基于用户部分特征的协同过滤算法①

李永超, 罗 军

(国防科技大学 计算机学院, 长沙 410073)

摘 要: 协同过滤算法作为推荐系统中应用最广泛的算法之一, 在大数据环境下面临严重的数据稀疏问题, 使得近邻选择的效果不佳, 直接影响了算法的推荐性能. 为了解决这一问题, 本文提出了一种基于用户部分特征的协同过滤算法(UPCF), 该算法首先基于评分偏差和项目流行度进行矩阵缺失值填充, 随后利用初始聚类中心优化的 K-means 算法对该填充矩阵进行项目聚类, 并利用用户在项目分类下的局部特征进行近邻集合构建, 最终采用基于用户的协同过滤算法获得推荐. 我们采用流行的 MAE 指标对算法在 MovieLens 数据集上进行评测. 实验表明, 与目前流行的协同过滤算法相比, 提出的 UPCF 算法在没有增加算法复杂性的前提下, 性能有近 10%的提升.

关键词: 项目流行度; 最近邻选择; 项目聚类; 协同过滤算法

Collaborative Filtering Algorithm Based on User Partial Feature

LI Yong-Chao, LUO Jun

(Department of Computer Science, National University of Defense Technology, Changsha 410073, China)

Abstract: As one of the most widely used algorithms in recommender system, the traditional collaborative filtering algorithm faces serious data sparseness problem in the big data trend, which leads to the ineffective in nearest neighbor selection, and restricts the performance of the algorithm. To address this problem, this paper proposes a collaborative filtering algorithm based on user partial feature(UPCF). In our method, it first rates the missing values based on rating bias and item popularity; and then clusters the items in the filled matrix with a K-means clustering algorithm of meliorated initial center. At last, it uses the user-based collaborative filtering algorithm with the user feature in item class to get the recommendations. The MAE measures on the MovieLens dataset shows that compared with the current popular algorithms, the performance of our UPCF algorithm improves about 10% without any increase of algorithm complexity.

Key words: item popularity; nearest neighbor selection; item clustering; collaborative filtering algorithm

1 引言

随着互联网的兴起, 互联网上的信息呈指数化增长, 人类进入了信息爆炸的大数据时代. 如何从浩瀚的数据信息中获取自己感兴趣的信息, 已成为人类面临的巨大难题. 于是无需用户提供明确需求, 仅通过用户历史行为主动帮助用户快速有效筛选信息的推荐系统应运而生[1]. 我们在互联网网站中看到的“猜你喜欢”, “大家都在看”, “看过的也看”, “你可能会感兴趣”等都是推荐技术的实际应用. 据报道, 早在 2002 年, 在线购物企业 Amazon 总销售额的 20%便源自它的推荐系统. 推荐技术在新闻领域更是产生了“今日头条”这样的不生产内容, 仅依靠推荐引擎便拥有 3.5 亿注册用户, 3500 万活跃用户的新兴科技媒体.

协同过滤作为推荐系统的主流技术之一, 主要包括基于用户的协同过滤推荐、基于项目的协同过滤推荐和基于矩阵分解的协同过滤推荐[2]. 而其中基于用户的协同过滤算法是目前在实际应用中最为成功的算法. 该算法首先通过用户间的共同评分项计算用户间的相似度, 然后根据用户间的相似度选择目标用户的近邻集合, 最后根据用户近邻集合对目标用户进行推

① 收稿时间:2016-07-01;收到修改稿时间:2016-08-31 [doi:10.15888/j.cnki.csa.005704]

荐. 最近邻选择作为该算法中最关键的步骤, 直接决定了推荐的质量. 然而在实际应用中由于数据集中项目的维度巨大, 大多数用户只会对极少数的项目进行评价, 从而导致用户评分数据的极端稀疏,不同用户间的共同评分项极少,用户间相似性计算的可靠性和准确性难以得到保证, 推荐算法的效果大打折扣.

为了解决稀疏性问题, 多种措施相继被提出. Ungar LH 等[3]首次提出基于用户聚类的协同过滤算法(UBCF), 通过用户聚类来降低最近邻搜索的数据规模, 增加最近邻可靠性. 黄裕洋等[4]根据评分数据的稀疏性情况, 提出了一种动态计算相似性的方法(HCFR). Xavier Amatriain[5]等提出在提前构建的专家集合中寻找用户近邻集合, 以确保用户的近邻对待预测项目有过评分记录. 黄创光等[6]提出了一种不确定近邻的协同过滤推荐算法(UNCF). 该算法通过不确定近邻因子及调和参数去计算基于用户和产品的预测评分并产生推荐. Koren Y[7]通过将矩阵分解和最近邻算法相结合, 大大提高了算法的推荐性能.

以上方法虽然从一定程度上减弱了数据稀疏对近邻选择带来的影响, 提高了协同过滤的推荐质量和效率, 但在最近邻计算的过程中, 对用户的相似性计算仍基于全局相似性, 没有充分考虑用户在不同项目类别下的兴趣差异. 正如世上没有完全相同的两片树叶一样, 在各个方面兴趣都相似的用户也难以寻找. 大多用户可能只在某个领域内兴趣相仿, 在其他领域内可能兴趣完全相悖. 因此本文提出了一种基于用户部分特征的协同过滤算法 UPCF, 该算法首先对填充矩阵进行项目聚类, 然后仅根据用户在该项目分类下的所有评价进行相似度矩阵构建, 降低数据维度的同时提升了最近邻计算的可靠性, 最后根据相似性矩阵进行近邻集合构建, 从而最终得到推荐结果.

2 问题定义及基本方法

基于用户的协同过滤算法基于以下假设: 如果用户之间对一些项目的评分比较相似, 则他们对其它项目的评分也将会比较相似. 协同过滤推荐系统首先搜索目标用户的若干近邻, 然后根据最近邻对项目的评分去预测目标用户对项目的评分, 从而产生推荐列表. 作为算法的输入, 数据源 $D=(U,I,R)$, 其中 $U=\{u_1,u_2,\cdots,u_m\}$是基本用户的集合, $|U|=m$; $I=\{i_1,i_2,\cdots,i_n\}$是项目集合, $|I|=n$. $m*n$ 阶矩阵 R 是用户对各项目的评分矩阵, 其中的元素 r_{ij} 表示 U 中第 i 个用户对 I 中第 j 个项目的评分. 基于用户的协同过滤算法主要包括以下三个步骤.

2.1 评分矩阵预处理

由于在实际应用中, 项目集 I 的维数 n 很大, 用户只能对极少数项目进行评价, 因此评分矩阵十分稀疏, 这对后面的相似性计算提出了很大的挑战. 合理的矩阵缺失值预测填充可以从一定程度上缓解稀疏性问题.

目前常用的缺失值预测方法包括评分中值、众数、用户评分均值、项目评分均值、采用奇异值分解填补近邻评分缺失值[7]以及基于近似项目预测评分值[8]等.

2.2 用户近邻集合构建

接下来, 我们在预处理过的用户评分矩阵上采用相似度计算方法, 计算用户之间的相似度, 形成用户的相似度矩阵. 协同过滤算法研究中最常用的相似度计算方法是相关相似度、余弦相似度和修正的余弦相似度, 它们的计算公式分别如下:

相关相似度:

$$\mathrm{sim}(u,v)=\frac{\sum_{i\in I_{uv}}(r_{ui}-\bar{R}_u)*(r_{vi}-\bar{R}_v)}{\sqrt{\sum_{i\in I_u}(r_{ui}-\bar{R}_u)^2}\sqrt{\sum_{i\in I_v}(r_{vi}-\bar{R}_v)^2}} \tag{1}$$

余弦相似度:

$$\mathrm{sim}(u,v)=\cos(u,v)=\frac{u*v}{\|u\|*\|v\|}=\frac{\sum_{i\in I_{uv}}r_{ui}*r_{vi}}{\sqrt{\sum_{i\in I_u}r_{ui}^2}\sqrt{\sum_{i\in I_v}r_{vi}^2}} \tag{2}$$

修正的余弦相似度:

$$\mathrm{sim}(u,v)=\frac{\sum_{i\in I_{uv}}(r_{ui}-\bar{R}_u)*(r_{vi}-\bar{R}_v)}{\sqrt{\sum_{i\in I_{uv}}(r_{ui}-\bar{R}_u)^2}\sqrt{\sum_{i\in I_{uv}}(r_{vi}-\bar{R}_v)^2}} \tag{3}$$

各公式中 u,v 表示用户 u、v. r_{ui} 表示用户 u 对项目 i 的评分, $\bar{R}_u$ 表示用户 u 的平均评分, I_u 、I_v 表示用户 u、v 已经评价过的项目集合, I_{uv} 表示用户 u 和用户 v 的共同评分项目集合.

相似矩阵构建结束后, 便可根据用户指定的最近邻筛选规则构建近邻集合, 常用的筛选规则包括指定近邻数量和设置相似度阈值.

2.3 物品推荐

利用上一步计算得到的近邻集合, 找到这个集合中的用户喜欢且目标用户没有听说过的物品推荐给用户. 具体而言, 我们利用公式(4)计算用户对指定项目

的预测评分.

$$\hat{r}_{ui}=\bar{R}_u+\frac{\sum_{v\in N_u}\mathrm{sim}(u,v)\cdot(r_{vi}-\bar{R}_v)}{\sum_{v\in N_u}|\mathrm{sim}(u,v)|} \quad (4)$$

其中 N_u 为用户 u 的最近邻集合, sim(u,v)为用户 u、v 的相似度, 其余符号与前面定义一致. 最终便得到了用户 u 关于项目 i 的预测评分 $\hat{r}_{ui}$.

3 基于用户部分特征的协同过滤算法(UPCF)

传统的基于用户的协同过滤算法在计算最近邻的过程中使用了用户的所有评分记录, 考察了用户的全局相似性. 然而在全部项目集上兴趣都相似的用户并不常见, 大多用户可能只在某一主题下兴趣相似, 而在其余项目分类中喜好完全不同. 因此传统的近邻集合构建往往选择了全局相对相似而舍弃了在某些领域内兴趣高度契合的用户. 为了解决这个问题, 本文提出了一种基于用户部分特征的协同过滤算法, 使得在最近邻选择时所需的相似度仅根据用户在该项目所在类内的评价信息计算获得. 算法详细流程如下所示.

3.1 未评分项目预测填充

为了缓解在项目聚类时矩阵的稀疏问题, 我们首先对评分矩阵进行缺失值预测填充. 考虑到热门项目对用户特征贡献度不大, 以及相对冷门项目而言, 用户接触到热门项目的概率大得多, 而如果用户未对热门项目进行反馈评价, 很可能是因为用户对该项目并不感兴趣. 而在推荐系统中, 项目流行度是衡量项目热门程度的主要指标, 它是指项目被用户反馈的总次数, 被反馈的次数越多代表项目流行度越高. 因此为了能够从一定程度上合理惩罚未评分热门项目, 我们引入了项目流行度权重系数 w, 在此次试验中, 我们采用以下公式计算项目流行度, 其中 $S(i)$表示项目 i 已被评分的总次数.

$$w=1/(\log(1+S(i))) \quad (5)$$

项目被评分总次数 $S(i)$越大则对应权重 w 越小, 预测评分则会相应降低. 最终我们采用如下方法进行缺失值预测填充.

$$\tilde{r}_{ui}=\bar{R}+b_u+b_i*w \quad (6)$$

其中 $\tilde{r}_{ui}$ 表示对 r_{ui} 的预测填充值, $\bar{R}$ 表示数据集中所有用户的评分均值, b_u表示用户 u 与用户评分均值的评分偏差, b_i表示项目 i 与用户评分均值的评分偏差, w 表示项目流行度权重系数.

3.2 项目聚类

缺失数据处理过后, 我们便可对项目进行聚类, 本次我们采用聚类算法中最经典的 K-means 算法进行项目聚类. 而传统的 K-means 算法对初始聚类中心非常敏感, 聚类结果随不同的初始输入而有较大波动. 为消除这种敏感性, 本文采用袁方等提出的优化初始聚类中心的改进 K-means 算法[9]进行聚类计算. 与传统聚类算法不同的是, 该算法在选取初始聚类中心时计算每个数据对象所在区域的密度, 选择相互距离最远的 k 个处于高密度区域的点作为初始聚类中心. 实验表明改进后的 K-means 算法能产生质量较高的聚类结果, 并且消除了对初始输入的敏感性.

过程 1. 基于项目的 kmeans 聚类

输入: 聚类数目 k, 最大迭代次数 $iter\_num$ 和用户评分数据填充矩阵 R

输出: k 个聚类

1) 计算以项目集I中每个项目i_j为中心, 包含常数 $Minpts$ 个数据对象的半径, 记为 i_j 的密度参数 ε . ε 越大, 说明数据对象所处区域的数据密度越低. 反之则说明数据对象所处区域的数据密度越高. 选取满足 $\varepsilon_1<\varepsilon_{\max}$ 的点 i_j 为高密度区域 D. 取 D 中处于最高密度区域的点作为第 1 个聚类中心 r_1; 取 D 中距离 r_1 最远的点作第 2 个聚类中心 r_2; 计算 D 中各数据对象 i_j 到 r_1, r_2 的距离 $d(i_j, r_1)$, $d(i_j, r_2)$, r_3 为满足 $max(min(d(i_j, r_1), d(i_j, r_2)),j=1,2\cdots n$ 的数据对象 i_j; r_m 为满足 $max(min(d(i_j, r_1), d(i_j, r_2)...d(i_j, r_{m-1})),j=1,2\cdots n$ 的数据对象 $i_j,i_j\in D$. 依此得到 k 个初始聚类中心. 记为集合 $center_{old}=\{r_1, \cdots, r_k\}$;

2) k 个聚类簇 $cluster_1$, $\cdots cluster_k$ 均初始化为空, 记为集合 $Cluster=(cluster_1, \cdots cluster_k)$

3) REPEAT

FOR each item i in I:

FOR each center $r_j$in $center_{old}$:

计算项目 i 和聚类中心 r_j 的相似性 sim(i,r_j);

sim(i,r_m)=max(sim(i,r_1),sim(i,r_2),$\cdots$,sim(i,r_k))

$cluster_m=cluster_m\cup$ i

Endfor

For each $cluster_m$ in $Cluster$:

计算 $cluster_m$ 的均值, 生成新的聚类中心 c_{newm}.

$Center_{new}=\{c_{new1},c_{new2},\cdots, c_{newk}\}$

Endfor

UTIL$Center_{old}$=(c_1, …c_k)和 $Center_{new}$=(c_1, …c_k)相同或达到最大迭代次数 *iter_num*.

4) 返回 *Cluster*.

算法的复杂性为 O($iter\_num * k * m * n$). 最终我们便得到了聚类后的项目.

3.3 推荐生成

为了保证预测的精确性, 避免提前引入误差, 我们在评分预测阶段采用原始用户评价矩阵而非填充矩阵. 并使用公式(1)计算待推荐用户在待推荐项目所在类内与其余用户的相似度, 构建用户相似性矩阵. 查找与用户相似度最大的 n 个最近邻. 使用公式(4)计算用户预测评分, 得到最终评分预测值, 算法过程如下.

过程 2. 评分预测

输入: 原始用户评价矩阵 R, 最近邻个数 n, 待预测评分用户 u, 项目 i, 项目 i 所在聚类簇 $cluster_j$=[i_l,i_j,…,i_p].

输出: 评分预测值 $\hat{r}_{ui}$

1) *simDict*={}

2) For user v in U:

IF v!=u:

simDict[v]=sim(u,v)

Endif

Endfor

3) N_u=sort(*simDict*)[:n]

4) $$\hat{r}_{ui}=\bar{R}_u+\frac{\sum_{v\in Nu}\mathrm{sim}(u,v)\cdot(r_{vi}-\bar{R}_v)}{\sum_{v\in Nu}|\mathrm{sim}(u,v)|}$$

其中, 此处 u,v 的特征向量为 u=(r_{uil} , r_{uij} ,…, r_{uip}), v= (r_{vil} , r_{vij} , … , r_{vip}). sim(u,v)我们采用公式(1)所提供的相关相似性计算. 算法的复杂度为 O($m * n / k$). 至此, 我们便可获得指定用户对指定项目的评分预测值, 为随后的推荐提供支持.

4 实验结果及分析

本次实验的硬件平台是配置 Intel pentium E58003.2 GHz CPU, 4G RAM, 操作系统为 ubuntu 14.04 的个人计算机, 所有程序均由 python 实现.

4.1 数据集

本文采用的实验数据集是目前衡量推荐算法质量常用的著名电影评分数据集 MovieLens 中的 100k 数据集(http://grouplens.org/datasets/movielens), 该数据集由美国明尼苏达大学 GroupLens 研究小组创建并维护. 该实验数据集共包含 930 个用户对 1682 部电影的 100000 条评价信息, 其中每个用户至少对 20 部电影进行了评分, 每个电影也都收到了用户评论. 该数据集的稀疏性为 1-100000/(943*1682) = 0.937. 数据集中用户评分范围是 1-5, 数值越大代表用户对该电影的兴趣越大. 本次实验按照 80%和 20%的比例随机的将数据集划分成为训练集和测试集, 随后进行 5-折交叉实验, 取五次试验的平均值作为最终结果.

4.2 评价标准

平均绝对误差 MAE(mean absolute error)是目前学术研究中应用广泛的推荐系统推荐质量评价标准. 其主要通过公式(7)计算测试集中用户实际评分和推荐算法根据训练集的训练预测值的差的绝对值均值, 平均绝对误差 MAE 越小, 推荐算法的质量越高. 其中 N 表示测试集的数据个数, p_i 为预测评分值, r_i 为测试集中的实际评分值.

$$MAE=\frac{\sum_{i=1}^{N}|p_i-r_i|}{N} \tag{7}$$

4.3 实验结果分析

我们首先研究聚类个数对本文算法性能的影响. 实验结果如图 1 所示, 其中横坐标表示聚类个数, 纵坐标表示 MAE 值, 最近邻个数统一取 n=30.

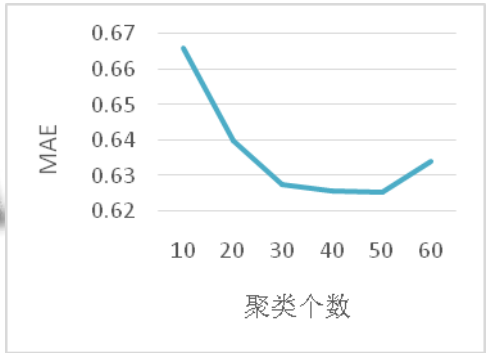

图 1 聚类个数对 MAE 值的影响

通过图 1 我们可以清晰看到刚开始, 随着聚类族数的增多, 算法性能不断提升, 当项目聚类个数为 50 时, 算法取得了最好的性能, 此后聚类族数的增多反而引起算法性能的下降. 接下来, 我们通过将本文所提出算法 UPCF 与传统的基于用户聚类的协同过滤算法(UBCF)[3]、综合用户和项目因素的协同过滤推荐算法 (HCFR)[4] 、基与不确定近邻的协同过滤算法 (UNCF)[6]的平均绝对误差 MAE 进行对比观测试验性

能. 为了缩短算法的运行时间, 聚类个数均设置为 20.

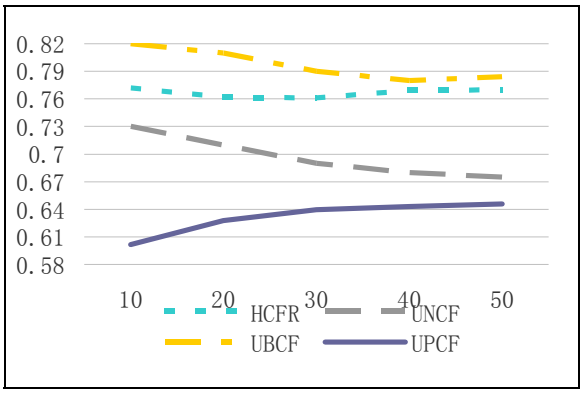

图 2 算法性能比较

由图 2 可以看出, 本文算法与其他算法在 MAE 值上有了 10%左右的提高, 特别是当近邻个数比较少的时候, 本文算法体现了非常好的推荐效果, 性能优势明显更好, 这充分说明本文提出算法最近邻选择的高效合理性. 我们也可以观察到, 当最近邻个数达到一定数量后, 所有算法的 MAE 性能趋于平稳, 这也反映出当近邻相似度不断减小时, 该近邻对算法的性能提升没有显著的影响.

此外, 本文算法在提高推荐质量的同时并没有带来算法复杂性的提升. 数据的预填充和项目聚类均可提前离线完成, 仅需根据需求隔一段时间更新. 而评分预测由于不再依赖用户全局特征, 单个评分预测的复杂性由 $o(m*n)$ 变为 $o(m*n/k)$, 其中 m 表示用户总个数, n 表示项目总个数, k 表示项目聚类个数. 用户还可根据实际需求, 并行的在各个项目类内进行用户评分预测.

5 总结

针对传统协同过滤算法中最近邻计算时所面临的稀疏性和准确性挑战, 本文提出了一种基于用户部分特征的协同过滤算法. 该算法采用基于评分偏差与项目流行度的思想进行缺失值填充, 并在最近邻构建时仅考虑用户在该项目分类下的特征, 动态的根据待预测项目筛选用户最近邻, 从而提高了推荐的质量. 此外, 由于仅需考虑用户类内特征, 该算法实现了一定程度的降维, 降低了算法的复杂性. 并可根据需要, 分布并发的计算各个类内用户的预测评分值, 一定程度上提高了算法的实时性.

但算法中项目分类以及用户最近邻选择时特征选择的准确性仍需要进一步研究改良, 所使用的聚类算法 K-means 的聚类效果仍不是十分理想. 此外, 在此次研究前期对推荐算法的了解中, 我们发现目前针对各种推荐算法模型的融合以及算法并行化的研究也成为业界的新热点. 而能够更好地反应用户兴趣的用户社交关系的引入[10,11]大大提高了协同过滤算法的近邻可靠性和准确性, 为算法的改良提供了新的方向. 如何将用户社交关系引入本文提出的算法, 进一步改善本文算法的性能, 将是下一阶段研究的重点.

参考文献

1 Park DH, Kim HK, Choi IY, Kim JK. A literature review and classification of recommender systems research. Expert Systems with Applications, 2012, 39(11): 10059–10072.

2 Bobadilla J, Ortega F, Hernando A, Gutiérrez A. Recommender system survey. Knowledge-Based Systems. 2013, 46: 109–132.

3 Ungar LH, Foster DP. Clustering methods for collaborative filtering. AAAI Workshop on Recommendation Systems, 1998, 1: 114–129.

4 黄裕洋,金远平.一种综合用户和项目因素的协同过滤推荐算法.东南大学学报(自然科学版),2010,40(5):917–921.

5 Amatriain X, Lathia N, Pujol JM, Kwak H, Oliver N. The wisdom of the few: A collaborative filtering approach based on expert opinions from the web. Proc. of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM. 2009. 532–539.

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

7 Koren Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2008. 426–434.

8 邓爱林,朱扬勇,施伯乐.基于项目评分预测的协同过滤推荐算法.软件学报,2003,14(9):1621–1628.

9 袁方,周志勇,宋鑫.初始聚类中心优化的 K-means 算法.计算机工程,2007,33(3):65–66.

10 Daly EM, Geyer W. Effective event discovery: Using cation and social information for scoping event recommendations. Proc. of the 5th ACM Conference on Recommender Systems. ACM. 2011. 277–280.

11 Guy I. Social recommender systems. Recommender Systems Handbook. Springer US, 2015: 511–543.