速反馈。GPU 计算的最新进展设法将神经网络的训练时间减少几个数量级。然而，协同过滤处理稀疏数据，GPU 被设计为在密集数据上执行良好。本文在深度学习框架 Tensorflow 中实现了该算法，本文在实验上展示了在 Movielens 数据集上的实验效果并同常见的协同过滤算法进行比较，见第五章实验部分。

本文提出了一种新的基于模型的协同过滤方法，称为协同自动编码器（CF-AE），用于 Top-N 推荐。CF-AE 假定观察到的任何用户项目交互都是用户的全部优选集的损坏版本。模型学习了可以最好地重建完整输入的用户项目偏好的潜在表示。在训练期间，将模型反馈给用户的项目集合的子集，并训练模型以恢复整个项目集合。在预测时，模型在给定现有偏好设置为输入的情况下向用户推荐新项目。对损坏的数据的训练有效地恢复共同偏好模式，表明这是一种有效的协同过滤方法。

本文遵循典型的 Top-N 推荐设置，并且仅考虑本文中的用户项目交互数据。CF-AE 概括了几个先前提出的协同过滤模型，但它的结构更灵活。它将非线性结合到模型中以实现更好的 Top-N 推荐结果。本文评估各种参数选择对模型组件的影响，并将其性能与先前的方法在真实数据集上进行比较。

本文的贡献可以总结如下：

1）提出了一个新的模型 CF-AE，它使用自动编码器框架制定 Top-N 推荐问题，并从隐性数据的输入中学习。与相关方法相比，CF-AE 在模型定义和目标函数中都是新颖的。

2）证明 CF-AE 是几个常见方法的泛化，但具有更灵活的结构。

3）进行深入的实验研究 CF-AE 中不同组件的选择的影响，实验结果表明，CF-AE 在一些共同的评估指标上有显着的优势，始终优于常见的 Top-N 推荐方法。

4.5 小结

本章详细介绍了结合自动编码器的协同过滤框架 CF-AE。首先明确了稀疏输入对于协同过滤是至关重要的，然后指出被学习的用户和项目表示之间的复杂关系。介绍了去噪自动编码器，由于自动编码器与协同过滤中的低秩矩阵因子分解是密切相关的，进而引出协同自动编码器架构。然后解释如何在其顶部添加额外的约束，以使网络处理稀疏输入。最后，详细介绍了完整的设计和训练过程。并且说明架构的可扩展性问题。

第5章 实验设计与评估

5.1 引言

本文的第 3 章和第 4 章详细介绍了 UEIFM 模型和协同自动编码器的设计过程，在第 3 章，重点介绍了融合多源数据的 UEIFM 模型以及基于矩阵分解算法的优化。在第 4 章，重点介绍了结合自动编码器的协同过滤推荐以及详细的设计架构。为了验证上述两章中模型和算法的有效性，本章对 UEIFM 模型和 CF-AE 框架进行分析和验证。为了体现模型泛化能力，在多个数据集中进行实验。基于 Spark 和 Tensorflow 深度学习平台来完成，制定了一系列已有的公认评价标准，来进行实验结果的评估。并和其他的 Top-N 推荐模型进行比较，来体现本文提出的模型的优越性。

5.2 实验平台

本文使用实验室中的 HP 工作站计算机作为实验机器。该计算机配置为 55G 内存，12 核 CPU。使用轻量级的虚拟化技术 Docker，然后在 Ubuntu 主机操作系统上创建容器。最后，完成部署 5 节点的 Spark 集群的准备。使用 HDFS（Hadoop 分布式文件系统）存储实验数据，并在没在节点上安装好深度学习库 tensorflow。并使用虚拟网络适配器 docker0 建立网络连接。这样构建了分布式并行计算的云环境平台。集群实验环境如图 5.1 所示。

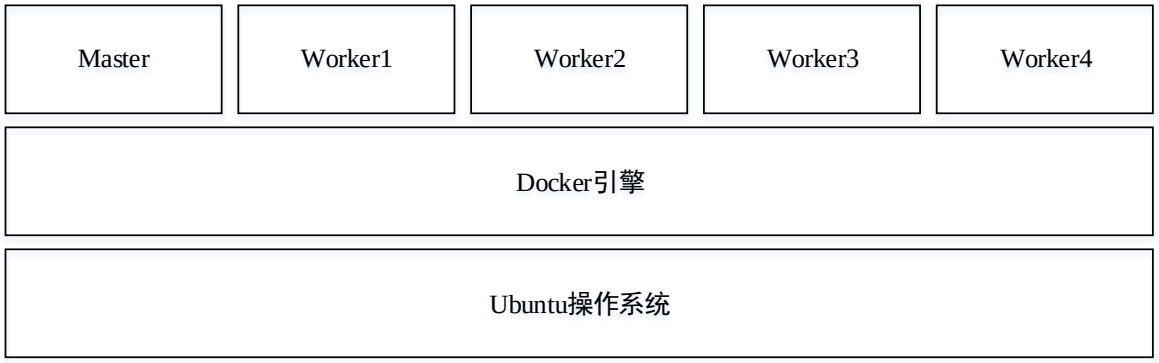


图 5. 1 基于 Docker 的 Spark 集群实验环境

5.3 实验数据集

本文使用 Movielens[66]作为实验数据集来模拟显式评级和隐式反馈数据，利用该评分数据的目的来模拟显式评分和隐式反馈数据，构成为隐式评级。当计算得分时，包括 Movielens-1m，Movielens-10m，Movielens-20m。例如，

Movielens-1m 包含来自 3883 部电影和 6040 用户的 1000209 评级。用户号码从 1 到 6040，电影号码从 1 到 3952，但实际上只有 2706 部电影被评级。单个用户至少评分 20 部电影，评级密度为 4.47%。Movielens 20m 数据集包含 20000263 个评分，27278 部电影和 138493 个用户。用户编号 1-138493，电影编号 1-131262。用户至少参与了 20 部电影的评分。根据统计，用户最多参与了 9254 部电影的评分。而实际只有 26744 部电影被评分，评分密度为 0.54%。详细信息如表 5.1 所示。

表 5. 1 Movielens 数据集详细信息

数据集	用户数量	项目数量(实际)	最小	最大	评分数量	密度
Movielens-1m	6040	3883(3706)	1	5	1000209	4.47%
Movielens-10m	69878	10681(10677)	0.5	5.0	10000054	1.34%
Movielens-20m	138493	27278(26744)	0.5	5.0	20000263	0.54%

Movielens 数据集的总数为 31000526。经过计算后，评分的分布如表 5.2 所示。可以看到，评分 4 是最多的，第二是评分 3。较低的评分比远低于较高的评分。实验时，使用 K 层交叉检验（5-fold crossvalidation），将数据集随机 8/2 分，即将 80%作为训练集，20%作为测试集。

表 5. 2 评分分布

评分	评分数量	比率
0.5	334113	1.08%
1	1121086	3.62%
1.5	397530	1.28%
2	2328860	7.51%
2.5	1253576	4.04%
3	6909066	22.29%
3.5	3079920	9.94%
4	8786747	28.34%
4.5	2119846	6.84%
5	4669782	15.06%

5.4 实验评估

对于 Top-N 推荐的推荐列表的评价指标有准确率（Precision），召回率（Recall），F1 指标（综合准确率和召回率），曲线下面积（Area Under the ROC curve，AUC），折扣累计利润（Normalized Discounted Cumulative Gain，NDCG）等。准确率（Mean Average Precison，MAP）或点击率预估

（Click Through Rate，CTR）也是常用的评价方法；当用于评分预测问题时，均方根误差（Root Mean Squared Error，RMSE）或平均绝对误差（Mean Absolute Error，MAE）是常见评价指标。

令 $R(u)$ 为评级模型（融合用户的显性反馈和隐性反馈信息）根据用户在训练集上的行为给用户作出的推荐列表，而 $T(u)$ 是用户在测试集上的行为列表。推荐结果的准确率定义为公式（5.1）：

$$\text{Precision} = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |R(u)|} \tag{5.1}$$

推荐结果的召回率定义为公式（5.2）：

$$\text{Recall} = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |T(u)|} \tag{5.2}$$

AUC 定义为公式（5.3）：

$$AUC = \frac{1}{|\mathrm{U}|} \sum_{u \in U} AUC(u) \tag{5.3}$$

在特定秩位置 p 处累积的折扣 CG 被定义为公式（5.4）：

$$DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{5.4}$$

然后，NDCG 的计算函数为公式（5.5）：

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{5.5}$$

5.5 多源数据融合模型与算法优化实验

统一显式和隐式反馈模型（UEIFM）融合显式和隐式反馈后，训练推荐模型，得到的是 Top-N 推荐。Top-N 推荐的准确率评估一般是通过准确率，召回率来度量。所以本文主要计算准确率，召回率，加上几个额外的辅助评价指标。最后，评估参数 P@N 和 R@N，还有 AUC，评价准确率 MAP 和 NDCG。为了评测 Top-N 推荐的准确率和召回率权衡的结果，在推荐列表长度为 N 时，计算出准确率/召回率，最后画出准确率/召回率曲线。

将本文的模型和 ItemKNN，UserKNN，WRMF 和 BPRMF 进行比较，计算 AUC，Precision，Recall，MAP 和 NDCG，结果如表 5.3 所示。AUC 的值好于 WRMF，因为本文的模型就是基于 ALS 进行改进的，稍小于 BPRMF 模型的 AUC。在准确率和召回率上，准确率无论是 prec@5 和 prec@10 都好于 WFMF, 在召回率的比较中，对比 WRMF 和 BPRMF 都有明显的减小，但是 MAP 很小

的原因是，综合多个实验数据集之后，数据有重叠的部分，减小了显式和隐式反馈在各自反馈的增强表现。NDCG 的方法叠加了在 Top-5 和 Top-10 的计算结果，所以比 WRMF 和 BPRMF 更好。

表 5. 3 实验结果，包括 AUC, Precision, Recall, MAP, NDCG

模型	AUC	P@5	P@10	MAP	R@5	R@10	NDCG
ItemKNN	0.9289	0.3140	0.2792	0.2226	0.0962	0.1673	0.5632
UserKNN	0.9295	0.3974	0.3335	0.2681	0.1380	0.2178	0.6105
WRMF	0.9558	0.3657	0.3097	0.2303	0.1159	0.1852	0.5672
BPRMF	0.9717	0.3036	0.2629	0.1964	0.0932	0.1556	0.5391
UEIFM	0.9610	0.3660	0.3435	0.1172	0.0569	0.1042	0.7296

表 5. 4 在不同的 N 值下 Precision，Recall 的实验结果

模型		N=5	N=10	N=15	N=20
UEIFM	precision	0.1245	0.1201	0.1165	0.1131
	recall	0.0475	0.0892	0.1274	0.1620
	NDCG	0.1267	0.1338	0.1449	0.1566
	precision/recall	2.6234	1.3461	0.9140	0.6983
WRALS	precision	0.0903	0.0854	0.0813	0.0779
	recall	0.0576	0.1073	0.1499	0.1884
	NDCG	0.0975	0.1109	0.1247	0.1377
	precision/recall	1.5676	0.7966	0.5423	0.4133

下面是评测 TopN 推荐的准确率和召回率权衡的结果，在推荐列表列表长度时，计算出 Top-N 推荐的准确率/召回率，然后得到准确率/召回率曲线图。在不同的 N 值下 Precision，Recall 的实验结果，如表 5.4 所示。实验结果如图 5.2 所示。

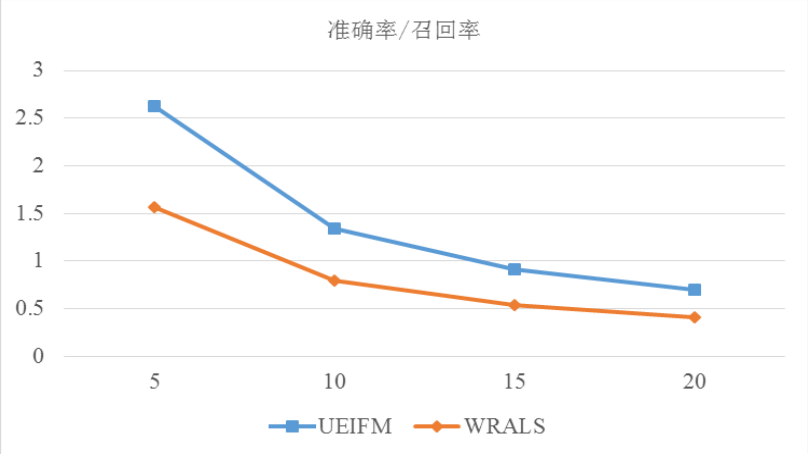


图 5. 2 准确率/召回率曲线图

5.6 结合自动编码器的协同过滤推荐实验

在进行协同自动编码器实验时，需要先定义基准测试模型，并且设置 CF-AE 模型的参数。首先对 Movielens 实验数据集（Movielens-1m，Movielens-10m，Movielens-20m）进行数据预处理。然后，对比其他推荐模型，完成实验对比过程。最后进行比较总结。

5.6.1 参数设置

本节训练四层自动编码器。第一编码层是输入大小的 10 倍，第二层是输入大小的 10 倍。超参数的设置如表 5.5 所示。解码层以相反的顺序具有相同的维度。使用扇入规则来初始化权重。网络是以 minibatch 为 20 的随机反向传播的方式来优化。去噪自动编码器的损失和稀疏需要强正则化，并且设置权重衰减。使用堆叠自动编码器进行训练。

表 5. 5 训练的超参数

超参数	外层	中间层	第 4 层去噪自动编码器
学习率	0.01	0.001	0.001
学习衰减率	0.1	0.1	0.1
权重衰减率	0.01	0.02	0.03
α	1	1	1
β	0.8	1	0.8
高斯噪声	0.1	0	0.15
掩码噪声	0.05	0.1	0.05

以下参数根据经验设置以提供良好的结果。用相同的参数在 Movielens 数据集上进行实验。

3 个 Movielens 的数据集分别是：Movielens-1M、Movielens-10M、Movielens-20M。对于每个数据集，保持那些与评级不低于 4 星，并将所有其他评级作为缺失条目。保留的那些评级被转换为 1。对 Movielens-1m 数据集进行数据预处理，过滤得到评分大于 4 和大于 5 的数据集，分别是 575281 和 226310。

该处理方法广泛用于以前的具有隐式反馈的推荐工作[14]。删除所有评分低于 5 的用户和项目。对于每个用户，随机保留测试集中的 20%的评分，并将其他评分放在训练集中。结果数据集的统计数据如表 5.6 所示。

表 5. 6 预处理后的 Movielens 数据集

数据集	用户数量	物品数量	总数量	评分数目	比率(%)
Movielens-1m	6040	3706	1m	226310	22.62
Movielens-10m	69878	10677	10m	1544812	15.45
Movielens-20m	138493	26744	20m	2898660	14.49

通过对训练数据集执行 5 折交叉验证，为所有的模型选择最佳超参数，然后使用最佳超参数在整个训练数据集上训练模型。

使用随机梯度下降（SGD）来学习所提出的方法，设置α=1 并尝试不同的步长β={1；0.1；0.01；0.001}，并获得每个模型的最佳结果。

5.6.2 评估指标

在进行 Top-N 项目推荐时，向每个用户呈现具有最高预测值，但在训练数据中推荐未被用户采用的 N 个项目。基于用户实际在测试数据中采用哪些项目来评估不同的方法。通过对所有用户的准确率和召回率进行平均来计算整个推荐系统的准确率和召回率。平均准确率（MAP）。平均准确率（AP）是一个排名的精度度量，给予正确推荐的排名最高的项目更大的值。准确率（MAP@N）被定义为所有用户的 AP 分数的平均值。

这些评估指标如果一个模型在一个度量上的性能优于另一个模型，则更有可能在另一个评估指标上产生更好的结果。重点关注 MAP@N 的评估结果，其中 N = {1，5，10}。

CF-AE 模型的主要组成部分包括映射函数的类型，损失函数。这些组件的不同选择导致模型的不同变体，其产生不同的 Top-N 推荐。对于映射函数，本文显示隐藏层和输出层上的恒等式函数和 sigmoid 函数的结果。（tanh 函数的结果与 sigmoid 函数的结果类似）。对于映射函数（在两个层上）和损失函数，存在 2^3 = 8 个组合选择。其中，logistic 损失函数需要值在 0 和 1 之间，因此它必须与输出层上的 sigmoid 函数相关联。最后说明，sigmoid 函数和 logistic 损失的组合等效于交叉熵损失。

在实验中，观察到成对目标函数没有比 CF-AE 的逐点目标更好。一个可能的原因是对于具有二元评分的隐式反馈，逐点损失函数足以分离由用户优选的那些项和不优选的那些项。换句话说，良好设计的逐点损失可以足够区分以建模用户在这些数据集中的偏好，并且不需要成对损失。本文省略了成对目标函数的结果。使用成对损失的 BPR 模型不能比使用逐点损失的 MF 更好。这可能是由于与 CF-AE 相同的原因。BPR 被设计为优化 AUC，而不是 Top-N 度量，例如准确率，召回率或平均准确率。因此，针对 Top-N 推荐的多个模型，成对损失函数对于所有的数据集可能并不是最好的。实验表明，最佳模型取决于数据集，应该根据数据映射函数、目标函数、损失函数的选择来确定。

潜在维度的数量。研究潜在维数 K 的数量的影响。在 Movielens 数据集上 1m，10m，20m 的实验结果，其中 K = {5，15，25，35，45，55}。在进行低秩矩阵分解时，K 的取值并不是越大越好，K 过小可能导致数据过拟合，K 过大可能导致欠拟合。所以需要观察 K 的取值来选择最好的结果。

5.6.3 模型比较

本节将 CF-AE 与许多流行的 Top-N 推荐方法进行比较。对比算法是：

1）POP：根据多少用户对其评分来推荐项目。

2）ITEM-CF[12]：使用 Jaccard 相似性度量，并将最近邻的数量设置为 50。

3）WR-MF[16]：目标函数对观察/正反馈分配高置信度，对缺失/负反馈分配低置信度，训练潜点因子模型与点对点目标函数是平方损失。计算加速技巧 ALS。实验表明 Log，Hinge 和交叉熵损失效果都不佳。

4）BPR-MF[14]：BPR 是基于隐式反馈的推荐方法。BPR-MF 具有成对对数损失函数的潜在因子模型。实验表明平方和铰链损失函数效果不佳。

对于所有比较方法，通过交叉验证仔细选择超参数，以确保公平比较。对于所有潜在因素模型（包括 WR-MF，BPR-MF 和 CF-AE），将潜在维度的数量设置为 5，15，25，35，45，55。

将协同自编码器模型 CF-AE 和 ItemPop（Pop），ItemCF，WR-MF 和 BPR-MF 进行比较，计算在 N=1，5，10 时的 Precision，Recall，MAP，在调整实验参数，得到最优的实验结果。如表 5.7，表 5.8 和表 5.9 所示，分别是在 Movielens-1m，Movielens-10m，Movielens-20m 数据集中的对比实验结果，在 Movielens-1m 中效果平均准确率最好的，WR-MF 最好；在 Movielens-1m 和 Movielens-20m 中，均好于其他模型。表明了 CF-AE 模型在数据量增大，数据稀疏性高的时候，准确性是好于其他模型的，并且不会出现过拟合问题。

表 5. 7 Movielens-1m 的对比实验结果

模型	P@1	P@5	P@10	R@1	R@5	R@10	MAP@5	MAP@10
Pop	0.0939	0.0808	0.0709	0.0173	0.0625	0.1162	0.0589	0.0541
Item-CF	0.1944	0.1324	0.1051	0.0353	0.1146	0.1770	0.1078	0.0967
WR-MF	0.2095	0.1546	0.1248	0.0382	0.1341	0.2103	0.1249	0.1141
BPR-MF	0.2167	0.1486	0.1196	0.0383	0.1247	0.1962	0.1210	0.1088
CF-AE	**0.2196**	**0.1514**	**0.1202**	**0.0401**	**0.1302**	**0.1994**	**0.1235**	**0.1116**

图 5.3-图 5.10 是 Movielens-1m 下各模型参数在准确率、召回率和平均准确率的详细对比图。

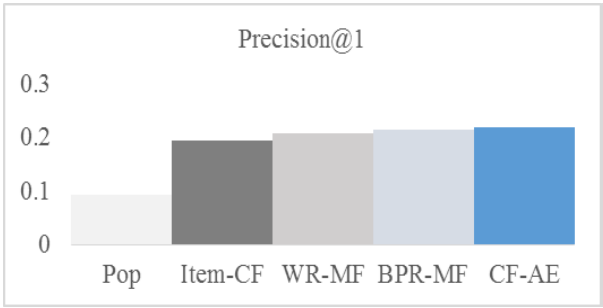


图 5. 3 Movielens-1m 下 Precision@1 对比

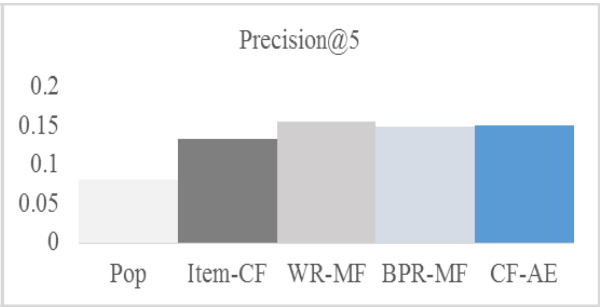


图 5. 4 Movielens-1m 下 Precision@5 对比

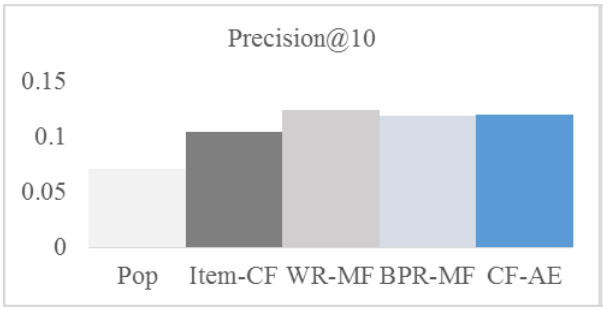


图 5. 5 Movielens-1m 下 Precision@10 对比

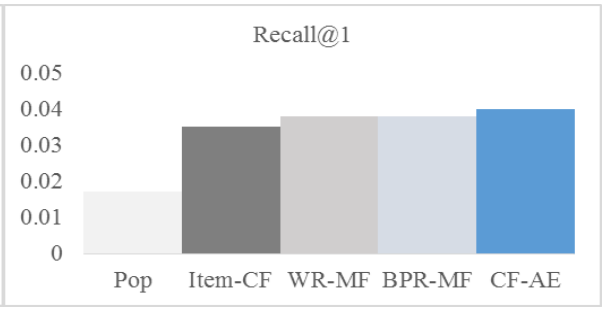


图 5. 6 Movielens-1m 下 Recall@1 对比

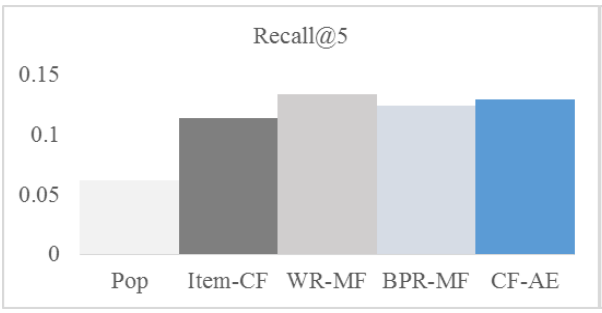


图 5. 7 Movielens-1m 下 Recall@5 对比

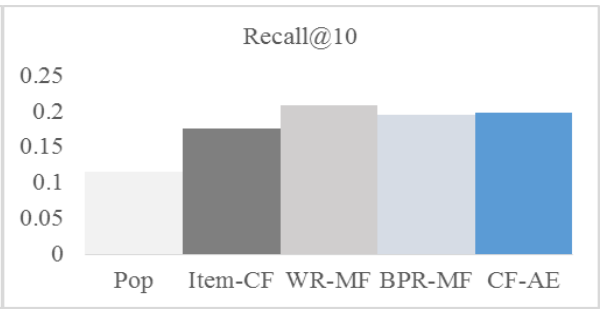


图 5. 8 Movielens-1m 下 Recall@10 对比

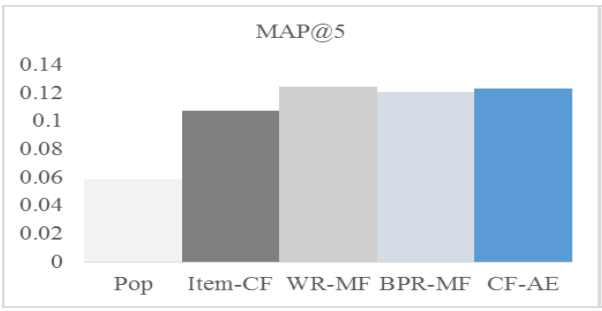


图 5. 9 Movielens-1m 下 MAP@5 对比

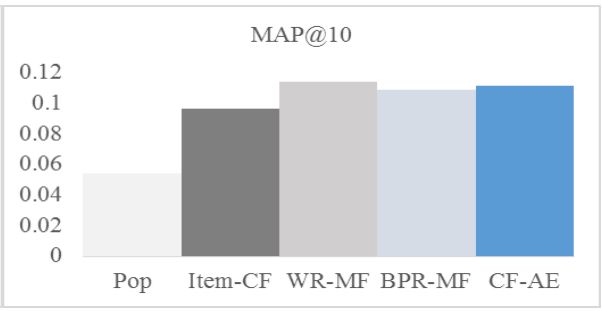


图 5. 10 Movielens-1m 下 MAP@10 对比

如表 5.8 所示，在 Movielens-10m 下各模型的实验结果数据表。其中 CF-AE 实验效果的最好的。在准确率和召回率方面均优于其他模型，WR-MF 实验结果好于 WR-MF。实验表明，WR-MF 和 BPR-MF 在迭代过程中会出现过拟合现象，并不会随着迭代次数的增加而准确率变大。但对于 CF-AE 并没有出现，迭代次数在 50 范围内，准确率都是提升的。

表 5. 8 Movielens-10m 的对比实验结果

模型	P@1	P@5	P@10	R@1	R@5	R@10	MAP@5	MAP@10
Pop	0.0777	0.0584	0.0486	0.0247	0.0852	0.1332	0.0566	0.0581
Item-CF	0.1678	0.1080	0.0817	0.0536	0.1571	0.2278	0.1157	0.1159
WR-MF	0.1509	0.1166	0.0949	0.0454	0.1612	0.2497	0.1140	0.1173
BPR-MF	0.1431	0.0986	0.0799	0.0421	0.1355	0.2129	0.0986	0.1005
CF-AE	**0.1763**	**0.1199**	**0.0943**	**0.0525**	**0.1645**	**0.2491**	**0.1216**	**0.1231**

图 5.11-图 5.18 是 Movielens-10m 下各模型参数在准确率、召回率和平均准确率的详细对比图。如图 5.11 所示，CF-AE 模型的准确率稍好于 WR-MF，Item-CF 的效果也很好，WR-MF 次之。总体来看，CF-AE 在准确率、召回率和平均准确率方面都有一点提高。但相比 Movielens-1m 的准确率是降低的。

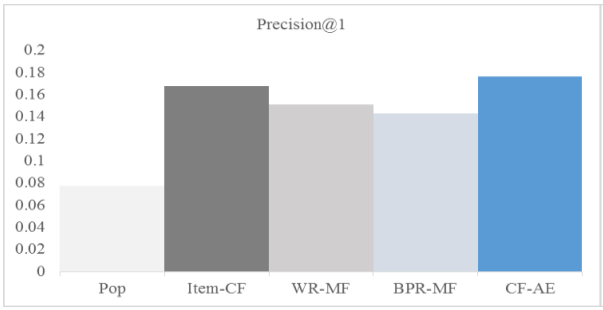


图 5. 11 Movielens-10m Precision@1 对比

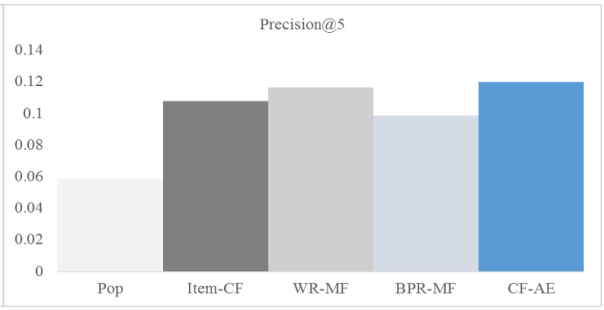


图 5. 12 Movielens-10m Precision@5 对比

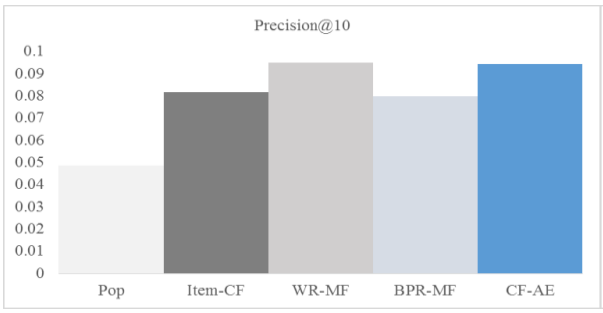


图 5. 13 Movielens-10m Precision@10 对比

图 5. 14 Movielens-10m Recall@1 对比

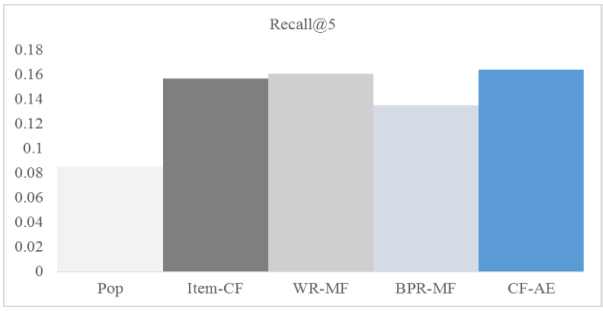


图 5. 15 Movielens-10m Precision@5 对比

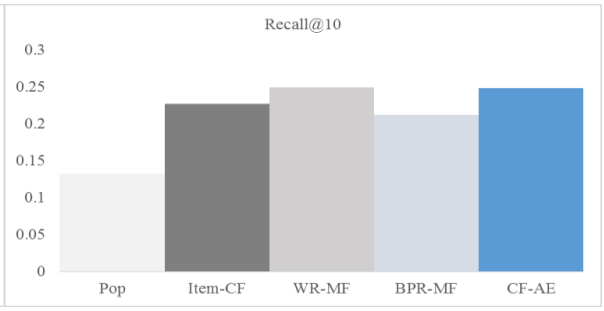


图 5. 16 Movielens-10m Recall@10 对比

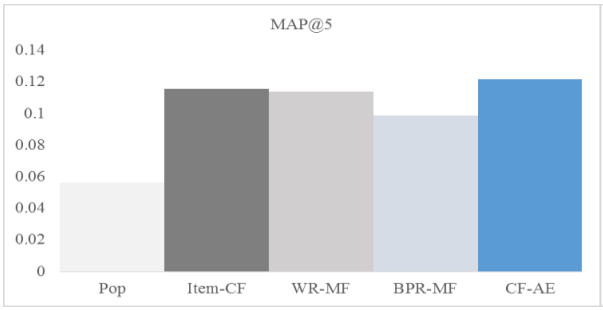


图 5. 17 Movielens-10m MAP@5 对比

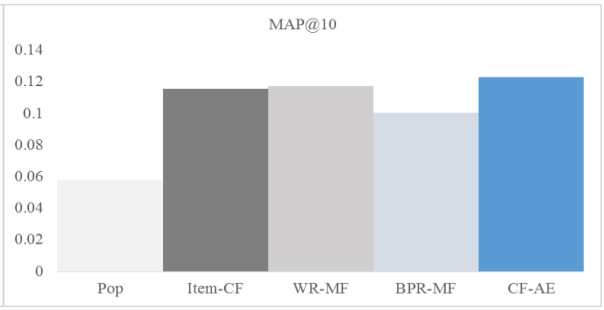


图 5. 18 Movielens-10m MAP@10 对比

如表 5.9 所示，在 Movielens-20m 数据集中不同模型的准确率，召回率和评价准确率的实验结果比较。CF-AE 是实验结果最好的模型。在准确率上比其他模型有明显的提升。在准确率和召回率方面均优于其他模型，WR-MF 实验结果好于 WR-MF。实验表明，WR-MF 和 BPR-MF 在迭代过程中也会出现过拟合现象，并不会随着迭代次数的增加而准确率变大。但对于 CF-AE 并没有出现，迭代次数在 50 范围内，准确率都是提升的。如图 5.19 所示，CF-AE 模型的准确率远好于其他模型，BPR-MF 次之。但相比 Movielens-1m 和 Movielens-10m

的准确率是降低的。

表 5. 9 Movielens-20m 的对比实验结果

模型	P@1	P@5	P@10	R@1	R@5	R@10	MAP@5	MAP@10
Pop	0.0779	0.0551	0.0462	0.0249	0.0785	0.1255	0.0540	0.0557
Item-CF	0.1026	0.0705	0.0602	0.0353	0.1051	0.1324	0.0786	0.0805
WR-MF	0.1251	0.0875	0.0709	0.0359	0.1204	0.1889	0.0858	0.0872
BPR-MF	0.1309	0.1032	0.0866	0.0401	0.1439	0.2297	0.0993	0.1040
CF-AE	0.1620	0.1117	0.0889	0.0467	0.1502	0.2333	0.1103	0.1122

图 5.19-图 5.26 是 Movielens-20m 下各模型参数在准确率、召回率和平均准确率的详细对比图。总体来看，CF-AE 在准确率和平均准确率方面对比其他模型都有一点提高。在召回率方面相比其他模型，提高不明显。

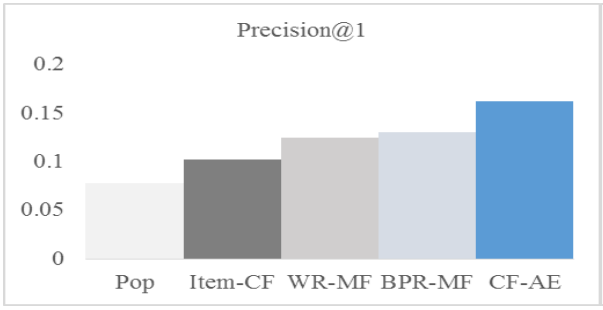


图 5. 19 Movielens-20m Precision@1 对比

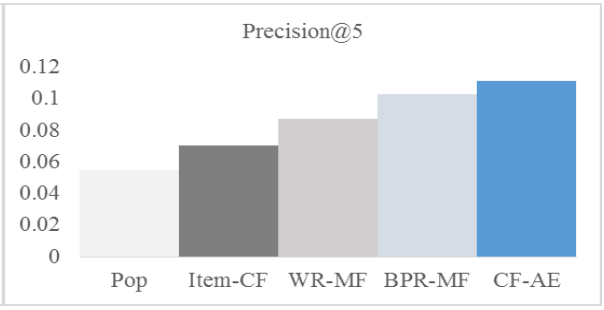


图 5. 20 ovielens-20m Precision@5 对比

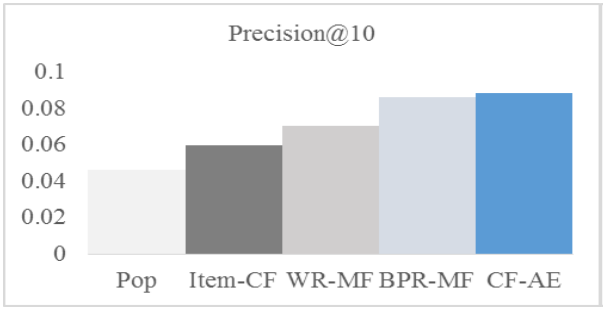


图 5. 21 Movielens-20m Precision@10 对比

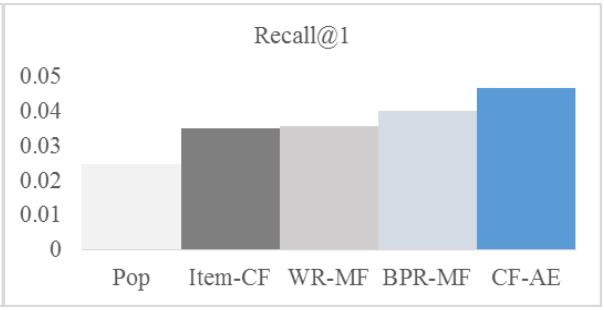


图 5. 22 Movielens-20m Recall@1 对比

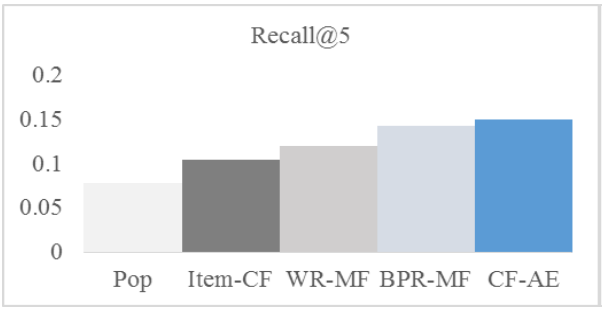


图 5. 23 Movielens-20m Recall@5 对比

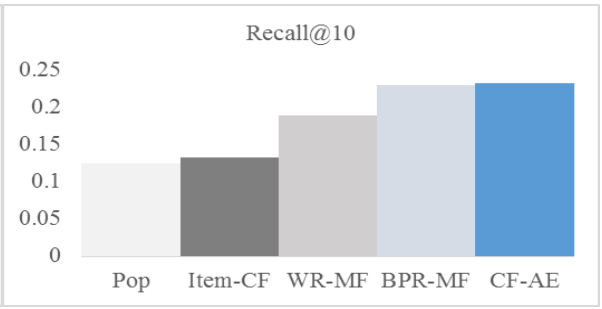
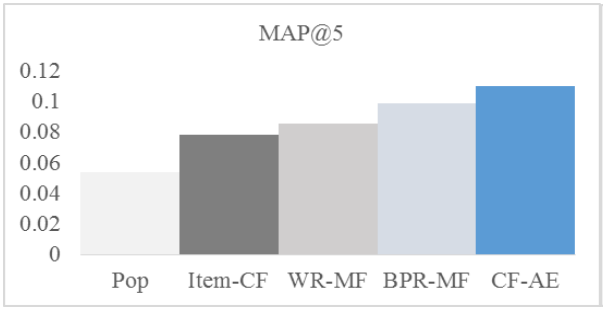


图 5. 24 Movielens-20m Recall@10 对比



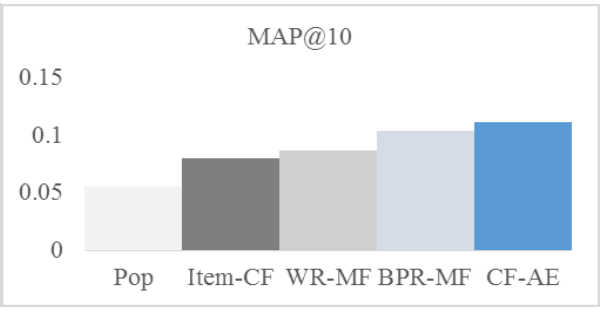


图 5. 25 Movielens-20m MAP@5 对比　　图 5. 26Movielens-20m MAP@10 对比

如表 5.10、表 5.11、表 5.12 所示，分别是在 Movielens-1m，Movielens-10m 和 Movielens-20m 下，不同的特征数下模型的评估 MAP@5 的值。特征数 K 的取值为 5，15，25，35，45，55。从图中，可以看到，性能随着 K 的增大而增加，但只有一点。当 K 变得足够大时，性能不再提高，并且实际上可能由于过度拟合而降低。在 Movielens-1m 数据集中，WE-MF 的实验结果好于 BPR-MF 和 CF-AE，在小数据集中，CF-AE 表现效果不好，并出现过拟合。但是，对于 CF-AE，随着 K 的增大，MAP@5 不断增大，并且增加的速度比其他模型更快。并且在 Movielens-20m 数据集中的效果好于 Movielens-10m 数据集。因此，随着数据规模的不断增大，CF-AE 的平均准确率表现越好，更适合做大规模数据分析。

比较 CF-AE 与其他几个 Top-N 推荐方法的 MAP@5 值。表 5.10、表 5.11 和表 5.12 分别显示了在 Movielens 数据集 1m，10m，20m 上所有模型的 MAP@5 分数。综合 3 个数据集，可以看出 CF-AE 在 1m 数据集上的表现差于 WR-MF，因为在用户和项目数量较少时，低秩矩阵分解的 WR-MF 方法更能准确进行项目推荐，而 CF-AE 相比之下效果稍差。并且，随着数据集 1m，10m，20m 的用户和项目数量增大，数据稀疏度变大，准确率逐步下降。

如表 5.10 所示，在 Movielens-1m 下不同的特征数下各模型的 MAP@5 值。如图 5.27 是可视化展示图。可以看到 WR-MF 在整体上表现出在不同的特征下效果好于 CF-AE，在用户和项目较少的情况 CF-AE 并不是最好的。

表 5. 10 在 Movielens-1m 不同的特征数下模型的 MAP@5 值

模型	5	15	25	35	45	55
Pop	0.0589	0.0589	0.0589	0.0589	0.0589	0.0589
Item-CF	0.1078	0.1078	0.1078	0.1078	0.1078	0.1078
BPR-MF	0.0997	0.1120	0.1171	0.1173	0.1183	0.1209
WR-MF	0.1051	0.1216	0.1229	0.1262	0.1259	0.1249
CF-AE	**0.0777**	**0.1067**	**0.1149**	**0.1230**	**0.1218**	**0.1235**

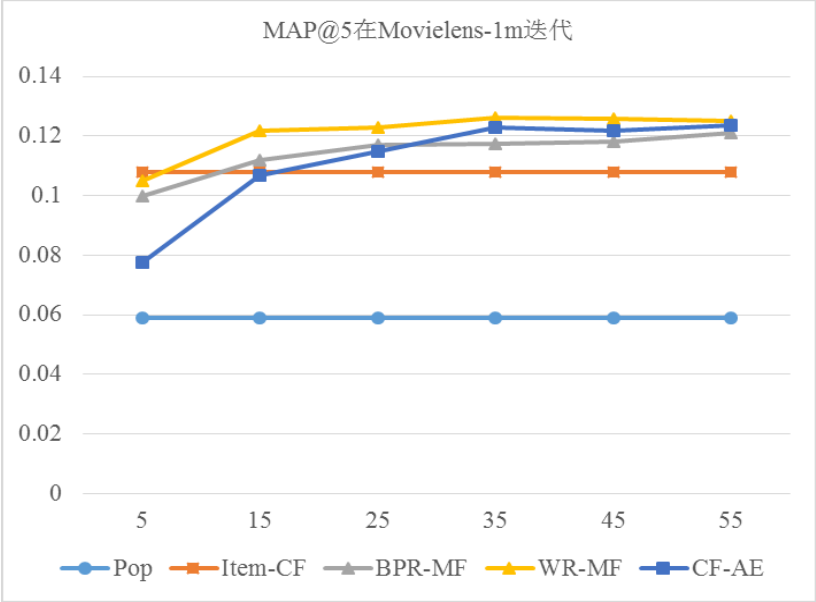


图 5. 27 Movielens-1m 下 MAP@5 对比

如表 5.11 所示，在 Movielens-10m 下不同的特征数下各模型的 MAP@5 值。如图 5.28 是可视化展示图。可以看到 CF-AE 在特征数少时 MAP@5 较低，但随着特征数的增大，表现远好于其他模型如 WR-MF 和 BPR-MF。相比 1m，随着用户和项目的增大，CF-AE 表现变好。

表 5. 11 在 Movielens-10m 不同的特征数下模型的 MAP@5 值

模型	5	15	25	35	45	55
Pop	0.0566	0.0566	0.0566	0.0566	0.0566	0.0566
Item-CF	0.1157	0.1157	0.1157	0.1157	0.1157	0.1157
BPR-MF	0.0783	0.0914	0.0958	0.0967	0.0983	0.0986
WR-MF	0.0842	0.1005	0.1057	0.1092	0.1121	0.1140
CF-AE	**0.0729**	**0.0981**	**0.1103**	**0.1168**	**0.1198**	**0.1216**

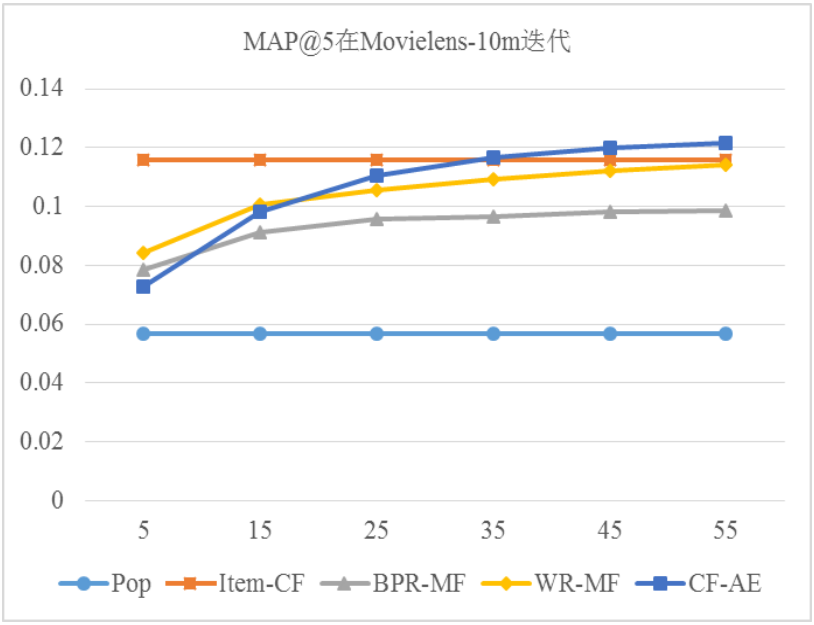


图 5. 28 Movielens-10m 下 MAP@5 对比

如表 5.12 所示，在 Movielens-20m 下不同的特征数下各模型的 MAP@5 值。如图 5.29 是可视化展示图。可以看到 CF-AE 在特征数少时 MAP@5 较低，但随着特征数的增大，表现远好于其他模型如 WR-MF 和 BPR-MF。相比 10m，随着用户和项目的进一步增大，CF-AE 表现更好。

表 5. 12 在 Movielens-20m 不同的特征数下模型的 MAP@5 值

模型	5	15	25	35	45	55
Pop	0.0540	0.0540	0.0540	0.0540	0.0540	0.0540
Item-CF	0.0786	0.0786	0.0786	0.0786	0.0786	0.0786
BPR-MF	0.0708	0.0809	0.0819	0.0837	0.0858	0.0849
WR-MF	0.0760	0.0875	0.0918	0.0945	0.0976	0.0993
CF-AE	**0.0663**	**0.0847**	**0.0960**	**0.1017**	**0.1086**	**0.1103**

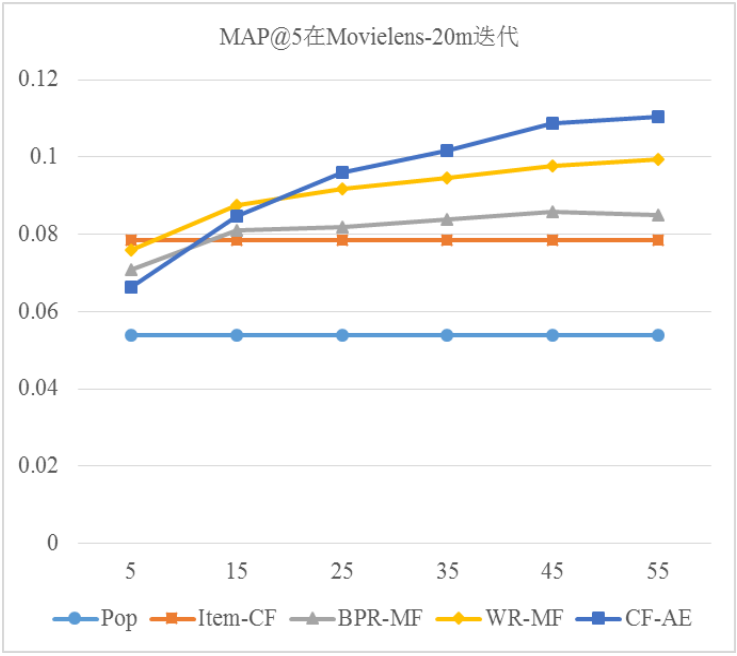


图 5. 29 Movielens-20m 下 MAP@5 对比

如表 5.13、表 5.14 和表 5.15 所示，CF-AE 在 Movielens-1m，Movielens-10m 和 Movielens-20m 数据集下，不同的特征数下的准确率（P@N）、召回率（R@N）和平均准确率（MAP@N）的评估值。K 的取值为 5，15，25，35，45，55。准确率随着 N 的取值（1，5，10）增大而增大。召回率率随着 N 的取值（1，5，10）增大而减小。平均准确率的变化同准确率逐渐增大。并且，随着用户和项目数量的增大，数据的稀疏度也变大，在相同特征下的准确率下降。

表 5. 13 CF-AE 在 Movielens-1m 不同的特征数下的评估值

评估	5	15	25	35	45	55
P@1	0.1465	0.1913	0.2091	0.2164	0.2173	0.2196
P@5	0.1015	0.1338	0.1434	0.1498	0.1505	0.1514
P@10	0.0850	0.1087	0.1146	0.1198	0.1197	0.1202
R@1	0.0241	0.03336	0.0371	0.04055	0.0393	0.0402
R@5	0.0806	0.1110	0.1213	0.1318	0.1274	0.1302
R@10	0.1371	0.1798	0.1900	0.2034	0.1982	0.1994
MAP@5	0.0777	0.1067	0.1149	0.1230	0.1218	0.1235
MAP@10	0.0693	0.0965	0.1038	0.1122	0.1101	0.1116

表 5. 14 CF-AE 在 Movielens-10m 不同的特征数下的评估值

评估	5	15	25	35	45	55
P@1	0.1021	0.1430	0.1598	0.1705	0.1749	0.1763
P@5	0.0774	0.0996	0.1092	0.1158	0.1181	0.1199
P@10	0.0636	0.0795	0.0865	0.0909	0.0932	0.0943
R@1	0.0290	0.0409	0.0472	0.0505	0.0521	0.0525
R@5	0.1051	0.1361	0.1516	0.1586	0.1617	0.1645
R@10	0.1687	0.2113	0.2301	0.2405	0.2462	0.2491
MAP@5	0.0729	0.0981	0.1103	0.1168	0.1198	0.1216
MAP@10	0.0746	0.0995	0.1118	0.1181	0.1213	0.1231

表 5. 15 CF-AE 在 Movielens-20m 不同的特征数下的评估值

评估指标	5	15	25	35	45	55
P@1	0.0934	0.1255	0.1425	0.1528	0.1594	0.1620
P@5	0.0705	0.0883	0.0980	0.1040	0.1095	0.1117
P@10	0.0584	0.0722	0.0792	0.0839	0.0867	0.0889
R@1	0.0271	0.0346	0.0405	0.0429	0.0462	0.0467
R@5	0.0960	0.1175	0.1310	0.1377	0.1474	0.1502
R@10	0.1552	0.1902	0.2078	0.2185	0.2259	0.2333
MAP@5	0.0663	0.0847	0.0960	0.1017	0.1086	0.1103
MAP@10	0.0679	0.0863	0.0976	0.1034	0.1099	0.1122

最后进行总结分析。MAP 和 Recall 的结果是一致的，即模型的性能排序几乎相同。例外情况是在 Movielens-10m 数据集，其中 WR-MF 获得比 BPR-MF 更好的 MAP@N 分数，但是 BPR 具有更好的 Recall@N 分数。

根据 MAP@10 和 Recall@10 的结果，CF-AE 始终优于其他比较方法。在 Movielens-20m 数据集上，CF-AE 在所有评估指标上具有大幅度的性能优于其他方法。CF-AE 的 MAP@10 得分和 Recall@10 得分比第二好的模型 MF 的得分高 10%左右。对于 Movielens-10m 数据集，ITEMCF 实现比其他方法（例如 MF，BPR）更好的结果，特别是对于 MAP@1 和 Recall@1 的度量。CF-AE 是唯一能够在度量 MAP@10 和 Recall@10 上好于 ITEM-CF 的模型，其中 CF-AE 的性能优于 ITEM-CF 大约 15%。在 Movielens-1m 数据集上，BPR-MF 实现比 MF-MF 更低的 MAP 值得分。其获得更好效果的数据集是在 Movielens-10m，但性能提升并不显著。

本文提出了用于 Top-N 推荐问题的协同自动编码器（CF-AE）。CF-AE 通过学习制定使用降噪自动编码器结构中用户项目的反馈数据分布在用户和项目的表示。对三个数据集进行了一组全面的实验，以研究模型组件的选择如何影响性能。除此之外，将 CF-AE 与其他几种 Top-N 推荐方法进行了比较，结果表明 CF-AE 在数据稀疏性高时优于其他方法。

5.7 小结

为了验证 UEIFM 模型和 CF-AE 框架的有效性，本章通过 Docker 容器技术，基于云计算环境搭建了基于 Spark 和 Tensorflow 的分布式实验平台，实现

数据的分布式存储。并介绍了对公共数据集的预处理方法，然后说明实验的评估指标。和几个已知的算法进行比较，验证了 UEIFM 多源数据融合模型。最后，验证了协同自动编码器框架，表明该算法在处理稀疏隐式反馈数据时的有效性。

结论

在信息大数据时代，用户的个性化需求不断提高，对于信息系统智能度的要求带来了很多挑战。面对大量的数据信息，如何帮助用户有效获取所需要的信息，有力改善信息过载问题，是数据研究人员的主要研究挑战之一。随着人们对推荐系统的需求越来越强烈，改善用户体验是推荐系统的根本目标。

推荐系统的任务是联系用户和产品，解决信息过载问题。当前大数据环境下的推荐系统成为主流，为了充分挖掘数据的价值，提高推荐的实时性和准确性，进一步有效地缓解信息过载的问题，系统和算法的可扩展性和性能问题都是面临的重要挑战。同时，由于数据产生的速度更快，数据高维且稀疏，内容采样渠道更多。多源数据在融合的过程中由于结构和采集方式的不同会引入噪声和冗余，更多的是非结构数据和半结构的数据。因此，有效利用多源数据成为获取有效推荐的必要前提。大数据环境比传统环境面临更加复杂的信息提供环境和数据特征，只有在充分、准确提取和预测用户产生的各种数据中蕴含的用户兴趣偏好后，才能有效生成准确度更高的推荐。大数据环境下数据产生的速度更快，数据高维稀疏，内容采样渠道更多。这些数据构成了多个数据来源，不同的数据类型和格式。推荐系统可以采集到丰富的用户显式反馈数据和隐式反馈数据，并随着 Web 应用和移动应用规模的不断扩大，数据规模也将变得越来越大。这使得推荐系统对数据处理效率的要求更高，丰富的数据使得用户对推荐系统准确性要求更高，同时用户要求具有实时性的要求获得良好的用户体验。为了达到推荐系统处理海量数据的要求，基于 Hadoop 和 Spark 大数据处理平台可以在海量数据中充分挖掘用户潜在的兴趣，为用户提供良好的服务。

本文以大数据环境下推荐系统的多源数据融合分析问题为背景，同时面临数据稀疏性问题和冷启动问题，设计了基于云计算 Docker 环境和大数据平台 Spark 和 Hadoop 的多源数据融合模型和协同过滤推荐算法，并实现了算法的并行化，结合深度神经网络，提高隐式反馈中进行 Top-N 推荐的等精度问题。本课题的具体工作内容如下：

1）有效地实现了算法的并行化。解决当前推荐系统面临的可扩展性问题，在云计算和大数据平台下采用分布式并行方法实现，可以有效地解决可扩展性和鲁棒性问题。

2）提出了适用于推荐的多源数据融合分析的模型。利用用户在互联网（包括微信，微博，网站等）访问的数据、数据库数据和日志数据等，提出统一显式与隐式反馈模型 UEIFM，通过对可观察用户选择行为数据集（隐式用户反馈数

据集）和评分（显式反馈数据），将项目推荐问题转化为优化问题，以提高推荐的准确率。

3）优化了基于模型的协同过滤算法。在 ALS 基础上进一步提出了基于分布式并行优化模型 UEIFM 并提供了 Spark 的实现，以处理大规模多源数据，最后，在实验中验证所提算法模型 UEIFM 及和并行化实现的有效性和可扩展性。

4）提出了结合自动编码器的协同过滤算法—协同自动编码器（CF-AE）。稀疏输入对于协同过滤是至关重要的，指出被学习的用户和项目表示之间的复杂关系。介绍了去噪自动编码器，并且自动编码器与协同过滤中的低秩矩阵因子分解是密切相关的，进而引出协同自动编码器架构。然后解释如何在其顶部添加额外的约束，以使网络处理稀疏输入。最后，详细介绍完整的设计和训练过程，并完成实验分析。并和几个常见的 Top-N 推荐算法进行比较，实验结果表明了 CF-AE 在推荐准确率的提升。

工作展望

推荐系统还面临着诸多挑战，推荐系统的功能就是评估效用函数用来预测用户是否会喜欢项目或者说喜欢项目的概率。一个好的推荐系统一定是根据用户喜好个性化地产生推荐列表。产生的推荐也是多样性的，代表着用户所有可能的兴趣。推荐一些用户不知道的东西，但可能会感兴趣的项目给用户。

如何结合上下文信息产生推荐，充分利用用户和项目的交互信息，根据位置，时间，天气等信息进行推荐，有效解释推荐的结果，提供更好的用户使用体验。间接的信息关联很多上下文信息，有可能有效地改善使用上下文的应用程序。然而，间接和上下文变量表示必须处理经常是噪声的更多数据并且创建更高维度的问题空间

多源信息融合的推荐也成为了研究的热点，利用评分数据和评论文本的融合，或者用户社交网络信息的融合。结合评论信息的评分模型，运用自然语言处理算法，提取语义分析，分析用户的情感倾向，得到用户对项目的偏好，最后得到更好的项目推荐结果。

冷启动问题是推荐系统面临的一大挑战，特别是在基于协同过滤的推荐方法中尤为值得研究。冷启动问题通常可采用混合推荐方法和协同过滤方法、用户人口统计信息如性别、年龄等，以及社会网络关系等来解决。

混合推荐模型始终是研究的热点，不同的混合模型有不同的亮点。单一方法无法解决推荐系统的所有问题，混合推荐模型更有效。混合的形式多种多样，如内容过滤和协同过滤的混合，基于项目和基于用户的协同过滤算法的混合和在线和离线方法的混合推荐方法等。

参考文献

[1] Gomez-Uribe C A, Hunt N. The Netflix recommender system: Algorithms, business value, and innovation. In: ACM Transactions on Management Information Systems (TMIS), 2015, 6(4): 13.

[2] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. Las Vegas: ACM, 2008, 426-434.

[3] Salakhutdinov R, Mnih A, Hinton G. Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th international conference on Machine learning. Corvalis: ACM, 2007, 791-798.

[4] Linden G, Smith B, York J. Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet computing, 2003, 7(1): 76-80.

[5] Lu Z, Dou Z, Lian J, et al. Content-Based Collaborative Filtering for News Topic Recommendation. In: The Twenty-Ninth AAAI Conference on Artificial Intelligence. Austin: AAAI, 2015, 217-223.

[6] 景民昌. 从 ACM RecSys' 2014 国际会议看推荐系统的热点和发展, 现代情报, 2015, 35(4): 41-45.

[7] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web. Hong Kong: ACM, 2001, 285-295.

[8] 朱扬勇, 孙 婧. 推荐系统研究进展. 计算机科学与探索, 2015, 9(5):513-525.

[9] Jain S, Grover A, Thakur P S, et al. Trends, problems and solutions of recommender system. In: International Conference on Computing, Communication & Automation (ICCCA). Noida: IEEE, 2015, 955-958.

[10] 王国霞, 刘贺平. 个性化推荐系统综述. 计算机工程与应用, 2012, 48(7): 66-76.

[11] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer, 2009, 42(8): 30-37.

[12] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web. Hong Kong: ACM, 2001, 285-295.

[13] Herlocker J L, Konstan J A, Borchers A, et al. An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. New York: ACM, 1999, 230-237.

[14] Krohn-Grimberghe A, Drumond L, Freudenthaler C, et al. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In: Proceedings of the fifth ACM international conference on Web search and data mining. Seattle: ACM, 2012, 173-182.

[15] Salakhutdinov R, Mnih A. Probabilistic Matrix Factorization. In: International Conference on Neural Information Processing Systems. Vancouver: Curran Associates Inc, 2007:1257-1264.

[16] Hu Y, Koren Y, Volinsky C. Collaborative filtering for implicit feedback datasets. In: Eighth IEEE International Conference on Data Mining. Pisa: IEEE, 2008, 263-272.

[17] Pan W, Chen L. GBPR: Group Preference Based Bayesian Personalized Ranking for One-Class Collaborative Filtering. In: 23rd International Joint Conference on Artificial Intelligence. Beijing: 2013, 2691-2697.

[18] Pan R, Zhou Y, Cao B, et al. One-class collaborative filtering. In: Proceedings of the 8th IEEE International Conference on Data Mining. Pisa: IEEE, 2008, 502-511.

[19] Rendle S, Freudenthaler C, Gantner Z, et al. BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence. Montreal: AUAI Press, 2009, 452-461.

[20] He R, McAuley J. VBPR: visual bayesian personalized ranking from implicit feedback. Computer Science, 2015.

[21] Takács G, Tikk D. Alternating least squares for personalized ranking. In: Proceedings of the sixth ACM conference on Recommender systems. Dublin: ACM, 2012, 83-90.

[22] Shi Y, Karatzoglou A, Baltrunas L, et al. CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In: Proceedings of the sixth ACM conference on Recommender systems. Dublin: ACM, 2012, 139-146.

[23] Recht B, Ré C. Parallel stochastic gradient algorithms for large-scale matrix completion. Mathematical Programming Computation, 2013, 5(2): 201-226.

[24] Zhuang Y, Chin W S, Juan Y C, et al. A fast parallel SGD for matrix factorization in shared memory systems. In: Proceedings of the 7th ACM conference on Recommender systems. Hong Kong: ACM, 2013, 249-256.

[25] Schelter S, Boden C, Schenck M, et al. Distributed matrix factorization with mapreduce using a series of broadcast-joins. In: Proceedings of the 7th ACM conference on Recommender systems. Hong Kong: ACM, 2013, 281-284.

[26] Pilászy I, Zibriczky D, Tikk D. Fast als-based matrix factorization for explicit and implicit feedback datasets. In: Proceedings of the fourth ACM conference on Recommender systems. Barcelona: ACM, 2010, 71-78.

[27] Yu H F, Hsieh C J, Si S, et al. Parallel matrix factorization for recommender systems. Knowledge and Information Systems, 2014, 41(3): 793-819.

[28] Rendle S, Gantner Z, Freudenthaler C, et al. Fast context-aware recommendations with factorization machines. In: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval. Beijing: ACM, 2011, 635-644.

[29] Ling G, Lyu M R, King I. Ratings meet reviews, a combined approach to recommend. In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 105-112.

[30] Sedhain S, Sanner S, Braziunas D, et al. Social collaborative filtering for cold-start recommendations. In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 345-348.

[31] Yang C, Sun M, Zhao W X, et al. A Neural Network Approach to Joint Modeling Social Networks and Mobile Trajectories. arXiv: 1606.08154, 2016.

[32] Saveski M, Mantrach A. Item cold-start recommendations: learning local collective embeddings. In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 89-96.

[33] Liu H, Goyal A, Walker T, et al. Improving the discriminative power of inferred content information using segmented virtual profile. In:

Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 97-104.

[34] Verstrepen K, Goethals B. Unifying nearest neighbors collaborative filtering. In: Proceedings of the 8th ACM RecSys. Silicon Valley: ACM, 2014, 177-184.

[35] Cheng C, Xia F, Zhang T, et al. Gradient boosting factorization machines.In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 265-272.

[36] Tang L, Jiang Y, Li L, et al. Ensemble contextual bandits for personalized recommendation. In: Proceedings of the 8th ACM RecSys. Silicon Valley: ACM, 2014, 73-80.

[37] Liu X, Aberer K. Towards a dynamic Top-N recommendation framework. In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 217-224.

[38] Yang S H, Long B, Smola A J, et al. Collaborative competitive filtering: learning recommender using context of user choice. In: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval. Beijing: ACM, 2011, 295-304.

[39] Hariri N, Mobasher B, Burke R. Context adaptation in interactive recommender systems. In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 41-48.

[40] Nguyen T V, Karatzoglou A, Baltrunas L. Gaussian process factorization machines for context-aware recommendations. In: Proceedings of the 37th international ACM SIGIR conference on Research & Development in Information Retrieval. New York: ACM, 2014, 63-72.

[41] Zhao T, McAuley J, King I. Leveraging social connections to improve personalized ranking for collaborative filtering. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management. Shanghai: ACM, 2014, 261-270.

[42] Bao Y, Fang H, Zhang J. TopicMF: Simultaneously Exploiting Ratings and Reviews for Recommendation. In: 28th AAAI Conference on Artificial Intelligence. Quebec: AAAI, 2014, 2-8.

[43] Stern D H, Herbrich R, Graepel T. Matchbox: large scale online bayesian recommendations. In: Proceedings of the 18th international conference on World Wide Web. Geneva: ACM, 2009, 111-120.

[44] Parra D, Karatzoglou A, Amatriain X, et al. Implicit feedback recommendation via implicit-to-explicit ordinal logistic regression mapping. Proceedings of the CARS, 2011.

[45] 梁天一, 梁永全, 樊健聪, 等. 基于用户兴趣模型的协同过滤推荐算法. 计算机应用与软件, 2014(11): 260-263.

[46] 肖强, 朱庆华, 郑华, 等. Hadoop 环境下的分布式协同过滤算法设计与实现. 现代图书情报技术, 2013, 29(1): 83-89.

[47] 孙天昊, 黎安能, 李明, 等. 基于 Hadoop 分布式改进聚类协同过滤推荐算法研究. 计算机工程与应用, 2015, 51(15): 124-128.

[48] 李改, 潘嵘, 李章凤, 等. 基于大数据集的协同过滤算法的并行化研究. 计算机工程与设计, 2012, 33(6): 2437-2441.

[49] 郑凤飞, 黄文培, 贾明正. 基于 Spark 的矩阵分解推荐算法. 计算机应用, 2015, 35(10): 2781-2783.

[50] 杨志伟. 基于 Spark 平台推荐系研究: [中国科学技术大学硕士学位论文]. 合肥: 中国科学技术大学, 2015, 2-10.

[51] 于娜娜, 王中杰. 基于 Spark 的协同过滤算法的研究. 系统仿真技术, 2016, 12(1): 40-45.

[52] Covington P, Adams J, Sargin E. Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems. Boston: ACM, 2016, 191-198.

[53] Dziugaite G K, Roy D M. Neural Network Matrix Factorization. arXiv preprint arXiv:1511.06443, 2015.

[54] Strub F, Mary J. Collaborative Filtering with Stacked Denoising AutoEncoders and Sparse Inputs. In: NIPS Workshop on Machine Learning for eCommerce. 2015.

[55] Kim D, Park C, Oh J, et al. Convolutional Matrix Factorization for Document Context-Aware Recommendation. In: Proceedings of the 10th ACM Conference on Recommender Systems. Boston: ACM, 2016, 233-240.

[56] H. Wang, N. Wang, D.Y. Yeung. Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2015, 1235–1244.

[57] Wang C, Blei D M. Collaborative topic modeling for recommending scientific articles. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Diego: DBLP, 2011, 448-456.

[58] Zheng Y, Tang B, Ding W, et al. A neural autoregressive approach to collaborative filtering. In: Proceedings of the 33nd International Conference on Machine Learning. New York: JMLR, 2016, 764-773.

[59] Wu Y, DuBois C, Zheng A X, et al. Collaborative denoising auto-encoders for Top-N recommender systems. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. San Francisco: ACM, 2016, 153-162.

[60] Strub F, Gaudel R, Mary J. Hybrid Recommender System based on Autoencoders. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. Boston: ACM, 2016: 11-16.

[61] Sedhain S, Menon A K, Sanner S, et al. Autorec: Autoencoders meet collaborative filtering. In: Proceedings of the 24th International Conference on World Wide Web. Florence: ACM, 2015, 111-112.

[62] Elkahky A M, Song Y, He X. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In: Proceedings of the 24th International Conference on World Wide Web. Florence: ACM, 2015, 278-288.

[63] Hidasi B, Karatzoglou A, Baltrunas L, et al. Session-based recommendations with recurrent neural networks. arXiv: 1511.06939, 2015.

[64] de Brébisson A, Simon É, Auvolat A, et al. Artificial neural networks applied to taxi destination prediction. arXiv:1508.00021, 2015.

[65] Liu Y, Xie M, Lakshmanan L V S. Recommending user generated item lists. In: Proceedings of the 8th ACM Conference on Recommender systems. Silicon Valley: ACM, 2014, 185-192.

[66] Movielens. http://grouplens.org/datasets/Movielens/, 2016.

致　谢

光阴似箭，岁月荏苒。又到了毕业的季节，回想过去 3 年的研究生生涯，一幕幕的学习和生活场景回荡在我的脑海里。从最初的懵懂少年成长为一名有理想追求的研究生，收获满满，既有老师们的谆谆教诲，还有同学们的友谊和无私帮助。三年的研究生生涯，让我深深体会到了学术研究的乐趣，我深深喜欢上实验室那充实而温馨的氛围。此时此刻，我借此机会，向帮助关心我的老师、同学表示衷心的感谢。

首先，我要感谢我的导师曾庆光教授和李仁发教授。李老师那宽广的胸襟、渊博的知识、严谨的治学态度和敢为人先的精神，榜样的力量一直激励着我不懈的努力。本论文的完成得益于李老师的悉心指导，从选题、实验和论文的撰写都得到了老师不遗余力的支持，值此论文完成之际，谨向李老师致以最诚挚的敬意和深深的谢意。李老师的身教言传和谆谆教诲，使我受益匪浅，能够成为李老师的学生，是我一生最大的荣幸，必将使我受益终生。

感谢徐成老师、李蕊老师、刘彦老师和杨科华老师。谢谢刘彦老师和李蕊老师在开题和中期检查对我的指导，提供了很多具有实际指导意义的建议，让我少走了很多弯路。

特别感谢长沙双菱电子科技公司的周总和徐部长，是公司提供了我在研二期间的实习机会，同时课题的问题来源离不开公司的实际需求，公司也提供了研究初期的实验环境，让我更加熟练地掌握了 Spark 和 Hadoop 平台。

谢谢课题组的朱立民、杨竞、查理佳、秦凯强等同学，在课题的研究过程中，我们朝夕相处、团结协作、互相勉励、共同进步，度过了美好快乐的研究生涯。

特别感谢女朋友屠晓涵同学，我们在一起探讨研究问题，相互学习，共同进步。我们相处时间最长的地方就是实验室，谢谢默默的陪伴和支持。

特别感谢我的父母，谢谢你们支持我，鼓励我。是你们的无私奉献让我取得今天的成绩。我会带着你们的期望继续奋斗，用更好的自己来回报你们。

最后，谨向审阅本论文及答辩组的评委老师们致以深深的敬意和诚挚的谢意。由于本人研究水平有限，文中难免有不足之处，恳请各位老师批评、指正。

刘四平 2017 年 5 月 1 日于长沙

附录 A 攻读硕士学位期间发表的学术论文

论文发表：

[1] 刘四平, 屠晓涵, 李仁发. Unifying explicit and implicit feedback for Top-N recommendation. ICBDA, 2017.（已录用）

[2] 屠晓涵, 刘四平, 李仁发. Improving Matrix Factorization Recommendations for Problems in Big Data. ICBDA, 2017.（已录用）

软件著作权：

[3] 屠晓涵, 刘四平, 李仁发. 基于 Spark 的数据分析流程管理系统 v1.0: 中国, No. 01216238. 2016-08-01

附录 B 攻读硕士学位期间所参与的项目

[1] 国家自然科学基金[61173036]：以汽车为例的CPS若干问题研究.

[2] 国家高技术研究发展计划(863)资助项目子项[2012AA01A301-01]：行业应用市场分析和电磁计算软件研发.