推荐系统用户相似度计算方法研究

重庆大学硕士学位论文

（学术学位）

学生姓名：朱文奇

指导老师：罗　军　副教授

专　　业：计算机系统结构

学科门类：工　学

重庆大学计算机学院

二〇一四年四月

Research on User Similarity Function of Recommender Systems

A Thesis Submitted to Chongqing University

in Partial Fulfillment of the Requirement for the

Master's Degree of Engineering

By

Zhu Wenqi

Supervised by Associate Prof. Luo Jun

Specialty：Computer System Architecture

College of Computer Science of

Chongqing University, Chongqing, China

April 2014

摘　要

随着互联网的数据规模急剧扩大，用户无法准确迅速地找到自己需要的信息，为解决日益严重的信息过载问题，多种技术方案应运而生，推荐系统就是其中的佼佼者。推荐系统是一种个性化的信息服务，能够很好地充当用户和信息资源之间的桥梁。推荐系统通过建立模型，对用户的需求进行描述，再通过某种推荐策略将特定的信息资源主动推荐给目标用户。

由于推荐系统具有个性化和智能化等特点，其在电子商务、社交网站和影音站点取得了巨大的成功，已经成为这些应用平台的核心子系统。基于协同过滤的推荐系统是应用最为广泛、研究最为深入的一类推荐系统，这类推荐系统算法的关键是寻找用户或者项目的邻居，邻居寻找的准确性决定了最后推荐结果的质量，而邻居的寻找依赖于用户或者项目相似度的计算，因此设计一个适合的相似度计算方法是推荐算法成功的关键问题。

本文首先介绍了推荐系统的概念和推荐系统的模型，同时详细探讨了用户研究和用户模型，并对常用的推荐系统架构进行了分析和介绍；然后介绍了常用的几种推荐算法并介绍了各自适用的场景和优缺点以及主要的相似度计算方法和这些相似度计算方法的局限，接着介绍了常用的实验数据集和评价指标。

传统的用户相似度计算方法中每个项目的权重是一样的，通过分析可以知道用户间共同高评分项目的权重应该高于用户间共同低评分项目的权重，再考虑上类群关系，就得到了本文提出的一种加权的用户相似度计算方法，这种方法解决了一些会导致寻找邻居准确性下降的问题。通过在 MovieLens 数据集上进行实验，与基于传统用户相似度计算方法的协同过滤算法相比较，实验结果表明，考虑了项目相似度权重的协同过滤推荐算法能显著提高评分预测的准确性和推荐结果的质量。

关键词：推荐系统，用户相似度，协同过滤，推荐算法，用户建模

ABSTRACT

With the rapid expansion of the data size on the Internet, users can accurately find the information they need quickly. To address the increasingly serious problem of information overload, a variety of solutions have emerged, and the recommendation system is one of the best. Recommendation system is a personalized information service, which can well serve as a bridge between users and information resources. Recommendation system describes the users' needs through building models, and then recommends specific information resources to the target users through some recommended strategies initiatively. Due to its personal and intelligent features, recommendation system has achieved great success in e-commerce, social networking sites and video sites, and has become the core subsystems of these application platforms. The recommendation system based on collaborative filtering is the most widely used and further studied. The key of this algorithm is to find users and project neighbors, and the accuracy determines the quality of the final results of the recommendation. Because of the neighbors' finding relies on similarity calculation of users and projects, the design of an appropriate calculation of similarity is the key issue of a successful recommendation algorithm.

This paper introduces the concept and model of recommendation system, and analyzes its architecture, while discussing the model of users in detail. Then this paper introduces several common recommendation algorithms and their applicable scenes, the advantages and disadvantages. After that, the paper also introduces the major similarity calculation methods and their limitations. At last, the commonly used experimental data sets and evaluation are introduced.

In traditional user similarity calculation methods, the weight for each item is the same. It is known by analyzing that the weight for co-high-rated programs among users is higher than which for low-rated programs. Considering the group relationship, a weighted similarity calculation method of user is been proposed. This solution will solve the issue that will lead to decrease the accuracy to find a neighbor. By comparing with the collaborative filtering algorithm based on traditional user similarity calculation method, the experiment, made on MovieLens data sets, shows that the collaborative filtering algorithm considering the item similarity can significantly improve score forecast accuracy and the quality of recommendation results.

Keywords： Recommender Systems; User Similarity; Collaborative Filtering; Recommendation Algorithm; User Modeling

目 录

1 绪 论

推荐系统是当下众多应用系统解决“信息过载”问题的首选方案，通过推荐系统的处理，系统主动将数据推荐给目标用户，充当了数据和用户间的桥梁[1]。随着未来应用系统的规模越来越大，推荐系统的地位也将越来越凸显。

本章介绍了论文的研究背景、研究意义和国内外研究现状与分析，同时说明了论文的研究内容和论文的结构安排。

1.1 研究背景及研究意义

1.1.1 研究背景

在日常生活中，人们对于去哪家餐厅吃饭，看哪部电影，去哪儿旅游，读哪本书等选择往往是通过朋友介绍或者阅读评论文章等途径决定的。大数据时代，互联网在提供产品或者服务的时候就会遇到这个问题：一方面，应用系统提供了海量的产品或者服务给用户，竭尽所能地展示这些产品或者服务；另一方面，用户在这些产品或者服务之前，不知道自己要什么。这个问题的根源是用户没有能力清楚描述自己的需求，同时应用系统只能被动地列出所有项目，无法匹配用户和具体产品或者服务。

为解决这个问题，众多的解决方案应运而生，其中最有影响力的包括分类目录和搜索引擎，其代表公司是雅虎和谷歌。这两类系统都很努力地试图解决上述问题：分类目录将系统中所有信息分类并按此编排目录，用户可借此迅速找到自己想要的信息；搜索引擎将所有信息按照关键字制成倒排索引，用户只要输入自己想要信息的关键字便可找到该信息。但是这两类方案还是被动地解决问题，用户需要准确地描述自己的需求，同时没有针对具体用户建立个性化的用户兴趣模型，无法挖掘用户的潜在兴趣[2]。

针对上述另类解决方案的缺点，学界在 20 多年前提出了推荐系统，然后很快地应用在商业网站上，取得了不错的效果。推荐系统根据用户的信息和历史行为记录，构造出用户的个性化模型，再依据特定的推荐算法向用户推荐其可能会感兴趣的项目。推荐系统和其他解决方案相比最大的特点就是其主动性，用户不需要对自己的需求有很准确的描述，系统主动地去预测用户的兴趣并将其推荐给用户。

推荐系统的最初设计的应用场景并不是电子商务，但近年来，由于推荐系统中信息过滤的特殊属性，推荐系统已经应用在任何规模的电子商务网站中，其为这些网站提供了海量的推荐结果[3]。推荐系统可以帮助企业决定向哪个客户出价，

实现一对一的营销战略目的。例如，用户搜索一件商品的名称，可能网站会弹出相关购物网站的广告，用户浏览了教育相关的网页，可能会受到教育培训机构发送的广告。基本上，由于推荐系统的设置，建议寻求者可以主动寻求得到推荐系统的推荐或者系统可以不经过提示推送推荐，例如亚马逊网站的畅销书信息。为了在第一种情况下得到个性化推荐，建议寻求者可以通过对一些项目评分已达到指定自己偏好的目的，推荐系统会分析这些偏好并利用这些偏好向用户提供个性化的建议。

推荐系统可以说是当前大型商业网站的必备模块，如亚马逊、淘宝网、和京东商城等，都建设了性能优良的推荐系统，通过成功地应用推荐系统，亚马逊成功将其销售额提高了三成。有的商业网站为了提高自己推荐系统的性能，甚至不惜重金，举办推荐系统比赛，如“Netflix Prize”和“百度电影推荐算法创新大赛”。同时推荐算法也是学术界研究的热点，特别是在机器学习和模式识别等领域，美国明尼苏达大学 GroupLens 小组首先研究电影推荐系统，该系统给用户推荐那些他们没有评分但很有可能会打高分的电影[4]。

随着大数据时代的到来，各商业公司对用户信息的挖掘越来越重视，对推荐系统的准确性的要求也越来越高，如何提高现有推荐算法的性能是研究的热点。

1.1.2 研究意义

推荐系统自从在上世纪 90 年代以一个独立的概念被提出来，至今已经获得了长足的进步，特别是在社交网络、电子商务、影音娱乐等领域，推荐系统已然成了核心组件[5]。如今互联网由被动向主动的转变有很大程度上归功与推荐系统的作用，推荐系统帮助用户节约搜索的时间成本，帮助内容提供商降低营销成本的同时增加销售额，让互联网环境更加成熟。具体来说，研究推荐系统有以下几点意义：

① 对于学术研究来说，推荐系统已经在很多领域取得了成功，但是想要把推荐系统应用到其他的领域还需要更多的研究。比如，在很多应用的环境里，用户的信息非常少，收集用户历史记录又很困难，这就会导致推荐系统的失效，如何使得推荐系统工作在这类环境中是很值得研究的[6]。

② 对于系统用户来说，推荐系统提高了应用系统的可用性，用户不会像无头苍蝇一样在终端面前无从下手，在推荐系统的引导下，用户能逐渐清楚自己的需求。研究更加智能、高效的推荐系统契合系统用户的需要。

③ 对于应用系统内容的提供商来说，推荐系统能够提升系统用户对于应用系统的满意度，从而提高网站的转化率，帮助提供商实现更好的经济效益。

④ 对于整个社会而言，研究推荐系统能够提高国家整体信息技术水平，缩小和先进国家之间的差距。同时推荐系统有助于解决信息不对称问题，让信息的流

动更加自由。

1.2 推荐系统研究现状

最早的推荐系统可以追溯到 Xerox 公司在 1992 年设计的协同过滤系统 Tapestry，其目的是为了解决 Palo Alto 研究中心信息过载的问题。同年，Goldberg 提出了推荐系统，1997 年 Resnick 和 Varian 首次在学术界正式提出推荐系统的定义，他们认为“推荐系统是用以帮助电子商务网站向用户提供商品和建议，促成用户的产品购买行为，模拟销售人员协助客户完成购买过程”[1]，这个定义直到现在都被广泛引用。推荐系统的发展离不开若干其他的领域的进步，比如信息检索、认知科学、管理科学等[7]。

推荐系统的核心是推荐算法，推荐算法决定了推荐系统如何工作和具体工作的策略，一般来说推荐算法分为两种类别，其中一类是基于内容过滤的推荐，另一类是基于协同过滤的推荐[8]。基于内容过滤的推荐算法是信息检索领域的重要研究内容，其中很多研究理论和研究方法都类似，由于基于内容的过滤算法较为直观，在很多领域已经得到了应用，比如 1985 年 Malone 在 MIT 设计的过滤电子邮件的原型系统 Information Lens、Balabanovic 在斯坦福建立的网站推荐系统 LIRA、Pazzani 在伯克利实现的基于内容过滤的推荐系统 Syskill Webert 等等。基于内容过滤的推荐算法用特征码描绘目标项目，达到挖缺目标项目内容的目的。由于互联网上的信息种类繁多，包括视频、音乐、图片等，要找到一个适用于所有内容的基于内容过滤的推荐算法不是一件容易的事，然而基于协同过滤的推荐算法能够巧妙地绕开这些问题[9]。

基于协同过滤的推荐算法的设计基于一个假设，那就是“和目标用户相似度高的用户，其感兴趣的物品目标用户也会感兴趣”，算法本身不考虑物品本身的内容属性，而是利用目标用户的社会化关系，借助了人与人之间的联系，这是也是“协同”二字的由来[10]。基于协同过滤的推荐算法根据目标用户的属性信息和对已评分项目的历史评分来计算用户之间的相关性，从而得到邻居用户，再把邻居用户高评分项目里那些目标用户没有评分的项目作为计算结果推荐给目标用户[11]。从上述描述可知两种推荐算法的推荐策略有非常大的不同，其中基于协同过滤的算法可能更占优势，首先，协同过滤算法不需要考虑项目的内容信息，不必花费计算资源在解析项目内容上，这就使得协同过滤算法有很广泛的适用性；其次，协同过滤算法是社会化搜索，体现了很强的社会性，其推荐结果也有很强的不可预测性，能够挖掘用户的潜在需求，而基于内容的推荐算法就不具有这种功能[12]。

随着互联网规模扩大，推荐系统的实用性增强，国外研究推荐系统的团队越

来越多。明尼苏达大学的 GroupLens 团队使用基于主动协同过滤的推荐算法开发了 GroupLens 推荐系统，并将其应用在 Usenet 新闻组中，后又推出了影响广泛的 MovieLens 推荐系统，其所用数据集 MovieLens 成了学术界研究推荐算法的常用数据集；美国贝尔通信研究所开发的 Video Recommender 同样是一个电影推荐系统，不同于 MovieLens 等推荐系统系统使用网页收集用户的评分数据，Video Recommender 使用电子邮件系统收集用户评分数据；卡内基梅隆大学和莲花公司联合设计了主动协作过滤系统 ACF，首次实现了文档的推荐服务，并应用在办公软件 Lotus Notes 中；加州大学伯克利分校 Ken Goldberg 教授设计并开发了笑话推荐系统 Jester，该系统使用 Eigentaste 协同过滤算法并通过主成分分析提高了算法的效率；麻省理工学院媒体实验室开发了音乐推荐系统 Ringo，Ringo 在传统协同推荐算法的基础上增加了将用户分类的步骤，在同一个类中向用户推荐音乐； 斯坦福大学数字图书馆项目组开发了推荐系统 FAB，FAB 推荐系统算法的设计同时使用了基于内容过滤的算法和基于协同过滤的算法，综合了二者的优点[1]。

由于国内互联网起步较晚，互联网用户相对较少，对推荐系统没有强烈的需求等原因导致初期研究推荐系统的团队不多。近些年互联网在我国获得了飞跃发展，互联网用户数也成了全球第一，学术界开始注意这个研究领域。清华大学曾春等人对推荐系统进行了全面的综述，对国内推荐系统的研究具有较大的启蒙意义[4]；复旦大学邓爱林等人对电子商务领域的协同过滤算法中的稀疏性问题进行了研究，提出了一种基于项目评分和聚类的协同过滤推荐算法[13]。中国科学院软件研究所吴湖等人针对协同推荐算法的实时性问题，通过对用户和项目两个方面进行聚类并结合加权非负矩阵分解算法实现了对未评分项目的评分预测，提高了推荐算法的性能[14]。

推荐系统作为当下的一个研究热点，国内外都有专门的研讨会，如 ACM 每年举办的国际会议 RecSys，会议每次召开都引起了全球相关研究者的参与和关注，已经成为推荐系统领域最权威的国际会议。ACM 下设若干个特殊兴趣小组，其中 SIGKDD 组织 WEBKDD 数据挖掘年会、SIGCHI 组织 CSCW 计算机辅助协作年会、SIGIR 每年组织 Workshop on Recommender Systems 研讨会。另外 AAAI、WPRSIUI 和 ECAI 也都有大量会议专门讨论推荐系统。

1.3 论文的主要工作

本文的研究对象是推荐系统，在文中对推荐系统做了系统性的概述，着重分析了几种推荐系统架构，然后介绍了常用的推荐算法和主要的相似度计算方法以及研究算法所用的实验数据集和评价指标，论文工作的核心重点是基于用户的系统过滤推荐系统中计算用户相似度的方法。传统的计算用户相似度方法存在着种

种的局限性，本文通过分析现有的相似度计算方法和协同过滤推荐系统的实际情况，对传统的相似度计算方法进行了改进，引进了更加合理的权重表示方法，最后通过实验分析和验证了该方法在协同过滤推荐算法中的有效性。

具体来说，本文的主要研究内容包括以下几点：

第一，对互联网的信息过载问题进行了分析，对于解决这个问题的信息检索技术和信息过滤技术进行了介绍，给出了推荐系统的定义和形式化描述，详细介绍了推荐系统的主要功能。

第二，详细分析了推荐系统构建过程中的步骤，包括用户研究、用户建模和最后的系统架构。不同类型的推荐系统，如基于内容的推荐系统和协同过滤推荐系统，其构建过程存在着很多不同，如用户的模型，本文详细探讨了这些类别不同所导致的具体构建系统使用的不同技术。

第三，根据推荐策略的不同把推荐算法分为了三类，对各个类别进行了分析对比，详细介绍了三种传统的相似度计算方法并介绍了其局限性。对于推荐算法的评价问题，总结了五种最常用的实验数据集并对实验所用的评价指标进行了介绍。

第四，详细分析了传统用户相似度计算方法在基于用户的协同过滤推荐系统中的不足之处，比如使用传统用户相似度计算方法来计算目标用户间的相似度时，目标用户间的共同已评分项目的权重都是一样的，在本文提出来的用户相似度计算方法中用户间的共同高评分项目的权重是高于共同低评分项目权重的；同时传统的相似度计算方法没有考虑项目的类群关系，本文的相似度计算方法通过引入项目相似度数据来实现类群关系。通过实验验证了该计算方法能有效提高推荐算法的准确性。

1.4 论文的结构安排

本文既包含推荐系统的研究理论又有算法的实验分析，在总结和分析推荐系统已有研究的基础上，针对先有推荐算法的不足，提出了相应的改进算法，然后通过实验验证了算法的有效性。本文分为五章，各章的具体内容安排如下：

第一章是绪论部分，包含推荐系统的研究背景、研究意义和研究现状，同时在上述概要性介绍的基础上简要说明了本文的主要研究工作。

第二章是推荐系统及推荐系统模型综述，主要介绍推荐系统的规范定义及形式化描述和推荐系统的结构组成，之后对推荐系统的一些关键技术进行详细说明，包括建模前的用户研究，对推荐系统的用户做了深入的思考，还介绍了用户模型，包括基于内容的推荐系统的用户模型 VSM 和协同过滤推荐系统的用户模型用户-项目评分矩阵。最后介绍了推荐系统的架构，并分析了架构的特点。

第三章是推荐算法及相似度计算方法，在前一章对推荐系统的定义及相关技术概述的基础上，根据推荐的策略对推荐算法进行了分类，并详细介绍了各类算法的工作原理、优缺点和各自适应的场景，同时对协同过滤推荐算法中常用的相似度计算方法进行了介绍，对各种相似度计算方法的局限性进行了说明。最后总结了研究推荐算法主要会用到的 5 个实验数据集和主要的评价指标。

第四章是本文提出的一种考虑物品相似权重的用户相似度计算方法及其分析，首先对传统的基于用户的协同过滤推荐算法进行了分析，概括了其中用户相似度计算方法所引起的问题，然后针对这些问题，从计算时项目的权重出发，提出了加权的用户相似度计算方法，并通过实验验证了改进的计算方法在提高推荐算法评分预测准确性方面的有效性。

第五章是总结与展望，对本文的工作做了回顾和总结，并对未来的研究工作做出展望。

2 推荐系统及推荐模型综述

推荐系统由信息检索和信息过滤技术发展而来，其中很多内容借鉴了二者，但在具体形式、功能和目的上与二者截然不同。为了方便后面的算法论述，本章将具体介绍推荐系统的规范定义和结构组成，之后将对推荐系统的一些关键技术进行详细说明。

2.1 推荐系统概述

想象一个这样的场景：你独自一个人逛书店，走到一个堆满某类书的书架旁，你对那类书籍很感兴趣，但是却无法决定购买哪一本，这个时候你可能会求助于店家，让店家推荐最合适的书。在日常生活中，人们经常需要在没有足够经验的情况下对未知事物做出选择，这个时候向旁人寻求建议或者咨询这个领域的专家是比较常见的行为，推荐系统就扮演了这样一个角色。Resnick 和 Varian 指出“推荐系统协助并加强这种自然的社会过程”，推荐系统帮助解决人们的信息过载困境并协助人们在专业知识不足的情况下做选择[1]。本节将介绍推荐系统的定义和推荐系统起到的作用。

2.1.1 推荐系统定义

相比数据库、搜索引擎等经典信息系统，推荐系统是比较新的研究方向，推荐系统作为独立的研究领域是在上个世纪 90 年代中期。近年来，对推荐系统的兴趣大幅增加，具体体现在如下几点：

① 推荐系统在热门网站如 Amazon.com、Taobao.com、YouTube、Netflix、Tripadvisor 等发挥了重要作用[15]。而且现在许多媒体公司开发和部署推荐系统并将之作为提供给用户的一种服务，例如电影租赁服务 Netflix 公司奖励一百万美元给能够最大幅度提高该公司推荐系统性能的团队。

② 有专门研究推荐系统的会议和研讨会。最权威的是成立于2007年，称得上推荐系统研究和应用领域年度盛事的RecSys（ACM Recommender Systems）大会，此外大会的研究领域还包括传统的领域包括数据库、自适应系统等[16]。这些会议中值得一提的是ACM SIGIR（Special Interest Group on Information Retrieval）、UMAP（Adaptation and Personalization）、User Modeling和SIGMOD（Special Interest Group on Management Of Data）。

③ 在世界各地的高等教育机构，本科生和研究生的课程都致力于研究推荐系统，推荐系统的专题研究报告在计算机科学会议里很受欢迎，同时不少关于推荐系统技术的书籍也已经出版[16]。

④ 在学术期刊上有专门针对推荐系统研究和发展的期号，这些学术期刊包括：AI Communications、IEEE Intelligent Systems、International Journal of Electronic Commerce、International Journal of Computer Science and Applications 、ACM Transactions on Computer-Human Interaction 和 ACM Transactions on Information Systems。

维基百科给推荐系统下了个这样的定义：推荐系统是一个预测用户对项目评分或者偏好的信息过滤系统的子类。不同的文献给出了不同的定义：

① 1998 年，Breese,Heckerman 等人认为“协同过滤或者推荐系统是一个使用包含用户偏好的数据库来预测新用户对其他主题或者产品的系统”[17]；

② 2000 年，Meteren 和 Someren 认为“推荐系统是一种特殊类型的信息过滤系统，其从大集合中挑选一个那些用户觉得有趣或有用的项目，可以被看作是一个分类系统”[18]；

③ 2002 年，Burke 认为“推荐系统是一个指导用户以个性化的方式在很大的选择空间选择自己感兴趣或觉得有用的物品的系统”[19]；

④ 2008 年，Konstan 认为“推荐系统帮助个人管理一个潜在并且巨大的选择建议的具体信息、产品或者个人的基于系统知识的偏好或当前的需求，并为系统用户组成的社区收集用户的喜好信息。”[20]；

⑤ 2008 年，Ting-Peng Liang 认为“推荐系统是一个能够分析用户历史行为和能为出现的新情况提供建议的信息系统”[21]。

从上述定义可以看出，站在不同的角度观察推荐系统可以得到很不一样的定义，但也有很多共同点：推荐系统是面向用户和物品的，其在用户和物品之间构建了一种多对多的映射关系。当用户进入系统时，推荐系统需要记录和分析用户的行为、兴趣、习惯和偏好等人性化信息，从这些个性化信息中挖掘和构造能够表示用户兴趣的模型，并能够根据让用户的兴趣偏好很好地匹配这个兴趣模型，然后根据某种过滤算法将符合用户兴趣的物品推荐给用户[22]。下面给出推荐系统的形式化定义：

1）系统用户集合 U

2）项目集合 T

3）U 中用户对 T 中项目的评分 R，R 是一定范围内的全序非负实数，通常以矩阵表示

4）效用函数 u()计算项目 t 对用户 c 的推荐度,推荐系统的目标是找到推荐度即 R 最大的那些项目，即

$$\forall c \in U, t' = \arg\max_{t \in T} u(\mathrm{c}, \mathrm{t})$$

2.1.2 推荐系统功能

任何学术的东西要想得到普及发展都应具有能提高社会生产力的前提，推荐系统也不例外。推荐系统可以说天生就带着商业的基因，由于能够实现互联网服务提供商的某些不可替代的要求[23]，推荐系统得到迅猛发展。总体来说服务提供商开发推荐系统基于如下一些目的：

① 增加销量。这可能是一个商业推荐系统最重要的功能了，拥有推荐系统的在线商店能够比没有推荐系统的在线商店多卖物品，因为推荐的物品有可能能够匹配用户的需要，所以这个目标是可能完成的。一般来说，从服务提供者的角度来看，引入推荐系统的主要目的就是提高系统的转化率，即让用户接受推荐的物品而不仅仅是浏览页面。

② 卖出更加多样的物品。推荐系统的另一个主要功能是能够向用户推荐那些用户喜欢而并不热门的物品。例如，对于网上书城 Amazon，书城想要卖出目录里的所有书籍而不单单是热门的畅销书，如果没有推荐系统这是不可能实现的，推荐系统能将不那么热门的书籍推荐给适合的用户。

③ 提高用户满意度。精心设计的推荐系统可以改善用户对网站或应用的体验，用户在和系统人机互动时会发现系统给他的推荐有趣并且是有理由的。有效、准确地推荐再加上一个友好的用户界面会提高用户对系统的主观评价，这反过来又会增加推荐被接受的可能性。

④ 提高用户忠诚度。用户会忠于一个重视并珍惜他们的网站，推荐系统利用和用户之前交互的信息来计算对用户的推荐，用户与网站交互的时间越长，推荐系统越能够精炼该用户的用户模型，系统对用户的偏好也越了解，越能够推荐匹配适合偏好的物品。

⑤ 更加了解用户的需求。推荐系统能够收集并描述用户的偏好信息，并以此产生推荐。服务提供商可以分析这些数据并以此管理自己的库存和生产。

2.2 推荐系统的架构和模型

推荐系统是一种信息系统，有输入和输出。推荐系统的输入是用户的元数据和历史行为数据，比如用户初始注册的一些信息和浏览系统的历史记录，输出即是针对用户的个性化的项目推荐列表。要开发一个推荐系统需要构建一个能够描述用户特征的模型，这是开发前非常重要的准备工作，用户模型决定了推荐系统的有效性[24]。本节先介绍如何对用户进行研究和推荐系统的用户模型，然后对推荐系统常用的架构做简要介绍和分析。

2.2.1 用户研究

充分理解用户是一个推荐系统成功的必要条件，洞察最终用户并以此构建用

户模型必须在推荐系统开发周期的早期阶段进行，因为这会影响到底选择哪些主要技术。以用户为中心的方法应用到任何项目的初始阶段都能够极大地减少重复设计、维护和客户支持，因此在建模之前需要对用户进行研究。

推荐系统不同于传统的信息系统，传统的信息系统的用户往往对自己的需求有大概了解，而推荐系统是主动去迎合用户，要把握用户的需求需要对用户进行多方面的研究。用户的信息包括好几个方面，比如用户的人口统计学信息（性别、年龄等）和偏好信息，因此用户研究也需要几种方式。用户的人口统计学信息可以在用户注册系统时留下的信息中找到，如若某些信息项没有甚至可以根据用户的行为信息进行分析，挖掘出这些信息项的值[25]。

最开始需要确定推荐系统的用户群，要弄清楚一些基本问题：用户的主要识别特征是什么、用户的技能水平如何和用户对类似系统的经验。我们专注于用户特征的识别是因为它对推荐系统的设计有独特的实用性。对于识别用户特征，可以收集不同用户组的人口统计学信息，如年龄、性别、工作区域、民族和语言，了解这些有助于设计者理解用户的动机、目标和期望[26]。推荐系统的设计者需要理解用户的目标并确认系统是否能够满足这些。例如亚马逊网站围绕提供给用户的商品和提供的购买建议。从用户的角度来看，他们使用这项服务的动机来完成两个目标：一个是为自己购买商品，另一个是为他人购买商品。然而亚马逊网站并不会区分这个目标，因此提供不了准确的推荐。另一个例子，一个配备有基于内容的推荐系统的搜索引擎可以为用户提供浏览网络和根据请求查找信息的机会。在这种情况下，用户可能是出于需要完成特定的目标，或他们的动机也许是简单的放松和花点时间浏览的乐趣。识别和理解用户的动机可以产生根本性的推荐效果和用户体验的改善，用户模型中所隐含的用户的见解将允许系统的设计者能够支持应用系统未来可能发生的变化[27]。

一个推荐系统要能达到用户的目标用户才会觉得对他有帮助，而这和用户之前对推荐系统的期望有很强的联系，所以需要分析用户的期望并以此把用户分类。

高期望的用户以目标为导向，专注于自己当前的任务是否能够完成，这意味着推荐系统必须产生“高质量的推荐”来满足这类用户。以新闻推送系统为例，该系统需要提供一个所有项目都令用户满意的列表，如果用户不能在前几个推荐中找到他们想要的个性化的新闻，他们必须一直向下滚动去寻找自己喜欢的新闻，在最坏的情形下，他们会拒绝使用这个一开始就让他们不满意的系统。

对于中等期望的用户，推荐系统可以产生“包含好的推荐”列表给他们。如果用户对于推荐系统有若干选择且具有灵活性，那么用户的期望就会降低。在这个用户群体中，有些用户纯粹是为了测试推荐系统能够有多匹配自己的偏好，有些用户在寻找契合自己偏好的建议，还有些用户是想从推荐系统中得到不同于自

己目前兴趣但是很新奇的建议。

低期望的用户，对于推荐系统的推荐仅仅是浏览一下，充满了机会主义。有研究表明对这种类型用户的推荐是非常困难的。

对用户而言，持有不同期望水平将导致对推荐系统不同的态度，如拒绝再次使用该系统或者觉得系统推荐的项目很有趣。用户的期望会随着使用系统的时间发生变化，而推荐系统的性能也会随时间演变。一般来说，推荐系统上线的时间越长，其针对用户的模型也愈加完善，推荐的准确性也会变高。然而用户的兴趣也可能发生变化，这就要求推荐系统有一定的灵活性，能够随用户的偏好的转变及时调整系统里的模型。

用户研究还涉及到一个问题：用户使用推荐系统的设备和环境。首先需要考虑的是用户使用什么设备接收系统的推荐。运行应用程序或服务的设备主要包括智能手机、台式电脑和未来智能电视的机顶盒。在当前无所不在的移动互联网环境下，用户所处环境是要考虑的因素。环境因素包括地理位置因素和时间因素，如用户是在工作还是购物。当用户乘火车旅行，给用户推荐娱乐性的视频绝对要比推荐新闻视频好得多。研究用户所处的社会环境也是重要的，一个例子是用户是单独还是和他人一起使用系统，这会影响许多设计决策如算法选择、数据收集的方法和推荐的呈现方式。对于多人一起使用系统的情形可以设计一个共享组配置文件机制以提供组推荐，例如把个体的喜好合并。然而即使在一个组中，仍需要给用户提供个性化的建议[28]。

还需要确定信息的收集方式，推荐系统通过隐式或显式的反馈方式来学习和建立用户的档案。在很多情况下，系统会要求用户明确地表示自己的喜好或者选择，并用这些信息得出对未来的建议，例如，用户可以把一个物品评为“喜欢”或“讨厌”，亦或者评为一到五个级别。虽然这种方法是让用户选择的最有效的方法，一些缺点仍然存在于这个策略中：① 用户必须参与提供相关反馈，增加了用户的负担；② 由于隐私或者其他情况，用户可能拒绝输入自己的偏好。系统通过显式方法获知用户偏好的方式包括自动地监视用户点击的超链接、在特定页面停留的时间、历史购买行为或者浏览的历史记录。同样的，系统使用隐式方法来构建用户的偏好也存在一些缺点，例如，用户长时间浏览一个特定页面不代表用户真正对该页面上的内容感兴趣。此前的研究已经证明在构建用户偏好数据库方面，显式方法和隐式方法拥有相同的效果。因此在要获得用户偏好的时候这两种方法往往联合一起使用。

研究所有这些问题需要做大量工作，但为了后期用户建立模型的顺利，仍需尽可能广泛地进行用户研究。

2.2.2 用户建模

在推荐系统的设计中有两个问题是最为关键的：其一是如何构建描述用户信息的模型，通过这个模型应该能够得出用户的偏好信息；其二是如何设计一个个性化的推荐算法，这个算法能够根据用户模型得出对具体用户的推荐。这两个问题有很强的联系，推荐算法依托于用户模型的选择，用户模型也决定了推荐算法的设计。本节主要介绍用户模型的相关内容。

完成了前期的用户研究工作，可以知道用户的信息主要包括人口统计学信息和兴趣信息，其中人口统计学信息可以在系统中得到，表示兴趣信息是用户建模的重点。

对于兴趣信息的收集，前文提到过分为隐式和显式，隐式的方式具有良好的透明性，更不容易引起用户反感，是更为常用的方式，用户操作应用系统的历史行为和上下文信息是信息收集的重点。确定了收集用户信息的方式后，需要对收集的信息做一定的研究，确定哪些信息项能够较为准确地表示用户的兴趣[29]。

用户模型可分为静态模型和动态模型，静态模型往往通过显式方式收集用户偏好信息，动态模型通过隐式的方法收集用户偏好信息，动态模型和静态模型相比较为灵活，使用更加广泛。根据推荐算法的不同还可以将用户模型分为基于内容的推荐系统的用户模型和协同过滤推荐系统的用户模型，下面对这两种推荐系统的用户模型分别进行介绍。

大多数基于内容的推荐系统使用相对简单的用户模型，如关键字匹配或者空间向量模型（VSM，Vector Space Model）。空间向量模型是文本文档的空间表示，在该模型中每个文件由一个 n 维空间组成，每一维对应于给定的词汇表中的一个关键词。使用这个模型需解析用户兴趣信息的内容，将用户的偏好表示成特征关键字，并对不同的关键字赋予不同的权重。具体的方法是根据前期的用户研究，列出所有能够影响用户偏好的关键字列表，再使用 TF-IDF（term frequency-inverse document frequency）方法对所有关键字加权。TF-IDF 加权方法的主要思想是如果一个关键字在一段信息中的频率很高并且在其他信息中很少出现，则该关键字具有很好的区分性，赋予高权重[30]。

对所有的用户信息进行处理后，得到一个含 n 个关键字的特征空间 $T=\{t_1, t_2, t_3, \cdots, t_i, \cdots, t_n\}$，用户 u 的用户模型 $W_u=\{w_{u1}, w_{u2}, w_{u3}, \cdots, w_{ui}, \cdots, w_{un}\}$可以表示为表 2.1 的形式：

表 2.1 关键字特征空间

Table 2.1 Keyword Feature Space

t_1	t_2	t_3	…	t_i	…	t_n
w_{u1}	w_{u2}	w_{u3}	…	w_{ui}	…	w_{un}

使用空间向量模型来表示用户的偏好信息既有优点也有缺点。优点是实现简单，容易理解。缺点是只是从关键字的频次层面进行建模，没有进一步考虑关键字的语义，这导致推荐系统缺乏联想性，无法发现用户潜在的兴趣。更大的不足之处是适用范围太小，只能用在类文本信息的用户建模，如果用户的信息项是无法解析的多媒体等格式的话，空间向量模型就不适用了。

基于协同过滤算法的推荐系统中的用户模型要复杂一些，在这个模型不考虑具体的物品信息，只采用用户对物品的评分来表现偏好信息，评分值一般为一定范围内的离散值，该值的高低放映了用户对物品兴趣的强与弱，因为模型不考虑物品的内容信息，因此该用户模型的适用范围非常广泛[31]。

用户对物品的评分由用户主动进行，因为系统中物品数量非常多，对一个用户而言，他评过分的项目的比例应该是非常低的，因此应该用一些策略鼓励用户积极的评分。用户对所有项目的评分组成一个评分向量，所有用户的评分向量组成一个矩阵，称为评分矩阵，因为用户已评分项目的比例小，因此该矩阵中绝大部分评分数值为零。例如对一个由 m 个用户、n 个物品项组成的协同过滤推荐系统，其评分矩阵 $R(m \times n)$表示为表 2.2 所示：

表 2.2 评分矩阵

Table 2.2 Rating Matrix

	t_1	t_2	t_3	…	t_j	…	t_n
c_1	r_{11}	r_{12}	r_{13}	…	r_{1j}	…	r_{1n}
c_2	r_{21}	r_{22}	r_{23}	…	r_{2j}	…	r_{2n}
c_3	r_{31}	r_{32}	r_{33}	…	r_{3j}	…	r_{3n}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
c_i	r_{i1}	r_{i2}	r_{i3}	…	r_{ij}	…	r_{in}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
c_m	r_{m1}	r_{m2}	r_{m3}	…	r_{mj}	…	r_{mn}

协同过滤的用户模型取得了很大成功，很多流行的推荐系统都是基于这个模型，比如 Amazon 和 Netflix。但是该模型也存在很明显的缺点，如因评分项目少产生的稀疏性问题和由过期评分引起的概念漂移问题[32]。

综合前两种用户模型而产生的新模型称为混合推荐用户模型，该模型克服二者的不足，发挥各自的优势，也是推荐系统用户建模的一个可选项。

2.2.3 推荐系统架构

架构是推荐系统设计的核心，架构的设计直接影响推荐系统的工作方式和推荐质量。架构在推荐系统的位置如图 2.1 所示：

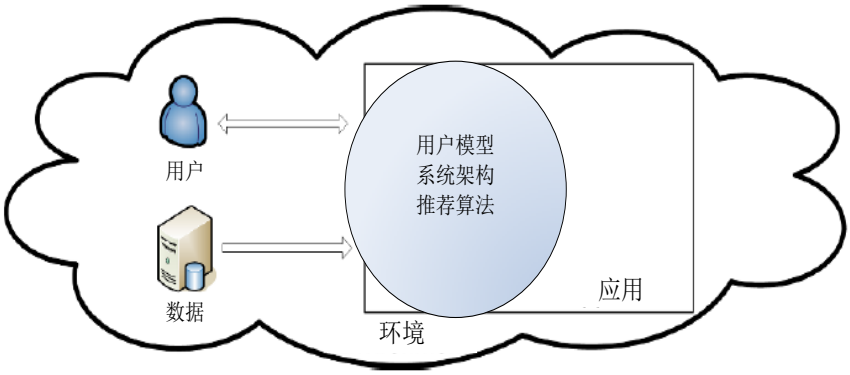

图 2.1 推荐系统环境

Fig.2.1 Environment of Recommender Systems

推荐系统向用户推荐物品有三种情况：推荐和用户已表示喜欢物品相似的物品、推荐和用户有相似偏好用户喜欢的物品、推荐包含用户偏好特征的物品。推荐系统的架构需要反映用户和物品联系起来的方式，根据推荐系统的不同，架构只实现一种联系方式。三种方式如图 2.2 所示：

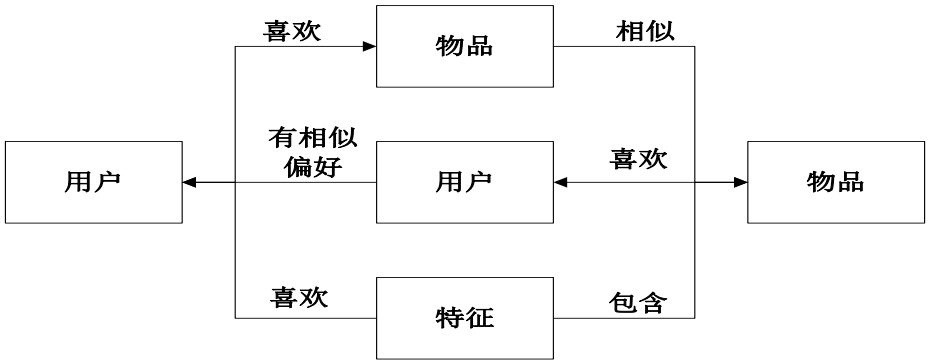

图 2.2 用户和物品的联系方式

Fig.2.2 Contact between Users and Items

根据各种不同的推荐策略，推荐系统为若干种，但是系统主要的工作流程不会有太大的不同，图 2.3 显示了推荐系统的基本工作流程：

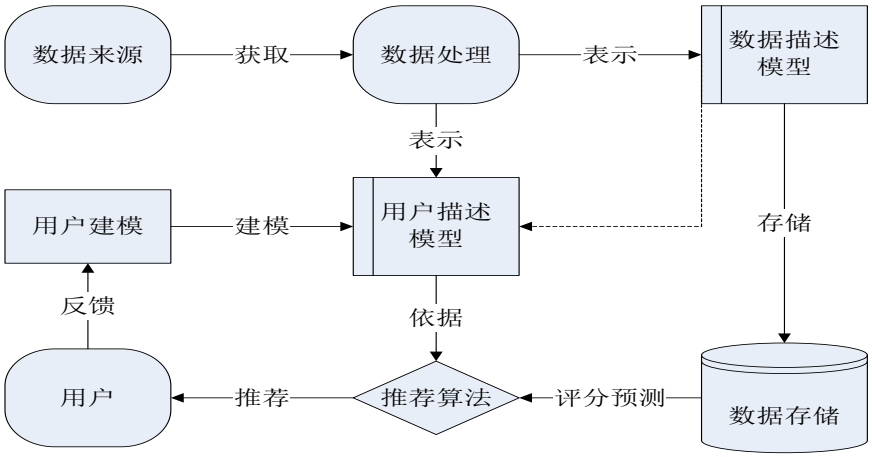

图 2.3 推荐系统工作流程

Fig.2.3 Workflow of Recommender Systems

图 2.4 给出了一个基于内容的推荐系统的架构，推荐过程分为三个步骤：内容分析、档案学习和过滤，各个步骤都是一个单独的组件。内容分析器提取项目结构相关的信息，并在接下来的处理步骤中表示项目。档案学习器收集用户偏好信息并以此构建用户配置文件，通常情况下这里使用机器学习算法来实现[33]。举例来说，一个网页推荐系统的档案学习器可以实现一个相关反馈方法，其中学习方法将正面和负面的向量和表示用户档案的原型向量联合起来。训练实例是在页面上由用户提供的正面或者负面的反馈。过滤组件利用建好的用户档案来提出相关推荐项目，其结果是一个二进制或者连续的相关性判断，后者的结果是一个用户可能喜欢的项目列表。

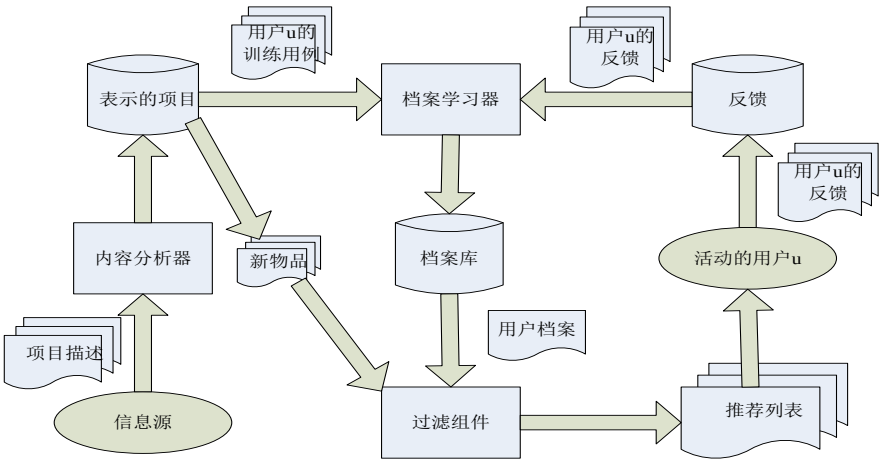

图 2.4 一个推荐系统架构

Fig.2.4 an Architecture of Recommender Systems

下面以 Netflix 的架构为例介绍一个具体的架构，Netflix 的推荐系统运行在 Amazon 的云服务器中，其架构分为三个部分：离线部分、接近在线部分和在线部分。其中在线部分为了能更快地响应最近发生的事件和用户的交互，对实时性要求高，这就使得这部分的算法不能太复杂，处理的数据不能太多。离线部分没有这方面的限制，整个架构的主要计算都放在这里。接近在线部分介于前两个部分之间，用于执行一些不要求实时性的在线计算任务，如图 2.5 所示：

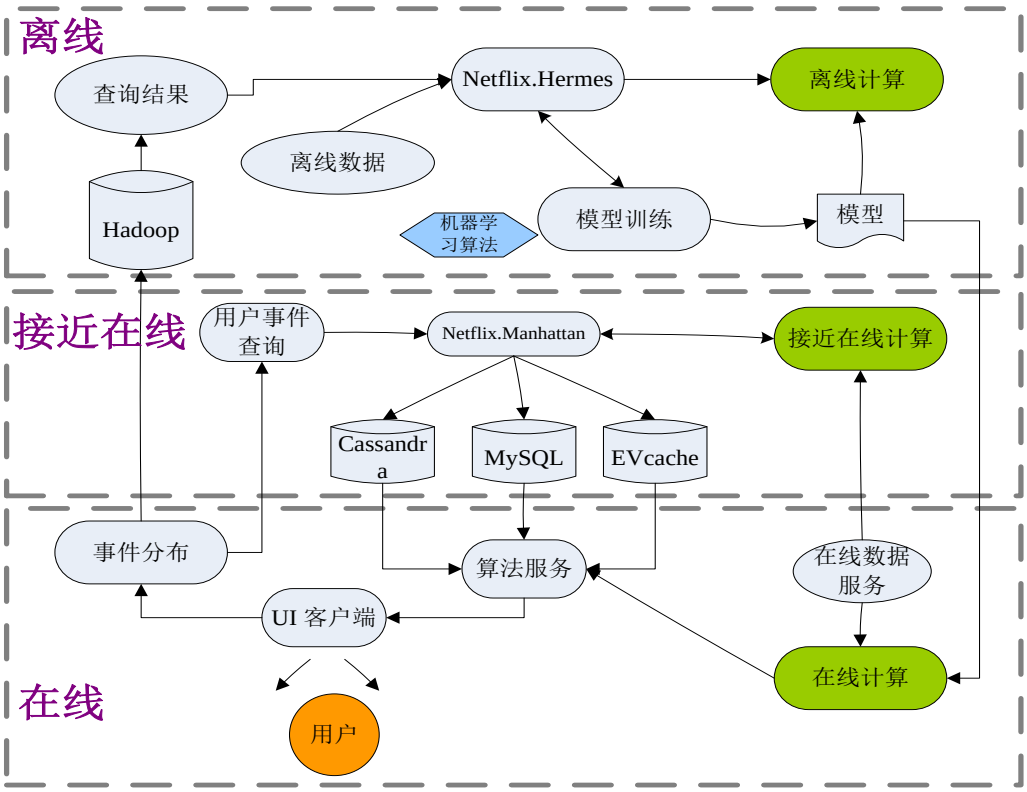

图 2.5 Netflix 推荐系统架构

Fig.2.5 Architecture of Netflix Recommender System

在这个个性化推荐架构最关键的地方是如何以无缝的方式连接离线、接近在线和在线三个部分。整个架构有三种输入：模型、数据和信号。模型是以离线方式训练完成的参数文件，数据是已处理完成的信息，信号是输入到系统中的新信息。

总的来说这个架构使用了复杂的机器学习算法，能够处理 Netflix 产生的大量数据。同时架构具有灵活性，新组件能较容易地加入进去。

2.3 本章小结

本章从讨论推荐系统的各种定义开始，介绍了推荐系统的功能和作用，接着

对推荐系统中的架构和模型做了简要的讨论，具体包括对用户的研究和之后针对用户建模的过程，最后介绍了一个基于内容的推荐系统的架构，并以一个实际架构（Netflix）的介绍结束。

3 推荐算法及相似度计算方法

本章介绍推荐算法，并根据常用分类详细介绍和比较了各类型算法的原理；然后全面介绍了本课题的研究核心：在推荐系统中可能用到的计算相似度的公式和方法，并着重分析了各个计算方法的特点和不足之处。

3.1 推荐算法

如果说推荐系统的架构是躯体，那么推荐算法就是这具躯体的灵魂。推荐算法的划分标准没有统一，但通常根据产生推荐的策略不同，可将其分为三类：基于内容的推荐算法、协同过滤推荐算法和混合推荐算法。本节将对这三种算法和各自的适用性及优缺点做详细的介绍。

3.1.1 基于内容的推荐算法

基于内容的推荐算法使用能代表用户偏好的文本信息来生成推荐，例如，对于文献推荐系统，基于内容的推荐算法会去分析论文的具体内容。同样，一个使用基于内容的推荐算法的电影推荐系统可能会利用电影的元数据，如电影名称、流派或者用户给出的文字评论等来做推荐[34]。

基于内容的方法被广泛用在信息检索和信息过滤领域中，这种方法分析用户配置文件中的文本信息来向用户推荐项目。用户配置文件中包含该用户的偏好和需求信息，这些信息被用来和系统中项目的属性相比较。基于内容的推荐算法做了如下假设：用户会喜欢和原来喜欢的物品相类似的项目。

基于内容的推荐算法使用了一种来自信息检索领域的技术：为关键字分配权重，目前常用的为关键字分配权重的方法是 TF-IDF。还有几个技术用在了基于内容的推荐算法里，包括贝叶斯分类器和人工神经网络。这些方法运用统计分析和机器学习技术来构建潜在的模型从而产生推荐。本论文介绍采用 TF-IDF 技术的基于内容的推荐算法。

TF-IDF 加权方法通过给词赋予权重来表示一个词的重要性，并结合了两种辅助参数：词频（*Term Frequency*）和逆文档频率（*Inverse Document Frequency*）。每个文档的共同矢量空间被称为向量空间模型，也称为词袋模型。文档 d 的向量表示 $\vec{V}(\mathrm{d})$ 包含文档中所有词的 TF-IDF 值，即 $\vec{V}(\mathrm{d})$ 中分组 i 是文档中第 i 个词的 TF-IDF 值。文档 d_1 和 d_2 之间相似度是通过它们文档向量间的余弦相似度来计算的。余弦相似度是通过计算 d_1 和 d_2 的文档向量间角度的余弦值来测定的，余弦相似度的具体内容在本章第二节进行介绍。

词频的分配策略是基于词 t 在文档 d 中出现的次数分配权重。在很多文档中都

经常出现的词不利于区分文档，逆文档频率策略提供了一个表示出现频次少的词的重要性的方法。一个频次少的词的 IDF 值高，相反频次高的词的 IDF 值很低，词 *t* 的 IDF 值计算公式如下：

$$idf_t = \log \frac{N}{df_t}$$

公式中的 N 是集合中文档的总数，df_t 是包含词 *t* 的文档在集合中的频率。对于文档 *d* 中词 *t* 的 TF-IDF 权重值的计算公式为：$tf_{t,d} \times idf_t$，其中 $tf_{t,d}$ 是词 *t* 在文档 *d* 中的词频，idf_t 是词 *t* 在整个集合中的逆文档频率。在使用 TF-IDF 方法计算得到词的权重后，基于内容的推荐算法再利用一个相似性度量方法来产生推荐。

使用 TF-IDF 的基于内容的推荐算法能够很好地产生高相似度推荐，但无法产生有新颖的推荐结果。使用 TF-IDF 的基于内容的推荐算法还有个缺点：无法分辨同义词，例如算法认为“土豆”和“马铃薯”相似度为零。相比后面要介绍的协同过滤推荐算法，基于内容的推荐算法可以向用户推荐那些刚加入到系统中的项目，而由于这些项目没有用户的评分，无法在协同过滤推荐系统中向用户推荐。

基于内容的推荐算法建立在信息检索技术的长期发展和积累之上，其实现的难度不大，主要的计算复杂度为 *O(mn)*（*m* 为项目数量，*n* 为特征向量空间规模），通常仅使用在线计算便能满足要求（协同过滤推荐算法计算量大，常常需要离线计算）。

无法很好地处理多媒体数据、推荐结果不具有联系性等问题没有很好的解决办法，这限制了基于内容的推荐算法的推广。针对这些问题协同过滤推荐算法应运而生。

3.1.2 协同过滤推荐算法

协同过滤算法最早是由 Goldberg 在 1992 年提出来的，是当今构建推荐系统最流行的算法[5]。不同于基于内容的推荐算法，协同过滤算法不分析项目的内容信息，而是以用户对项目的评分为依据来产生推荐，经常被称为社会化推荐或者社会过滤。在协同过滤推荐系统中，对用户的推荐是基于用户先前对项目的评分和社会化网络中用户之间的相似度[35]。协同过滤以协同过滤技术为基础，寻找邻居用户（与目标用户兴趣相似），并将邻居用户喜好的项目推荐给目标用户。协同过滤推荐算法的社会学基础是人与人之间的口碑原则：用户可能会听从其他用户的建议去喜好或者厌恶一个项目，其他用户的建议可以表示成系统里的一个评分矩阵，具体内容在下文介绍。

系统给用户的推荐基于其他用户对项目的显式或者隐式的评分，用户对项目的显式评分有预定数值范围，很多系统将范围设置为 1 到 5 之间的整数值，其中评分 1 表示用户对项目及其讨厌，评分 5 表示用户对项目非常喜爱。隐式评分从

用户的购买记录、浏览记录和时间日志分析中得出，近年来使用用户标签来作为隐式评分的手段也非常热门。

协同过滤推荐算法通过用户-项目评分矩阵来表示用户对项目的偏好信息，矩阵中的行和列分别表示用户和项目，矩阵中的每个条目以数值的形式表示。在一个包含 m 个用户 n 个项目的系统里，每个用户 i 都对应一个对项目评分的向量 I_l，这些评分是通过显式或者隐式的方式从用户的评论、购买记录或者浏览历史中收集来的，I_l 中元素的个数为零到 n。因为用户不可能对系统中所有项目都评分或者用户不愿意评分，矩阵中有非常多的空项。协同过滤推荐算法的任务就是去寻找用户可能喜欢的项目列表，或者预测用户对一个未评分项目的评分，其输出可能是一个 top-N 推荐列表或者是一个评分值，评分预测暗示用户对其未评分项目的喜好程度，top-N 推荐列表是在用户未评分项目中用户最有可能喜欢的前 N 个项目，具体的过程如图 3.1。

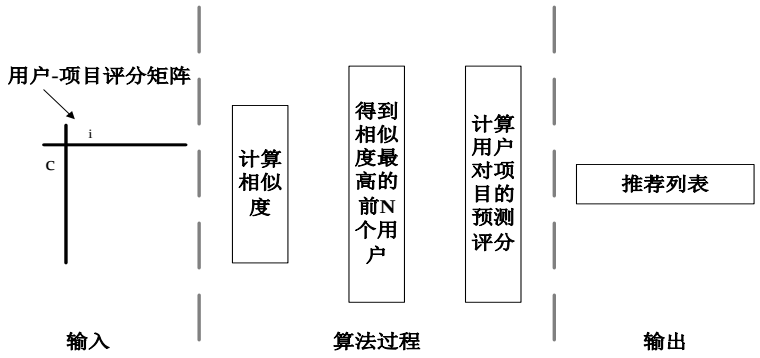

图 3.1 协同过滤推荐算法过程

Fig.3.1 Process of Collaborative Filtering Recommendation Algorithm

寻找和目标用户相似度最高的前 N 个用户是协同过滤推荐算法必须要做的步骤，不同的推荐算法，具体实现这个步骤的方法不一样。在大多数方法中，用户间相似度的计算基于他们的共同已评分项目之间的相似度。

根据推荐策略的不同，可将协同过滤推荐算法分为基于内存的协同过滤推荐算法和基于模型的协同过滤推荐算法。基于模型的协同推荐算法使用多种机器学习算法和数据挖掘模型来构建了一个评分预测模型，算法首先会对系统的评分数据进行分析，然后将分析好的数据通过评分预测模型进行处理来实现评分预测。一般的研究认为基于内存的协同过滤推荐算法优于基于模型的协同过滤推荐算法[36]。本论文研究的领域属于基于内存的协同过滤推荐算法，下面对该算法进行介绍。

基于内存的协同过滤算法利用系统内存中已有的用户评分数据，再通过一定的启发式方法实现评分预测，因此这种算法也被称为启发式协同推荐算法。根据

用户或项目的角度不同，基于内存的协同过滤算法可以分为基于用户的协同过滤算法（User-Based Collaborative Filtering）和基于项目的协同过滤算法（Item-Based Collaborative Filtering）[37]。

基于用户的协同过滤算法的主要步骤是寻找各用户的邻居用户，并根据邻居对自己未评分项目的评分来进行评分预测，其知算法的关键是寻找邻居的方法和根据邻居产生推荐的规则。

根据前文对邻居的介绍可知邻居是和目标用户相似度最高的 k 个用户，k 的具体数目由系统的特性决定，实际中往往会通过实验来确定。相似度的度量方法是这个步骤的核心，常使用的方法包括皮尔逊相关系数、夹角余弦相似度和 Jaccard 系数等，各个方法的计算公式和特性分析将在本章的第二节进行分析。在确定邻居用户后，需要根据邻居已评分且目标用户未评分的项目的评分值进行评分预测，评分预测的值通过效用函数 u 得到，常用的效用函数是将所有邻居用户的评分来进行聚合得到的。假设 Tab 为用户 u_a 和用户 u_b 的共同已评分项目集合，用户 u_a 关于目标项目 t_t 的邻居用户集合为 T_a^t，sim（a,b）为用户 u_a 和用户 u_b 的用户相似度，$\overline{r_i}$ 为用户对项目 t_t 的平均评分，那么用户 u_a 的评分预测函数表达式为：

$$u(a,t)=\underset{u_i\in T_a^t}{\text{aggregate}}(r_{it})=\overline{r_a}+\frac{\sum_{u_i\in T_a^t} sim(a,i)\times(r_{i,t}-\overline{r_i})}{\sum_{u_i\in T_a^t}|sim(a,i)|}$$

从上式可以看出基于用户的协同过滤算法的评分推荐是通过用户的邻居对项目的评分加权求和得到，而用户与邻居间的相似度作为该评分的权重，由此可见用户间相似度对基于用户的协同过滤算法的重要性。对于一个 m 个用户，n 个项目的推荐系统，计算用户间相似度的复杂度一般需要 O(mn)，当 m 和 n 数量级比较大时，这会很费时间，而且用户和用户之间的相似关系并不稳定，用户间相似度可能需要经常更新，因此基于用户的协同过滤算法并不适用于超大规模的推荐系统。针对这个问题，Sarwar 等人于 2001 年提出了基于项目的协同过滤算法，虽然计算项目间相似度的复杂度也达到了 O(mn)，但是项目间的相似度比用户间的相似度更加稳定，更加适用于现今大规模应用系统，比如 Amazon 的推荐系统用的就是基于项目的协同过滤算法[38]。

事实上，基于项目的协同过滤算法和基于用户的协同过滤算法的目的是一样的，只是解决问题的角度不同。基于项目的协同过滤算法假设用户未来的信息会保持和以往一致，因此算法着重挖掘项目间的相似度。基于项目的协同过滤算法的依据也是用户-项目评分矩阵，但分析的是矩阵的纵向数据，即项目的评分向量，通过各种相似度计算方法计算项目间相似度[39]。

对于目标用户 u_c 的未访问项目 t_s，在 u_c 的已访问项目中选择对于 t_s 的邻居项

目集合 $T_c^s = \{t_i | t_i \in T \wedge r_{ci} \neq 0\}$，通过加权求和，评分预测的表达式为：

$$r_{cs} = \underset{t_i \in T_c^s}{aggregate}(\mathrm{r}_{ci}) = \frac{\sum_{t_i \in T_c^s} sim(\mathrm{s},\mathrm{i})\,\mathrm{r}_{ci}}{\sum_{t_i \in T_c^s} |sim(\mathrm{s},\mathrm{i})|}$$

其具体的 top-n 推荐列表的构造过程为：首先对于目标用户 u_c 的已访问项目集合 $T_c = \{t_s | t_s \in T \wedge r_{cs} \neq 0\}$，根据项目相似度将 t_s 的邻居项目 T_s 加入到候选项目集合 $T_c^{'} = (\bigcup_{t_s \in T_c} T_s) - T_c$ 中，求出累加相似度 $sim(t_{s'}, T_c) = \sum_{t_s \in T_c} sim(s', s)$，选择 $sim(t_{s'}, T_c)$ 最大的 n 个未访问项目构建 top-n 推荐列表。

从上述介绍可看出无论是基于用户的协同过滤算法还是基于项目的协同过滤算法，相似度的计算都起着至关重要的作用。

协同过滤算法能很好地工作在项目是多媒体内容的环境中，例如在电影推荐系统中评分可以用来表示用户的偏好，但电影的文本信息是不可用的。在这种环境下使用基于内容的过滤算法无法生成高质量的推荐结果。协同过滤推荐系统能够产生令用户意外的推荐结果，这种推荐结果会使得用户对此产生兴趣。这种推荐的产生不是通过用户模型中文本匹配的结果，而是来自查找邻居用户的喜好，或者说是基于用户过去的偏好但不是过去偏好的内容信息。正是这个优点使得协同过滤算法被广泛使用在推荐系统中。

协同过滤推荐算法在发展过程中，存在两个问题：冷启动问题和稀疏性问题。如果没有处理好这些问题将导致协同过滤推荐系统的性能大大降低。

冷启动问题是指系统刚上线时系统中用户比较少，某些项目甚至一个评分都没有，根据协同过滤算法的原理可知这类项目无法被推荐给用户。事实上，冷启动问题存在于推荐系统的整个生命周期中，每当新项目的加入都会出现该问题。冷启动问题的关键是协同过滤推荐算法对评分数据过于依赖，一个解决方案是使用项目的文本信息和评分数据一起构成数据库，这样推荐结果就不单单依赖项目评分了[40]。

由于系统中项目数量往往非常多，用户的已评分项目数往往很少，结果就是用户-项目评分矩阵中有效数值的比例很低，一般评分矩阵的稀疏度达到 99%以上，寻找邻居用户是协同过滤推荐算法的关键步骤，评分矩阵的高稀疏度将导致确定邻居用户非常困难，这就是稀疏性问题，稀疏性问题严重影响推荐系统的推荐质量，是协同过滤推荐算法研究的热点问题。

3.1.3 混合推荐算法

从上文的分析介绍可知道基于内容的推荐算法和协同过滤推荐算法都是通过

单一的推荐策略来实现推荐的，各种都具有相互之间无法替代的优点，同时也单一策略有无法弥补的缺点。基于内容的推荐算法依赖于项目的文本信息，而协同过滤算法则没有这个依赖关系；协同过滤推荐算法无法很好地处理新项目的推荐，而基于内容的推荐算法不区分新老项目；基于内容的推荐算法无法实现联想性推荐，协同过滤算法能利用社会性信息很好地推荐联想性项目。许多推荐系统为了克服这两类算法的限制使用了两者结合的算法，称为混合推荐算法。不同的混合推荐算法用不同方法来结合基于内容的推荐算法和协同过滤推荐算法的产生推荐的方式，其中的一些结合策略如下：

① 两种算法分别独立执行，然后再加权组合各自的推荐结果形成混合推荐算法的结果，加权方法采用动态线性组合或者基于统计的组合。这种策略是混合推荐算法最简单的方式，能够充分发挥两种算法的优势，但是算法的时间复杂度较高，而且分别执行的两种算法需要统一的评分范围。例如 DailyLearner 系统根据用户的反馈向用户推荐新闻，其推荐系统所使用的方法就是使用这两种算法各自产生推荐，再根据置信度水平来选择推荐结果。

② 使用基于内容的推荐算法来辅助协同过滤推荐算法。Melville 等人在 2002 年提出并测试了这种称为内容驱动协同过滤推荐算法的方法[41]，该方法克服了单纯的协同过滤推荐算法的冷启动和稀疏性问题，内容驱动协同过滤推荐算法用一个基于内容的推荐算法来预测一个基于用户现有评分的伪用户评分向量，所有用户的伪用户评分向量被用于形成伪用户项目评分矩阵。然后再使用协同过滤算法在伪用户项目评分矩阵中找到相似用户来完成推荐。

③ 使用协同过滤推荐算法来辅助基于内容的推荐算法。这种混合推荐算法是 Soboroff 等人在 1999 年提出来的[42]，该算法使用协同过滤推荐算法的方法来构建基于内容的用户模型，并使用隐性语义索引（Latent Semantic Indexing）来将用户模型构建为词向量。隐性语义索引是使用潜在类别来查找用户和项目之间联系的一种基于概率模型的算法。

④ 在两种算法中相互使用对方的特征。如协同过滤算法可以集成基于内容的推荐算法所使用的用户和项目的内容信息，这样做可以提高对用户偏好信息的区分度，提高协同过滤算法的表达用户需求的能力。

⑤ 结合基于内容的推荐算法和协同过滤推荐算法的特点的统一模型来构建混合推荐算法。这个算法是 C.Basu 等人在 1998 年提出来的，他们使用评分信息和从项目信息中获取的文本信息来做归纳学习算法，这种算法既使用了协同过滤推荐算法中的评分信息，又用了基于内容的推荐算法中的内容文本信息[43]。

⑥ 对两种算法各种的特征放大，将一种算法的初级推荐模型应用到另一种算法中，称为元层次组合方法。比如可以将基于内容的推荐算法的用户模型作为协

同过滤推荐算法计算相似度的根据，这样做可以解决协同过滤推荐算法的稀疏性问题。这种混合推荐中一种算法需要另一种算法的元数据，是把两种算法结合的最紧密的混合方法。

⑦ 分阶段混合方法。将两种算法分为前后阶段，前阶段以自己的方法产生初级推荐结果，若该结果没有达到要求则再由后阶段对前阶段的推荐结果进行再处理。

3.2 主要的相似度计算方法

协同过滤算法在推荐系统的流行同时取得了飞跃的发展，成为主流的推荐算法。相似度的计算是协同过滤推荐算法最为关键的部分，对于基于用户的协同过滤算法，相似度的计算决定了邻居用户选择的准确性，同时也在效用函数中作为权值直接导致了最后的推荐结果，因此相似度计算方法的选择是推荐系统构建的核心问题。有很多相似度计算方法已被协同过滤推荐算法使用，由于各种推荐系统的实际情况不一样导致其用户-项目评分矩阵特征的不同，所以在选择相似度计算方法之前应该全面仔细地分析具体用户-项目评分矩阵的特征，再选择具有针对性的计算方法。

对于基于用户的协同过滤推荐算法来说，是将两个用户的共同已评分项目的评分向量 $\vec{a}$ 和 $\vec{b}$ 的相似度 w_{ab} 来作为用户 a 和用户 b 的相似度。类似的，基于项目的协同过滤推荐算法是将共同评分过项目 i 和项目 j 的用户向量 $\vec{i}$ 和 $\vec{j}$ 来作为项目 i 和项目 j 的相似度。

本节介绍几种常用的相似度计算方法并在此基础上分析各种计算方法的特点和局限性。

3.2.1 基于皮尔逊相关系数的相似度

皮尔逊相关系数（Pearson Correlation Coefficient）是统计学里线性回归模型中的概念，用来衡量两个向量的线性相关性，是一个无量纲指标，其取值范围介于-1到 1 之间，若值为 0 则表示两向量不相关，值为负数表示两向量负相关，值为正数表示两向量正相关。皮尔逊相关系数作为一种度量向量间相关性的方法，经常用在计算机领域中计算实体的相似度。

在基于用户的协同过滤推荐算法中，用基于皮尔逊相关系数的相似度计算用户间相似度的公式如下所示，其中 PCCSim(a,b) 表示用户 a 和用户 b 之间的基于皮尔逊相关系数的相似度，T_{ab} 表示用户 a 和用户 b 的共同已评分项目集合，r_{ai} 和 r_{bi} 分别表示用户 a 和用户 b 对项目 i 的评分，$\overline{r_a}$ 和 $\overline{r_b}$ 分别表示用户 a 和用户 b 对评分矩阵中所有项目的评分均值：

$$\mathrm{PCCSim}(a,b)=\frac{\mathrm{cov}(R_a,R_b)}{\sigma(R_a)\sigma(R_b)}=\frac{\sum_{t_i\in T_{ab}}(r_{ai}-\overline{r_a})(r_{bi}-\overline{r_b})}{\sqrt{\sum_{t_i\in T_{ab}}(r_{ai}-\overline{r_a})^2}\sqrt{\sum_{t_i\in T_{ab}}(r_{bi}-\overline{r_b})^2}}$$

同样的，在基于项目的协同过滤推荐算法中，用基于皮尔逊相关系数的相似度计算项目间相似度的公式如下所示，其中 PCCSim(u, v) 表示项目 a 和项目 b 之间的基于皮尔逊相关系数的项目相似度，T_{uv} 表示对项目 u 和项目 v 已评分的所有用户的集合，r_{iu} 和 r_{iv} 分别表示用户 i 对项目 u 和项目 v 的评分，$\overline{r_u}$ 和 $\overline{r_v}$ 分别表示所有用户对项目 u 和项目 v 的评分均值：

$$\mathrm{PCCSim}(u,v)=\frac{\mathrm{cov}(R_u,R_v)}{\sigma(R_u)\sigma(R_v)}=\frac{\sum_{t_i\in T_{uv}}(r_{iu}-\overline{r_u})(r_{iv}-\overline{r_v})}{\sqrt{\sum_{t_i\in T_{uv}}(r_{iu}-\overline{r_u})^2}\sqrt{\sum_{t_i\in T_{uv}}(r_{iv}-\overline{r_v})^2}}$$

图 3.2 和图 3.3 分别显示了基于用户的协同过滤推荐算法中用户相似度的计算和基于项目的协同过滤推荐算法中项目相似度的计算，用户-项目评分矩阵中一行表示一个用户向量，一列表示一个项目向量。计算用户间的相似度就是计算相应的两个行向量的相关系数，计算项目间的相似度就是计算相应的两个列向量的相关系数。

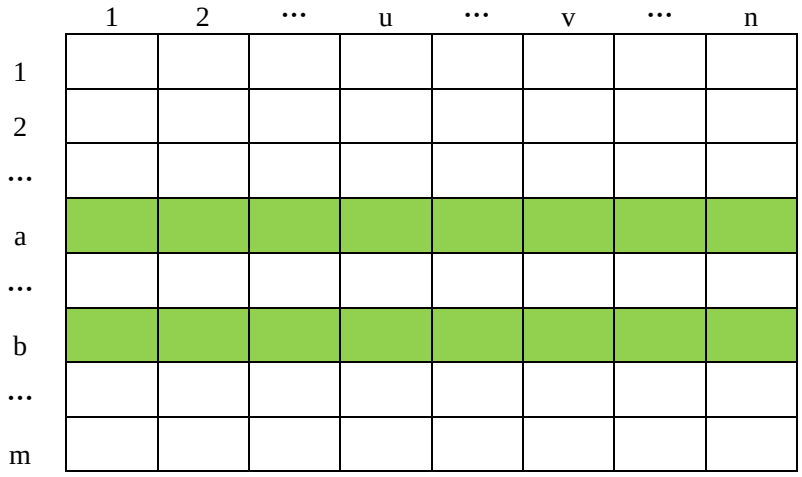

图 3.2 基于用户的协同过滤算法中用户相似度的计算

Fig.3.2 Calculation of User Similarity in Collaborative Filtering Algorithm Based on Users

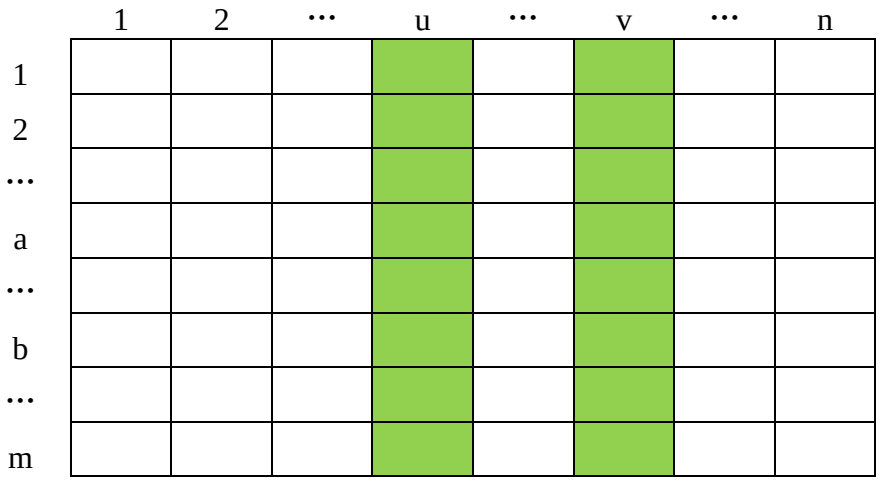

图 3.3 基于项目的协同过滤算法中项目相似度的计算

Fig.3.3 Calculation of User Similarity in Collaborative Filtering Algorithm Based on Items

3.2.2 基于夹角余弦的相似度

在基于协同过滤的推荐算法中，用户或项目间的相似度是通过相对应的向量间的相似度来计算的，而欧几里德空间中就存在计算向量间相似度的方法，包括欧氏距离和向量夹角的余弦相似度。欧氏距离是最简单的度量相似度的方法，表示向量空间中点间的绝对距离，其值和向量的数值大小直接相关，而向量间的夹角度量的向量在空间中方向上的差异，是一个相对的差异。图 3.4 描述了欧式距离和夹角余弦相似度直接的不同，保持 A 点不动，B 点沿着向量的方向移动，那么向量$\overrightarrow{OA}$和向量$\overrightarrow{OB}$间的欧氏距离会随着增大或减小，而两向量间的夹角没有变化，即两向量间的余弦相似度保持不变。因此可以知道欧式距离描述绝对差异，比较适合比较各个维度的具体差异，余弦相似度从向量的相对差异——方向上区分向量间的差异，用来度量协同过滤推荐系统中的用户相似度更加合理。

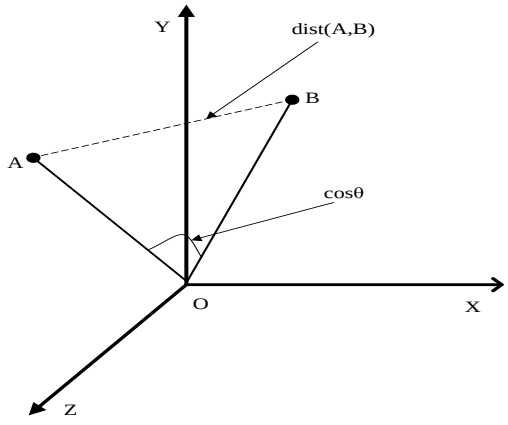

图 3.4 两点距离

Fig.3.4 Distance between Two Point

在一个 n 个元素组成的 n 维欧几里德空间中，夹角余弦相似度的定义是任意两个向量之间夹角的余弦值。在实际使用前需要对空间中所有向量无量纲化并使空间中元素取值为正，夹角余弦相似度的取值范围介于 0 到 1 之间，该值越大表示两向量之间的夹角越小，即两向量的相似度越大，当取值为 1 时两向量完全相同。

余弦相似度起初来源于信息检索领域，在该领域每个文本项目有对应的词频向量，向量空间模型通过计算任两个词频向量的余弦相似度来度量相应文本项目间的相似度。

把夹角余弦相似度用到用户-项目评分矩阵中，如对于基于用户的协同过滤推荐算法可以把用户对所有项目的评分作为一个用户向量，若系统中有 m 个用户，则可以把用户相似度计算问题替换成在 m 维用户向量空间中计算向量间的夹角余弦值，其值越大，用户间的相似度越高。

在基于用户的协同过滤推荐算法中，用基于夹角余弦的相似度计算用户相似度的公式如下所示，其中 $COSSim(a,b)$ 表示用户 a 和用户 b 之间的基于夹角余弦的相似度，T_{ab} 表示用户 a 和用户 b 的共同已评分项目集合，r_{ai} 和 r_{bi} 分别表示用户 a 和用户 b 对项目 i 的评分，$\overrightarrow{R_a}$ 和 $\overrightarrow{R_b}$ 分别表示对项目 u 和项目 v 评分的所有用户组成的向量：

$$COSSim(a,b)=\frac{\overrightarrow{R_a}\cdot\overrightarrow{R_b}}{\left\|\overrightarrow{R_a}\right\|\left\|\overrightarrow{R_b}\right\|}=\frac{\sum_{t_i\in T_{ab}} r_{ai}r_{bi}}{\sqrt{\sum_{t_i\in T_{ab}} r_{ai}^{\,2}}\sqrt{\sum_{t_i\in T_{ab}} r_{bi}^{\,2}}}$$

相应的，在基于项目的协同过滤推荐算法中，用基于夹角余弦的相似度计算项目相似度的公式如下所示，其中 $COSSim(u,v)$ 表示项目 u 和项目 v 之间的基于夹角余弦的相似度，T_{uv} 表示对项目 u 和项目 v 已评分的所有用户的集合，r_{iu} 和 r_{iv} 分别表示用户 i 对项目 u 和项目 v 的评分，$\overrightarrow{R_u}$ 和 $\overrightarrow{R_v}$ 分别表示对项目 u 和项目 v 评分的所有用户组成的向量：

$$COSSim(u,v)=\frac{\overrightarrow{R_u}\cdot\overrightarrow{R_v}}{\left\|\overrightarrow{R_u}\right\|\left\|\overrightarrow{R_v}\right\|}=\frac{\sum_{t_i\in T_{uv}} r_{iu}r_{iv}}{\sqrt{\sum_{t_i\in T_{uv}} r_{iu}^{\,2}}\sqrt{\sum_{t_i\in T_{uv}} r_{iv}^{\,2}}}$$

在实际的应用环境中，因为推荐系统中各个用户对项目的评分标准不同，比如用户 a 倾向于打高分，用户 b 倾向于打低分，对于同一个用户 a 和用户 b 都喜好的项目 i，用户 a 打出的分往往高于用户 b，这时夹角余弦相似度就不能很好地处理这种状况。这时如果把用户向量中的每个评分都减去该用户的均值评分，就能修正这个问题，同时这样做之后夹角余弦相似度又演化为基于皮尔逊相关系数的

相似度，而具体选用哪种相似度计算方法需要针对具体问题具体分析。

3.2.3 基于 Jaccard 相关系数的相似度

Jaccard 相关系数由 Jaccard 和 Paul 在 1901 年提出，最初在统计学中用来比较样本数据中集合的相似度和差异度，其值为两集合的交集大小和并集大小的比值：

$$\mathrm{JACSim(A,B)}=\frac{|A\cap B|}{|A\cup B|}$$

Jaccard 相关系数很适合度量具有离散维度特征向量间的相似度，协同过滤推荐系统里很大部分数据的表现形式是离散型的，比如用户对项目的评分，常常是 1 到 5 的整数，或者是有些协同过滤推荐系统使用的喜欢和不喜欢二值标注的模式。图 3.5 展示了一个使用二值标注模式得到的用户-项目评分矩阵，其中“1”、“0”和“-1”分别表示“喜好”、未标记和“不喜欢”。

	1	2	…	u	…	v	…	n
1	0	1		-1		0		
2	1	1		0		0		1
…								
a	-1	1		0		1		0
…								
b	0	-1		1		-1		0
…								
m	-1	-1		0		-1		0

图 3.5 一个二值标注模式的用户-向量评分矩阵

Fig.3.5 A Binary Label Mode Users-Vector Rating Matrix

根据 Jaccard 相关系数的定义可知图 3.5 中的评分矩阵中用户 a 和用户 b 的相似度的计算公式为：

$$\mathrm{JACSim}(a,b)=\frac{\sum_{i=1}^{n}1_{bool}\{\mathrm{r}_{ai}=\mathrm{r}_{bi}\wedge\mathrm{r}_{ai}\neq 0\}}{\sum_{i=1}^{n}1_{bool}\{\mathrm{r}_{ai}\neq 0\wedge\mathrm{r}_{bi}\neq 0\}}$$

其中指数函数$1_{bool}\{\mathrm{x}\}$的定义为：$1_{bool}\{\mathrm{x}\}=\begin{cases}1 & \text{while x is true}\\ 0 & \text{while x is false}\end{cases}$，公式中$\mathrm{r}_{ai}$和

r_{bi} 分别为用户 a 和用户 b 对项目 i 的标注。

3.2.4 传统相似度计算方法的局限

上面介绍的三种相似度计算方法在协同过滤推荐算法中取得了巨大的成功，不过随着应用环境的多样化和应用研究的深入，这些方法也被发现存在一些局限性，比如这三个方法都没有很好的办法来解决稀疏性问题，稀疏性问题会严重影响用户间相似度的计算，甚至根本无法计算。

当用户间共同已评分项目只有一个时，使用 Jaccard 相关系数来度量用户相似度是非常不准确地，并且这个时候使用夹角余弦相似度计算的结果一直是 1。当用户对用户间共同已评分项目的评分都一样时，比如两个用户向量 $\vec{a}$ 和 $\vec{b}$ 分别为（1,1,1,1,1）和（5,5,5,5,5），可以知道这两个用户的相似度是很低的，基于皮尔逊相关系数的相似度无法计算，夹角余弦相似度的计算结果一直为 1，这时这两个相似度计算方法都无法准确度量用户间真实的相似性。

皮尔逊相关系数度量的是用户向量间的线性相关程度，当用此方法计算得到两个用户向量的相似度很高时，但用欧氏距离计算后可能得到相反的结果。比如，假设两个用户向量 $\vec{a}$ 和 $\vec{b}$ 分别为（2,1,2,1,2）和（5,4,5,4,5），则使用基于皮尔逊相关系数的方法得到用户间相似度为 1，而实际上这两个用户并不相似。同时当处理非线性情况时，夹角余弦相似度和皮尔逊相关系数的处理能力会大幅度减弱，当项目数很少时，这两种方法也无法准确地反映用户间的相似关系。

在有些情况下，基于皮尔逊相关系数的相似度和夹角余弦相似度的计算结果会偏差很大。当两用户向量分别为（1,0,0）和（4,3,2）时，两用户的相似度是很小的，但是用皮尔逊相关系数计算后的结果却很大；当两用户向量分别为（3,5,4）和（5,4,5）时，两用户较为相似，而用皮尔逊相关系数得到的结果却比较小；当用户 a、用户 b 和用户 c 的用户向量分别为（1,1）、（5,5）和（1,2）时，用户 a 和用户 c 非常相似，但用余弦相似度计算得到的结果小于用户 a 和用户 b 间的计算结果。

3.3 推荐算法评价

推荐算法的评价方法是比较推荐算法优劣的依据，因此研究推荐算法评价是是推荐系统研究的重要组成部分。一般推荐算法的改进过程是：首先根据评价方法得到现有算法的评价指标数据，然后根据这些数据进行针对性的改进。随着推荐系统的应用领域逐渐扩大，导致对推荐算法的评价缺乏统一的标准，不同的应用环境需要特征不同的推荐算法，在开发算法的时候需要选择不同的实验数据集并且对推荐算法的评价需要选择具有实用性、针对性强的指标。因此推荐算法的

评价主要体现在两方面：实验数据集的选择和评价指标的选择。本节先后介绍了推荐算法实验常用的数据集和评价指标。

3.3.1 实验数据集

推荐算法的评价需要的实验数据集可以是实际收集的真实数据集，也可以是用经过加工的具有某些特征的数据集，这两种数据集各有特点。加工过的数据集一致性很好，能够人为去控制数据的特性，但加工过的实验数据集一般只用来评价特定应用领域的算法。真实数据集能够反映推荐系统的真实数据分布，但需要很长的时间去收集，因此形成了一些通用的真实数据集。

推荐算法经过长时间的发展，研究人员收集了大量的真实数据，这些真实数据能在一定程度上反映各个应用领域中真实数据的特征，因此具有比较强的代表性。研究人员可以根据实际的环境来选择这些真实数据集，常用真实数据集包括：

① MovieLens 电影评分数据集：该数据集是由美国明尼苏达大学的 GroupLens 研究小组创建并维护，GroupLens 研究小组从 MovieLens 电影评分网站收集评分数据并将其划分为三个集合，每个数据集都包括电影信息、用户信息、用户对电影的评分和评分的时间戳。用户评分的取值范围介于 1 至 5 之间，评分越高表示用户对电影的偏好程度越高。目前公布的三个评分数据集中，第一个也是规模最小的数据集包含 943 位匿名用户对 1682 部电影的十万条评分数据，其中每个用户都至少对 20 部电影进行了评分；第二个较大的数据集包含 6040 位匿名用户对 3952 部电影的一百万条评分数据，其中每个用户都至少对 20 部电影进行了评分；规模最大的数据集包含 71567 位匿名用户对 10681 部电影的一千万条评分数据，该数据集还包含用户对电影的十万条标签数据。MovieLens 数据集是协同过滤推荐算法研究领域普遍采用的实验数据库，本文对算法的研究也采用了 MovieLens 数据集作为实验数据集。

② Netflix 电影评分数据集：该数据集来源于美国著名电影网站 Netflix，Netflix 在 2006 年开展了 Netflix Prize 推荐系统大赛，公布了一个由 480189 位用户对 17770 部电影的上亿条评分的数据集，并斥资一百万美元奖金希望能够使基于此数据集的推荐算法准确度提高 10%以上。数据集包含了用户打分的时间信息，该时间跨度从 1999 到 2005 年。数据集里的评分取值范围介于 1 到 5 之间，是个离散值，分数越高表明用户对电影越喜欢。该数据集是研究中所使用数据集中规模最大的一个，最新的版本不对外开放。

③ Jester 笑话评分数据集：Jester 数据集由加州大学伯克利分校的 Goldberg 等人通过 Jester 网站收集，该网站向用户推荐笑话，用户可以给笑话打分。数据集以用户-笑话评分矩阵形式储存，该矩阵包含 73421 位匿名用户对 100 个笑话的四百一十万条评分数据，评分的取值范围介于-10 到 10 之间，是个连续值。Jester 数

据集结构简单，数据稠密，也是较为常用的实验数据集。

④ Delicious 书签数据集：Delicious 是目前世界上最大的书签网站，网站的用户可以对互联网上的任何一个页面打标签。该数据集由德国的研究人员公布，数据集每行是一个四元组，包含用户名、网页地址、时间和书签。数据集非常庞大，包括 66 万名用户的 250 万个标签，在实验中往往只取其中用户对特定域名的数据，常抽取的域名包括 wiki、Sourceforge、wsj 等。

⑤ Book-Crossing 图书评分数据集：该数据集是由 Cai-Nicolas Ziegler 等人在 2004 年通过网络爬虫从 Book-Crossing 图书社区上采集的用户对图书的评分信息。该数据集包含了用户和图书的基本信息及用户对图书的评分，共 278858 位用户对 271379 本图书的上百万条评分数据，其中包括显性评分和隐形评分，评分的取值范围介于 1 至 10 之间，是离散值。

3.3.2 评价指标

推荐算法的评价指标有很多，主要针对推荐质量和推荐速度。推荐速度的评价指标包括运行时间和时间复杂度，运行时间的计算从推荐算法运行开始，直到推荐结果产生了为止。算法时间复杂度指的是算法中基本操作的重复次数。相对来说人们对推荐质量更为关心，常用的评价推荐质量的指标有：准确性、覆盖度、多样性、惊喜度等。下面一一介绍这些指标。

① 准确性：推荐算法的准确性描述的是算法结果和用户真实偏好间的相似程度，是推荐算法评价指标中最重要的一个指标。准确性度量有量化指标，但由于推荐算法的功能和目的多样性，导致准确性的度量标准并没有统一。根据推荐算法任务的不同，准确性度量可以分为预测准确性和分类准确性，预测准确性针对的是评分预测型的推荐算法，分类准确性针对的是二元推荐型推荐算法。

预测准确性是使用最为广泛的评价标准，常用的度量标准包括 MAE（平均绝对误差）、NMAE（归一化平均绝对误差）和 RMSE（均方根误差）。MAE 统计预测评分和实际评分间的实际差距，计算后得到的 MAE 值越小，预测的准确性越高。对于真实评分 $R=\{r_1, r_1, \cdots, r_n\}$，推荐算法的预测评分为 $P=\{p_1, p_1, \cdots, p_n\}$，那么 MAE 的计算表达式为：

$$MAE = \frac{\sum_{i=1}^{n} |p_i - r_i|}{n}$$

MAE 计算简单，实用性强，是大部分研究所用的评价标准，本文的算法评价使用的也是 MAE 标准。由于各种实验数据集的评分数值范围不一样，比如 MovieLens 数据集的评分数值范围介于 1 至 5 之间，Jester 数据集的评分数值范围介于-10 至 10 之间，会导致使用不同数据集进行实验的算法计算 MAE 后的结果不具有可比性。为了克服这种缺陷，Goldberg 等人提出了 NMAE，该标准对 MAE

的结果进行了归一化处理，得到了一个相对的准确性度量数值。假如评分的最大值和最小值分别为 r_{max} 和 r_{min}，那么 NMAE 的计算表达式为：

$$NMAE = \frac{MAE}{r_{\max} - r_{\min}} = \frac{\sum_{i=1}^{n} |p_i - r_i|}{n \times (r_{\max} - r_{\min})}$$

度量预测准确性的方法还有 RMSE，其是预测值与真实值偏差的平方和除以评分个数的平方根，其计算表达式为：

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\mathrm{p}_i - \mathrm{r}_i)^2}{n}}$$

对于二元型推荐，算法只考虑用户可能喜好的项目，而不关心用户喜好该项目的程度，度量这种算法用的是分类准确性方法。分类准确性统计算法推荐正确和错误的次数，其度量指标和分类算法使用的很类似，包括准确率（Precision）和召回率（Recall）。确率量化了符合用户偏好的推荐结果所占的比例，表示为有效推荐和所有推荐的比值。对于用户 u，R_u 是推荐结果的集合，T_u 是用户喜好的集合，准准确率的计算表达式为：

$$\mathrm{Re}\,call = \frac{\sum_u |R_u \cap T_u|}{\sum_u |T_u|}$$

召回率能够反映推荐的全面性，是提供给用户的符合其偏好的项目数与所有推荐结果的比值，其表达式为：

$$\mathrm{Pr}\,ecision = \frac{\sum_u |R_u \cap T_u|}{\sum_u |R_u|}$$

② 覆盖度：覆盖度（Coverage）反映推荐系统在解决长尾问题方面的成效，一个好的推荐算法应该推荐更多长尾中的并符合用户偏好的项目给用户。覆盖度度量推荐给用户的项目占所有项目的比重，假设 U 是所有用户的集合，I 是所有项目的集合，R_u 是推荐给用户 u 的推荐结果集合，覆盖度的计算表达式为：

$$Coverage = \frac{\left|\bigcup_{u \in U} R_u\right|}{|I|}$$

覆盖度高的算法，其推荐系统能够更好地挖掘长尾中的项目，在商业系统中就能够实现更好的商业效益，更可贵的是覆盖度和准确性之间并不存在相关性，同时提高这两个指标是推荐算法努力的方向。

③ 多样性：多样性（Diversity）高的推荐结果能够提高用户对系统使用的满意度，而传统的推荐算法的多样性都是很低的，针对这个问题，研究人员提出了很多多样性高的推荐算法，并且也提出了度量多样性的方法。

多样性的度量是通过计算用户的推荐结果里的项目两两之间的相似度来实现的，如果一个推荐结果集合中项目两两相似度越高，那么该推荐的多样性越低，

反之多样性越高。假设 R_u 是用户 u 的推荐项目集合，N 是集合 R_u 中项目的个数，U 表示用户集合，sim(I,j)表示项目 i 和项目 j 之间的相似度，那么用户 u 的推荐结果的多样性的计算表达式为：

$$Diversity(\mathrm{u}) = \frac{\sum_{i,j \in R_u, i \neq j} (1 - \mathrm{sim}(\mathrm{i},\mathrm{j}))}{\frac{1}{2} N(\mathrm{N}-1)}$$

整个推荐算法的多样性的计算表达式为：

$$Diversity = \frac{1}{|U|} \sum_{u \in U} Diversity(\mathrm{u})$$

惊喜度：系统系统里的惊喜度度量的是用户对于推荐结果感到非常意外，同时又觉得非常喜欢的程度。随着商业互联网竞争的加剧，现在的推荐系统都很关注能够提高用户体验的惊喜度。惊喜度意味着推荐结果不仅要准确，而且要新颖，目前还没很好的量化公式来计算这个指标，一般的方法是用户调查，让用户来对推荐结果的惊喜度进行评分。

3.4 本章小结

本章首先介绍了推荐算法的三个类别，其中着重介绍了协同过滤推荐算法，接着对主要的相似度计算方法做了介绍并指出了各种计算方法的局限性，最后介绍了推荐算法评价所使用的实验数据集和评价指标。

4 考虑物品相似权重的用户相似度计算方法及其分析

本文在上面的章节中介绍了推荐算法和相似度的计算方法，通过介绍可以知道相似度计算方法在协同过滤推荐算法中起着非常重要的作用。但是上面章节中介绍的相似度计算方法仍然存在着一些不足，本章针对这些不足提出了一种改进的用户相似度计算方法，并通过实验分析证明了该方法的有效性。

4.1 引言

协同过滤推荐算法是目前研究和使用很广泛的一类算法，它模拟了现实生活中人们分享爱好的方式，具有社会学理论基础，是当前社会化网络环境下主要使用的推荐算法。协同过滤推荐算法分为基于用户的协同过滤推荐算法和基于项目的协同过滤推荐算法[44]。

基于用户的协同过滤推荐算法根据用户-项目评分矩阵来计算用户间的相似度，把其中相似度最高的 N 个用户确定为目标用户的邻居，然后根据效用函数得到邻居用户高评分且目标用户未评分的项目，再将这些项目推荐给目标用户。基于项目的协同过滤推荐算法的计算过程和此类似，首先计算项目间的相似度，找到邻居项目，再把目标项目推荐给对邻居项目高评分且对目标项目未评分的用户[45]。对于基于用户的协同过滤推荐算法，由于其中寻找邻居用户和计算效用函数步骤中都用到了计算得到用户相似度，相似度的计算方法对推荐结果影响非常大。

传统的用户间相似度计算方法依据的是两个目标用户的共同已评分项目的评分，这些共同已评分项目在计算中的权重是一样的。然而这些计算方法没有针对后面的步骤如寻找邻居用户做特别的处理。比如，两个用户对共同已评分项目中的项目都给予同样的低评分，用传统的用户相似度计算方法计算后这两个用户的相似度会很高，推荐系统关心的是用户可能会喜欢的项目，两个用户都讨厌一些项目不代表一个用户会喜欢另外一个用户喜欢的项目，因此用户间共同高评分项目和用户间共同低评分项目应该给予不同的权重。同时传统的用户相似度计算方法没有考虑项目间的类群关系，不同类群中的项目的权重也应该不同。这些问题都将会降低推荐系统的推荐质量。

在以往的研究中，基于用户的协同过滤推荐算法和基于项目的协同过滤推荐算法很少有交集，因为两者看待问题的角度不一样。针对上面提到的问题，本章提出了一个考虑用户间共同已评分项目相似权重的用户协同过滤算法，其中用到了基于项目的协同过滤推荐算法中的项目相似度，用户相似度计算方法的核心思想是增加一个衡量项目在用户间共同评分项目集合中相似权重的因子。

下面的章节先分析这些问题，然后给出一个新的计算用户相似度的方法，并给出计算方法的算法代码，最后给出并分析了一个在 MovieLens 数据集中的实验得出的结果。

4.2 考虑物品相似权重的用户相似度

基于用户的协同过滤推荐算法需要计算用户间的相似度，并选择相似度最大的 top-k 个用户作为目标用户的邻居，然后通过评分聚合函数 u 的计算获得用户对未评分项目的预测评分，选取预测评分项目中最大的 n 个未访问项目推荐给用户。用户 u_a 对项目 t_t 的评分聚合函数 u(a,t)通过下面的公式计算：

$$u(a,t)=\overline{r_a}+\frac{\sum_{u_i\in T_a^s} sim(a,i)\times(r_{i,t}-\overline{r_i})}{\sum_{u_i\in T_a^s}\left|sim(a,i)\right|}$$

用户相似度决定了用户邻居的选择，同时也是评分聚合函数中的权重因子，因此用户相似度的计算在基于用户的协同过滤推荐算法中占有重要地位。然而目前的用户相似度计算方法存在两个问题：第一，推荐系统最后向用户推荐的是预测评分最大的 n 个未访问项目，在计算用户相似度时，用户间共同低评分项目对最后的推荐结果没有帮助，反而会间接降低用户间高评分项目的权重；第二，项目是分类群的，对于用户间共同已评分项目中的一个类群，类群中共同高评分项目越多，该类群就越有可能为两用户共同喜好。如能在计算用户相似度的时候考虑这两个问题，相应地提高某些项目的权重，那么就可能改善基于用户的协同推荐算法的性能。

针对上述问题，本文提出两个推断：

推断 1：用户间共同高评分项目的权重应大于共同低评分项目的权重。

推断 2：在用户间的共同高评分项目中，属于同一个类群的项目越多，那么该类群越有可能被两用户同时所喜好，相应地，该类群中项目的权重应该更大。

下一小节给出项目加权的方法和加权后的用户相似度计算表达式。

4.2.1 共同已评分项目加权方法

在加权之前需要根据用户-项目评分矩阵计算出系统中项目间相似度矩阵 H_{nn}，其中项目相似度数据根据传统相似度计算方法计算得来。在计算用户 u_a 和用户 u_b 的相似度时，假设 T_{ab} 为两用户的共同已评分项目集合，p 是 T_{ab} 中项目的个数，$r_{a,i}$ 是用户 u_a 对项目 t_i 的评分，$r_{b,i}$ 是用户 u_b 对项目 t_i 的评分，$h_{i,j}$ 是项目 t_i 和项目 t_j 的相似度，对于项目 $t_i\in T_{ab}$，若只考虑推断 1，则其权值可以为：

$$w_1(i)=\frac{(r_{a,i}+r_{b,i})}{2}$$

加上两个用户对项目的评分之和的权值后，两用户对共同已评分项目的评分越高，权值就越大，起到了提高共同高评分项目权重的同时降低共同低评分项目权重的作用。

若只考虑推断 2，则其权值可以为：

$$w_2(i)=\frac{\sum_{t_j\in T_{ab}且j\neq i} h_{i,j}}{p-1}$$

式中的“ $\sum_{t_j\in T_{ab}且j\neq i} h_{i,j}$ ”是两用户共同已评分项目 t_i 与其他所有 T_{ab} 中的项目的相似度之和，这个值越大说明该项目所属类群在 T_{ab} 中拥有越多的项目，同时这个类群就越可能是两用户共同喜好的类群，相应的这个类群中的项目也获得了更高的权重。

若同时考虑推荐 1 和推断 2，则权重可以是上面两个权重 W_1 和 W_2 的乘积，即：

$$w_3(i)=\frac{(r_{a,i}+r_{b,i})}{2}\times\frac{\sum_{t_j\in T_{ab}且j\neq i} h_{i,j}}{p-1}$$

这样得到的权重可以兼顾推断 1 和推断 2，但具体的效果需要经过实验验证。W_1 加权方法不需要用到项目相似度，其算法的复杂度比 W_2 和 W_3 加权方法大大简化，适合在对算法复杂度要求较高的环境下使用。

4.2.2 加权用户相似度计算方法

考虑了上述权值之后，用户相似度计算公式需要做相应的调整，下面具体针对皮尔逊相关系数和夹角余弦相似度为例，将相似度计算表达式修改为：

① 加权后的皮尔逊相关系数：

$$PCCSim(a,b;w)=\frac{\mathrm{cov}(R_a,R_b;w)}{\sigma(R_a;w)\sigma(R_b;w)}$$

$$=\frac{\sum_{t_i\in T_{ab}} w_{a,b}(i)(r_{a,i}-m(a;w))(r_{b,i}-m(b;w))}{\sqrt{\sum_{t_i\in T_{ab}} w_{a,b}(i)(r_{a,i}-m(a;w))^2}\sqrt{\sum_{t_i\in T_{ab}} w_{a,b}(i)(r_{b,i}-m(b;w))^2}}$$

其中 m(a;w)为用户 u_a 对 T_{ab} 中物品评分加权后的平均值：$m(a;w)=\frac{\sum_{t_i\in T_{ab}} w_{a,b}(i)r_{a,i}}{\sum_{t_i\in T_{ab}} w_{a,b}(i)}$。

② 加权后的夹角余弦相似度：

$$COSSim(a,b;w)=\frac{\sum_{t_i\in T_{ab}} w_{a,b}(i)r_{a,i}r_{b,i}}{\sqrt{\sum_{t_i\in T_{ab}} w_{a,b}(i)r_{a,i}^{\ 2}}\sqrt{\sum_{t_i\in T_{ab}} w_{a,b}(i)r_{b,i}^{\ 2}}}$$

下面用一个例子来展示上文提出的用户相似度计算方法，方法使用 W_3 加权方法。假设有四个用户 u_a、u_b、u_c 和 u_d，计算 Sim(a,b)、Sim(a,c)和 Sim(a,d)。为计算方便，使 $T_{ab}=T_{ac}=\{t_1, t_2, t_3, t_4, t_5, t_6, t_7\}$，$T_{ad}=\{t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}\}$，用户对这些项目的评分见表 4.1 和表 4.2。T_{ab}（T_{ac}）中项目的项目相似度见表 4.3。T_{ad} 中项目的项目相似度见表 4.4。表 4.5 显示的是使用传统皮尔洛相关系数和使用 W_3 作为权值的皮尔洛相关系数计算出来的用户相似度。对于用户 u_a、u_b 和用户 u_a、u_c，$T_{ab}=T_{ac}$，u_a 和 u_c 之间共同高评分项目较多，u_a 和 u_b 之间共同低评分较多，若选定 u_c 为 u_a 的邻居来推荐项目，那么效果不会很好。但是如表 4.5 中数据所示，Sim(a,b)=0.8074，Sim(a,c)= 0.8189，传统用户相似度计算方法并不能很好地区分这两种情况。而使用本文提出的计算方法得到 Sim(a,b;w)= 0.5313，Sim(a,c;w)= 0.8641，加权后的用户相似度计算方法增加了用户间共同高评分项目的权值，使得类似于 u_a 和 u_c 这样的情形无法得到高相似度。计算结果表明本文提出的相似度计算方法能将这两种情况的不同很好地表现出来，且 Sim(a,c;w)更高，符合现实情形。对于用户 u_a、u_c 和用户 u_a、u_d，虽然 $T_{ac}\neq T_{ad}$，但是 u_d 对项目的评分数据和 u_c 对项目的评分数值相同，从表 4.3 中可看出$\{t_1$、$t_2\}$、$\{t_3$、$t_5\}$、$\{t_4$、$t_6\}$和 t_7 可能是四个类型的项目，而从表 4.4 中可看出$\{t_8$、t_9、t_{11}、t_{12}、$t_{14}\}$和$\{t_{10}$、$t_{13}\}$可能是两个类型的项目，因此如果用户 u_a 和 u_d 对$\{t_8$、t_9、t_{11}、t_{12}、$t_{14}\}$中的一个项目有较高的评分，则这个项目的权值应该大于两用户对$\{t_{10}$、$t_{13}\}$中的一个项目有较高评分时的那个项目的权值，同理 T_{ad} 中$\{t_8$、t_9、t_{11}、t_{12}、$t_{14}\}$的权值应该大于 T_{ac} 中项目的权值。从表 4.5 中可知用传统用户相似度计算方法得出 Sim(a,d)=0.8252，没能反映出这个情况，而本文提出的计算方法得到 Sim(a,d;w)= 0. 9172，大于 Sim(a,c)= 0.8189，计算结果符合预期。

表 4.1 用户 u_a、u_b 和 u_c 对项目$\{t_1, t_2, t_3, t_4, t_5, t_6, t_7\}$的评分

Table 4.1 Rating on Items $\{t_1, t_2, t_3, t_4, t_5, t_6, t_7\}$ By u_a,u_b and u_c

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
u_a	1	2	4.5	5	0.5	5	3
u_b	1	1.5	3	2	1	3	2.5
u_c	3	4	4.5	4.5	3	5	1

表 4.2 用户 u_d 对{ t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}}的评分

Table 4.2 Rating on Items { t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}} By u_d

	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}
u_d	3	4	4.5	4.5	3	5	1

表 4.3 项目{t_1, t_2, t_3, t_4, t_5, t_6, t_7}间项目相似度

Table 4.3 Items Similarity between {t_1, t_2, t_3, t_4, t_5, t_6, t_7}

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
t_1	-						
t_2	0.93	-					
t_3	0.54	0.61	-				
t_4	0.48	0.51	0.29	-			
t_5	0.49	0.29	0.89	0.59	-		
t_6	0.46	0.53	0.43	0.97	0.52	-	
t_7	0.32	0.10	0.41	0.22	0.17	0.49	-

表 4.4 项目{ t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}}间项目相似度

Table 4.4 Items Similarity between { t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}}

	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}
t_8	-						
t_9	0.94	-					
t_{10}	0.33	0.17	-				
t_{11}	0.95	0.90	0.28	-			
t_{12}	0.92	0.95	0.48	0.92	-		
t_{13}	0.42	0.35	0.91	0.23	0.38	-	
t_{14}	0.93	0.88	0.18	0.92	0.94	0.27	-

表 4.5 相似度计算结果

Table 4.5 Similarity Calculation Results

	Similarity	Similarity(proposed)
Sim(a,b)	0.8074	0.5313
Sim(a,c)	0.8189	0.8641
Sim(a,d)	0.8252	0.9172

4.3 算法过程及分析

加权的相似度计算方法的重点在求权值的过程，计算之前需要得到项目的相似度矩阵，具体计算方法可以采用基于内容的相似度计算方法也可以使用基于项目的协同过滤推荐算法来得到。下面就本章提出的 W_3 加权方法给出算法过程描述和分析。

算法 Weight(): 计算项目相似权重

输入：用户 u_a、用户 u_b、用户-项目评分矩阵 ***R***、项目相似度矩阵 ***H***、共同评分项目 T_{ab}

输出：$W_3(i)$（其中 $t_i \in T_{ab}$）

Step 1.　p=0，$T_1=T_{ab}$；

Step 2. 从 T_1 中取出 t_i，$T_1=T_1-\{t_i\}$；　p++；

Step 3.　$T_2=T_{ab}$，$W_{a,b}(i)=0$，q=1；

Step 4. 从 ***H*** 矩阵中取出 $h_{i,k}$，其中 $t_k \in T_2$ 且 k≠i；

Step 5. q++;$T_2=T_2-\{t_k\}$；

Step 6. $W_3(i)=W_3(i)+(h_{i,k})$；

Step 7. 若 q< $|T_{ab}|$，转 Step 4；

Step 8.　$W_3(i)=\frac{W_3(i)}{|T_{ab}|-1}\times\frac{(r_{a,i}+r_{b,i})}{2}$；

Step 9.若 p< $|T_{ab}|$，转 Step 2；否则计算完成。

按照上面的算法得到权值后，然后就是计算加权后的用户相似度，其公式已经在上文给出，这里不再赘述其计算过程。下面给出加权后的协同过滤推荐算法的形式化描述，其中的 $w_{a,b}(i)$表示调用上面提到的 Weight()函数：

输入：

用户集合：$U=\{u_a \mid 1\leqslant a\leqslant m\}$；

项目集合：$T=\{t_c \mid 1\leqslant b\leqslant n\}$；

用户-项目评分矩阵：$R_{mn}=\{r_{ac} \mid (u_a\in U)\wedge(t_c\in T)\}$;

邻居个数：k;

输出：

评分预测矩阵：$\hat{R}_{mn}=\{\hat{r}_{ac} \mid (u_a\in U)\wedge(t_c\in T)\wedge(r_{ac}=0)\}$;

/*计算用户相似度*/

For each a in U

　For each b in U

　　If　a≠b then

T_a=getUserRatingSet(R,a);

T_b=getUserRatingSet(R,b);

$T_{ab}=T_a \cap T_b$;

$$m(a;w)=\frac{\sum_{t_i \in T_{ab}} w_{a,b}(i) r_{a,i}}{\sum_{t_i \in T_{ab}} w_{a,b}(i)};$$

$$Sim(u_a,u_b)=\frac{\sum_{t_i \in T_{ab}} w_{a,b}(i)(r_{a,i}-m(a;w))(r_{b,i}-m(b;w))}{\sqrt{\sum_{t_i \in T_{ab}} w_{a,b}(i)(r_{a,i}-m(a;w))^2}\sqrt{\sum_{t_i \in T_{ab}} w_{a,b}(i)(r_{b,i}-m(b;w))^2}};$$

or

$$Sim(u_a,u_b)=\frac{\sum_{t_i \in T_{ab}} w_{a,b}(i) r_{a,i} r_{b,i}}{\sqrt{\sum_{t_i \in T_{ab}} w_{a,b}(i) r_{a,i}^{\ 2}}\sqrt{\sum_{t_i \in T_{ab}} w_{a,b}(i) r_{b,i}^{\ 2}}};$$

End if

End for

End for

/*将用户按照相似度排序，并选择用户的邻居*/

For each a in U

$User_a$=sortCadidateUserSet(U-{u_a},Sim,a);

$NUser_a$=getNeighborUser($User_a$,k);

$Item_a$=getUnRatingItem(R,u_a);

For each c in $Item_a$

$$\hat{r}_{ac}=\overline{r_a}+\frac{\sum_{u_i \in NUser_a} sim(a,i)\times(r_{i,c}-\overline{r_i})}{\sum_{u_i \in NUser_a}|sim(a,i)|};$$

End for

End for

计算用户对于一个项目的评分预测时，整个算法过程中项目相似度矩阵只需计算一次，所以其时间复杂度对整个算法影响不大。分析上述算法过程，可知影响时间复杂度的主要是项目相似度求和，若系统中用户数为 m，项目数为 n，k 为算法设定的用户邻居数目，则计算权值的时间复杂度为 $O(n^2)$，计算用户相似度的时间复杂度为 $O(mn^2)$，评分预测的时间复杂度为 O(kn)，整个推荐算法的时间复杂度为 $O(n^2)+ O(mn^2)+ O(kn)$。

4.4 实验结果与分析

本章采用 MovieLens 数据集来评测考虑项目相似权重后的用户相似度计算方法的性能。数据集中用户对电影评分的区间为[1,5]，评分间距为 0.5。评分越高，表明用户对该电影越感兴趣。实验分别选取了其中的 2000 和 10000 条评分数据做为实验数据集，2000 条评分数据的数据集包括 23 个用户和 688 部电影，10000 条评分数据的数据集包括 80 个用户和 2329 部电影，每个用户至少对 20 部电影进行了评分。评分预测评价指标使用平均绝对误差 MAE（Mean Absolute Error）。实验方案采用十折交叉验证法（10-fold cross-validation）[14]，把数据集分为不相交的十份，轮流将其中的九份作为训练集，其余一份作为测试集，算法执行十次，每次的测试集不相同，最后的实验结果采用十次计算结果的平均值。邻居数量从 4 取到 40，间隔为 4。本章提出的加权方法由两部分组成，其中第一部分“$(r_{a,i}+r_{b,i})/2$”即 W_1 加权方法计算简单，所花费的时间和空间代价相比完整的加权方法 W_3 小的多，因此具有不错的研究价值。实验通过 MATLAB 完成，W_1 加权方法的实验结果如图 4.1 和图 4.2 所示，W_3 加权方法的实验结果如图 4.3 和图 4.4 所示。

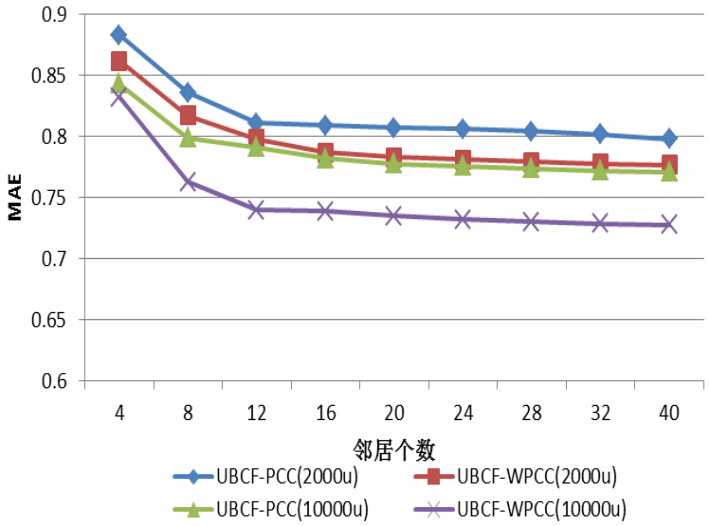

图 4.1 基于 W_1 加权皮尔逊相关系数协同算法的 MAE

Fig.4.1 MAE of W_1 weighted pearson correlation coefficient based collaborative algorithm

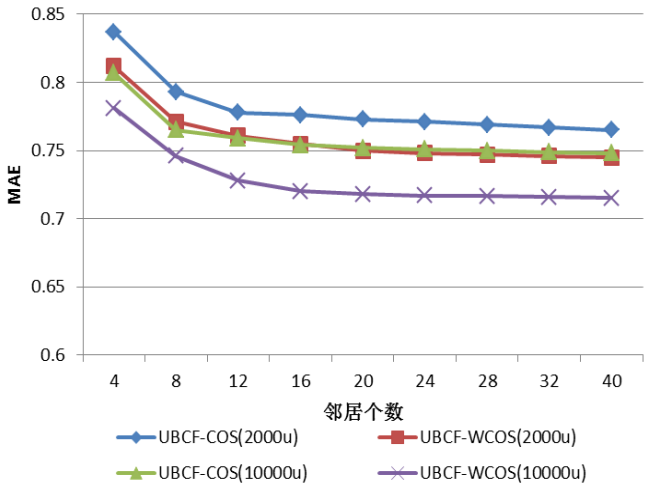

图 4.2 基于 W_1 加权余弦相似度协同算法的 MAE

Fig.4.2 MAE of W_1 weighted cosine similarity based collaborative algorithm

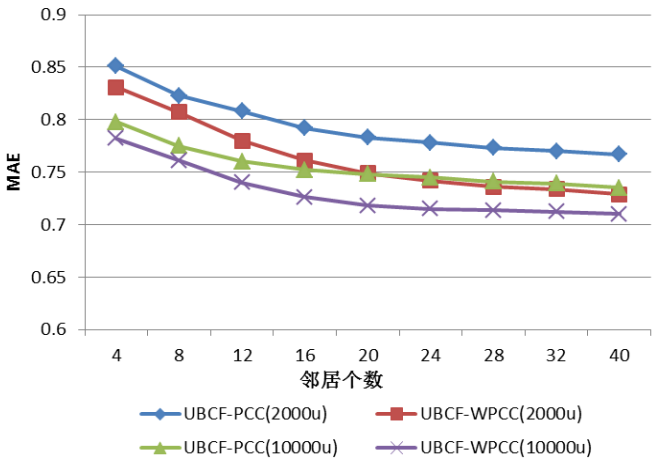

图 4.3 基于 W_3 加权皮尔逊相关系数协同算法的 MAE

Fig.4.3 MAE of W_3 weighted pearson correlation coefficient based collaborative algorithm

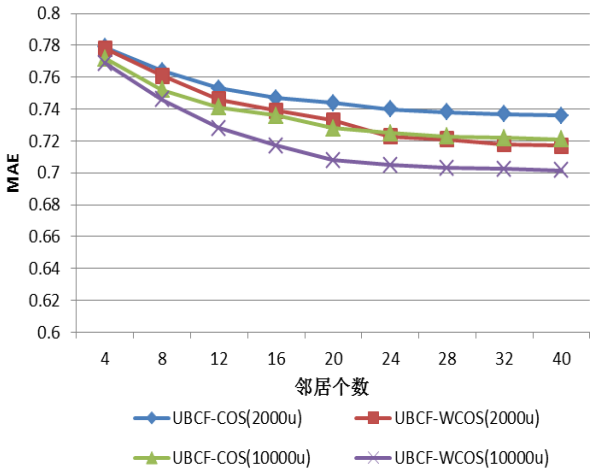

图 4.4 基于 W_3 加权余弦相似度协同算法的 MAE

Fig.4.4 MAE of W_3 weighted cosine similarity based collaborative algorithm

比对上面的 4 个实验结果图可知，无论是使用皮尔逊相关系数还是使用余弦夹角相似度，基于 W_3 加权方法的协同过滤算法的 MAE 都小于基于 W_1 加权方法的协同过滤算法的 MAE，即 W_3 加权方法能更明显地提高协同过滤推荐算法的准确度。下面着重分析基于 W_3 加权方法的协同过滤推荐算法。

从图 4.3 中曲线的形状可以看出，当使用皮尔洛相关系数时，随着邻居数量的增加，不论是传统的基于用户的协同过滤算法（UBCF-PCC）还是使用本文提出的加权后的用户相似度计算方法的协同过滤算法（UBCF-WPCC）的 MAE 都降低，推荐性能都有提高。当邻居数量相同时，UBCF-WPCC 比 UBCF-PCC 的 MAE 更小，这个结果在用户数为 2000 和 10000 时都相同。

图 4.4 是采用余弦相似度计算用户相似度的协同过滤算法的实验结果，也满足上面的结论，即 UBCF-WCOS 比 UBCF-COS 的 MAE 更小。一般来说，采用余弦相似度来计算用户相似度的协同过滤算法比采用皮尔洛相关系数来计算用户相似度的协同过滤算法的性能更好[16]，这也能从上面两个图中看出来：相同邻居数量和用户规模下，UBCF-COS 的 MAE 要小于 UBCF-PCC。但是 UBCF-WPCC 的 MAE 小于 UBCF-COS，接近于 UBCF-WCOS 的 MAE，这说明本文提出的加权方法对采用皮尔洛相关系数的计算方法提升更大。

从图 4.3 和图 4.4 中能看出随着邻居规模的增大，所有算法的性能都有提高，但系统付出的时间代价也将大幅增加，因此一个适宜的邻居规模是影响推荐系统的重要参数，从本文实验结果可看出，邻居数量在大于 20 的时候，MAE 降低的不明显，因此邻居数量可选 10 至 20 之间。

4.5 本章小结

本章首先介绍了传统用户相似度在基于用户的协同过滤推荐算法中的存在的问题，并对这些问题的成因做了分析，上述内容是本章提出的改进用户相似度计算方法的基础。用户相似度的计算在基于用户的协同过滤算法中占有重要的地位，传统用户相似度计算方法没有区分用户间共同高评分项目和用户间低评分项目，并且没有考虑共同已评分项目的类群关系。针对上述问题，本章提出了一种考虑项目间相似权重的用户相似度计算方法，该方法给每个参加运算的项目加上了一个权值，通过实验验证，使用该用户相似度计算方法的协同过滤推荐算法能显著提高算法评分预测的准确性。

5 总结与展望

5.1 总结

随着互联网规模的发展，越来越多的问题开始显现，信息过载就是其中比较突出的一个问题。信息过载是指系统中的信息过于丰富超过了人们的接受能力，同时人们也无法在信息的海洋中找到自己真正需要的资源。推荐系统是解决这个问题的理想技术，本文对推荐系统中的相似度进行了研究，主要的工作包括：

第一章介绍了推荐系统的研究背景、研究意义和研究现状，并对全文的结构进行了说明。

第二章从信息个性化的角度对推荐系统进行了形式化的定义，并详细归纳了推荐系统的功能。推荐系统作为一种个性化的信息过滤系统，对用户信息需求的描述是至关重要的问题，本章详细介绍了用户研究和用户建模，对其中涉及的用户调查和用户模型进行了分析，然后简要介绍了推荐系统的架构并分析了架构的各个组成部件。

第三章根据推荐策略介绍了推荐算法的三种分类，分析了各个类别推荐算法的特点和适用环境，然后介绍了计算用户相似度的三种方法及分析了各自的局限，最后总结了研究协同过滤推荐算法要用到的实验数据集及算法的评价指标。

第四章介绍了目前传统用户相似度存在的问题，并对这些问题的成因做了分析，上述内容是本文提出的改进用户相似度计算方法的基础。传统用户相似度计算方法没有区分用户间共同高评分项目和用户间低评分项目，并且没有考虑共同已评分项目的类群关系。针对上述问题，提出了一种考虑项目间相似权重的用户相似度计算方法，该方法给每个参加运算的项目加上了一个权值，通过实验验证，使用该用户相似度计算方法的协同过滤推荐算法能显著提高算法评分预测的准确性。

5.2 展望

在可见的未来信息过载问题还将存在，推荐系统仍将是研究的热点。本文对推荐系统的研究存在一些不足，比如考虑了项目间近似的类群关系，而没有考虑用户间兴趣的广度，用户间那些共同喜好的类群中的项目获得了较高的权重，这可能会降低推荐系统的覆盖率和多样性。以后的工作可以围绕如何提高使用了本文提出的推荐算法的覆盖率来展开，同时可以考虑将本文的方法应用到基于项目的协同过滤推荐算法中去。

致 谢

转眼间，在重庆大学的三年一晃而过，短暂的时光却留下了青春最美好的记忆。身边一直有陪伴着我的良师益友，一起学习，一起出游，留下了太多的欢声笑语，或是感动、幸福的泪水。在此，向所有陪伴着我，帮助我的人们表示真诚的感谢。

首先要感谢我的导师罗军副教授，罗老师不论是在学习上还是生活上都给予了我最大的关爱！每当遇到学业和工作上的艰难选择，恩师都会悉心的指导，与我共同分析问题，并最终支持我的选择。论文从选题到构思，再到撰写、修改，最后定稿，都是在恩师的倾力指导下完成的。恩师广博的学识素养、严谨的学术风范、乐观的处事态度将是我毕生追随的模范，我将用自己的行动来回报恩师的培育之恩，在以后的工作和生活中，努力拼搏，有所成就。很喜欢听罗老师讲话，每次都会感到平静，总觉得受益良多。罗老师是个豁达、谦和、不拘小节的人，罗老师对人生的态度是我一生学习的榜样。

然后很感谢陪伴我三年的同门张俊勇、王秋菊、陈飞，回想这三年，我们一起在主教学习、做项目，这里留下了太多美好的回忆，非常感谢他们在学习和生活上给我的帮助。

非常感谢我的室友同时也是我的同乡吴幸良，三年的友谊终身不忘！

另外，还要感谢我的亲人、朋友以及师兄弟姐妹们，谢谢你们的帮助、支持和鼓励。

最后要感谢的是养育我的父母，因为有你们的存在，我才有了前进奋斗的动力，才有不怕失败不怕困难的决心。

真心的感谢所有在我的生命中出现的人，你们每个人都给了我独一无二的人生际遇，都让我的生活更加精彩，谢谢你们！

朱文奇

二 O 一四年四月 于重庆

参考文献

[1] P Resnick,H R Varian. Recommender systems[J], Commun. ACM, 1997, 40(3):56-58.

[2] J. B. Schafer, J. A. Konstan,J. Riedl. E-Commerce Recommendation Applications,Data Min. Knowl. Discov.,vol. 5, iss. 1-2,pp. 115-153,2001.

[3] Adomavicius,Gediminas.Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions[J].IEEE Transactions on Knowledge and Data Engineering,2005,17(6):734-749.

[4] 曾春,邢春晓,周立柱.个性化服务技术综述,软件学报,vol.13, iss. 10,pp. 1952-1961, 2002.

[5] D. Goldberg, D. Nichols, B. M. Oki,D. Terry. Using collaborative filtering to weave an information tapestry, Commun. ACM, vol. 35,iss. 12, pp. 61-70,1992.

[6] G. Adomavicius, A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions,IEEE Trans, on Knowl. and Data Eng., vol. 17,iss. 6,pp. 734-749,2005.

[7] C. Manning, P. Raghavan, Sch, H. tze. Introduction to Information Retrieval.Cambridge University Press, 2008.

[8] Hill, W., Stead, L., Rosenstein, M., Furnas, G. Recommending and evaluating choice in a virtual community of use. In Conference on Human Factors in Computing Systems-CHI'95. May 1995.

[9] U. Shardanand, P. Maes. Social information filtering: algorithms for automating "word of mouth", presented at the Proceedings of the SIGCHI conference on Human factors in computing systems,Denver, Colorado,United States, 1995.

[10] U Shardanand, P Maes. Social information filtering: algorithms for automating "word of mouth"[C]. Proceeding of the SIGCHI conference on Human factors in computing systems, Denver, Colarado, United States, 1995.

[11] C. Basu, H. Hirsh,W. Cohen. Recommendation as Classification: Using Social and Content-Based Information in Recommendation, in AAAI, pp.714-720, 1998.

[12] Linden, G., Smith, B., York, J. Amazon.com recommendations: Item-to-Item collaborative filtering. IEEE Internet Computing. Jan./Feb. 2003, 7(1): 76-80.

[13] 邓爱林,左子叶,朱扬勇.基于项目聚类的协同过滤推荐算法,小型微型计算机系统,vol. 25,iss. 9,p. 6,2004.

[14] 吴湖,王永吉,王哲,王秀利,杜栓柱.两阶段联合聚类协同过滤算法,软件学报,vol. 21, iss. 5, pp. 1042-1054, 2010.

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

[16] Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems An Introduction.Cambridge University Press (2010)

[17] Breese, J.S., Heckerman, D., Kadie, C.M.: Empirical analysis of predictive algorithms for collaborative filtering. In: UAI, pp. 43–52 (1998)

[18] Van Meteren, Robin, and Maarten Van Someren. "Using content-based filtering for recommendation." Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop. 2000.

[19] Burke, R.: Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction 12(4), 331–370 (2002)

[20] Konstan, Joseph A. "Introduction to recommender systems." Proceedings of the 2008 ACM SIGMOD international conference on Management of data. ACM, 2008.

[21] Liang, Ting-Peng. "Recommendation systems for decision support: An editorial introduction." Decision Support Systems 45.3 (2008): 385-386.

[22] 许海玲, 吴潇, 李晓东, 等.互联网推荐系统比较研究[J]. 软件学报, 2009,20(2):350-362.

[23] Linyuan Lü, Matúš Medo, Chi Ho Yeung, Yi-Cheng Zhang ,Zi-Ke Zhang, Tao Zhou. Recommender systems[J]. Physics Reports, 2012,519(1):1-49.

[24] Goldberg, K., Roeder, T., Gupta, D., Perkins, C. Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval. Jul. 2001, 4(2): 133-151.

[25] Adomavicius G, Tuzhilin A. Towards the Next Generation of Recommender Systems: a Survey of the State-of-the-art and Possible Extensions[J]. IEEE Trans on Knowledge and Data Engineering,2005,17(6):734-749.

[26] J L Herlocker, J A Konstan, A Borchers, J Riedl. An algorithmic framework for performing collaborative filtering[C]. Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval,1999, 230-237.

[27] 黄创光, 印鉴, 汪静, 等. 不确定近邻的协同过滤推荐算法[J]. 计算机学报,2010,33(8):1369-1377.

[28] Keunho Choi, Yongmoo Suh. A new similarity function for selecting neighbors for each target item in collaborative filtering[J]. Knowledge-Based Systems, 2013,37:146-153.

[29] Massa, P., Avesani, P. Trust-aware recommender systems. Proceedings of the 2007 ACM Conference on Recommender system. 2007: 17-24.

[30] 徐海玲,吴潇,李晓东,阎保平.互联网推荐系统比较研究,软件学报,vol.20, iss. 2, pp. 350-362, 2009.

[31] A Albadvi, M Shahbazi. A hybrid recommendation technique based on product category attributes[J]. Expert Systems with Applications, 2009,36(9):11480-11488.

[32] Jeong Buhwan, Lee Jaewook, Cho Hyunbo. An iterative semi-explicit rating method for building collaborative recommender systems[J]. Expert Systems with Applications,2009,36(3):6181-6186.

[33] P Resnick, N Iacovou, M Suchak, P Bergstrom, J Riedl. GroupLens: an open architecture for collaborative filtering of netnews[C]. Proceedings of the 1994 ACM conference on Computer supported cooperative work, Chapel Hill, North Carolina, United States,pp, 1994,175-186.

[34] T Q Leea, Y Parkb, Yong-Tae Parkc. A time-based approach to effective recommender systems using implicit feedback[J]. Expert Systems with Applications,2008,34(4):3055-3062.

[35] ROBIN B "Hybrid recommender systems: survey and experiments" Department of information Systems and Decision Sciences, Califomia State University.

[36] SongJie Gong. A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering[J]. Journal of Software,2010,5(7):745-752.

[37] R Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection[C], Proceedings of the 14^{th} international joint conference on Artifical intelligence,1995,2.

[38] Sarwar B, Karypis G, Konstan J, Riedl J.Item-Based collaborative filtering recommendation algorithms[C]. Proceedings of the 10th International World Wide Web Conference,2001:285~295.

[39] Arazy, O., Kumar, N., Shapira, B.: Improving social recommender systems. IT Professional 11(4), 38–44 (2009)

[40] S.-T. Park,D. Pennock,O. Madani, N. Good,D. DeCoste. Nai've filterbots for robust cold-start recommendations, presented at the Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, Philadelphia, PA, USA,2006.

[41] Melville, Prem, Raymond J. Mooney, and Ramadass Nagarajan. "Content-boosted collaborative filtering for improved recommendations." AAAI/IAAI. 2002.

[42] Soboroff, Ian, and Charles Nicholas. "Combining content and collaboration in text filtering." Proceedings of the IJCAI. Vol. 99. 1999.

[43] Basu, Chumki, Haym Hirsh, and William Cohen. "Recommendation as classification: Using social and content-based information in recommendation." AAAI/IAAI.1998.

[44] Berkovsky, S.: Mediation of User Models: for Enhanced Personalization in Recommender Systems. VDM Verlag (2009)

[45] Burke, R.: Hybrid web recommender systems. In: The AdaptiveWeb, pp. 377–408. Springer

Berlin / Heidelberg (2007)

[46] Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems An Introduction.Cambridge University Press (2010)

附 录

A.作者在攻读学位期间发表的论文目录：

[1] 罗军，朱文奇. 考虑物品相似权重的用户相似度计算方法. 计算机工程与应用（已录用待发表）.

B.作者在攻读学位期间取得的科研成果目录：

[1] 重庆市城投路桥管理有限公司“路桥数字中心”的设计和研发.

推荐系统用户相似度计算方法研究

作者：朱文奇
学位授予单位：重庆大学

引用本文格式：朱文奇 推荐系统用户相似度计算方法研究[学位论文]硕士 2014