基于标签权重评分的推荐模型及算法研究

孔欣欣 苏本昌 王宏志 高 宏 李建中
(哈尔滨工业大学计算机科学与技术学院 哈尔滨 150001)

摘 要 推荐系统已经被越来越频繁地应用到电子商务网站与一些社交网站,在提高用户满意度的同时也带来了巨大的商业利益.然而,当前的推荐算法由于原始数据的不完整性以及算法本身处理数据的特殊性,导致推荐效果不理想.例如,某些推荐系统会产生冷启动、复杂兴趣推荐困难、解释性差等问题.为此,该文提出一种基于标签权重评分的推荐系统模型(Label-WeightRating based Recommendation, LWR),旨在使用一种较为简洁的方式——标签权重评分来获取用户最准确的评价和需求,并通过改进当前的一些推荐算法来处理标签权重评分数据,从而生成对用户的推荐,最后以标签权重评分的形式向用户展示推荐结果并作出合理的解释.扩展实验中,通过电影推荐实验,证明了该文技术的有效性和可行性.

关键词 推荐系统;标签;标签权重评分;数据挖掘;人工智能
中图法分类号 TP391 **DOI号** 10.11897/SP.J.1016.2017.01440

Research on the Modeling and Related Algorithms of Label-Weight Rating Based Recommendation System

KONG Xin-Xin SU Ben-Chang WANG Hong-Zhi GAO Hong LI Jian-Zhong
(*School of Computer Science and Technology, Harbin Institute of Technology, Harbin* 150001)

Abstract Recommendation System has been frequently applied into various e-commerce websites and social networking sites. With improving users' satisfaction, recommendation system has also brought huge commercial interests. However, as the original data is incomplete and some recommendation algorithms have their own special way of processing data, current recommendation system sometimes cannot work very well. For example, some recommendation systems are bothered with cold-start problem, difficult for complex interest recommendation problem, poor interpretability and so on. Consequently, in the paper, we propose a recommendation system modeling based on label-weight rating. In this system, first we will get the most accurate evaluation and demanding information of users in a more concise way—label-weight rating method. Then we will generate recommendations using improved existing recommendation algorithm. Finally, we will show the recommendations to the users in the form of label-weight rating and make reasonable explanation to users. In the extended experiments we design a series of movie recommendations experiments to prove the effectiveness and feasibility of the modeling.

Keywords recommendation system; label; label-weight rating; data mining; artificiall intelligence

收稿日期:2014-06-27;最终修改稿收到日期:2015-06-15.本课题得到国家自然科学基金(61472099,61003046)、国家"九七三"重点基础研究发展规划项目基金(2012CB316200)和国家科技支撑计划(2015BAH10F00)资助.**孔欣欣**,女,1994年生,硕士研究生,主要研究方向为数据质量.E-mail:xinxinkonghit@163.com.**苏本昌**,男,1989年生,硕士研究生,主要研究方向为数据质量.**王宏志**(通信作者),男,1978年生,博士,副教授,博士生导师,主要研究领域为大数据管理、数据质量、XML数据管理等.E-mail:wangzh@hit.edu.cn.**高 宏**,女,1966年生,博士,教授,博士生导师,主要研究领域为无线传感器网络、物联网、海量数据管理和数据挖掘.**李建中**,男,1950年生,教授,博士生导师,主要研究领域为无线传感器网络、物联网、数据库和海量数据管理.

1 引　言

推荐系统[1-3]的主要任务是通过分析用户信息、物品信息或其他辅助信息，获得用户对物品的偏好特征，并据此为用户进行物品推荐.

当前的推荐算法主要包括以下 3 种[4]：基于内容的算法、基于协同过滤的算法和基于标签的方法.

基于内容的算法[5-6](Content-Based Algorithm，CB)通过为每个物品抽取内容特征来描述该物品，通过用户过去所喜好物品的特征描述用户的偏好特征，通过计算用户与物品之间的相关性进行推荐.

基于协同过滤的算法[7-8](Collaborative Filter Algorithm，CF)有两种情况：一种是通过对不同用户对相同物品的行为分析找出相似用户，根据相似用户的偏好对指定用户进行物品推荐，这称为基于用户的协同过滤推荐(User-based Recommendation)；另外一种是通过对相同用户对不同物品的行为分析找出相似物品，根据相似物品的相似度为指定用户进行推荐，这称为基于物品的协同过滤推荐(Item-based Recommendation).

基于标签的方法[9-10](Tag-Based Algorithm，TB)通过分析用户的标签偏好、物品的标签特征，基于二者的相似性为用户进行物品推荐. 其本质是通过引入标签，形成用户-标签-物品三元关系，其中标签来源于 Web2.0 环境下用户对物品的描述.

以上 3 种方法在当前推荐系统中已得到广泛应用，然而它们有以下缺陷：

(1)冷启动问题[11-13]. 当推荐系统中加入了新的用户，由于没有该用户历史偏好数据(如 CB 算法和 CF 算法)或标签数据(如 TB 算法)，以致无法为用户进行有效的推荐.

(2)复杂兴趣推荐困难. 当用户的兴趣突然发生变化或者多个用户共用一个账户时，用户的兴趣就变得复杂. 以上 3 种方法对用户历史兴趣依赖过重，很难适应这种情况，推荐也就变得不准确.

(3)可解释性差问题. 为提高用户满意度，推荐系统在进行物品推荐的同时会提供解释来说明推荐原因. 推荐解释的方式与所使用的推荐算法有直接关系. CB 算法会提供抽取的内容特征来作解释，但是物品的特征一般很难提取. 比如电影推荐，很有可能从两部不同电影描述信息中提取出相同的演员导演的信息，这样的推荐解释缺乏区分度和信服力. CF 算法提供与所推荐物品相似的物品作为说明或者提供相同偏好的用户作为解释. 这样的推荐解释的不足之处在于它默认相似用户偏好同一物品是基于相同的理由，这显然是不准确的. 比如用户 A 和用户 B 都喜欢“阿甘正传”，而用户 A 是因为喜欢“幽默”，用户 B 是因为喜欢“汤姆汉克斯”. 如果向 A 推荐一部电影，解释为“B 也喜欢”，就不合适了. TB 算法会为推荐的物品提供标签解释，但是不同的物品可能具有相同的标签，这时区分度就不大，会影响用户满意度. 比如电影“美国队长”具有标签“科幻”“剧情”两个标签，电影“黑暗骑士”也具有“科幻”“剧情”两个标签，然而看过的人知道“美国队长”中科幻元素更强些，“黑暗骑士”的剧情更胜一筹，所以仅仅有标签还是不够.

针对以上问题，本文提出了一种基于标签权重评分的推荐系统模型(Label-WeightRating based Recommendation，LWR). 标签权重评分(Label-Weight Rating，LWR)是对传统标签的一种扩展，我们通过为每个标签配以相应的评分，来描述该物品或用户在该标签上的权重. 同时，该方法较以往的方法还能最大化地降低客观因素对用户评分的影响. 例如某用户可能本来很喜欢 a 餐馆，但最近一次在该餐馆就餐时发生过不愉快的事情，则该用户在对该餐馆打分时极可能给出较低的分数，这就使得评分出现了偏差[14]. 而当前提出的方法可以较为公正客观地解决这一问题，例如可以将对餐馆的标签评分划分为：饭菜质量、用餐环境、餐厅服务. 此时用户可以对每一项打分，因为这种细分能够最大化地降低客观因素对用户打分的影响，使得评分更为准确、真实.

本文第 2 节提出标签权重评分推荐模型；第 3 节设计标签权重评分推荐算法；第 4 节进行相关实验及其结果分析；第 5 节总结全文.

2 系统模型

这一节我们介绍了基于标签权重评分推荐系统模型. 首先我们给出了标签权重评分数据表示，然后给出了推荐系统架构及其数据处理流程，最后说明了本文模型在解决冷启动问题、复杂兴趣推荐问题、可解释性差问题上的优越性.

2.1 数据表示

定义 1.　标签.

标签是用来描述物品特征的，我们把标签定义为 $t=(t^1,t^2,\cdots,t^p)$，其中 t^k 为标签的第 k 个基本属

性,可以是标签名称、词性等.

定义 2. 标签权重.

本文在传统标签的基础上进行了扩展,即在描述物品时不仅给出标签特征,还会给出该物品在特征上的权重,即我们用标签权重代替标签对物品进行描述.我们定义标签权重为 $s=(t,tagRating)$,其中 t 为标签属性,$tagRating$ 为权重属性.

定义 3. 基于标签权重评分的数据表示.

在基于标签权重评分的推荐系统中数据表示可以描述为一个五元组

$$M:=(U,I,R,S,Y)$$

其中:U 为用户集合,$\forall u\in U,u=(u^1,u^2,\cdots,u^m)$,其中 $u^k(k=1,2,\cdots,m)$ 为用户的第 k 个基本属性.例如,当 $m=3$ 时,$u=(1,$"Ben"$,24)$表示编号为 1 的用户姓名为"Ben",年龄为 24.

I 为物品集合,$\forall i\in I,i=(i^1,i^2,\cdots,i^k,\cdots,i^n)$,其中 i^k 为物品的第 k 个基本属性.例如,当 $n=3$ 时,$i=(3,$"美国队长 2","2014")表示编号为 3 的物品名字为"美国队长 2",上映年份为 2014.

R 为用户对物品的偏好评分集合,$\forall r\in R,r=(r^1,r^2,\cdots,r^k,\cdots,r^l)$,其中 r^k 表示用户 u 对物品 i 的偏好评分的第 k 个基本属性,比如评分时间、地点等上下文信息.在本文中,我们不考虑上下文信息对推荐结果的影响.所以,为简洁起见,我们省略评分数据的大多属性,只保留数值评分.例如常用 $R=\{1,2,3,4,5\}$.

S 为用户对物品的标签权重评分集合,$\forall s\in S$,$s=(t,tagRating)$,其中 t 为标签属性,$tagRating$ 为权重属性,故 S 表示带有权重的标签的集合,且满足 $S\subseteq T\times R$,其中 T 为用户对物品的标签集合.

Y 为 U,I,R,S 上的一个四元关系,即满足关系 $Y\subseteq U\times I\times R\times S$,故 Y 表示用户对物品的偏好评分及标签权重评分构成的四元组的集合.

例如,$y=(u,i,r,(t,tagRating))\in Y$,表示用户 u 对物品 i 的偏好评分为 r,同时指定物品 i 的一个标签权重评分 $s=(t,tagRating)$.其中,t 是用户 u 认为物品 i 具有的其中一个特征,并且认为物品 i 在该标签特征上的权重为 $tagRating$.

定义 4. 基于标签权重评分的相关符号定义.

基于此模型定义推荐系统中的其他数据要素表示如下.

(1) $s(u,i)\in S$ 表示用户 u 对物品 i 的其中一个标签权重评分;

(2) $S(u,i)\subseteq S$ 表示用户 u 对物品 i 的标签权重评分集合;

(3) $S(U,I)=\{s(u,i)\mid u\in U,i\in I\}$ 表示用户集合 U 中的用户对物品集合 I 中的物品的标签权重评分集合;

(4) $s(u)\in S$ 表示用户 u 的其中一个标签权重特征;

(5) $S(u)\subseteq S$ 表示用户 u 的标签权重特征集合;

(6) $S(U)=\{S(u)\mid u\in U\}$ 表示用户集合 U 中的用户的标签权重特征集合;

(7) $s(i)\in S$ 表示物品 i 的其中一个标签权重特征;

(8) $S(i)\subseteq S$ 表示物品 i 的标签权重特征集合;

(9) $S(I)=\{S(i)\mid i\in I\}$ 表示物品集合 I 中的物品的标签权重特征集合;

(10) $r(u,i)\in R$ 表示用户 u 对物品 i 的偏好评分;

(11) $r(i)\in R$ 表示物品 i 的整体评分;

(12) $R(I)=\{r(i)\}$ 表示物品集合 I 的整体评分;

(13) $R(U,I)=\{r(u,i)\mid u\in U,i\in I\}$ 表示用户集合 U 中的用户对物品集合 I 中的物品的偏好评分集合.

2.2 系统架构

基于标签权重评分的推荐系统架构如图 1 所示,主要分为数据源模块、推荐引擎模块、推荐结果处理模块和用户反馈模块.具体解释如下:

数据源模块.数据源 D 是推荐系统进行推荐的依据来源,主要包括用户信息集合 U、物品信息集合 I、用户对物品的偏好评分信息 R、用户对物品的标签权重评分 S.数据源模块的主要任务是对数据源 D 的获取以及预处理.

推荐引擎模块.推荐引擎 E 是推荐系统的核心,其主要作用是使用推荐算法处理分析来自数据源 D 的信息,根据一定的推荐标准为用户推荐最需要的物品集合 $I(U)$.

推荐结果处理模块.在推荐引擎计算出对用户的初始物品推荐列表后,我们要对推荐物品进行过滤、排名以及加上相应的标签权重向用户解释推荐这些物品的原因.

用户反馈模块.用户在看到推荐系统提供的推荐的物品以及推荐解释后,可以进行相应的用户信息反馈,用户反馈模块收集并处理反馈信息,并传送给推荐引擎模块完成新一轮的推荐.

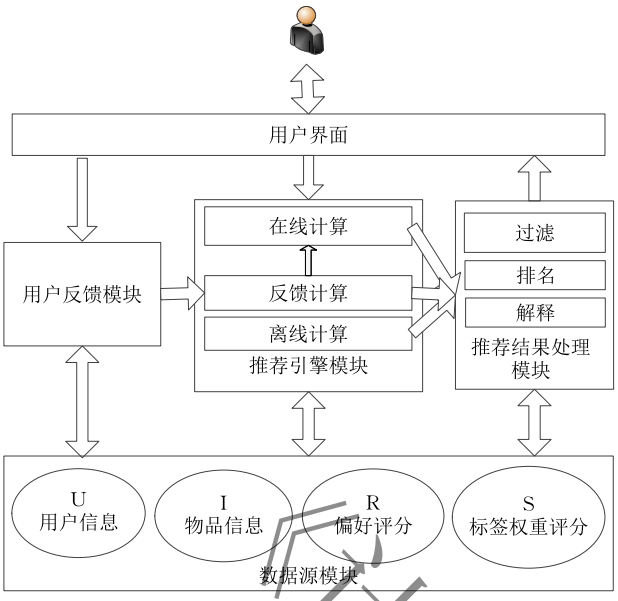

图 1　推荐系统基本架构

2.3　数据处理流程

针对当前推荐系统存在的冷启动、复杂兴趣推荐困难、可解释性差这 3 个缺陷，我们给出在上述系统架构下的 3 种基本推荐方式及其数据处理流程如下.

2.3.1　离线计算推荐

离线计算推荐，指的是在获得大量用户的评分数据后，通过离线计算为用户进行物品推荐，然后在用户登陆系统时进行结果展示. 这种推荐把大量计算放在线下，避免了让用户长时间等待.

具体流程如图 2. 在获得用户的基本信息和偏好数据后，数据源模块首先进行预处理，然后调用相应的推荐算法进行计算，生成对每个用户的物品推荐并把结果存储起来. 最后在用户使用推荐系统时将推荐的物品通过推荐结果处理模块展示，并且以标签权重评分形式阐述推荐原因.

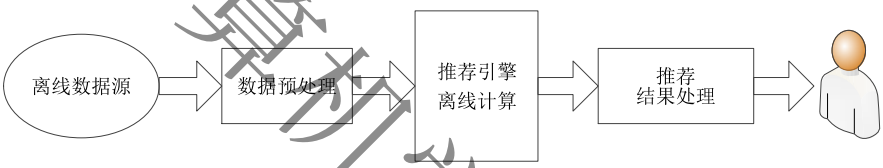

图 2　离线计算推荐数据流程

2.3.2　在线计算推荐

本文提出的在线算法不同于传统意义上的在线算法，传统意义上的在线算法是在解决一个问题时事先不知道问题的所有输入数据，是序列化地一个个地处理输入，并在有限的已知条件下做出最优选择. 而本文中的在线含义类似于 QQ、飞信等通讯工具的在线概念，即强调用户在在线状态下，能够在线实时输入信息，并进行实时地反馈.

在线计算推荐，指的是用户在登录推荐系统后，以标签权重评分的形式表达出当前的兴趣需求，系统获取当前用户偏好数据，并令用户选择是否考虑历史兴趣，若是，则结合离线计算的结果，进行在线计算，若否，则只根据当前用户偏好数据进行在线计算，最后将推荐结果进行展示.

具体流程如图 3. 不管是新用户还是老用户，都可以通过标签权重评分表达当前的偏好需求. 推荐系统获得当前的偏好数据之后首先进行预处理，然后根据用户是否依赖历史偏好数据的选择，来决定是否获取离线计算的结果，然后调用在线推荐算法，进行用户的物品推荐. 另外，在线计算的结果计算后会加入到用户的历史偏好数据中.

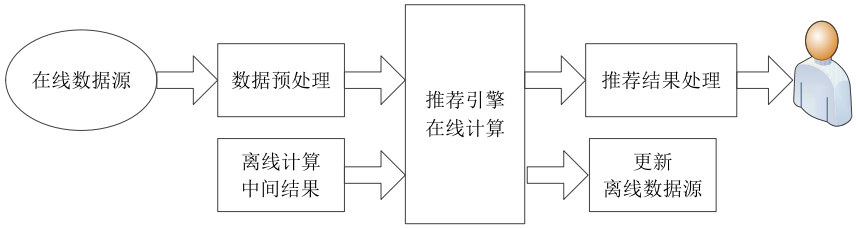

图 3　在线计算推荐数据流程

2.3.3　反馈计算推荐

反馈计算推荐，指的是用户在看到推荐结果时通过标签权重评分机制进行反馈，即重新对系统推荐的物品进行评分以及给相应的标签进行评分. 系统获得反馈后，获取并更新与该用户相关的离线数据，结合离线计算结果形成新一轮的推荐.

具体流程如图 4. 反馈计算推荐的整个过程与在线计算推荐是相似的，不同的地方是，在获得用户的反馈数据后，为了进行更好地推荐需要把反馈数据更新到与该用户相关的历史数据中去，然后根据更新后的数据，结合离线计算的结果进行在线计算推荐.

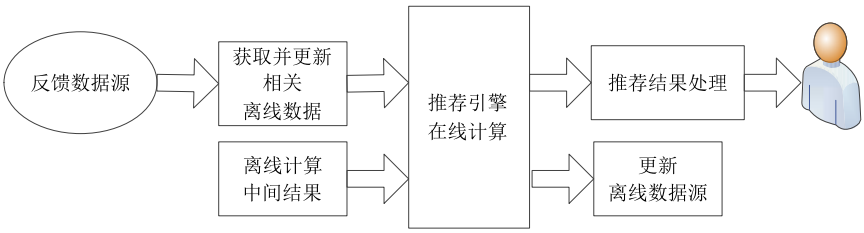

图 4 反馈计算推荐数据流程

2.4 模型优越性阐述

这一小节，我们介绍如何利用上述 3 种推荐方式来解决冷启动、复杂兴趣推荐困难、可解释性差这 3 个问题.

由于没有新用户的历史数据，无法进行离线计算，不能作出有效的推荐，也就是会出现冷启动问题. 但基于本文模型，我们允许用户通过标签权重评分机制准确表达自己的兴趣偏好，同时，我们会对标签权重进行说明：1.0 表示非常不喜欢，2.0 表示不太喜欢，3.0 表示一般喜欢，4.0 表示很喜欢，5.0 表示非常喜欢. 系统把新获取的数据作为用户的标签权重特征，调用在线计算推荐算法进行物品推荐.

例如，用户 u 可以选择如下标签并赋予相应的权重来表达自己的标签权重特征

$$S(u)=\{(\text{“科幻”},5.0),(\text{“超级英雄”},4.0),(\text{“人性”},4.0),(\text{“情节”},3.0)\}$$

对于用户兴趣突然发生变化或多人共用一个账户的复杂兴趣推荐问题，据笔者所知，当前还没有很有效的在线计算方法能够解决. 但与解决冷启动问题相似，在本文模型下，我们允许用户通过标签权重评分机制准确表达当前的兴趣偏好，系统把新获取的标签权重评分数据作为用户的临时标签权重特征，调用在线计算推荐算法进行物品推荐.

可以看出，冷启动问题和复杂兴趣推荐问题，都可以在本文模型下通过在线计算解决，当然在线计算也需要使用离线计算的结果，至于具体的在线计算和离线计算推荐算法我们将在下一章介绍.

另外，对于当前推荐系统可解释性差的问题，在本文模型下，我们提供基于标签权重评分的推荐解释. 这样可以让用户更详细地了解所推荐物品的特征以及在这些特征上的权重，在提高用户满意度的同时，可以让用户作出准确地反馈. 系统接收到反馈，经过用户反馈模块进行数据预处理，然后调用反馈计算推荐算法进行物品推荐. 例如可以为用户推荐物品 i 并提供推荐解释为

$$S(u)=\{(\text{“科幻”},5.0),(\text{“超级英雄”},5.0),(\text{“人性”},3.0),(\text{“情节”},4.0)\}$$

虽然该方法使得用户操作复杂度增加，但是很多用户还是愿意对自己的倾向做细致区分并做出选择，以提高检索或者推荐结果的精度. 以淘宝网(www.taobao.com)和豆瓣网(www.douban.com)为例，淘宝网用户为了购买到最符合自身需求的物品，他们很乐意使用标签的方式来明确表明自身需求，他们为了追求更高的准确度而可以进一步进行复杂操作，同时他们也愿意进行反馈评价，故该方法适用于淘宝网这类电子商务网站. 在豆瓣网中标签应用很广泛，豆瓣用户为了找到志同道合的瓣友、看感兴趣的帖子而使用标签，他们为了找出符合自身品味的作品而愿意进行较为复杂的标签操作. 因此，该方法具有一定的实用价值.

3 算 法

本节我们将基于标签权重评分推荐模型进行相关算法的研究与设计.

3.1 节我们介绍了数据源分解算法，目的是通过数据预处理获得用户与物品的标签权重特征；3.2 节给出基于矩阵填充的离线计算推荐算法；3.3 节给出基于聚类的在线计算推荐算法；3.4 节给出基于标签权重评分的反馈计算推荐算法.

3.1 数据源分解算法

数据源分解算法用在数据源模块，通过分解获得的用户标签权重特征将用于相似用户的计算以及其他用户标签权重特征的计算. 物品标签权重特征将用于相似物品的计算以及作为物品的推荐解释提供给用户.

3.1.1　用户标签权重特征求解算法

从评分数据 $D=\{(u,i,r,s)\mid y=(u,i,r,s)\in Y\}$ 求解用户的标签权重特征，其基本思想是：如果用户对物品的偏好评分比较高，那么我们有理由认为用户对其给予该物品的具有较高评分的标签特征更为看重. 于是我们就在计算用户的标签权重特征时把该标签的权重按照一定的规则提高.

具体流程如算法 1 所示.

算法 1.　用户标签权重特征求解算法.

输入：评分数据 $D=\{(u,i,r,s)\mid y=(u,i,r,s)\in Y\}$

输出：用户的标签权重特征集合 $S(U)=\{S(u)\mid u\in U\}$

算法过程：

1. $U=\{u\mid(u,i,r,s)\in D\}$//参与标签权重计算的用户
2. $S(U)=\{S(u)\}$，$S(u)=\varnothing$//初始标签权重特征为空
3. foreach $(u,i,r,s)\in D$ do
4. if $r\geq threshold$ then //偏好评分很高
   $t=getTag(s)$
   $tagRating=getTagRating(s)$
5. if $tagRating\geq threshold$ then
   $weightUp(S(u),t,r,tagRating)$
6. else//偏好评分很低
7. if $tagRating\geq threshold$ then
   $weightDown(S(u),t,r,tagRating)$
8. $normalize(S(u))$
9. return $S(U)$

算法流程如下：首先对所有参与计算的用户的标签权重集合初始化(1～2 行)，对于 D 中的任意一个元组进行循环迭代(3～7 行). 如果当前用户 u 对物品 i 的偏好评分超过了一个阈值 $threshold$，并且此时该元组所对应的标签的权重超过了阈值 $threshold$，我们就相应地提升用户 u 对该物品 i 的当前标签的权重(4～5 行). 反之如果用户 u 对物品的整体偏好得分低于阈值 $threshold$(6 行)，且当前标签特征在物品 i 中所占的权重超过了阈值 $threshold$，我们就相应地降低该特征的权重(7 行). 迭代地判定每个元组，直到所有元组判定完毕(3～7 行)，最后规范化 $S(u)$(8 行)，返回所有参与标签权重计算的用户的标签权重集合 $S(U)$(9 行).

由于只需要遍历一遍数据源，该分解算法的时间复杂度为 $O(N)$. 其中，N 为用户评分记录的数目.

3.1.2　物品标签权重特征求解算法

从评分数据 $D=\{(u,i,r,s)\mid y=(u,i,r,s)\in Y\}$ 求解物品的标签权重特征，其基本思想是：物品被描述次数较多的标签权重特征应该被赋予较大的权重. 这里简单地把被描述的特征次数作为权重加减，也可以把标签权重评分也考虑作为特征权重加入到物品的标签权重特征.

具体流程如算法 2 所示.

算法 2.　物品标签权重特征求解算法.

输入：评分数据 $D=\{(u_1,i_1,r_1,s_1),(u_2,i_2,r_2,s_2),\cdots,(u_N,i_N,r_N,s_N)\}$

输出：物品的标签权重特征集合 $S(I)=\{S(i)\mid i\in I\}$

算法过程：

1. $I=\{i\mid(u,i,r,s)\in D\}$//参与标签权重计算的物品
2. $S(I)=\{S(i)\}$，$S(i)=\varnothing$//初始标签权重特征为空
3. foreach $(u,i,r,s)\in D$ do
4. $t=getTag(s)$
   $tagRating=getTagRating(s)$
5. $addToSlider(S(i),t,r,tagRating)$//增加相应特征的权重
6. $normalize(S(i))$
7. return $S(I)$

算法流程如下：首先对计算的物品标签权重特征初始化为空(1～2 行)，对于 D 中的任意一个元组，进行循环判断(3～5 行). 首先分离出物品 i 的标签特征 t 和该标签特征的权重 $tagRating$(4 行)，并将当前的特征权重累计添加到物品 i 的相应标签权重特征 $s(i)$中(5 行). 迭代地判定每个元组，直到所有元组判定完毕(3～5 行)，最后规范化 $S(i)$(6 行)，返回得出所有参与标签权重计算的物品的标签权重特征 $S(I)$(7 行).

同算法 1 一样，算法的时间复杂度为 $O(N)$.

3.2　离线计算推荐算法

离线推荐算法应用在推荐引擎的离线计算模块. 本文的离线推荐算法利用了奇异值分解的性质，所以首先介绍下奇异值分解相关知识.

3.2.1　关于奇异值分解

奇异值分解[15]是线性代数中的一种重要的矩阵分解. 对于 1 个 $m\times n$ 的矩阵 $\boldsymbol{M}$ 的可以分解为 3 个矩阵相乘：$m\times m$ 的矩阵 $\boldsymbol{W}$，$m\times n$ 的矩阵 $\boldsymbol{\Sigma}$ 的、$n\times n$ 的矩阵 $\boldsymbol{V}$ 的转置. 表示如下

$$\boldsymbol{M}_{m\times n}=\boldsymbol{W}_{m\times m}\times\boldsymbol{\Sigma}_{m\times n}\times\boldsymbol{V}^{\mathrm{T}}_{n\times n}$$

其中：$\boldsymbol{\Sigma}=\begin{bmatrix}\sigma_1 & 0 & 0\\ 0 & \sigma_2 & 0\\ \cdots & \cdots & \cdots\end{bmatrix}$ 是奇异值矩阵，对角元素 $\sigma_1,\sigma_2,\cdots$，被称为矩阵 $\boldsymbol{M}$ 的奇异值，非负递减，其他元素为 0.

$\boldsymbol{W}=(w_1,w_2,\cdots,w_i,\cdots,w_m)$，$w_i$ 是矩阵 $\boldsymbol{M}$ 的左奇异向量，描述了 $\boldsymbol{M}$ 的特征，对应 σ_i；而且 σ_i 的值越

大表示 $\boldsymbol{M}$ 的特征 u_i 的重要性越大;其实,u_i 是方阵 $\boldsymbol{M}\boldsymbol{M}^{\mathrm{T}}$ 的特征向量.

$\boldsymbol{V}=(v_1,v_2,\cdots,v_i,\cdots,v_n)$,$v_i$ 是矩阵 $\boldsymbol{M}$ 的右奇异向量,同样描述了 $\boldsymbol{M}$ 的特征,对应 σ_i;而且 σ_i 的值越大表示 $\boldsymbol{M}$ 的特征 v_i 的重要性越大;其实,v_i 是方阵 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}$ 的特征向量.

在实际应用中,我们可以去掉那些重要性比较小的奇异向量,只保留 r 个奇异值、对应的 r 个左奇异向量以及对应的 r 个右奇异向量. 即

$$\hat{\boldsymbol{\Sigma}}=\begin{bmatrix}\sigma_1 & 0 & 0\\ 0 & \sigma_2 & 0\\ \cdots & \cdots & \sigma_r\end{bmatrix},\ \hat{\boldsymbol{W}}=(u_1,u_2,\cdots,u_r),$$

$$\hat{\boldsymbol{V}}=(v_1,v_2,\cdots,v_r)$$

于是我们得到原来矩阵的近似矩阵:

$$\boldsymbol{M}_{m\times n}\approx\hat{\boldsymbol{M}}_{m\times n}=\hat{\boldsymbol{W}}_{m\times r}\times\hat{\boldsymbol{\Sigma}}_{r\times r}\times\hat{\boldsymbol{V}}^{\mathrm{T}}_{r\times n}.$$

3.2.2 基于矩阵填充的物品推荐算法

离线计算推荐算法基本思想是:利用奇异值分解提取偏好评分矩阵的主要特征;并根据这些特征计算用户之间的相似度;根据用户相似度找到指定用户的 K 近邻;根据 K 近邻的评分数据填充指定用户的未评分数据,得到近似评分矩阵;根据近似评分矩阵的评分数据进行物品推荐.

具体流程如算法 3 所示.

算法 3. 基于矩阵填充的推荐算法.

输入:用户对物品的偏好评分矩阵 $\boldsymbol{R}(U,I)$
为每位用户选择 K 个邻居

输出:为每位用户推荐的物品矩阵 $\boldsymbol{I}(U)$

算法过程:

1. $\boldsymbol{I}(U)=\{\boldsymbol{I}(u)\}$,$\boldsymbol{I}(u)=\varnothing$//推荐物品列表为空
2. $\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}=SVD(\boldsymbol{R}(U,I))$//奇异值分解
3. $\boldsymbol{W}=(w_1,w_2,\cdots,w_m)$//得到左奇异向量
   $\boldsymbol{V}=(v_1,v_2,\cdots,v_n)$//得到右奇异向量
   $\boldsymbol{W}\approx\hat{\boldsymbol{W}}=(w_1,w_2,\cdots,w_r)$//近似左奇异向量
   $\boldsymbol{V}\approx\hat{\boldsymbol{V}}=(v_1,v_2,\cdots,v_r)$//近似左奇异向量
4. $\boldsymbol{R}(U,I)\approx\hat{\boldsymbol{R}}(U,I)=\hat{\boldsymbol{W}}\hat{\boldsymbol{\Sigma}}\hat{\boldsymbol{V}}^{\mathrm{T}}$//求得近似矩阵
5. foreach $u\in U$ do//计算用户之间相似度
6. for each $u_j\in U$ do $\alpha_j=sim(u,u_j)$
7. find K neighbors $N(u)$ which has highest similarity//找到 K 近邻
8. foreach $u\in U$ do//为每个用户推荐物品
9. for $j\leftarrow 1$ to r do //填充未评分特征
10. if u^j is blank in original $\boldsymbol{R}(U,I)$ then
11. $u^j=\sum_k\alpha_k\cdot u_k^j/\sum_k\alpha_k$//填充
12. $\boldsymbol{I}(u)\leftarrow L$ items which have top ratings //推荐
13. $\boldsymbol{I}(U)\leftarrow\boldsymbol{I}(U)+\boldsymbol{I}(u)$ //推荐结果离线存储
14. return $\boldsymbol{I}(U)$

算法流程如下:首先对推荐物品列表初始化为空(1 行),接着利用奇异值分解提取偏好评分矩阵的 r 个主要特征(2~3 行),利用奇异值分解求得近似矩阵(4 行). 对于每一个用户,循环计算与其他用户的相似度,并据用户相似度找到该用户的 K 近邻,直到每个用户均计算出了自己的 K 近邻(5~7 行). 对于每一个用户,根据该用户的 K 近邻的相似度,依次补充该用户的 r 个偏好评分特征矩阵中的未评分数据,得到完整的近似评分矩阵(8~11 行). 根据近似评分矩阵为用户推荐前 L 个排名的物品,并且将结果进行离线存储,最后返回物品推荐列表 $I(U)$(12~14 行).

用户相似度采用余弦相似度[16]进行计算,公式如下

$$sim(u_i,u_j)=\frac{u_i\cdot u_j}{\|u_i\|\cdot\|u_j\|}=\frac{\sum_{k=1}u_i^k\cdot u_j^k}{\sqrt{\sum_{k=1}(u_i^k)^2}\cdot\sqrt{\sum_{k=1}(u_j^k)^2}}$$

其中,u_i^k 表示用户 i 的第 k 个属性.

在计算出为用户推荐的物品后,推荐结果处理模块把我们上一节计算出的物品标签权重特征作为推荐原因,与物品推荐列表一起展示给用户.

对于 $m\times n$ 的矩阵,该算法时间复杂度为 $O(m^3)$. 考虑到该算法是在进行离线计算时运用的,所以对系统的性能影响不大. 下面我们给出在线计算推荐算法.

3.3 在线计算推荐算法

在线计算推荐算法应用在推荐引擎的在线计算推荐模块. 为了减少在线推荐所用时间,本文首先对用户进行两级聚类. 通过两级聚类,为指定用户进行推荐在线推荐时,只需要比较该用户与各类代表用户的相似度从而找到该用户所属的类,然后在类内部计算推荐.

3.3.1 两级聚类算法

两级聚类算法先利用用户的注册信息进行粗粒度聚类,再利用用户的标签权重特征进行细粒度聚类. 两级聚类所采用的算法是一致的,只是所利用的数据不同.

聚类算法基本思想是:初始每个用户单独成一个集合,利用用户注册信息特征向量进行粗粒度聚类或者利用用户标签权重特征进行细粒度聚类,计

算任意两个用户集合的相似度，然后从中选出相似度最大的两个用户集合进行合并，并作为一个用户集合参与下一轮的合并，直至用户集合只剩下我们所需要的个数. 另外，在聚类过程中，我们还会求出每个类的类代表，以便以后计算用户与类的相似度.

具体流程如算法 4 所示.

算法 4.　聚类算法.

输入：待聚类的用户集合 $U=\{u_1,u_2,\cdots,u_n\}$

需要生成的用户相似类的个数 k

输出：完成聚类的属性集合 $U'=\{U_1,U_2,\cdots,U_k\}$

算法过程：

1. $U'=\{U_i\}$，$U_i=\{u_i\}$//每个用户单独成类
2. $simpool=\varnothing$//记录相似对及其相似度
3. foreach $u_i\in U$ do
4. for each $u_j\in U$ do
5. $sim(U_i,U_j)\leftarrow sim(u_i,u_j)$//相似度
6. $simpool\leftarrow simpool+(U_i,U_j,sim(U_i,U_j))$
7. while $|U'|>k$ do
8. $U_P\leftarrow U_i+U_j$//合并相似度最大的类
9. $simpool\leftarrow simpool-(U_i,U_j,sim(U_i,U_j))$
10. for each $U_q\in U'$ do

$sim(U_p,U_q)\leftarrow(sim(U_q,U_i)+sim(U_q,U_j))/2$

11. $simpool\leftarrow simpool-(U_q,U_i,sim(U_q,U_i))$
12. $simpool\leftarrow simpool-(U_q,U_j,sim(U_q,U_j))$
13. $simpool\leftarrow simpool+(U_p,U_q,sim(U_p,U_q))$
14. $U'\leftarrow U'-U_i-U_j$//更新相似类集合
15. $U'\leftarrow U'+U_p$
16. return U'

算法流程如下：首先将每个用户初始化为一个单独的类，并令 $simpool$ 初始化为空(1～2 行)，其中 $simpool$ 用来存储所有类组成的相似对和相似度的集合，通过循环迭代，计算出每两个类之间的相似度，并将相应的相似对和相似度存储到 $simpool$ 中(3～6 行). 当类的个数大于 k 时，循环迭代(8～15 行)，寻找相似度最大的两个初始类合并为一个新类(8 行)，且从 $simpool$ 中去除这两个初始类组成的相似类和相似对(9 行). 并迭代更新 $simpool$(10～13 行)，计算每个类与这个新类的相似度——与两个初始类的相似度的均值(10 行)，且从 $simpool$ 中去除所有与这两个初始类之一相关的相似类和相似对，同时添加该类与新类组成的相似对和相似度(11～13 行)，直到扫描完所有的类(10～13 行). 同时从相似类集合中去除进行合并的两个类，而将新类添加到集合中(14～15 行)，故每次迭代会使得生成的相似类集合个数减 1，直到最终用户相似类类别为 k 为止. 最后返回 k 个用户相似类集合 U'(16 行).

在粗粒度聚类中，利用注册信息进行计算用户相似度时，我们采用 Jaccard 距离[17]相似度

$$sim(u_i,u_j)=\text{Jaccard}(u_i,u_j)=\frac{|u_i\cap u_j|}{|u_i\cup u_j|}$$

其中：$|u_i\cap u_j|$ 表示用户 u_i 和用户 u_j 的共有信息个数；$|u_i\cup u_j|$ 表示用户 u_i 和 u_j 的总的属性个数.

在细粒度聚类中，利用标签权重特征进行计算用户相似度时，我们采用余弦距离相似度

$$sim(u_i,u_j)=\frac{\boldsymbol{u}_i\cdot\boldsymbol{u}_j}{\|\boldsymbol{u}_i\|\cdot\|\boldsymbol{u}_j\|}$$

其中：$\boldsymbol{u}_i$ 表示用户 u_i 的标签权重特征向量，在 3.1 节计算得出.

我们利用类内部各用户信息的平均值计算出类代表用户 $\boldsymbol{u}$. 粗粒度类代表的属性信息使用类内部在该属性上出现最多的属性值. 细粒度类代表使用类内部用户标签权重特征的平均值作为代表用户的标签权重特征.

算法复杂度为 $O(n^2)$，其中 n 是用户的个数.

3.3.2　基于聚类的物品推荐算法

基于聚类的物品推荐算法基本思想是：在收到标签权重评分后，将其作为用户的临时标签权重特征，首先根据用户注册信息，通过计算与各个粗粒度类代表的相似度找到所属粗粒度类，再根据标签权重特征，通过计算与各个细粒度代表的相似度找到其所属的细粒度类，然后在细粒度内部找邻居，通过判定与类内用户的相似度，得出 K 个最近邻用户，并根据 K 近邻用户对物品的评分信息计算该用户的物品评分信息，最后根据该用户的评分数据进行物品推荐.

具体流程如算法 5 所示.

算法 5.　基于聚类的物品推荐算法.

输入：粗粒度聚类结果 $\boldsymbol{U}'=\{U_1,U_2,\cdots,U_j\}$

细粒度聚类结果 $\boldsymbol{U}''=\{U_j\,|\,U_j=\{U_{jk}\,|\,k=1,2,\cdots\}\}$ 为每位用户选择 K 个邻居

输出：为用户 u 推荐的物品推荐列表 $I(u)$

算法过程：

1. $\boldsymbol{I}(u)=\varnothing$//初始推荐物品列表为空
2. foreach $U_i\in U'$ do
3. $\alpha_i=sim(u,\boldsymbol{u}_i)$//与粗粒度类代表相似度
4. find U_j which has highest similarity
5. foreach $U_{jk}\in U_j$ do
6. $\alpha_{jk}=sim4(u,U_{jk})$//与细粒度类代表相似度
7. find U_{jl} which has highest similarity
8. for each $u_m\in U_{jl}$ do

9. $\alpha_m = sim5(u, u_m)$ //与类内用户相似度

10. find K neighbors $N(u)$

$$u^j = \sum_n^K \alpha_n \cdot u_n^j / \sum_n^K \alpha_n$$ //由邻居填充用户评分

11. $\boldsymbol{I}(u) \leftarrow L$ items which have top ratings

12. return $\boldsymbol{I}(u)$

算法流程如下:首先初始化物品推荐列表为空(1 行),依次计算出当前用户与每个粗粒度类代表的相似度(2～3 行),比较得出与当前用户相似度最大的粗粒度类(4 行).依次计算当前用户与这个粗粒度类内部的每个细粒度类代表的相似度(5～6 行),比较得出与当前用户相似度最大的细粒度类(7 行).循环迭代计算当前用户与该细粒度类内用户的相似度(8～9 行),并找出该用户的 K 近邻用户,根据 K 近邻的评分信息计算出该用户的用户评分信息(10 行).并为该用户推荐前 L 个排名的物品,最后返回物品推荐列表 $I(u)$(11～12 行).

为指定新用户进行物品推荐,需要先找到其所属的粗粒度类与细粒度类,然后找到类内部的邻居,根据邻居生成评分.所以算法时间复杂度为 $O(K_1 \cdot K_2 \cdot K_3)$.其中,$K_1$ 为粗粒度类个数,K_2 为细粒度类个数,K_3 为类内邻居数.

3.4 反馈计算推荐算法

反馈计算推荐算法应用在推荐引擎的反馈计算模块.算法基本思想是:收到用户的反馈时,首先计算该用户的真实评分与预估评分之间的差距,以该用户与邻居的相似度作为权重对邻居未评分数据进行调整.然后根据调整后的用户数据调用在线计算推荐算法进行推荐.

具体流程如算法 6 所示.

算法 6. 基于标签权重评分的用户反馈算法.

输入:用户反馈数据

$D' = \{(u,i,r,s_1),(u,i,r,s_2),\cdots,(u,i,r,s_k)\}$

用户 u 的邻居 $N(u)$

输出:为用户 u 推荐的物品推荐列表 $\boldsymbol{I}(u)$

算法过程:

1. $\boldsymbol{I}(u) = \varnothing$ //初始推荐物品列表为空

2. $\Delta r = \hat{r} - r$ //计算偏好评分误差

3. foreach $(u,i,r,s_j) \in D'$ do //标签权重评分误差

4. $\Delta s_j = getTagRating(\hat{s}_j) - getTagRating(s_j)$

5. foreach $u_k \in N(u)$ do//更新邻居评分

6. $\hat{r}_k = \hat{r}_k - \frac{\alpha_k}{\sum \alpha_m} \cdot \Delta r$//更新偏好评分

7. foreach $(u,i,r,s_j) \in D'$ do//更新标签权重评分

8. $update(getTagRating(\hat{s}_j) - \frac{\alpha_k}{\sum \alpha_m} \cdot \Delta s_j)$

9. $\hat{r} = \hat{r} - \Delta r$//更新该用户偏好评分

10. foreach $(u,i,r,s_j) \in D'$ do//该用户标签权重评分

11. $update(getTagRating(\hat{s}_j) - \Delta s_j)$

12. $\boldsymbol{I}(u)$ = 调用在线计算推荐算法($S(u)$)

13. return $\boldsymbol{I}(u)$

算法流程如下:首先初始化用户的推荐物品列表为空(1 行),再根据用户的预估评分 $\hat{r}$,用户的真实评分 r,计算出当前用户的预估评分与真实评分之间的差距 $\Delta r = \hat{r} - r$(2 行),对于用户的标签权重,依次循环迭代计算出当前用户的每个预估标签权重评分与真实标签评分的差距 Δs_j(3～4 行).对于该用户的每个邻居,迭代更新邻居的偏好评分和标签权重评分(5～8 行).每次迭代中,根据 Δr 及当前用户与邻居的近似度 α_k,更新该邻居的偏好评分(6 行).根据 Δs_j 及当前用户与邻居的近似度 α_k,循环更新当前邻居的每个标签权重评分值(7～8 行).直到所有邻居评分均更新为真实值为止.最后更新当前用户的偏好评分(9 行),并迭代更新当前用户的每个标签权重评分为真实评分(10～11 行).最后调用在线推荐算法计算出推荐物品列表 $\boldsymbol{I}(u)$,并返回推荐物品列表 $\boldsymbol{I}(u)$(12～13 行).

当收到用户的反馈,我们计算该用户的反馈评分与我们的预估评分之间的差距,然后只更新该用户邻居的相关评分数据,算法复杂度为 $O(1)$.

该算法避免了大规模数据的重新计算,而只调整与反馈用户相关的数据,花费时间少,适合在线推荐.

这一节我们介绍了在基于标签权重评分模型下的相关算法,分别应对离线计算推荐、在线计算推荐、反馈计算推荐 3 种基本推荐情况.如 2.4 节所述,这样就可以很好地解决冷启动、复杂兴趣推荐难、可解释性差这 3 个当前推荐系统所具有的问题.下面我们通过实验验证本文模型及算法的有效性.

4 实验验证

基于标签权重评分模型我们实现了一个电影推荐系统.开发工具为 MyEclipse10.6,运行环境为 ubuntu 12.04-32 位系统,计算机采用 3.10 GHz Intel(R) Core(TM) i5 2400 CPU 和 4 GB 内存.

实验中我们使用的数据集是 GroupLens 实验室提供的 MovieLens 的电影评分数据集.该数据集包括 1 000 209 个偏好评分记录和 95 580 个标签描

述记录，有 6040 个用户，3952 部电影和 1127 个标签.

通过 3.1 节数据预处理我们得到用户对物品的偏好评分以及用户和电影的标签权重评分. 其中，偏好评分从 1～5 分别表示为很不喜欢、不喜欢、一般、很喜欢和非常喜欢. 同样标签权重评分从 1～5 表示电影或用户与标签的关联度为：没有关联、很少关联、一般关联、很大关联和非常有关联.

4.1 离线计算推荐实验

在离线计算推荐电影的实验中，我们将用户对电影的偏好评分数据集均匀分成 M 份(本文 $M=8$)，将每个子集数据分别做一次验证集，其余 $M-1$ 份子集数据作为训练集，从而得到 M 个模型，用这 M 个模型最终的验证集的分类准确率的平均数作为整体的性能评价指标. 然后调用 3.2 节的基于矩阵填充的物品推荐算法为用户生成电影推荐列表. 在生成电影推荐列表时，我们采用如下规则：只有在用户对电影的预估评分≥3 分时才被推荐给用户.

我们采用的评测系统性能的指标是准确率、召回率和覆盖率[18]. 一个系统推荐覆盖率越高，系统给用户推荐的商品种类就越多，覆盖多样新颖的物品的可能性就越大. 如果一个推荐算法总是推荐给用户流行的商品，那么它的覆盖率往往很低，通常也是多样性和新颖性很低的推荐[18]. 当覆盖率相对较高时，多样性也会较高，新颖性也不会过低. 故覆盖率能有效反映推荐的多样性和新颖性指标，故采用覆盖率来间接双重反映系统的多样新颖性. 令 $\boldsymbol{I}(u)$ 为我们为用户 u 推荐的电影列表，$\boldsymbol{D}(u)$ 为测试集中用户 u 评分在 3 分以上的电影列表.

那么准确率和召回率的计算如下

$$Precision = \frac{\sum_{u\in U} |\boldsymbol{I}(u) \cap \boldsymbol{D}(u)|}{|\boldsymbol{I}(u)|}$$

$$Recall = \frac{\sum_{u\in U} |\boldsymbol{I}(u) \cap \boldsymbol{D}(u)|}{|\boldsymbol{D}(u)|}$$

覆盖率是衡量推荐系统推荐的物品在总的物品种类中所占比例的指标. 在本文中覆盖率计算如下

$$Coverage = \frac{|\bigcup_{u\in U} \boldsymbol{I}(u)|}{|\boldsymbol{I}|}$$

其中，U 为进行推荐的用户集合.

如 3.2 节所述，基于邻居评分信息为用户填充未评分数据，所以邻居数目 K 是一个很关键的参数. 我们通过改变 K 进行了对比实验. 实验结果如表 1 所示.

表 1　基于矩阵填充的推荐算法在不同 K 参数下的性能

K	准确率/%	召回率/%	覆盖率/%
5	16.99	8.21	51.33
10	20.59	9.95	41.49
20	22.99	11.11	33.17
40	24.50	11.83	25.87
80	**25.20**	**12.17**	**20.29**
160	24.90	12.03	15.21

从表 1 可以看到，选择 $K=80$ 左右会获得比较高的准确率和召回率. 另外可以看出，随着邻居数目的增加，覆盖率会不断降低，这是因为邻居数目的增加会导致为用户推荐的电影越来越倾向于热门电影，一些冷门电影就得不到推荐，覆盖率就会下降.

为了说明本文离线推荐算法(LWR-Offline)的优越性，我们选择 $K=80$，$L=10$，并且与现有的推荐算法 UPCC、IPCC、PMF、QSA 做了对比实验. 这 4 种推荐算法的简介如下.

UPCC[19] 这个方法使用皮尔逊相关系数对用户进行聚类，然后基于相似用户进行物品推荐.

IPCC[8] 这个方法使用皮尔逊相关系数对物品进行聚类，然后基于相似物品进行物品推荐.

PMF[7] 这个方法是基于用户、物品与标签的三元关系的协同过滤算法.

QSA[4] 这个方法基于用户-物品-标签-评分四阶张量的语义分析进行物品推荐.

对比实验结果如表 2 所示.

表 2　离线推荐算法与当前算法性能对比

Algorithm	准确率/%	召回率/%	覆盖率/%
UPCC	15.79	7.18	18.33
IPCC	15.86	7.95	19.15
PMF	19.89	9.83	19.17
QSA	21.43	11.96	20.02
LWR-Offline	**25.20**	**12.17**	**20.29**

可以发现，本文模型在准确率、召回率和覆盖率都优于参与对比的其他方法. 其中，5 种算法在覆盖率上都比较接近. 由于 PMF、QSA 还有本文算法加入了标签这一数据，所以在准确率和召回率上要高于简单的基于用户或电影的协同过滤推荐，而本文的基于标签权重评分的算法不仅加入了标签数据，还加入了对标签的权重描述，所以在性能上要优于 PMF 和 QSA.

因此，从表 2 的对比实验可以看出，尽管该方法用户操作设置较为复杂，但是它并没有降低离线模型的准确度，该方法的准确率、召回率、覆盖率相比

其它经典推荐算法均得到了提高.因为采用这种基于标签权重评分的模型相比以往的方法,离线训练数据中增加了标签权重值,同时结合用户的海量评分信息,能够对用户的品味做出更细化的划分,因此离线模型的准确度会更高.

4.2 在线计算推荐实验

这部分实验中我们结合离线计算的结果,调用在线计算推荐算法,为只有注册信息的新用户或者突然改变兴趣爱好的老用户进行推荐,即主要为了解决冷启动问题与复杂兴趣推荐问题.

实验设计如下:随机抽取 N(本文 $N=1000$)个用户,通过 3.1.1 节数据预处理获得这些用户的标签权重评分;从数据集中抽出这些用户的评分信息作为测试集,清除这些用户在数据集中的评分信息但保留注册信息,剩余数据作为训练集;把选中的 N 个用户看作新用户,当这些新用户登录至推荐系统后,把他们的标签权重评分输入系统以表达当前的兴趣要求,调用在线推荐算法,首先根据用户的注册信息找到所属粗粒度类,再根据用户标签权重评分在细粒度内找邻居,根据邻居对物品的评分信息计算出该用户的物品评分信息,最后进行物品推荐.数据集中 N 个用户的评分信息为测试集,且测试集中用户 u 评分在 3 分以上的电影列表为 $\boldsymbol{D}(u)$.通过在线推荐算法得到的物品推荐为 $\boldsymbol{I}(u)$.准确率、召回率、覆盖率计算公式同离线计算推荐实验.

计算 N 个用户的平均准确率和召回率.与离线计算推荐、Radom 推荐、Popular 推荐进行对比实验.其中,Radom 推荐是指为每次用户随机推荐 L(本文 $L=10$)部电影;Popular 推荐是指每次为用户推荐整体评分最高的 L(本文 $L=10$)部电影.结果如表 3 所示.

表 3 在线推荐算法性能

Algorithm	准确率/%	召回率/%	覆盖率/%
Random	0.765	0.455	100.00
Popular	10.730	5.950	3.10
LWR-Offline	25.200	12.170	20.29
LWR-Online	**18.150**	**10.080**	**16.17**

从表 3 可以发现,从多个角度分析,在线计算推荐算法(LWR-Online)在性能上与离线计算推荐算法是有一定差距,这是因为输入数据并没用到被推荐用户的本身的偏好评分信息,而只用了用户的标签权重特征.但是该算法的准确率和召回率远远高于 Random 推荐算法和 Popular 推荐算法,所以对于解决冷启动和复杂兴趣推荐问题还是很有帮助的.另外,对比研究前沿的算法[20]实验数据,针对数据集 MovieLens 时,该算法 $Recall\in[1\%,5\%]$,从召回率的角度分析,LWR-Online 算法明显优于该算法.对比算法 NBI[21]的实验数据,$Precision=16.2\%$,从精确度角度分析,LWR-Online 算法也优于 NBI 算法.故 LWR-Online 具有非常好的性能.另外,为了提高在线推荐算法的性能我们加入了用户反馈机制,我们将通过用户反馈计算推荐实验证明该机制的有效性.

4.3 反馈计算推荐实验

实验设计如下:随机抽取 N(本文 $N=1000$)个用户,首先进行 3.2 节的在线计算推荐实验,不同的是每次为用户推荐 L(本文 $L=10$)部电影,同时提供标签权重评分形式的推荐解释,即为每个电影配以 L(本文 $L=10$)个标签,并且每个标签都会有相应的权重表明该电影与该标签的关联性.然后用户可以通过标签权重评分对推荐的电影进行反馈,系统接收反馈,调用反馈计算推荐算法进行下一轮的推荐.

每一轮实验我们将数据集的 N 个用户的评分信息作为测试集,且测试集中用户 u 评分在 3 分以上的电影列表为 $\boldsymbol{D}(u)$,每次调用在线反馈机制后得出的物品推荐为 $\boldsymbol{I}(u)$.推荐准确率计算方式同离线计算推荐实验.每次调用我们都将计算平均绝对误差 MAE、推荐的准确率作为衡量用户反馈性能的指标.MAE 计算如下

$$MAE=\frac{\sum_{u,i\in D}|r(u,i)-\hat{r}(u,i)|}{|D|}$$

对比实验结果如表 4 所示.

表 4 反馈推荐算法在多轮反馈中的性能对比

Algorithm	MAE	Precision/%
Round1	1.15	18.63
Round2	1.03	20.22
Round3	0.95	23.17
Round4	0.93	23.45
Round5	0.95	23.44

从表 4 中可以看出,从第 1 轮到第 5 轮,我们通过用户反馈调整推荐使得平均绝对误差 MAE 有了 17% 的提升,准确率有了 25% 的提升,而且通过 5 轮反馈,对新用户的在线计算推荐的准确率已经非常接近对老用户的离线计算推荐,这再次体现出本文模型在解决冷启动与复杂兴趣推荐问题上的优越性.

另外,为了检验用户对基于标签权重评分的推荐解释的满意度,我们进行了一场用户问卷调查.我

们对 65 名用户进行问卷调查：每位用户被推荐一个自己比较喜欢的电影，并且会被提供 4 种推荐解释；每位用户要求对这些推荐进行评分，来表达他们对这些推荐解释的满意程度. 我们通过计算用户的平均评分来对比用户满意度. 评分从 1～5 分别表示非常不满意、不太满意、一般满意、很满意和非常满意.

调查结果如表 5 所示.

表 5　推荐解释的用户问卷调查

类型	平均分	排名
基于相似用户	3.6	4
基于相似物品	4.2	3
基于标签	4.4	2
基于标签权重评分	4.5	1

从表 5 中可以看出，用户对基于标签权重评分的推荐解释更满意，因为这样的解释不仅能清楚地给出推荐物品的特征，而且能描述物品在这些特征上的权重. 该调查展现了本文所提出的模型在解决推荐原因可解释性差问题上的优越性.

通过本模型的离线推荐算法、在线推荐算法及反馈推荐算法较相关推荐算法的实验对比，可以看出本模型的准确率、召回率均较高，这体现了本方法的性能上的优越. 同时表 5 说明用户对基于标签权重评分的推荐系统解释很满意，这表明本文提出方法的合理性.

5　总　结

为了解决当前推荐系统存在的冷启动、复杂兴趣推荐困难、可解释性差 3 个问题，本文提出了基于标签权重评分的推荐系统模型，分析了在该模型下的 3 种推荐方式在解决这 3 个问题上的优越性. 为了实现这 3 种推荐方式，我们进行了相关算法的深入研究. 最后通过实验验证了本文技术的有效性.

未来拟对基于标签权重的推荐系统的效率和有效性进行进一步优化，使之适应大规模数据. 另外一项未来的研究工作是如何提高离线训练数据的收集质量，进一步提高模型的准确度.

参　考　文　献

[1] Burke R. Knowledge-based recommender systems. Encyclopedia of Library and Information Science, 2000, 4: 2000

[2] Lü L, Medo M, Yeung C H, et al. Recommender systems. Physics Reports, 2012, 519(1): 1-49

[3] Liu Jian-Guo, Zhou Tao, Wang Bing-Hong. The research progress of personal recommender systems. Progress in Natural Science, 2009, 19(1): 1-15(in Chinese)
(刘建国，周涛，汪秉宏. 个性化推荐系统的研究进展. 自然科学进展，2009，19(1)：1-15)

[4] Wei C, Hsu W, Lee M L. A unified framework for recommendations based on quaternary semantic analysis//Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval. Beijing, China, 2011: 1023-1032

[5] Balabanovic M, Shoham Y. Fab: Content-based, collaborative recommendation//Proceedings of the Communications of the ACM. Zurich, Switzerland, 1997: 66-72

[6] Mooney R J, Roy L. Content-based book recommending using learning for text categorization//Proceedings of the 5th ACM Conference on Digital Libraries. San Antonio, USA, 2000: 195-204

[7] Salakhutdinov R, Mnih A. Probabilistic matrix factorization. Advances in Neural Information Processing Systems, 2008: 1257-1264

[8] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms//Proceedings of the 10th International Conference on World Wide Web. Hong Kong, China, 2001: 285-295

[9] Tso-Sutter K H L, Marinho L B, Schmidt-Thieme L. Tag-aware recommender systems by fusion of collaborative filtering algorithms//Proceedings of the 2008 ACM Symposium on Applied Computing. Fortaleza, Brazil, 2008: 1995-1999

[10] Zhang Z K, Zhou T, Zhang Y C. Tag-aware recommender systems: A state-of-the-art survey. Journal of Computer Science & Technology, 2011, 26(5): 767-777

[11] Zhang Z K, Liu C, Zhang Y C, et al. Solving the cold-start problem in recommender systems with social tags. EPL (Europhysics Letters), 2010, 92(2): 28002-28007(6).

[12] Schein A I, Popescul A, Ungar L H, et al. Methods and metrics for cold-start recommendations//Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Tampere, Finland, 2002: 253-260

[13] Lin J, Sugiyama K, Kan M Y, et al. Addressing cold-start in app recommendation: Latent user models constructed from twitter followers//Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 2013: 283-292

[14] Qu Yi-Heng, He Jia-Peng, Liang Zhou-Yang. The application of multidimensional scoring criteria in recommender systems. Collective Economy of China, 2008(21): 85-86(in Chinese)
(曲懿恒，何嘉鹏，梁周扬. 多维评分标准在推荐系统中的应用. 中国集体经济，2008(21)：85-86)

[15] Herlocker J L, Konstan J A, Riedl J. Explaining collaborative filtering recommendations//Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work. Philadelphia, USA, 2000: 241-250

[16] Tintarev N. Explanations of recommendations//Proceedings of the 2007 ACM Conference on Recommender Systems. Minneapolis, USA, 2007: 203-206

[17] Chen W, Hsu W, Lee M L. Tagcloud-based explanation with feedback for recommender systems//Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 2013: 945-948

[18] Zhu Yu-Xiao, Lü Lin-Yuan. Evaluation metrics for recommender systems. Journal of University of Electronic Science and Technology of China, 2012, 41(2): 163-175(in Chinese)
(朱郁筱, 吕琳媛. 推荐系统评价指标综述. 电子科技大学学报, 2012, 41(2): 163-175)

[19] Resnick P, Iacovou N, Suchak M, et al. GroupLens: An open architecture for collaborative filtering of netnews//Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work. Chapel Hill, USA, 1994: 175-186

[20] Zhang Z K, Zhou T, Zhang Y C. Personalized recommendation via integrated diffusion on user—item—tag tripartite graphs. Physica A: Statistical Mechanics and its Applications, 2010, 389(1): 179-186

[21] Zhou T, Ren J, Medo M, et al. Bipartite network projection and personal recommendation. Physical Review E, 2007, 76(4): 70-80

KONG Xin-Xin, born in 1994, M. S. candidate. Her research interests focus on data quality.

SU Ben-Chang, born in 1989, M. S. candidate. His research interests focus on data quality.

WANG Hong-Zhi, born in 1978, Ph. D., associate professor. His research interests include big data management, data quality, XML data management.

GAO Hong, born in 1966, Ph. D., professor, Ph. D. supervisor. Her research interests include wireless sensor networks, cyber-physical systems, massive data management and data mining.

LI Jian-Zhong, born in 1950, professor, Ph. D. supervisor. His research interests include wireless sensor networks, cyber-physical system, database, massive data processing etc.

Background

Recommendation System has been frequently applied into people's real-life. With improving users' acceptance, recommendation system has brought huge commercial interests. However, as the original data is incomplete and some recommendation algorithms have their own special ways of processing data, current recommendation system sometimes cannot work very well. For example, some recommendation systems are bothered with cold-start problem, difficult for complex interest recommendation problem, poor interpretability and so on. There already have existed some methods on solving these problems. However, they all have their own restrictions. As far as I know, there haven't had any effective methods to solve complex-interest recommendations problem.

In the paper, we propose a label-weight rating based recommendation system model. In the model, we will get the precise information of users' need by the simple way of using label-weight rating. Then we will generate recommendations using improved existing recommendation algorithm. At last, we will show the recommendations to the users in the form of label-weight rating. In addition, it is feasible for users to give their' feedback to the system and get more accuracy recommendations.

This work is supported in part by NGFR 973 grant 2012CB316200, NSFC grant 61472099, 61003046 and National Sci-Tech Support Plan 2015BAH10F00.