doi: 10.12068/j.issn.1005-3026.2018.08.008

基于二分网络社团划分的推荐算法

陈东明, 严燕斌, 黄新宇, 王冬琦
(东北大学 软件学院, 辽宁 沈阳 110169)

摘 要: 传统的基于用户的协同过滤(User-based CF)推荐算法的推荐效率随着数据的不断增加而降低. 本文在User-based CF算法中引入二分网络社团发现理论,提出一种基于二分网络社团划分的推荐算法(RACD). 首先通过用户与项目之间的关系建立用户-项目二分网络,然后通过RACD对该网络进行社团划分,得到用户的社团信息,最后通过同一社团中的其他用户对目标用户进行项目的推荐. 在经典网络数据集上的实验结果表明,RACD能够有效提高推荐系统实时推荐效率.

关 键 词: 推荐算法;二分网络;社团划分;协同过滤;复杂网络
中图分类号: TP 391 **文献标志码**: A **文章编号**: 1005-3026(2018)08-1103-05

Recommendation Algorithm Based on Community Detection in Bipartite Networks

CHEN Dong-ming, *YAN Yan-bin*, *HUANG Xin-yu*, *WANG Dong-qi*
(School of Software, Northeastern University, Shenyang 110169, China. Corresponding author: HUANG Xin-yu, E-mail: neuhxy@163.com)

Abstract: The efficiency of traditional user-based collaborative filtering (user-based CF) recommendation algorithm is reduced with data increasing. This paper proposes a recommendation algorithm based on community detection (RACD) in bipartite networks by introducing bipartite network community detection theory into user-based CF recommendation algorithm. Firstly, the user-item rating matrix is mapped into user-item bipartite network. Then, the community information of each user is obtained by using RACD to divide the user-item network. Finally, the items are recommended to the target user according to other users in the same community. Experiments on real-world classic network datasets show that the RACD can effectively improve real-time recommendation efficiency of the recommendation system.

Key words: recommendation algorithm; bipartite network; community detection; collaborative filtering; complex network

随着互联网技术的发展,网络中的数据呈指数型增长,虽然以Google、百度为代表的信息检索系统能够帮助用户获取很多的信息,但不同用户检索出的结果是相同的. 为了满足不同用户的差异化需求,个性化推荐系统得到推广,它的出现使得用户不再迷茫于浏览各种各样的网页,而是在系统的协助下快速发现感兴趣的信息[1]. 目前,国内外很多大型网站都使用了个性化推荐技术,例如:Amazon,GroupLens,淘宝,百度等.

对推荐系统的推荐质量起决定作用的是推荐算法,例如:Schedl等[2]通过分析用户的年龄、性别、国籍等属性,以及用户听音乐的频率、时间等行为提出了一种音乐推荐算法;Zheng等[3]通过分析学生的历史借阅信息和图书之间的相似性等信息提出了一种图书推荐算法. 迄今为止,推荐系统使用的推荐算法主要有协同过滤(collaborative filtering, CF)算法[4-5]、基于内容(content-based)的推荐算法[6-7]、混合推荐算法[8-10]等. User-based CF算法[11]是CF算法中的一种,传统的User-based CF算法将所有用户对项目的评分数

收稿日期: 2017-03-30
基金项目: 辽宁省自然科学基金资助项目(20170540320); 辽宁省教育厅科学研究项目(L20150167).
作者简介: 陈东明(1968-),男,安徽怀宁人,东北大学教授.

据转换为评分矩阵,然后通过该矩阵获取用户之间的相似性,进而根据相似性高的用户完成项目的推荐. 但是,实际数据的稀疏性影响了推荐精度和效率. 因此,学者们提出了不同的优化算法[12]. 近些年,有些学者提出了基于社团划分的协同推荐算法[13],但是这些算法中的社团划分算法需要事先知道社团的数量或只适合小规模的网络,其适用性有局限性. 实际上,推荐系统中存在的网络数据大多数可以抽象成二分网络,如"用户 - 商品"网络,但针对这样的二分网络先进行社团划分,然后再推荐的算法并不多. 文献[14]提出了一种基于结构相似性的二分网络社团划分算法(BSSCD 算法),该算法将相似性高的节点划分到同一个社团中,而 User-based CF 算法的核心就是寻找相似用户集合;因此,本文将 BSSCD 算法与 User-based CF 算法进行结合,提出一种基于二分网络社团划分的推荐算法(称为 RACD),该算法能够有效提高推荐系统实时推荐的效率.

1 模型和方法

大多数推荐系统中的数据都可以抽象成二分网络,"用户 - 商品"网络、"用户 - 电影"网络等都属于二分网络. 这些网络的用户之间往往会有潜在的联系,如行为相似. 社团划分算法能够发现网络中可能存在的用户群体,推荐算法可以基于这些群体进行产品的推荐. 社团划分算法与推荐算法之间存在着必然的联系.

社团划分算法可以在大规模网络中挖掘出节点所属的社团信息,同一社团中节点之间的联系比较密切. 例如在依据兴趣爱好构建的社交网络中进行社团划分,那些兴趣爱好相似的人往往被划分到同一个社团. 另一方面,User-based CF 算法中最核心的部分就是寻找相似用户集合,从社团的角度来说,就是寻找与目标用户属于同一社团的其他用户,然后针对同一社团中的其他用户进行产品推荐. 因此,在推荐算法中根据已划分的社团结构寻找目标用户的相似用户集合,返回一个数量级更小并且更加精确的与目标用户相似的用户集合. 返回的相似性用户集合形成内部连接紧密的社团结构,根据该相似性用户集进行推荐,能够改善推荐精度和效率.

二分网络社团划分(BSSCD)算法根据二分网络的结构特征计算二分网络中每个节点的相似性,然后通过聚类的方式将网络中相似性高的节点划分到同一个社团中. BSSCD 算法的步骤如下:

```
Begin
    Initialization: Obtain the structure of the network, and get the initial node v = getFirstNode();
step1: /* 获得每个节点的最相似节点 */
    getMaxSimilartityNodeSet(){
      while(one of the neighbor nodes of v has not been visited){
          set_tag.add(node);
          s = getMaxSimilarity(node);
          set.add(node);
          set.add(s);
          list.add(set);
          node = v.getNextNeigborNode();}}
step2: /* 从邻居节点扩展节点集合 */
    for (Node node: set_tag){
      if(node.hasAllVisited()){
          nodelist1 = combine(set_tag,list);
    }else{
          go to step1;}}
    In the same way, get nodelist2;
step3: /* 合并社团 */
    mergeCommunity(nodelist1,nodelist2){
    for(List list: nodelist2){
                community = getMaxSimilarityCommunity(list,nodelist1);
          merge(list,community);}}
End
```

2 RACD 的思想与实现

2.1 算法思想

User-based CF 算法在电子商务网站中得到广泛应用,但仍存在一些问题,数据稀疏性就是其中之一. 例如:假设某个电影推荐网站中包含几百万部电影,而平均每个用户评价过的电影只有几百部,显然,得到的"用户 - 电影"矩阵十分稀疏. 基于这样的稀疏矩阵给用户推荐电影,推荐的质量并不高. 如果用户和电影的数量不断增加,数据的稀疏程度也会随之增加,推荐算法的效率就会更低. User-based CF 算法的推荐过程见图 1.

User-based CF 算法中的一个关键步骤是获取每个用户的相似用户集合. 图 2 是由用户和项目(比如商品,电影等)所构成的一个简单的二分网络,网络中的节点有用户和项目,连边表示用户

对项目的“动作”,比如购买行为、观看行为等.

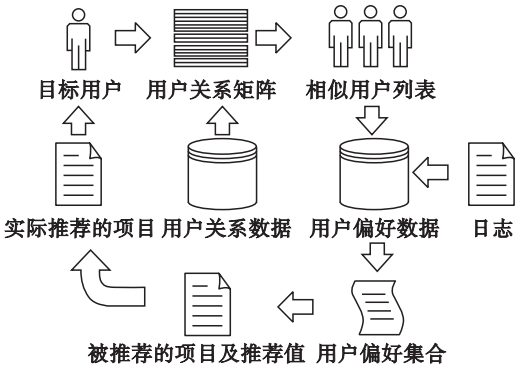

图 1　基于用户的协同过滤的推荐过程

Fig. 1　Recommendation process of user-based collaborative filtering

图 2 中,当获取用户 5 的相似用户集合时,User-based CF 算法首先通过用户 5 与其邻居用户的相似性来获取其邻居用户集合,用户 5 的邻居用户包括用户 4、用户 6、用户 1 和用户 2. 假设选取用户 4、用户 6、用户 1 和用户 2 作为用户的相似用户集合,那么最终系统可能会根据用户 4 推荐项目 4 给用户 5;根据用户 6 推荐项目 7 给用户 5;根据用户 1 和用户 2 推荐项目 2 和项目 3 给用户 5. 当计算项目的推荐度时,项目 4 和项目 7 的推荐度要远高于项目 2 和项目 3,因为通过相似性计算,用户 5 与用户 4 和用户 6 的相似度高于与用户 1 和用户 2 的相似度. 在实际中,推荐系统也会优先推荐用户感兴趣程度更高的产品.

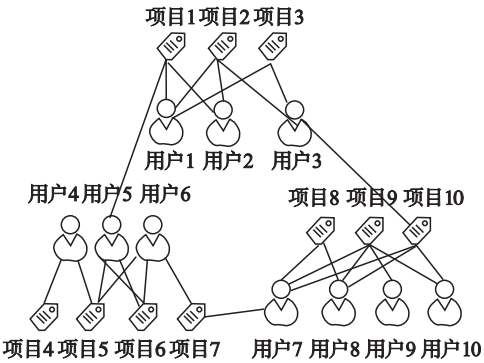

图 2　用户和项目构成的网络

Fig. 2　Network composed of users and items

社团划分的目的是把那些联系紧密的节点划分到同一个社团中,在社交网络中可以理解成把用户偏好类似的一些用户划分到同一个社团中. 所以,在获取目标用户的相似用户集合之前,可以通过社团划分算法提前获得目标用户的社团信息,同一社团中的用户与目标用户之间的相似度相对较高,这样能够过滤掉一些与目标用户不在同一社团(即相似度很低)的用户. 使用 BSSCD 算法对图 2 所示的网络进行社团划分,会得到如下划分结果:

社团 1:用户 1,用户 2,用户 3,项目 1,项目 2,项目 3;社团 2:用户 4,用户 5,用户 6,项目 4,项目 5,项目 6,项目 7;社团 3:用户 7,用户 8,用户 9,用户 10,项目 8,项目 9,项目 10.

在用户 5 的邻居用户中,用户 4 和用户 6 与用户 5 在同一个社团中,而用户 1 和用户 2 却在另一个社团中,这也证明了用户 5 与用户 4 和用户 6 的相似度高于用户 1 和用户 2. 所以,在推荐算法中,事先通过社团划分算法对数据集进行过滤处理,然后根据社团划分的结果去获取目标用户 i 的相似用户集,不但可以过滤掉那些虽是目标用户 i 的邻居但相似度却不高(不在同一个社团中)的用户,得到一个更精确的相似用户集,而且能够减少一些不必要的计算开销,从而提高推荐系统实时推荐的效率.

2.2　RACD 描述

输入:用户对项目的评分数据,目标用户 i.

输出:给目标用户 i 进行项目推荐的列表.

1) 根据用户与项目之间的关系建立“用户 – 项目”二分网络;

2) 使用 BSSCD 算法对二分网络 B 进行社团划分,得到用户所在的社团;

3) 建立用户对项目的评分矩阵;

4) 得到目标用户的相似用户集合;

5) 计算与目标用户 i 相似度较高的前 k 个用户作为推荐用户集合;

6) 通过相似用户集合得到所有推荐项目以及项目的推荐度,按照项目推荐度的高低返回目标用户 i 的项目推荐列表.

RACD 的推荐模型如图 3 所示.

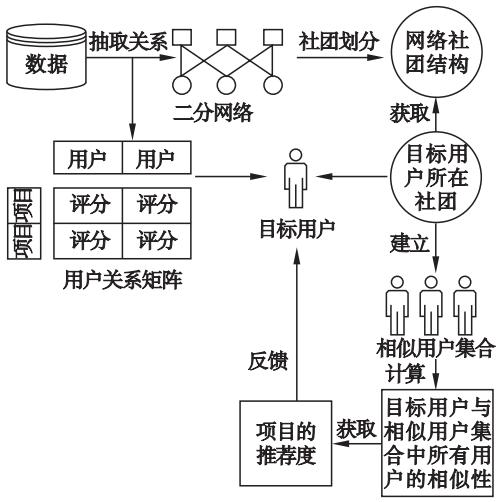

图 3　RACD 的推荐模型

Fig. 3　Recommendation model of RACD

2.3 RACD 实现

获取相似性用户集和建立推荐列表是算法的两个核心步骤.

2.3.1 获取推荐对象的相似用户集

首先通过 BSSCD 算法得到用户所在的社团,社团划分结果保存为 Communities 字典;然后在同一社团内部获得推荐对象的相似用户集合.

```
def getNeighbors(uId, user_dict, item_user, communities):
    n = []
    user_communityId = communities[uId]
    for item in user_dict[uId]:
        for neighbor in item_user[item[0]]:
        neighbor_communityId = communities[n]
            (if neighbor ! = uId and neighbor not
            in neighbors and user_communityId ==
            neighbor_communityId):
            #添加相似邻居节点
                    neighbors.append(neighbor)
    neighbors_dist = []
    for i in neighbors:
    #获取相似性
        dist = getSimilarity(user_dict[uId],
            user_dict[i]
        neighbors_dist.append([dist, neighbor])
    #按照相似性从高到低排序
    neighbors_dist.sort(reverse = True)
    return neighbors_dist  #返回相似用户
```

2.3.2 建立推荐列表

```
def recommendByUserFC(neighbors, k):
    #建立推荐字典
    recommand_dict = {}
    for neighbor in neighbors:
        neighbor_user_id = neighbor[1]
        items = user_dict[neighbor_user_id]
        for item in items:
            if item[0] not in recommand_dict:
                recommand_dict[item[0]] =
                neighbor[0]
            else:
                recommand_dict[item[0]] += =
                neighbor[0]
    #建立推荐列表
    recommand_list = []
    for key in recommand_dict:
        recommand_list.append([recommand_dict
        [key], key])
    recommand_list.sort(reverse = True)
return [k[1] for k in recommand_list]
```

3 实验及其结果与分析

实验平台使用 Intel(R) Core(TM) i7 - 4790 @3.60 GHz 四核处理器,8 GB DDR3 内存,Windows10 64 位操作系统,使用 Python 3.5.1 编程语言实现.

采用 GroupLens 提供的 MovieLens 数据集[15],该数据集包含 943 个用户和 1 682 部电影的 100 000 条评分记录.

本实验中使用的数据分为 3 个数量级,分别是 5 000 条、2 万条和 10 万条评分数据. 首先针对各个数据集进行处理,形成二分网络,然后使用 BSSCD 算法对该网络进行社团划分,得到每个用户的社团信息,算法运行时间为:5 000 条数据,21.48 s;2 万条数据,56.50 s;10 万条数据,231.13 s. 可见,随着数据量的增大,BSSCD 算法的处理时间成倍地增加. 但是,在 RACD 的推荐模型中,BSSCD 算法对 MovieLens 数据的处理过程一般是在线下完成的,社团数据保存在文件中,在系统推荐时,通过读取该文件获得用户的社团数据. 所以,在实时推荐时,系统消耗的时间是读取用户社团信息文件的时间,而不是社团划分消耗的时间.

给定用户 i,通过运行 RACD 给用户 i 推荐的前 10 部电影的结果如图 4 所示.

movie id	movie name	release
79	Fugitive, The(1993)	01 - 01 - 1993
222	Star Trek: First Contact(1996)	22 - 11 - 1996
95	Aladdin(1992)	01 - 01 - 1992
153	Fish Called Wanda, A(1988)	01 - 01 - 1988
173	princess Bride, The(1987)	01 - 01 - 1987
403	Vatman(1989)	01 - 01 - 1989
294	Liar Liar(1997)	21 - 03 - 1997
258	Contact(1997)	11 - 07 - 1997
118	Twister(1996)	10 - 05 - 1996
433	Heathers(1989)	01 - 01 - 1989

图 4 用户 i 的推荐结果

Fig. 4 Recommendation result for user i

由于 RACD 在 User-based CF 基础上考虑了社团信息,因此推荐结果相对而言更精确.

在不同数据量下, User-based CF 算法与在其

基础上作了改进的 RACD 的运行时间如表 1 所示. 运行时间是完成对数据集中所有用户进行推荐所花费的时间.

表 1　运行时间对比

Table 1　Comparison of running time　　s

数据量	5×10^3	5×10^4	1×10^5
RACD 算法	1.45	14.96	31.13
User-based CF 算法	3.71	57.85	145.35

与 User-based CF 算法相比, RACD 的运行时间大幅减少,从而验证了在推荐系统进行实时推荐时, RACD 的推荐效率比 User-based CF 算法的推荐效率更高.

4　结　　语

为了解决传统推荐算法存在的一些关键性的问题,提高推荐系统的性能,本文把二分网络社团发现算法与推荐算法相结合,提出了基于二分网络社团划分的推荐算法,对 User-based CF 算法进行了改进. 采用社团划分算法对数据进行预处理,得到一个数量级更小并且更加精确的相似用户集合,提高了数据的相关性,从而减少算法的计算开销. 与 User-based CF 算法相比,本文算法的实时推荐效率更高.

参考文献:

[1] Adomavicius G, Zhang J. Stability of recommendation algorithms [J]. *ACM Transactions on Information Systems*, 2010,30(4):47 – 54.

[2] Schedl M, Hauger D, Farrahi K, et al. On the influence of user characteristics on music recommendation algorithms [C]//European Conference on Information Retrieval. Vienna,2015:339 – 345.

[3] 郑祥云,陈志刚,黄瑞,等. 基于主题模型的个性化图书推荐算法[J]. 计算机应用,2015,35(9):2569 – 2573.
(Zheng Xiang-yun, Chen Zhi-gang, Huang Rui, et al. Personalized book recommendation algorithm based on topic model [J]. *Journal of Computer Applications*,2015,35(9): 2569 – 2573.)

[4] Cai X, Bain M, Krzywicki A, et al. Collaborative filtering for people to people recommendation in social networks[C]//AI 2010: Advances in Artificial Intelligence. Berlin: Springer, 2010:476 – 485.

[5] Chee S H S, Han J, Wang K. RecTree: an efficient collaborative filtering method [J]. *Lecture Notes in Computer Science*,2001,2114:141 – 151.

[6] Balabanovi M, Shoham Y. Fab: content-based, collaborative recommendation [J]. *Communications of the ACM*,1997,40 (3):66 – 72.

[7] Lops P, de Gemmis M, Semeraro G. Content-based recommender systems: state of the art and trends [M]// Recommender Systems Handbook. [S. l.]: Springer US, 2011:73 – 105.

[8] Soboroff I M, Nicholas C K. Combining content and collaboration in text filtering [C/OL]//Proceedings of the IJCAI' 99 Workshop on Machine Learning in Information Filtering,1999. [2017 – 01 – 23]. http://citeseerx. ist. psu. edu/viewdoc/download? doi = 10. 1. 1. 640. 3677 & rep = rep1 & type = pdf.

[9] Girardi R, Marinho L B. A domain model of Web recommender systems based on usage mining and collaborative filtering [J]. *Requirements Engineering*, 2007, 12(1):23 – 40.

[10] Papagelis M, Plexousakis D. Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents [J]. *Engineering Applications of Artificial Intelligence*,2005,18(7):781 – 789.

[11] Zhao Z D, Shang M. User-based collaborative-filtering recommendation algorithms on hadoop [C]// IEEE International Conference on Knowledge Discovery and Data Mining. Phuket,2010:478 – 481.

[12] 孟桓羽,刘真,王芳,等. 基于图和改进 K 近邻模型的高效协同过滤推荐算法[J]. 计算机研究与发展,2017,54(7): 1426 – 1438.
(Meng Huan-yu, Liu Zhen, Wang Fang, et al. An efficient collaborative filtering algorithm based on graph model and improved KNN [J]. *Journal of Computer Research and Development*,2017,54(7):1426 – 1438.)

[13] Wang Q, Li W, Zhang X, et al. Academic paper recommendation based on community detection in citation-collaboration networks [C]//Asia-Pacific Web Conference: Web Technologies and Applications. Suzhou, 2016: 124 – 136.

[14] Chen D M, Yan Y B, Wang D Q, et al. Community detection algorithm based on structural similarity for bipartite networks [C]// Proceedings of 7th IEEE International Conference on Software Engineering and Service Science. Beijing, 2016: 173 – 177.

[15] Vozalis M G, Margaritis K G. Using SVD and demographic data for the enhancement of generalized collaborative filtering [J]. *Information Sciences*,2007,177(15):3017 – 3037.