小 型 微 型 计 算 机 系 统
Journal of Chinese Computer Systems

2016 年 3 月 第 3 期
Vol. 37 No. 3 2016

基于用户的协同过滤算法的推荐效率和个性化改进

王 成[1,2],朱志刚[1],张玉侠[1],苏芳芳[1]

[1](华侨大学 计算机科学与技术学院,福建 厦门 361021)
[2](西安交通大学 机械强度与振动国家重点实验室,西安 710049)
E-mail:wangcheng@hqu.edu.cn

摘 要: 针对传统的基于用户的协同过滤算法存在的推荐效率、精度和个性化低的问题,提出一种改进方法.该方法在计算用户评分矩阵时,考虑到用户评分矩阵稀疏性,建立项目-用户的倒查表,只计算有相同评分项的用户之间的相似度,避免了传统方法中对所有用户计算两两用户相似度的庞大工作量.该方法在计算用户相似度时,考虑到项目的热门程度不同,"惩罚"了用户共同兴趣列表中的热门项目,避免了传统方法中赋予所有项目相同权值对推荐结果个性化的负面影响.本文在详细分析了改进的用户协同过滤算法的原理和优点,给出了其推荐步骤流程图.在 Movielens100K 和 HetRec2011-movielens-2k 公开数据集上,十折交叉验证的结果表明,改进后的算法节约了运行时间,提高了推荐算法的效率和个性化.

关 键 词: 基于用户的协同过滤;个性化推荐;相似度计算;用户评分矩阵;数据稀疏性;项目-用户倒查表;十折交叉验证
中图分类号: TP393 **文献标识码:** A **文 章 编 号:** 1000-1220(2016)03-0428-05

Improvement in Recommendation Efficiency and Personalized of User-based Collaborative Filtering Algorithm

WANG Cheng[1,2], ZHU Zhi-gang[1], ZHANG Yu-xia[1], SU Fang-fang[1]
[1](College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China)
[2](State Key Laboratory for Strength and Vibration, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: Aiming at low efficiency, precision and personalized of recommendation existing in traditional user-based collaborative filtering algorithm, an improvement recommendation algorithm is put forward in this paper. Considering data sparseness of user-rating-data matrix, this improvement recommendation algorithm introduces items—users inversion table, and only calculates users' similarity of whom with the same rating items in user-rating-data matrix. Therefore, it can avoid huge workload, which is brought by the pair-wise users' similarity calculation in traditional method. Considering different hot degree of items in users' similarity calculation, this improvement recommendation algorithm punishes the influence of popular items on users common interest lists for similarity. Therefore, it can avoid negative influence of popular items in recommendation results personalized, which is brought by giving the same weight of all items in traditional method. After analyzing principles and advantages of improved user-based collaborative filtering algorithm, new recommendation steps and flowchart are given. The experiment results of 10-fold cross-validation in Movielens100K and HetRec 2011 open datasets shows that improved algorithm could save running time, increase efficiency and personalization of recommendation.

Key words: user-based collaborative filtering; personalized recommendation; similarity calculation; user-rating-data matrix; data sparseness; items—users inversion table; 10-fold cross-validation

1 引 言

在网络信息过载和信息爆炸的背景下,个性化推荐技术利用用户和商品信息,可以高效、自动化地指导用户在电子商务网站中方便地浏览、购物而得到广泛应用,成为商业和学术研究的热点[1],例如,可以根据用户的资料为用户推荐他们可能感兴趣的图书、音乐、商品等项目的豆瓣网、淘宝网等.在推荐系统中,基于用户的协同过滤是应用最早和使用最为广泛、最为成功的推荐算法,但也存在的稀疏性、效率、精确性、冷启动和可扩展性等问题[2].

在实际的电子商务系统中,有成千上万的商品,而在这些商品中被用户评价过的商品只占了不到 1%,导致了用户-项目矩阵的稀疏.传统协同过滤算法在计算用户相似度的时候,需要计算两个用户对同一个项目的评分,这种方法的时间复杂度达到了 $O(n^2)$,这使得在用户数比较大时会非常耗时,难以找到最近邻居,进而影响到推荐的质量.

传统的基于用户的协同过滤算法中未考虑到项目的热门程度不同,在计算用户相似度时,赋予所有项目相同权值.John S. Breese 曾提出[3],如果两个用户购买过同一热门项目,但不能以此认为他们兴趣相似.因此,热门项目将对推荐

收稿日期:2014-12-22 收修改稿日期:2015-03-17 基金项目:国家自然科学基金项目(51305142,61103170)资助;厦门市科技计划项目(3502Z20143041)资助;福建省自然科学基金计划项目(2014J01191)资助;中国博士后科学基金第 55 批面上项目(2014M552429)资助;华侨大学引进人才科研启动项目(12BS217)资助. 作者简介:**王 成**,男,1984 年生,博士,副教授,研究方向为智能信息处理;**朱志刚**,男,1993 年生,研究方向为智能电子商务;**张玉侠**,男,1990 年生,硕士研究生,研究方向为智能电子商务;**苏芳芳**,女,1991 年生,硕士研究生,研究方向为智能电子商务.

结果的个性化造成负面影响.

2 改进的用户协同过滤算法

2.1 建立倒查表解决数据稀疏性问题

用户-项目矩阵的稀疏性如表 1 所示,用户 B 和 C 对项目 d 没有评分,而这在实际的电子商务系统中普遍存在.

表 1 用户-项目矩阵的稀疏性

Table 1 Data sparseness of user-rating-data matrix

用户	评分过的项目		
A	a	b	d
B	a	c	
C	b	e	
D	c	d	e

本文中,不去计算那些用户相似度为 0,即两用户之间没有对任何一个项目同时评分的情况. 建立项目到用户的倒查表,即对每个项目都保存对该项目评分过的用户列表. 假设用户 A 和用户 B 同时评论过 K 个项目,就有 Count[A][B] = K,从而可以倒查每个项目对应的用户列表,最终得到所有用户之间不为 0 的 Count[A][B]. 以表 1 的数据为例,建立的倒查表如表 2 所示.

表 2 项目到用户的倒查表

Table 2 Items—users inversion table

项目	对该项目评分过的用户	
a	A	B
b	A	C
c	B	D
d	A	D
e	C	D

2.2 对用户相似度的改进公式提高推荐精度和个性化

传统的基于用户的协同过滤算法中,未考虑到项目的热门程度不同,在计算用户相似度时,赋予所有项目相同权值. 例如,采用余弦相似度来计算两个用户 A 和 B 的兴趣相似度,N(A)表示用户 A 曾经评分过的项目集合,|N(A)|表示集合 N(A)中元素个数,N(B)表示用户 B 曾经评分过的项目集合,|N(B)|表示集合 N(B)中元素个数,计算公式[4] 如(1)所示:

$$w_{AB} = \frac{|N(A) \cap N(B)|}{\sqrt{|N(A)||N(B)|}} \tag{1}$$

利用余弦相似度的公式(1),如表 2 所示,用户 A 和用户 B 对同一项目 a 有过评分,则余弦相似度中的分子部分就加 1,可以得到用户 A 和用户 B 的兴趣相似度为:

$$w_{AB} = \frac{|\{a,b,d\} \cap \{a,c\}|}{\sqrt{|\{a,b,d\}||\{a,c\}|}} = \frac{1}{\sqrt{6}} \tag{2}$$

John S. Breese 曾提出[3],如果两个用户购买过同一热门项目,但不能以此认为他们兴趣相似. 因此,热门项目将对推荐结果的个性化造成负面影响. 以图书为例,如果两个用户都买过《新华字典》,并不能说明他们兴趣相似,因为《新华字典》是基本的汉语工具书,大部分人都有购买的需求. 但是如果俩人买过同一本冷门书籍,则可以认为他们兴趣相似. N(i)表示所有评分项目中包含项目 i 的集合,|N(i)|表示集合 N(i)中元素个数,利用 John S. Breese 提出的思想对用户相似度的公式进行改进:

$$w_{uv} = \frac{\sum_{i \in N(u) \cap N(v)} \frac{1}{\log(1 + |N(i)|)}}{\sqrt{|N(u)||N(v)|}} \tag{3}$$

公式(3)中对项目元素个数|N(i)|求对数后再取倒数,降低了热门项目对用户相似度计算的影响,“惩罚”了用户 u,v 共同兴趣列表中,热门项目对他们相似度的影响,有助于提高推荐结果的精度和个性化.

2.3 改进的用户协同过滤算法流程

1) 将数据集分为 M 份,选出其中的一份作为测试集,将剩下的 M-1 份作为训练集. 通常为了保证评测指标不是过拟合的结果,利用十折交叉法进行交叉验证,需要进行 M 次实验,并且每次使用不同的测试集,然后将 M 次实验测出的评测指标的平均值作为最终的评测指标[5].

2) 进行基于用户的协同过滤算法[6]:就是先找到和用户 A 有相似兴趣的其他用户,然后把那些用户喜好的、而用户 A 不知道的项目推荐给 A:

第 1 步. 找到和目标用户兴趣相似的用户集合.

第 2 步. 给用户推荐和他兴趣最相似的 K 个用户评价过的项目. 以下公式计算用户 u 对项目 m 的感兴趣程度:

$$p(u,m) = \sum_{v \in S(u,K) \cap N(m)} w_{uv} r_{vm} \tag{4}$$

其中,S(u,K)包含和用户 u 兴趣最接近的 K 个用户,N(m)是对项目 m 评分过的用户集合,w_{uv}是用户 u 和用户 v 的兴趣相似度,r_{vm}代表用户 v 对项目 m 的兴趣(因为使用的是单一行为的隐反馈数据[7],所以 $r_{vm}=1$).

改进后的算法流程如下页图 1 所示.

3 实验验证和结果分析

3.1 Movielens100K 和 HetRec2011-movielens-2k 实验数据集

Movielens100K 数据集[1] 包括了 943 个用户对 1682 部电影的 100000 条评分记录,评分范围是 1-5,数值越大表示评价越高,其中 u. data 表格式如表 3 所示.

表 3 u. data 表的格式

Table 3 Format of the u. data table

UserID	MovieID	Rating	Timestamp
用户编号	电影编号	评分	时间戳

本文使用其 u. data 表内的数据,主要对其中的 UserID,MovieID,Rating 三个字段进行计算.

HetRec2011-movielens-2k 数据集[2],其中包括了 2113 个用户对 10197 部电影的 855598 条评分记录,其中的 user_ratedmovies 表信息如下页表 4 所示.

和Movielens100K数据集一样,只采用其中的UserID,

[1] http://grouplens.org/datasets/movielens/

[2] http://grouplens.org/datasets/hetrec-2011/

MovieID,Rating 三个字段进行计算.

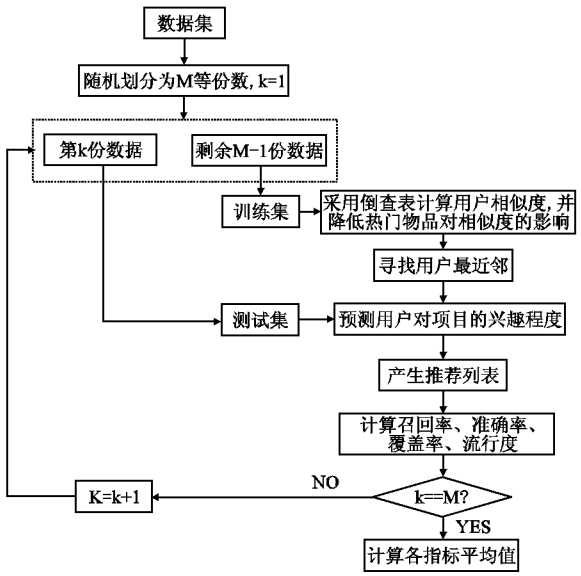

图 1 改进的基于用户的协同过滤算法流程

Fig. 1 Flow chart of improved user-based CF algorithm

3.2 精度和个性化评价指标

设向用户 u 推荐的 N 个电影集合为 R(u),测试集中用户真实评分过的电影集合为 T(u). 推荐系统的个性化评测指

表 4 user_ratedmovies 表的格式

Table 4 Format of the user_ratedmovies table

	MovieID	Rating	date_day	date_month
用户编号	电影编号	评分	日	月
date_year	date_hour	date_minute	date_second	
年	时	分	秒	

标一般有召回率、准确率、覆盖率和新颖度[10]等:

1) 召回率:又称"查全率",描述有多少比例的用户-电影评分记录包含在最终的推荐列表中,其计算公式为:

$$\text{Recall} = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |T(u)|} \tag{5}$$

2) 准确率:又称"精度"、"正确率",描述最终的推荐列表中有多少比例是已经发生过的用户-电影评分记录. 准确率的计算公式为:

$$Precision = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |R(u)|} \tag{6}$$

3) 覆盖率表示最终的推荐列表中包含多大比例的电影,反应了算法发现长尾的能力,覆盖率越高,说明算法越能将长尾推荐给用户. 覆盖率的计算公式为:

$$coverage = \frac{|R(u)|}{|M|} \tag{7}$$

M 为全部电影的集合,R(u)为向用户推荐的电影集合.

4) 流行度:本课题中流行度即为评价某项目的用户数. 计算公式为:

$$\text{Popularity} = \frac{\log_e(1+P)}{N} \tag{8}$$

其中 P 为平均流行度,N 为推荐的电影数.

3.3 实验环境参数介绍

因为重点是对隐反馈数据进行分析,所以预测用户会不会对某部电影评分,而不是预测用户在准备评分的前提下会评多少分,进行离线实验[8]. 实验的评价指标:有召回率、准确率、覆盖率、流行度 4 个指标. 利用十折交叉法[9]进行交叉验证,这里实验次数取 M = 8.

实验环境:处理器:Intel(R) Core(TM) i5CPU 2.67Hz,内存:4G,硬盘 500G.

运行环境:Win7(64 位)操作系统;开发语言为 Python 语言,版本 2.7.3;开发 IDLE:Python 自带 IDLE,版本:2.7.3.

4 实验结果及其分析

4.1 不同 K 值对推荐效果的影响

实验中的参数:K 值,即与目标用户相似的其他用户数目. 取用不同的 K 值会对推荐结果产生不同的影响[11],可以通过不同的取值结果找到该算法中效果最好的 K 取值.

以 K 为参数,Movielens100K 为数据集,运行的结果如下:

```
>>> ========== RESTART ==========
>>>
  K    召回率     准确率     覆盖率     流行度
 10   16.777%   19.629%   31.084%   5.367
 30   18.961%   22.185%   20.602%   5.522
 50   19.523%   22.842%   16.986%   5.572
 70   19.215%   22.481%   15.241%   5.605
 90   19.034%   22.269%   14.036%   5.623
110   18.680%   21.856%   13.193%   5.635
130   18.345%   21.463%   12.470%   5.647
150   18.372%   21.495%   11.928%   5.657
```

图 2 使用 Movielens100K 数据集的运行结果

Fig. 2 Running result of using Movielens100K dataset

从图 2 得到的运行结果可以看出,取不同的 K 值得到的召回率、准确率、覆盖率和流行度都不相同,且不成线性关系.

召回率和准确率:K 值在 50 左右会得到比较高的召回率和准确率,因此选取合适范围内的 K 对于获得高推荐精度比较重要.

流行度:因为 K 决定了在推荐时参考多少和目标用户兴趣相似的用户,K 越大,则参考的人越多,结果自然就越来越趋近于全局热门的电影.

覆盖率:随着 K 值的增长,覆盖率越来越低. 这是因为随着 K 的增大,流行度越来越大,参考的和目标兴趣相似的用户就越多,结果就越趋近于热门电影,从而对长尾电影的推荐就越少,造成了覆盖率的降低.

4.2 采用传统计算用户相似度的方法与采用倒查表方法运行时间的对比

这里第一个时间为程序初始时间,第二个时间为程序运行结束的时间,两者时间差即为程序运行所使用的时间.

从下页图 3 和图 4 的运行结果不难算出,采用传统方法计算用户相似度的运行时间为约 160 秒,而采用倒查表计算用户相似度的运行时间为约 145 秒. 通过采用倒查表,减少了对没有相同评分项的用户之间相似度的计算,从而节约了大

约 10% 的时间. 这在一定程度上解决了基于用户的协同过滤算法的稀疏性问题.

```
>>> ========== RESTART==========
>>>
  K    召回率    准确率    覆盖率    流行度
 10  16.768%  19.618%  31.084%  5.367
 30  18.961%  22.185%  20.602%  5.522
 50  19.523%  22.842%  16.988%  5.572
 70  19.224%  22.492%  15.241%  5.605
 90  19.034%  22.269%  14.036%  5.623
110  18.680%  21.856%  13.193%  5.635
130  18.345%  21.463%  12.470%  5.647
150  18.399%  21.527%  11.928%  5.657
2014-04-30 23:05:21
2014-04-30 23:08:00
```

图 3　采用传统方法计算用户相似度的运行时间

Fig. 3　Running time of calculating users similarity by using traditional methods

```
>>> ========== RESTART ==========
>>>
  K    召回率    准确率    覆盖率    流行度
 10  16.777%  19.629%  31.084%  5.367
 30  18.961%  22.185%  20.602%  5.522
 50  19.523%  22.842%  16.988%  5.572
 70  19.215%  22.481%  15.241%  5.605
 90  19.034%  22.269%  14.036%  5.623
110  18.680%  21.856%  13.193%  5.635
130  18.345%  21.463%  12.470%  5.647
150  18.372%  21.495%  11.928%  5.657
2014-04-30 23:09:26
2014-04-30 23:11:51
```

图 4　采用倒查表计算用户相似度的运行时间

Fig. 4　Running time of calculating users similarity by using inversion table

4.3　采用用户相似度改进方法后的对比

与图 3 的改进前的运行结果相比,采用 John S. Breese 提出的对用户相似度的改进公式(3)后,对项目元素个数 |N(i)| 求对数后再取倒数,大大降低了热门项目对用户相似

```
>>> ========== RESTART==========
>>>
  K    召回率    准确率    覆盖率    流行度
 10  16.659%  19.491%  32.711%  5.331
 30  18.988%  22.216%  21.687%  5.498
 50  19.541%  22.863%  17.590%  5.557
 70  19.550%  22.874%  15.904%  5.589
 90  19.052%  22.291%  14.639%  5.612
110  18.780%  21.972%  13.614%  5.628
130  18.517%  21.665%  12.831%  5.639
150  18.417%  21.548%  12.349%  5.650
```

图 5　采用用户相似度改进方法后的运行结果

Fig. 5　Running result of calculating users similarity by using improved method

度计算的影响,“惩罚”了用户共同兴趣列表中热门项目对相似度的影响,这使得推荐结果的召回率由原来的最高 19.523% 提升到 19.550%,准确率由原来的最高 22.842% 提升到 22.874%、覆盖率由原来的 31.084% 提升到 32.711%,进而提高了推荐的质量,能更好地实现推荐系统的个性化.

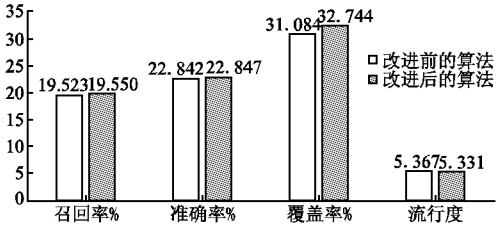

图 6　采用用户相似度改进方法后与改进前的对比

Fig. 6　Comparison result between calculating users similarity after by using improved method and before

4.4　使用 Movielens100K 数据集与 HetRec 2011 数据集对比

从图 2 与图 7 的对比结果可以看出:使用不同的数据集,总体来说推荐效果最好时对应的 K 值不同,Movielens100K 在 K = 50 左右获得较好的推荐,而 HetRec 2011 数据集在 K = 150 左右获得较好的推荐效果.

```
>>> ========== RESTART ==========
>>>
  K    召回率    准确率    覆盖率    流行度
 30  7.195%  32.248%  5.201%  6.908
 50  7.588%  34.009%  3.965%  6.955
 70  7.724%  34.619%  3.248%  6.981
 90  7.877%  35.305%  2.879%  6.997
110  7.931%  35.547%  2.610%  7.008
130  7.950%  35.632%  2.491%  7.015
150  7.968%  35.712%  2.391%  7.021
170  7.948%  35.622%  2.301%  7.026
190  7.927%  35.528%  2.182%  7.031
210  7.891%  35.367%  2.112%  7.035
```

图 7　使用 HetRec 2011 数据集的运行结果

Fig. 7　Running result of using HetRec 2011 dataset

换用大数据集后,召回率(查全率)有所下降,但是推荐的精度——准确率有了很大的提升. 如果希望推荐得到的内容越多越好,则是在追求召回率;如果希望推荐的内容中,真正想要的越多越好,则是在追求准确率. 根据本文的实验结果,两者不能两全,但是从用户体验的角度来看,推荐的内容有越多是自己真正感兴趣的,用户的体验会越好. 从这一点讲,在较大数据集上,基于用户的协同过滤推荐算法较为有优势.

使用大数据集后,覆盖率降低,表明在较大数据集上发现长尾的能力有所下降;而流行度增加,表明算法所推荐的项目随着数据集的增大,越来越接近热门项目.

4.5　采用不同计算相似度方法的结果对比

本文分别采用 Jaccard 系数、欧氏距离和皮尔逊系数计算出用户的相似度,得出最后的召回率、准确率、覆盖率和流行度四个指标的值,并将三种方法与利用余弦相似度计算的结果在如下图中列出:

通过图 8-图 11(见下页)的对比可以得出,利用余弦相似度计算的结果召回率、准确率、覆盖率都比其他方法要高,同时流行度也较低.

5 结 论

本文通过引入了项目-用户倒查表,只计算有相同评分项的用户之间的相似度,避免了传统方法计算任意两两用户相似度的复杂工作量,节约了运行时间;同时,利用"惩罚"用户

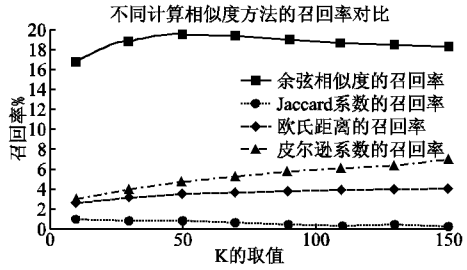

图 8 不同相似度计算方法对比_召回率

Fig. 8 Comparison result among calculating users similarity by different methods _recall ratio

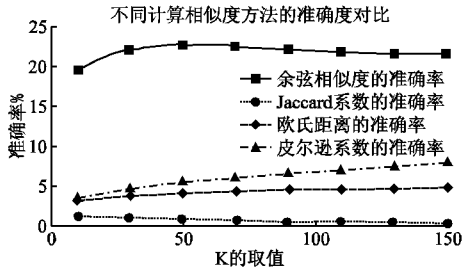

图 9 不同相似度计算方法对比_准确率

Fig. 9 Comparison result among calculating users similarity by different methods _accuracy

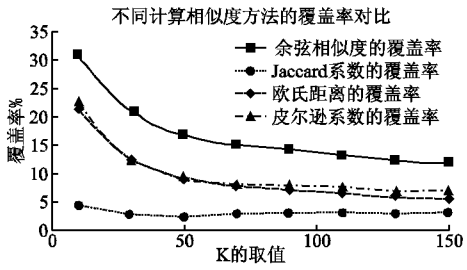

图 10 不同相似度计算方法对比_覆盖率

Fig. 10 Comparison result among calculating users similarity by different methods _coverage

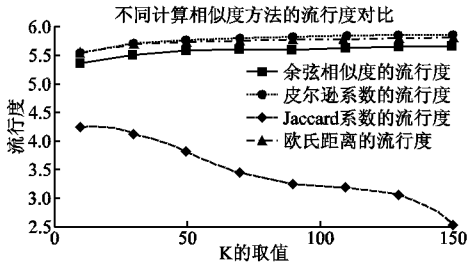

图 11 不同相似度计算方法对比_流行度

Fig. 11 Comparison result among calculating users similarity by different methods _popularity

共同兴趣列表中热门项目对用户相似度的影响这一方法,改进了用户相似度的计算,进而可以更好地实现推荐系统的个性化.最后,本文根据在不同数据集下推荐算法的评价指标比较,分析了对算法质量的影响因素,以及产生影响的原因;对比了不同计算用户相似度的方法,得出采用余弦相似度方法得出的推荐结果的平均效果最好的结论.

References:

[1] Zhu Xia, Song Ai-bo, Dong Fang, et al. A collaborative filtering recommendation mechanism for cloud computing [J]. Journal of Computer Research and Development, 2014, 51(10): 2255-2269.

[2] Wen Wu, Liang He, Jing Yang. Evaluating recommender systems [C]. Digital Information Management (ICDIM), 2012 Seventh International Conference on, 2012: 56-61.

[3] John S Breese, David Heckerman, Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering [M]. Morgan Kaufmann Publishers, 1998.

[4] Zhang Jin. Collaborative filtering algorithm analysis and research in E-commerce recommendation system [D]. Beijing: Capital University of Economics and Business, 2012.

[5] Leng Ya-jun, Lu Qing, Liang Chang-yong. Survey of recommendation based on collaborative filtering [J]. Pattern Recognition and Artificial Intelligence, 2014, 27(8): 720-734.

[6] Shi Feng-xian, Chen En-hong. Combining the Items' discriminabilities on user interests for collaborative filtering [J]. Jounal of Chinese Computer Systems, 2012, 33(6): 1533-1536.

[7] Yin Jian, Wang Zhi-sheng, Li Qi, et al. Personalized recommendation based on large-scale implicit feedback [J]. Journal of Software, 2014, 25(9): 1953-1966 .

[8] Chen Nuo-yan. The design and implementation of recommadation system based on personalized recommendation engine combination [D]. Guangzhou: South China University of Technology, 2012.

[9] Liu Bei-lin. A study of personalized recommendation evaluation based on customer satisfaction in E-commerce [J]. Computer Science and Service System (CSSS), 2011 International Conference, 2011: 129-135.

[10] Liu Jian-guo, Zhou Tao, Guo Qiang, et al. Overview of the evaluated algorithms for the personal recommendation systems [J]. Complex Systems and Complexity Science, 2009, 6(3): 1-10.

[11] Zhang Jin-bo, Lin Zhi-qing, Xiao Bo, et al. An optimized item-based collaborative filtering recommendation algorithm [C]. Network Infrastructure and Digital Content, 2009, IC-NIDC 2009, IEEE International Conference on, 2009: 414-418.

附中文参考文献:

[1] 朱 夏,宋爱波,东 方,等.云计算环境下基于协同过滤的个性化推荐机制[J].计算机研究与发展,2014,51(10):2255-2269.

[4] 张 进.电子商务推荐系统中协同过滤算法的分析与研究[D].北京:首都经济贸易大学,2012.

[5] 冷亚军,陆 青,梁昌勇.协同过滤推荐技术综述[J].模式识别与人工智能,2014,27(8):720-734.

[6] 施凤仙,陈恩红.结合项目区分用户兴趣度的协同过滤算法[J].小型微型计算机系统,2012,33(6):1533-1536.

[7] 印 鉴,王智圣,李 琪,等.基于大规模隐式反馈的个性化推荐[J].软件学报,2014,25(9):1953-1966.

[8] 陈诺言.基于个性化推荐引擎组合的推荐系统的设计与实现[D].广州:华南理工大学,2012.

[10] 刘建国,周 涛,郭 强,等.个性化推荐系统评价方法综述[J].复杂系统与复杂性科学,2009,6(3):1-10.