计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(10):86−91 [doi: 10.15888/j.cnki.csa.007118]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 Spark 并行化改进混合地点推荐①

蒲　鑫[1,2], 孟祥茹[2], 高　岑[2], 王美吉[2], 刘锦扬[3]

[1](中国科学院大学 计算机科学与技术学院, 北京 100049)
[2](中国科学院 沈阳计算技术研究所, 沈阳 110168)
[3](成都信息工程大学 统计学院, 成都 610103)
通讯作者: 蒲　鑫, E-mail: puxin16@mails.ucas.ac.cn

摘　要: 推荐算法是数据挖掘中最重要的算法之一. 地点推荐是推荐系统的重要研究内容. 针对目前地点推荐面临的数据稀疏、冷启动、个性化程度低等问题, 设计并实现了基于 Spark 并行化处理的改进混合地点推荐模型. 该算法融合了基于内容的推荐和基于协同过滤的推荐, 结合了用户当前的偏好和其他用户的意见. 使用基于用户-地点属性偏好的矩阵填充方式, 以此改善数据稀疏性问题; 同时, 对于海量数据, 系统采用 Spark 分布式集群实现并行计算, 缩短了模型训练时间. 实验结果表明, 与其他推荐算法相比, 该算法能有效改善数据稀疏性、提升推荐效果.

关键词: 地点推荐; 混合模型; 数据填充; 协同过滤; Spark

引用格式: 蒲鑫,孟祥茹,高岑,王美吉,刘锦扬.基于 Spark 并行化改进混合地点推荐.计算机系统应用,2019,28(10):86–91. http://www.c-s-a.org.cn/1003-3254/7118.html

Improving Hybrid Location Recommendation System Based on Spark Parallelization

PU Xin[1,2], MENG Xiang-Ru[2], GAO Cen[2], WANG Mei-Ji[2], LIU Jin-Yang[3]

[1](School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China)
[2](Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)
[3](School of Statistics, Chengdu University of Information Technology, Chengdu 610103, China)

Abstract: The recommendation algorithm is one of the most important algorithms in data mining. Location recommendation is an important research content of the recommendation system. Aiming at the problems of sparse data, cold start and low degree of personalization, the improved hybrid location recommendation algorithm based on Spark parallelization is designed and implemented. The algorithm combines content-based recommendations and collaborative filtering-based recommendations, combines the user's current preferences with the opinions of other users. We improve data sparsity by using a matrix fill based on user preferences for location attributes; Also, for massive data, the system uses Spark distributed cluster to realize parallel computing, which shortens the model training time. Experimental results show that compared with other recommended algorithms, the proposed algorithm can effectively improve data sparsity and improve recommendation.

Key words: location recommendation; hybrid model; data filling; collaborative filtering; Spark

在信息爆炸的时代, 从海量数据中找到用户感兴趣的信息是一件非常困难的事情, 推荐系统的任务就是挖掘数据中所隐藏的模式. 基于位置服务的应用在移动设备上也快速发展, 这些应用通过定位接口采样了大量的地点签到数据[1]. 从而实现面向用户的地点推荐.

本文研究的主要内容就是地点推荐算法. 根据利用的信息不同, 推荐算法可以分为基于内容[2]和基于协同过滤[3]的推荐两类算法. 基于内容的推荐算法提取用

① 收稿时间: 2019-02-22; 修改时间: 2019-03-14, 2019-04-15; 采用时间: 2019-04-26; csa 在线出版时间: 2019-10-15

户或者项目的特征构建用户偏好文档, 通过计算项目和用户偏好文档的相似度来推荐, 推荐的是满足用户本身的偏好. 与基于内容的推荐算法不同, 基于协同过滤的推荐算法考虑的是与他相似的用户的意见, 它的核心思想是相似的用户有类似的偏好. 这两类推荐算法有各自的优缺点. 基于内容的推荐算法能够很好地反映用户对项目的偏好, 可以为具有特定爱好的用户推荐, 但用户偏好文档根据用户的历史数据构建, 不能发现潜在偏好; 而基于协同过滤的推荐算法使用相似用户预测评分, 利用其他用户的意见, 可以发现潜在偏好, 但存在冷启动[4,5]、稀疏性、推荐效果依赖于已评分项的多少和准确性的问题.

根据以上分析, 本文提出了一个综合利用这两种推荐模型的地点推荐模型, 综合利用了两种推荐算法都有各自的优点. 此模型基于用户偏好文档和用户地点签到矩阵, 使用基于内容的推荐算法满足用户的个性化需求, 而基于协同过滤的推荐算法则可以利用其他用户的意见, 发掘用户的潜在偏好. 同时, 同时为应对海量数据的挑战, 使用 Spark 平台完成模型的并行化训练.

1 混合推荐模型

冷启动[6]和稀疏性问题是地点推荐最突出的问题. 在地点推荐中, 协同过滤算法是基于用户-地点签到矩阵实现的. 不同于传统商品推荐, 地点推荐中的大量用户访问的地点非常有限, 而且用户的签到记录中没有负样本, 从而导致用户-地点签到矩阵稀疏性非常高. 再者, 如果仅仅只使用协同过滤算法来实现推荐系统, 则只会利用到用户的历史偏好, 也会有冷启动和个性化程度低的问题. 使用基于内容和基于协同过滤的混合推荐算法不仅能有效改善稀疏性, 而且也能兼顾个性化推荐. 本文选择基于用户属性偏好文档和用户-地点评分矩阵模型组合的方式, 前者容易计算, 后者的改进填充方法可改善稀疏性问题.

本文提出的推荐模型, 结合两种推荐算法的优势, 利用个人偏好和地点的属性信息来填充签到矩阵, 大大改善稀疏性和有效性; 利用用户的输入约束条件实现了个性化推荐, 使用协同过滤考虑了相似用户的意见. 模型的推荐过程分为两个子过程: 基于内容推荐和基于协同过滤的推荐环节. 整个系统的各个流程如图 1 所示.

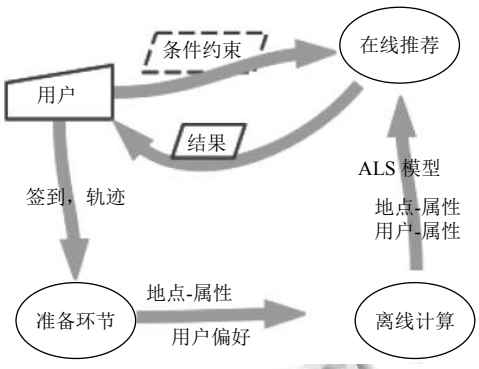


图 1 模型基本环节图

本文提出的地点推荐模型分为准备环节、离线计算和在线推荐 3 个基本环节:

(1) 准备环节: 为地点推荐模型的训练准备数据. 原始数据经过 ETL 加工清洗, 得到目标数据, 主要是地点签到矩阵、用户偏好文档.

(2) 离线计算: 主要实现模型的建立. 首先由地点-签到矩阵计算出两个矩阵: 地点-属性和用户-属性矩阵, 然后用这两个矩阵采用基于地点属性的矩阵填充方法填充用户-地点签到矩阵, 最后训练基于 ALS 的协同过滤算法模型.

(3) 在线推荐: 在线环节负责搜集实时的场景, 如用户的输入约束, 利用训练好的模型做推荐.

地点推荐模型的在线推荐环节根据用户输入的约束条件将符合条件的地点推荐给用户, 这个过程跟特定的用户有关. 如图 2 所示.

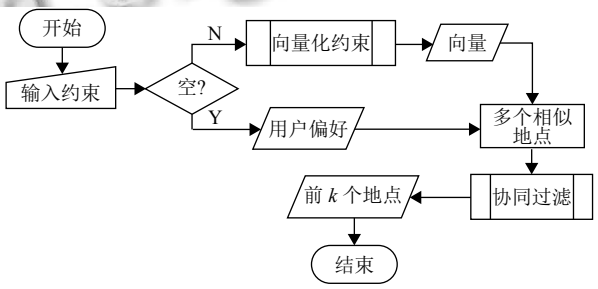


图 2 在线推荐环节流程图

(1) 首先根据用户输入约束交互式约束 IC, 这是用户感兴趣的地点的属性要求. 如果用户有输入则进入 (2). 否则进入 (4).

(2) 将用户的 *IC* 向量化, 转换成属性约束向量 *CV*.

(3) 根据得到 *CV* 结合地点-属性矩阵选出满足一定条件的地点集合 *C*, 进入 (5).

(4) 根据用户偏好文档结合地点-属性矩阵选出满足一定条件的地点集合 C, 进入 (5).

(5) 使用训练好的模型为集合 C 作最终的评分, 将前 k 个地点作为最终结果返回.

系统首先读取用户输入的条件, 然后将用户的输入向量化, 再利用建立的地点-属性矩阵就可以筛选出候选集合 C, 计算方式就是将约束向量与代表地点的属性向量相乘, 如果结果不为 0 就是满足条件的地点; 没有输入约束的条件下, 使用用户偏好文档来构成向量, 最后将满足结果的地点组成集合 C. 本文使用的倒查表的方式实现的用户偏好文档, 描述了用户对于地点的属性偏好, 倒查表建立的基础是用户的地点描述信息, 示意图如图 3 所示.

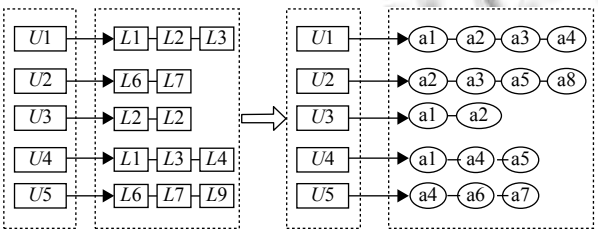


图 3 描述用户偏好的倒查表

如图 3 所示, 模型将用户地点签到数据中的地点描述信息转换为用户属性偏好, 图中的椭圆代表标签属性, 直角矩形代表的是用户, 圆角矩形代表用户的签到描述信息.

在计算对候选地点的评分时, 直接使用的是离线阶段训练好的 ALS 模型计算评分并排序. 评分计算公式如下:

$$P_{U,I} = P_U Q_I = \sum_{k=1}^{K} P_{U,k} Q_{k,I} \tag{1}$$

式 (1) 中, $P_{U,I}$ 表示用户 U 对地点 I 的评分, P_U 模型中代表用户 U 的隐含偏好向量, Q_I 表示地点 I 的隐含特征向量. K 表示模型使用的隐含因子数.

模型除了构建用户的地点属性偏好文档和用户-地点签到矩阵 (表 1) 外, 还有 2 个矩阵, 分别是地点-属性矩阵 (表 2)、用户-属性偏好矩阵 (表 3), 地点-属性矩阵表明了一个地点所具有的属性信息, 用户-属性矩阵代表了用户对地点属性的偏好情况, 通过这两个矩阵来填充用户-签到矩阵. 总体的流程是根据输入的约束得到用户的属性偏好, 根据偏好得到相应满足的地点集合 C, 从这些地点中使用基于模型的协同过滤计算出评分并最终地点给用户, 因此本文提到的模型前阶段使用的基于内容的推荐, 实现了个性化需求, 后面的环节使用的协同过滤, 来挖掘用户的历史偏好.

表 1 用户-地点签到矩阵

用户	$L1$	$L2$	$L3$	$L4$
$U1$	8	9	8	8
$U2$	1	1	8	2
$U3$	6	9	4	3
$U4$	1	0	3	1

表 2 地点-属性矩阵

地点	餐馆	公园	电玩	WIFI
$L1$	0	1	1	0
$L2$	1	1	1	1
$L3$	1	0	1	1
$L4$	1	1	0	1

表 3 用户-属性偏好矩阵

用户	餐馆	公园	电玩	WIFI
$U1$	8	8	8	8
$U2$	4	4	3	4
$U3$	5	4	6	5
$U4$	2	2	2	2

表 1 中的数字表示用户对地点的评分, 也就是签到次数. “0”表示用户没有在该地点签到过. 鉴于每个用户的签到频率不一致, 这里没有采取统一的归一化处理.

表 2 中的数字”1”表示地点具有该属性, “0”表示该地点没有这个属性. 通过分析用户地点评分及地点所具有的属性信息, 可以得到用户描述用户对一个具体的地点属性的兴趣度的用户-地点属性偏好矩阵, 如表 3 所示.

模型会维持这 3 个矩阵, 地点-属性矩阵的信息来自于用户地点签到信息, 再根据用户-地点签到矩阵和地点-属性矩阵推算出用户-属性偏好矩阵, 使用用户-属性偏好矩阵来填充用户-地点签到矩阵. 由于用户在签到时都会带有地点的标签信息, 从而导致地点的属性相比之下容易获得, 通过用户对地点属性评分可以预测用户对具有该属性的地点的评分, 具体的计算流程如 1.1 节所述, 这样极大的解决了数据稀疏的问题, 提高了推荐的准确度.

1.1 基于用户-地点属性偏好的数据填充方式

在地点推荐中的一个很大问题就是地点数量很大, 造成评分矩阵很稀疏, 目前很多填充方法都没有考虑

用户的偏好, 缺乏可靠性. 在数据很稀疏的情况下能够造成准确率严重下降, 鉴于地点推荐数据的特殊性, 本文利用了用户签到数据中的地点的属性标签, 利用这些数据可以提取出用户的属性偏好. 如表 3 所示, 用户-地点属性矩阵就是用户的偏好的体现. 这也是用户历史偏好的体现, 利用了用户的偏好的填充方法能取得更好的推荐效果.

为了利用用户的历史偏好, 需要从用户-地点评分矩阵中总结出用户的地点属性偏好, 这就需要利用地点的属性信息, 所以本文所提的推荐模型建立了地点-属性矩阵, 此矩阵容易建立, 因为地点签到数据中包含地点的属性信息, 如表 3. 也就是根据表 1 的用户-地点签到矩阵和表 2 的地点-属性矩阵计算出表 3 的用户-属性评分矩阵, 再利用表 3 的数据填充表 1 的空白项. 具体计算过程如下:

计算方法如式 (2) 所示. 表 1 中的数字为用户对地点的签到次数, 表 2 中 1 表示地点具有该属性. 计算方法如式 (1), 然后根据用户对地点属性的评分对用户-地点矩阵进行填充.

$$a_u = \frac{\sum_{i \in I_{u,a}} R_{u,i}}{|I_{u,a}|} \tag{2}$$

式 (2) 中, a_u为用户 u 对地点属性a的评分; $I_{u,a}$为用户 u 已评分且包含属性 a 的地点集合, $|I_{u,a}|$为该集合中的元素个数; $R_{u,i}$为用户 u 对地点 i 的评分. 通过计算得到用户-地点属性偏好矩阵, 如表 3.

在对一个用户评分缺失项进行填充时需要充分考虑地点的属性信息, 使用地点的属性评分和用户平均评分的综合, 具体计算式 (3):

$$r_{u,i} = \lambda \frac{\sum_{a \in A_i} r_{u,a}}{|A_i|} + (1-\lambda) \frac{\sum_{j \in Q} b_j}{|Q|} \tag{3}$$

其中, $r_{u,i}$表示用户 u 对地点 i 的评分, 也就是需要填充的评分; A_i表示地点 i 包含的属性集合, $|A_i|$为该集合的元素个数; $r_{u,a}$为用户 u 对属性 a 的评分; Q 表示用户的评分项目中除去A_i的集合, b_j为用户对地点 j 的评分; λ参数表示用户历史偏好和当前要填充的地点的相似度, 此值越高代表利用地点属性信息的程度越高, 引入 λ的目的是综合考虑当一个地点的属性标签也很少时的情况, 这时还是需要引入用户的平均签到次数作为评分. 具体计算公式如式 (4):

$$\lambda = \frac{|L_i \cap A_u|}{|A|} \tag{4}$$

其中, L_i表示地点 i 的属性集合, A_u表示用户-属性矩阵中用户 u 的偏好属性集合, A 表示整个系统的地点属性集合, 当地点的属性和用户的偏好越相近时λ越大. 按照步骤可以计算出用户 U=4 对 $L2$ 的评分为 2.

2 Spark 计算并行化

Spark 是一个基于内存计算的开源的集群计算框架. 与 MapReduce 相比, 它具有负载均衡、自动容错和容易扩展等优点. Spark 的核心是 RDD (Resilient Distributed Datasets), 他是一种只读的并行数据结构, 具有很高的可扩放性. Spark 包括的组件有: Spark SQL、Spark streaming、Mlib 和 GraphX, 这些组件使得 Spark 形成大数据一站式解决平台[7].

Spark MLlib 当前支持基于模型的协同过滤, 也是一种隐语义模型[8], 其核心问题是矩阵分解, 用户和项目通过一小组隐语义因子进行表达, 并且这些因子也用于预测缺失的元素. MLlib 使用交替最小二乘法 (ALS) 来学习这些隐性因子. 是将用户-项目评分矩阵分解为用户-隐含特征偏好矩阵和隐含特征-项目矩阵, 即:

$$R_{(m \times n)} = X^{\mathrm{T}}_{(m \times k)} Y_{(k \times n)} \tag{5}$$

其中, $R_{(m \times n)}$代表用户-项目评分矩阵, $X_{(m \times k)}$代表用户-隐含特征偏好矩阵, $Y_{(k \times n)}$表示隐含特征-项目矩阵, 其中$k = min(m,n)$, 为了使X和Y的乘积尽可能逼近R, 采用误差平方和最小作为损失函数:

$$L(X,Y) = \sum_{i=1}^{n} \left(r_{u,i} - x_u^{\mathrm{T}} y_i\right)^2 \tag{6}$$

式 (6) 中, 表示第 u 个用户对第 i 个项目的评分, 本文中的用户在此地点的签到次数, x_u表示用户 u 的偏好特征向量, y_i 表示地点 i 的隐含特征向量, $x_u^{\mathrm{T}} y_i$为用户 u 对地点评分的近似, 为防止过拟合, 加入正则化项:

$$L(X,Y) = \sum_{i=1}^{n} \left(r_{u,i} - x_u^{\mathrm{T}} y_i\right)^2 + \lambda\left(|x_u|^2 + |y_i|^2\right) \tag{7}$$

采用梯度下降迭代计算, 当均方根误差变化小于指定阈值或迭代次数达到一定时, 迭代结束. ALS 算法功能强大,效果理想而且被证明相对容易并行化, 所以模型特别适合在 Spark 框架下训练.

3 实验与分析

3.1 实验环境

本文的实验环境: 采用 Hadoop HDFS 作为底层存储、Hadoop YARN 作为集群资源管理的 Spark 分布

式集群平台. 主机处理器: Intel(R)Core(TM)i7-7700 CPU, 核心数为 8; 内存: DDR4 32 GB 内存. 通过 VMware 实现分布式平台.平台由 4 个 CentOS 操作系统的节点组成, 1 个节点为 Master 节点, 3 个节点为 Slave 节点.

JAVA 环境为 JDK1.8.0_201, 分布式系统基础架构为 Hadoop2.7.3; 大数据处理并行框架为 Spark2.4.0; 推荐算法开发语言为 Python2.7.12.

3.2 数据来源与评价指标

Foursquare 是著名的社交网站, 为用户提供了基于位置的签到服务. 使用 Foursquare, 用户能在任何一个地点签到, 比如说地点是餐馆, 公园等. 每一个签到发布后, Foursquare 的签到会包含位置的物理位置和用户的描绘信息, 如地点评价等. 由于 Foursquare 网站中, 不提供数据的下载, 所以本文利用已有的 Twitter, Facebook 账号与 Foursquare 相连. 通过 Twitter 提供的 API 接口, 抓取签到信息. 本文抓取了从 2017 年 10 月到 2018 年 12 月的 15 102 513 条签到数据. 共有 8690 个用户, 地点数目 75 662 个. 将数据集按照 3∶1 的比例划分为训练集和测试集.

为了评估推荐算法的准确度, 选择了推荐系统中常用的平均绝对误差 (MAE) 和归一化折损累计增益 (NDCG) 作为评价指标[9], MAE 反映的是预测值和真实值间的差距, NDCG 则是反映对多个候选项目的预测排序情况的优劣, 计算方式如式 (8) 所示.

$$NDCG_u(u)=\frac{NCG_n(u)}{\max DCG_n(u)} \tag{8}$$

式中, $NDCG_u(u)$为用户 u 候选项目预测值排序的 $NCG_n(u)$值比上实际情况的$\max DCG_n(u)$值, $DCG_n(u)$值由式 (9) 计算得出.

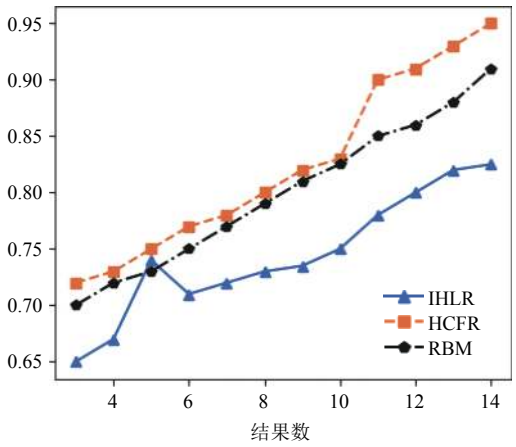
$$DCG_u(u)=r(u,i_1)+\sum_{k=2}^{n}\frac{r(u,i_k)}{\log_2 k} \tag{9}$$

3.3 实验结果分析

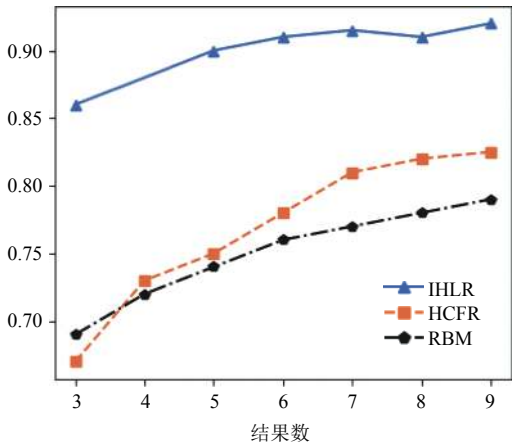
本文选择了几个典型的算法作为比较, 选择的算法为传统的混合协同过滤算法 (HCFR)、基于受限波尔兹曼机的推荐算法 (RBM)[10].

3 种模型在 MAE 和 NDCG 这两个指标上的实验效果如图 4 所示.

如图 4 所示, 图中横坐标表示推荐的结果数, 纵坐标是相应的测评值, 可以看出, 本文所提出的改进的推荐算法在表现效果上都要高于其他两个对比算法, 表现最差的是传统的协同过滤算法, 这是因为 HCFR 适用于商品推荐, 没有考虑地点推荐的特殊性, 也没有在数据填充上做改进. 本文模型所使用的算法融合了用户的历史偏好和个性化需求, 而且使用了用户的偏好信息来填充矩阵, 填充的结果更加有效, 大大地改善了数据稀疏性问题, 相比于其他两种算法, 更能体现地点的特征, 更适合地点推荐场景.



(a) MAE



(b) NDCG

图 4 实验结果

4 总结

本文提出了一种融合用户当前需求和历史偏好的个性化混合地点推荐系统, 并在推荐的核心阶段做了改进, 通过提取用户对地点的属性偏好, 填充用户-地点签到矩阵, 有效地改善了数据稀疏问题; 在协同过滤阶段采用了基于模型的地协同过滤算法, 最后将 IHLR 和 HCFR 和 RBM 进行了对比, 表明本文所提的 IHLR 方法在效果上能表现得更好, 接下来的工作是将用户之间的好友关系用到系统中, 使得推荐结果更加

准确, 此外, 系统运行在流行的 Spark 分布式平台, 适合做海量数据的推荐任务.

参考文献

1 郝日佩. 基于用户签到数据的个性化地点推荐算法研究[硕士学位论文]. 西安: 西安电子科技大学, 2017.

2 刘俊杰, 苏贵斌. 数据挖掘推荐算法. 计算机产品与流通, 2018, (10): 99.

3 李嵩, 李书琴, 刘斌. 改进的协同过滤算法及其并行化实现. 计算机工程与设计, 2018, 39(12): 3853–3859.

4 孙小华. 协同过滤系统的稀疏性与冷启动问题研究[博士学位论文]. 杭州: 浙江大学, 2005.

5 Bobadilla J, Ortega F, Hernando A, *et al*. A collaborative filtering approach to mitigate the new user cold start problem. Knowledge-Based Systems, 2012, 26: 225–238. [doi: 10.1016/j.knosys.2011.07.021]

6 陈妍, 洪蕾, 李广水, 等. 关于推荐系统中冷启动问题的研究. 中国高新区, 2018, (14): 29–31.

7 Shanahan J, Dai L. Large scale distributed data science from scratch using apache spark 2.0. Proceedings of the 26th International Conference on World Wide Web Companion. Perth, Australia. 2017. 955–957.

8 Shen YL, Jin RM. Learning personal + social latent factor model for social recommendation. Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Beijing, China. 2012. 1303–1311.

9 Polatidis N, Georgiadis CK. A dynamic multi-level collaborative filtering method for improved recommendations. Computer Standards & Interfaces, 2017, 51: 14–21.

10 马宗学. 基于改进受限玻尔兹曼机推荐算法的分布式实现[硕士学位论文]. 兰州: 兰州大学, 2015.