协同过滤推荐算法对比分析与优化应用①

张学钱, 林世平, 郭 昆

(福州大学 数学与计算机科学学院, 福州 350108)

摘 要: 协同过滤推荐算法分为基于内存和基于模型的推荐算法, 协同过滤推荐算法存在数据稀疏性、可扩展性、冷启动等问题. 通过基于用户、基于项目协同过滤推荐算法以及 SVD、Slope-One、KNN 等基于模型协同过滤推荐算法对比分析. 提出加入特征向量维度优化的 SVD 算法, 通过降维改善数据稀疏性问题. 利用 Hadoop 分布式平台改善推荐算法可扩展性问题. 基于 MovieLens 数据集实验结果表明, 引入基于 Boolean 相似性计算方法的推荐效果更优, 引入数量权重和标准差权重的优化 Slope-One 算法和引入特征向量维度的优化 SVD 算法推荐效果更优.

关键词: 协同过滤; 相似性; Hadoop; Slope-One; SVD

Collaborative Filtering Recommendation Algorithm Analysis and Optimization Applications

ZHANG Xue-Qian, LIN Shi-Ping, GUO Kun

(College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: The collaborative filtering recommendation algorithm is divided into user-based and item-based recommendation algorithms. Collaborative filtering recommendation algorithm had data-sparseness and scalability and cold-start problems. This paper mainly studied the collaborative filtering recommendation algorithm based on the users or Items and SVD, Slope-One, KNN. The optimization of SVD algorithm which considers the dimension of the feature space used dimension reduction to improve data-sparseness problem. Using the Hadoop distribution platform to improve the scalability problem. Experimental result shows that the similarity computation method based on Boolean data has better result and the optimization of Slope-One and SVD algorithm have better recommendation result based on MovieLens data set.

Key words: collaborative filtering; similarity; Hadoop; Slope-One; SVD

随着信息化时代的高速发展和移动互联网的普及, 互联网海量网络信息利用率低及用户在面对海量数据束手无策, 称为“信息过载”问题. 个性化推荐系统利用用户历史信息构建推荐模型向用户推荐感兴趣的信息, 个性化推荐系统在电子商务、电影推荐、音乐、视频推荐等个性化服务领域取得巨大成功[1]. 推荐算法主要有: 基于内容推荐算法、协同过滤推荐算法、基于社交网络推荐算法等[2], 本文主要研究内容是协同过滤推荐算法, 通过对比各基础协同过滤推荐算法推荐效果并提出推荐算法的优化方法, 基于电影数据集 MovieLensfe 进行实验分析结果[3].

信息时代进入“PB”为单位的数据新时代, 大数据具有数量大、种类多、处理速度快、价值高等特征, 著名照片分享网站 Flickr 有 1.3 亿张照片、博客网站 Technorati 有 4100 万篇博文和 25 亿各链接. 数据集规模呈几何增长, 传统推荐引擎受资源限制和推荐算法可扩展性限制, 利用分布式计算处理大规模数据成为研究热点. 结合 Hadoop 分布式云平台, 利用 HDFS 分

① 基金项目:国家自然科学基金(61300104)

收稿时间:2014-09-13;收到修改稿时间:2014-10-20

布式文件系统和MapReduce编程框架搭建分布式云计算平台, 实现大数据集推荐系统[4-6].

1 相关工作

文章[7]提出利用机器学习算法在多核 Hadoop 平台实现并行化推荐系统. 协同过滤推荐算法存在数据稀疏性问题, 文章[8]采用修正的条件概率方法计算项目之间相似性, 改善数据的稀疏性问题. 文章[9]利用降维方法来进行优化, 分别采用 LSI、SVD 对电子商务平台快速推荐. 文章[10]提出一种基于紧邻评分填补的协同过滤推荐算法, 对原始评分矩阵进行全局降维并利用近邻评分对缺失值进行填充. 文章[11]提出混合推荐思路, 利用特征加权与交换混合策略, 通过对数据集分割交叉验证, 优化推荐性能并提高推荐系统健壮性. 文章[12]提出特征增加与加权混合方法, 用户评分数和项目被评分数为主要特征, 通过二次回归, 利用随机梯度下降法求解确定线性函数. 文章[13]提出电影相似性权重, corr 的平方除以(1-corr)的平方或利用电影之间的均方差距离 Mes-10 次幂来计算相似性权重, 优化相似性计算. 文章[14]主要内容提出利用模型最小化总体计算代价确定最近邻居集, 利用用户显示反馈和隐式反馈扩展模型, 并结合因式分解方法, 提高算法推荐准确率. 文章[15]基于项目 KNN 推荐算法利用非负二次优化, 结合最小二乘法方法最小化误差的平方.

2 协同过滤推荐算法

协同过滤推荐算法核心思想是基于用户-项目评分矩阵, 计算用户或项目之间相似性来收集最近邻居集, 根据最近邻居集预测目标用户对目标项目的评分值. 本文主要工作研究基于用户和基于项目的协同过滤推荐算法, 结合不同相似性计算方法. 还有 SVD、KNN、Slope-One 等推荐算法进行实验对比, 比较 MAE、准确率、召回率、F-measure 评价函数, 提出 Slope-One 算法和 SVD 算法的优化方法.

2.1 SVD 推荐算法及优化

奇异值分解(SVD), 通过降低用户-项目评分矩阵维度来改善评分矩阵稀疏性问题[9,10]. SVD 是矩阵分解技术, 将矩阵 R 分解为 U、S、V 三个矩阵, 同时构造矩阵 R: $SVD(R)=[U、S、V]$, 其中 U 是 RRT 的特征向量矩阵、V 是 RTR 的特征向量矩阵、S 是 RTR 和 RRT 的共同特征值从大到小的排序[10]. 将矩阵 R 进行 SVD 分解后, 将 U、S、V 降到 K 维方阵. SVD 推荐算法优化方法可以对用户行为进行分析与建模, 并与 SVD++模型进行融合; 还可以利用评分矩阵的差分矩阵来表征局部结构信息, 并作为新的目标函数来优化 SVD 推荐算法; 将奇异值分解和符号数据分析方法结合起来运用到推荐系统中.

本文引入特征空间维度 n_f, 代表模型当中隐藏变量数量. 设定不同隐藏变量数使得评价函数 *MAE* 取值越小, 但存在特征空间维度取值过大会导致数据过拟合问题, 通过 *MAE* 评价函数添加吉洪诺夫正则化来避免过拟合问题. 本文提出正则化加权选择最小二乘法迭代算法来解决近似低阶问题. $MAE(R,U,I)$表示预测评分与真实评分值平均绝对误差, 正则化 *MAE* 公式:

$$MAE'(R,U,I)=MAE(R,U,I)+(\|U\Gamma u\|^2+\|I\Gamma i\|^2)$$

由于评分矩阵稀疏性, 标准 SVD 算法无法获取完整的用户特征矩阵 U 和项目特征矩阵 I. 本文利用交替最小二乘法解决低阶矩阵分解问题, 具体步骤如下:

步骤 1: 利用电影评分值平均值初始化矩阵第一行, 利用少量随机数填充剩余矩阵缺失值.

步骤 2: 固定矩阵 I, 通过最小化目标评价函数确定用户矩阵 U.

步骤 3: 固定矩阵 U, 通过最小化目标评价函数类似方法确定项目矩阵 I.

步骤 4: 重复步骤 2 和步骤 3 直到满足停止条件.

当正则化矩阵是非奇异的, 步骤 2 和步骤 3 都将有唯一解. 评价函数 *MAE* 计算误差是单调递减且有下界, 因此该排序具有收敛性. 通过设置停止条件来计算训练集数据的 *MAE* 值. 用户矩阵 U 和项目矩阵 I 更新后, 如果 *MAE* 变化值小于 0.001, 迭代终止, 完成 SVD 推荐算法的优化.

2.2 基本 Slope-One 推荐算法及优化

Slope-One 推荐算法本质是基于项目的协同过滤推荐算法, 通过新项目与用户评分过项目之间平均评分差值来预测用户对新项目的评分. Slope-One 通过简单的线性回归模型来对新 Item 进行评分, 不需要计算 Item 之间的相似度. 通过线性公式 $Y=X+d$ 由项目 X 的评分预测项目 Y 的评分, 参数 d 即两两项目之间的平均评分差值, 算法具体步骤如下:

步骤 1: 用户 U 对任意两个项目 i、j 的评分值 R_{ui}、R_{uj}, 评分值符合线性关系 $R_{ui}=R_{uj}+d_{ij}$, 公式 1 表示项目

间的差值.

$$d_{ij}=\sum_{u\in U_i\cap U_j}\frac{R_{ui}-R_{uj}}{\left|U_i\cap U_j\right|} \tag{1}$$

步骤 2: 如果 R_{ui} 是用户 u 评分的项目, 利用公示 $R_{uj}(i)=R_{ui}+d_{ij}$ 可以预测用户 u 相对于项目 i 对项目 j 的预测评分值. 公式 2 表示用户 u 对项目 j 的预测评分值取算术平均值作为最终预测评分.

$$\hat{R}_{uj}=\frac{1}{|S_u|}\sum\nolimits_{i\in S_u}(R_{ui}-d_{ij}) \tag{2}$$

目前对于 Slope-One 推荐算法优化有文章[3]提出基于动态近邻优化算法, 只对当前活跃用户 K 紧邻进行筛选, 用于保证与当前活跃用户相似用户的评分参与推荐; 引入描述关键字的语义相似度, 利用关键字相似性度量项目间的相似程度, 并结合该用户对其他项目的评分; Slope-One 算法空间复杂度过高, 提出并行 Slope-One 算法 PSL; 通过相似性动态选择邻居集然后基于相关项目对生产邻居集排名, 最后利用线性回归模型预测评分; 还有提出利用聚类算法对用户评分数据进行分类, 再利用 Slope-One 推荐算法预测评分值.

2.3 Slope-One 优化

由于基本 Slope-One 算法仅通过简单线性方法对于差值的可靠性和用于计算平均差值的数据量缺少考虑, 基于大量评分数据计算平均差值可靠性更高, 本文提出同时加入数量和标准差的权重. 引入数量权重, 在计算项目差值时所用到的数据量大小作为权重, 平均差值变成加权平均差值. 基于标准差的权重是通过对评分值差值的标准差来计算权重大小, 如果两个物品评分的差值对于大量评分数据是相近的, 具有更高的可靠性, 因此赋予更高的权重. 通过实验结果对比基本 Slope-One 算法、基于数量权重 Slope-One 算法、基于数量和基于标准差权重 Slope-One 算法的推荐效果. 公式 3 是加入数量权重推荐算法公式, 公式 4 是同时加入数量和标准差权重推荐算法公式, 公式 5 表示数量权重与标准差之间的关系, 标准差越大数量值越小.

$$p(u)_j=\frac{\sum_{k\in I(u)-\{j\}}(dev_{jk}+u_k)\cdot card(R_{j,k}(u))}{\sum_{k\in I(u)-\{j\}}card(R_{j,k}(u))} \tag{3}$$

$$p'(u)_j=\frac{\sum_{k\in I(u)-\{j\}}(dev_{jk}+u_k)\cdot card'(R_{j,k}(u))}{\sum_{k\in I(u)-\{j\}}card'(R_{j,k}(u))} \tag{4}$$

$$card'(R_{j,k}(u))=\frac{card(R_{j,k}(u))}{1+\mathrm{dev}_{jk}} \tag{5}$$

$card(R_{j,k}(u))$表示训练集用户对项目 i 和项目 j 共同评分的用户数量, $k\in I(u)-\{j\}$表示排除对项目 j 评分过的用户. Slope-One 需要预处理所有项目对之间的评分差值的计算, 通过离线计算项目之间的差值, 在线推荐效率更高.

3 基于Hadoop协同过滤推荐算法

3.1 MapReduce 分布式编程思想

本文提出利用构建项目的共现矩阵, 建立用户对项目的评分矩阵, 通过矩阵计算实现基于项目的协同过滤推荐算法分布式计算. 通过计算该推荐算法 10 万、100 万条数据集的推荐时间, 计算单机与分布式的加速比.

3.2 基于项目的分布式推荐算法

算法流程主要由 3 部分组成, 如图 1 所示. 共现矩阵用于描述项目间关联的方阵, 行数和列数等于项目数. 利用用户评分矩阵中不同项目对共同出现的次数填充矩阵, 如表 1 所示. 共现矩阵的共现关系类似于项目之间相似性方法, 两个项目共同出现次数越多表示项目之间相似性或者相关性越高. 构建共现矩阵利用用户-项目评分矩阵当中用户是否对项目评分过的隐式信息.

图 1　基于项目的分布式推荐算法流程

表 1　共现矩阵

	Item1	Item2	Item3	Item4
Item1	5	4	3	2
Item2	3	3	4	5
Item3	3	2	4	5
Item4	4	3	2	3

3.3 基于 MapReduce 的算法实现

步骤 1:　按用户 ID 进行分组, 计算所有项目出现的列表, 得到用户对项目的评分矩阵. Map 阶段输入数据采用 userID:itemID1 itemID2 itemID3 的形式, 输出新的键值对: 用户 ID 及其对应的项目 ID, 合并每个用户所对应的项目 ID. Reduce 阶段利用全部项目为用户构造一个用户的 ID 及该用户的评分向量:

ItemID[UserID1:value1,UserID2:value2,UserID3:value3 …].

步骤 2: 计算共现关系, 输入是步骤 1 的输出 ItemID[UserID1:value1,UserID2:value2,UserID3:value3 …], Map 阶段根据用户的评分值来决定所有的共现关系, 并为每一个共现关系生产一个项目 ID 对. Reduce 阶段为每个项目 ID 统计对应的全部共现关系, 构造一个出现次数的新向量.

步骤 3: 推荐结果, 共现矩阵乘积用户评分向量, 得到结果向量: UserID [ItemID 1:value1, ItemID 2:value2 …]. 项目 value 值最大值的 ItemID 为最佳推荐项目, 根据 value 值从大到小排序形成推荐列表.

4 实验与分析

4.1 实验方案

本次测试试验数据集采用电影推荐网站提供的评分数据集 MovieLens[2], 分值从 1 到 5, 分值越高表明用户对电影评价越高. 通过基于用户、基于项目、优化 Slope-One、优化 SVD、基于 KNN 线性插值等推荐算法实验对比.

相似性计算最常用三种方法: 皮尔逊、欧式距离、余弦. 本文结合隐式反馈计算最近邻居集以提高预测准确度, 相比与前面使用基于启发式相似性方法, 因此引入用户是否评分过的隐式信息, 利用基于 Boolean 类型的相似性计算方法: 谷本系数和对数似然两种方法.

实验数据集划分为 80%训练集、20%测试集, 通过实验分析评价标准 MAE、precision、recall 以及 F-measure 值大小. 基于 Hadoop 协同过滤推荐算法通过实验数据集评分数分别为 10 万、100 万运行时间与单机运行时间之间的加速比.

4.2 评价标准

推荐系统评价标准主要有: 平均绝对差值(MAE)、准确率(Precision)与召回率(Recall)等. MAE 值越小推荐系统评分预测越准确, 推荐效果越好. 准确率(Precision)与召回率(Recall)是针对 Top-N 推荐评价标准. 准确率与召回率相互制约, 利用综合推荐评价标准 F-measure 进行评定. 加速比是基于 Hadoop 平台运行时间与基于单机算法运行时间之间的比值.

4.3 实验结果

4.3.1 基于用户协同过滤推荐算法

基于用户协同过滤推荐算法, 通过固定邻域值结合各相似性方法计算 MAE、Precision、Recall、F-measure 值. 随着最近邻居集数的增大 MAE 值越小, 当最近邻居集数在 200-300 之间 MAE 值最优. 基于评分值大小皮尔逊相似性方法具有更优推荐效果, 基于隐式反馈的 Boolean 类型谷本系数与对数似然相似性方法推荐效果相近, 但都优于基于具体评分值相似性计算方法.

表 2 不同相似性方法计算平均 MAE 值

相似性计算方法	平均 MAE 值
皮尔逊相关系数	0.862
欧式距离	0.849
余弦相似性	0.906
谷本系数	0.815
对数似然	0.814

基于用户协同过滤推荐算法结合各相似性方法计算 Precision、Recall、F-measure, 综合评价 F-measure 如图 2 所示. Precision、Recall、F-measure 值大小与邻居数量成正比关系, 其中基于隐式 Boolean 类型的谷本系数相似性和对数似然相似性计算 F-measure 值优于其他方法计算结果, 因为两种相似性方法在计算时只考虑用户是否对该项目评分过(0 或 1)而不考虑用户对项目的具体评分值, 准确率与召回率结果更优.

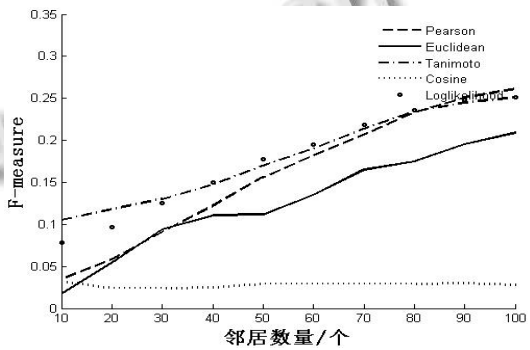

图 2 综合评价标准: F-measure

4.3.2 基于项目协同过滤推荐算法

基于项目协同过滤推荐算法结合各相似性方法计算 MAE 值结果如图 3 所示. 各相似性方法计算 MAE 值相差不大, 但基于用户隐式信息 Boolean 类型相似性计算方法计算的 MAE 值结果更优. 同时基于项目协同过滤推荐算法比基于用户协同过滤推荐算法具有更好的稳定性.

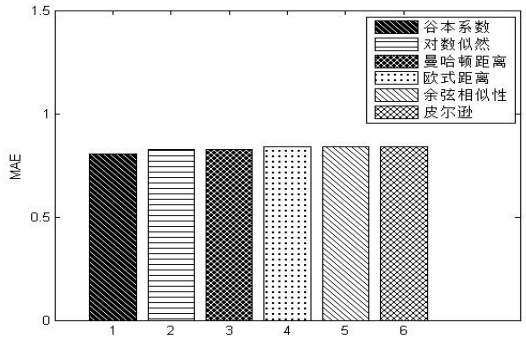

图 3　基于项目协同过滤推荐算法 MAE 值

4.3.3 Slope-One 算法

基本 Slope-One 推荐算法和加入数量权重和标准差权重优化 Slope-One 算法, 通过实验测试数据集大小分别为 100k、1M、10M. 实验结果如图 4 所示, 基于数量和标准差权重的推荐效果优于仅基于数量或者基本 slope-One 算法. 随着数据集规模增大, Slope-One 算法推荐效果更佳, 因为用户之间差值通过更多用户评分值计算的, 具有更高稳定性和可靠性. 实验结果显示同时加入数量和标准差权重比基本 Slope-One 算法推荐效果有所提高.

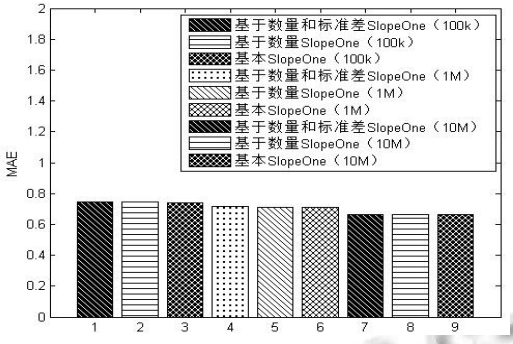

图 4　Slope-One 算法 MAE 值

4.3.4 SVD 推荐算法

该方法随着迭代次数增加或隐藏特征数增加不存在数据过拟合现象. a 代表 γ=0.3, b 代表 γ=0.4, c 代表 γ=0.5. 固定隐藏特征维度、不同迭代次数、不同正则化参数 γ. 实验结果如图 5 所示, 随着迭代次数和 γ 增大, MAE 值越小并趋于收敛.

通过固定迭代次数、选取不同特征维度、不同正则化参数 γ 进行实验, 实验结果如图 6 所示, 随着隐藏特征维度和 γ 值增大 MAE 取值越小, 随着特征维度越大推荐效果趋于相近.

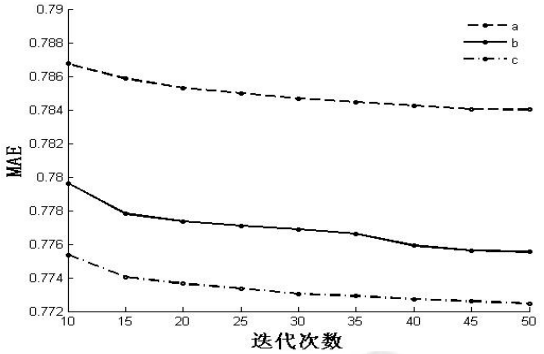

图 5　固定特征维度

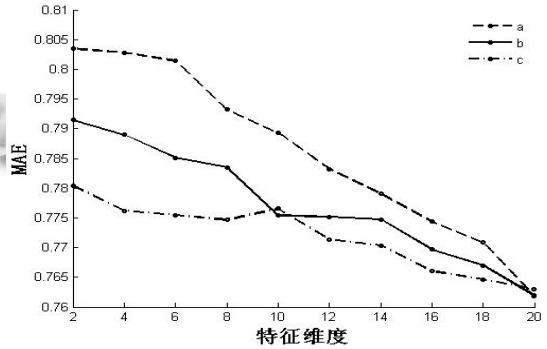

图 6　固定迭代次数

SVD 优化算法和项目协同过滤推荐算法对比: 计算 SVD 推荐算法推荐 10 个项目的绝对误差值—MAE 与基于项目协同过滤推荐算法对比, 分别基于 10 万、100 万、1000 万三种类型数据集计算 MAE. MAE 值优化百分比分别为: 6.56%、9.44%、9.54%.

具体如表 3 所示.

表 3　优化 SVD 与基于项目推荐算法 MAE 值对比

数据集 / MAE	10 万	100 万	1000 万
SVD	0.769	0.729	0.673
平均—Item	0.823	0.805	0.744
皮尔逊—Item	0.840	0.780	0.710
欧式距离—Item	0.836	0.829	0.769
谷本系数—Item	0.794	0.789	0.731
对数似然—Item	0.825	0.824	0.768

4.3.5 基于 Hadoop 分布式平台

通过测试计算评分数据 100000 和 1000209 的数据, 单机和 Hadoop 平台推荐不同用户数所花费的时间加速比, 推荐用户数分别为 1、2、4、8、16. 设置 Map/Reduce 工作数为 2、3、4, 计算评分数 10 万条加速比平均值分别为 1.92、2.67、3.59, 计算评分数 100

万条加速比平均值分别为 1.95、2.86、3.87, 具体如图 7 所示. 从图中看出加速比与 Map/Reduce 工作数并非成线性关系, 这是由于本文测试试验数据集大小的限制, 未能充分利用 Hadoop 平台资源, 并且在初始化 Map/Reduce 阶段消耗一定的时间.

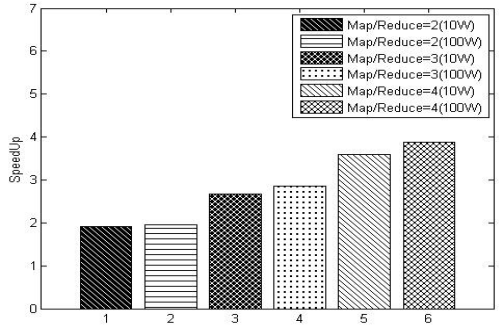

图 7 加速比

总结对比各推荐算法, 优化 SVD 和优化 Slope-One 推荐效果优于其他几种推荐算法, SVD 通过降维的方法改善数据稀疏性, 用户之间相似性更高, 预测评分值与真实评分值平均误差更小. 不同推荐算法各有优缺点, 对于不同应用场景选取最佳推荐算法进行个性化推荐.

5 结语

通过对协同过滤推荐算法的深入研究, 分别对基于用户和基于项目协同过滤推荐算法结合不同相似性计算方法实验分析, 利用隐式反馈的 Boolean 类型相似性方法的推荐结果优于基于具体评分值相似性方法. 通过对比传统协同过滤推荐算法、Slope-One、KNN、SVD 等推荐算法的推荐效果, Slope-One 算法同时加入数量和标准差的权重优化, SVD 算法引入特征空间维度优化推荐结果. 最后实现基于 Hadoop 分布式协同过滤推荐算法. 由于各推荐算法都存在优缺点, 如何将不同算法的优点强化缺点弱化是优化推荐效果的重点, 将机器学习方法与推荐系统相结合进行混合推荐的工作目前取得一定的进展, 还需要深入研究混合推荐算法.

参考文献

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

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

3 孙丽梅,李晶皎,孙焕良.基于动态 k 近邻的 SlopeOne 协同过滤推荐算法.计算机科学与探索,2011,5(9):857–864.

4 Zhao ZD, Shang MS. User-based collaborative–filtering recommendation algorithms on Hadoop. 3rd International Conference on Knowledge Discovery and Data Mining. IEEE. 2010. 478–481.

5 Jiang J, Lu J, Zhang G, et al. Scaling-up item-based collaborative filtering recommendation algorithm based on Hadoop. Proc. of the 2011 IEEE World Congress on Services (SERVICES '11). IEEE. 2011. 490–497.

6 Schelter S, Boden C, Markl V. Scalable similarity–based neighborhood methods with mapreduce. Proc. of the sixth ACM Conference on Recommender Systems. ACM. 2012. 163–170.

7 Chu C, Kim SK, Lin YA, et al. Map-reduce for machine learning on multicore. Advances in Neural Information Processing Systems, 2007, 19: 281.

8 周军锋,汤显,郭景峰.一种优化的协同过滤推荐算法.计算机研究与发展,2004,41(10):1842–1847.

9 Sarwar B, Karypis G, Konstan J, et al. Application of dimensionality reduction in recommender system--a case study. Minnesota Univ Minneapolis Dept of Computer Science, 2000.

10 冷亚军,梁昌勇,陆青,等.基于近邻评分填补的协同过滤推荐算法.计算机工程,2012,38(21):56–58,66.

11 Dooms S, De Pessemier T, Martens L. Offline optimization for user–specific hybrid recommender systems. Multimedia Tools and Applications, 2013: 1–24.

12 Sill J, Takács G, Mackey L, et al. Feature–weighted linear stacking. arXiv preprint arXiv: 0911.0460, 2009.

13 Bell RM, Koren Y. Improved neighborhood–based collaborative filtering. KDD Cup and Workshop at the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2007.

14 Koren Y. Factor in the neighbors: Scalable and accurate collaborative filtering. ACM Trans. on Knowledge Discovery from Data (TKDD), 2010, 4(1): 1.

15 Bell RM, Koren Y. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. 2007. Seventh IEEE International Conference on. IEEE Data Mining(ICDM 2007). 2007. 43–52.