

# 基于 Spark 的推荐系统的设计与实现

李 星,李 涛

(南京邮电大学 通信与信息工程学院,江苏 南京 210003)

**摘 要:**推荐系统是数据挖掘的一个重要部分,能够实现海量数据信息的快速、全面、准确过滤。然而基于以往传统单个主机模式实现的推荐算法其计算过程耗费的时间过长,已经不能满足当前商业时代快速可靠的技术追求。大数据平台 Spark 分布式计算框架通过引入 RDD(弹性分布式数据集)的概念以及基于内存的运算模式,能够更好地适应大数据挖掘这一应用场景。推荐算法在实现过程中存在多次迭代计算,Spark 计算框架的使用可以极大提升推荐系统的运算效率。文中利用 Spark 平台设计了一个基于物品的协同过滤(Item-CF)算法的商品推荐系统,并将其应用在 MovieLens 数据集上运行测试。实验结果表明,该系统能够提高推荐精确度并降低运算时间。

**关键词:**大数据;Spark 平台;推荐系统;协同过滤(CF);数据挖掘

**中图分类号:**TP302

**文献标识码:**A

**文章编号:**1673-629X(2018)10-0194-05

**doi:**10.3969/j.issn.1673-629X.2018.10.040

## Design and Implementation of Recommendation System Based on Spark

LI Xing, LI Tao

(School of Communication and Information Technology, Nanjing University of Posts and  
Telecommunications, Nanjing 210003, China)

**Abstract:** The recommendation system is an important part of data mining, which can realize the rapid, comprehensive and accurate filtering for a large number of data. However, it takes a lot of time to realize the proposed algorithm based on the traditional single-machine model, which cannot meet the fast and reliable business needs in today's business era. The Spark distributed computing framework of big data platform can better adapt to big data mining by introducing the concept of RDD (resilient distributed datasets) and based on memory computing mode. The recommendation algorithm has many iterative calculations in the implementation process, and the use of the Spark calculation framework can greatly enhance the efficiency of the recommended system. We use the Spark platform to design a product recommendation system based on item-based collaborative filtering (Item-CF) algorithm, which is applied to run a test on the MovieLens data set. The experiment shows that the system can improve the recommendation accuracy and reduce the operation time.

**Key words:** big data; Spark; recommendation system; collaborative filtering (CF); data mining

## 0 引言

在当今大数据时代<sup>[1]</sup>背景下,处在互联网浪潮之中的人们能够以前所未有的方式获得更多、更全、更丰富的信息。当前逐步从信息匮乏的阶段慢慢过渡到信息爆炸的阶段,从而进入到一个信息过载的时代,人们在享受丰富信息福利的同时,也经常陷入到数据的海洋中无所适从。信息过滤是人们必须要面对的问题,而推荐系统一直发挥着至关重要的作用。推荐系统不仅可以使用户更方便快捷地发现切合自身需求的有用信息,而且能够保证信息更加准确地推送给潜在用户,

达到企业与消费者的共赢<sup>[2]</sup>。在推荐系统中影响其性能高低最重要的因素就是其推荐算法的效率,而推荐系统中最为经典的算法就是协同过滤算法<sup>[3]</sup>,该算法是基于用户-物品行为历史数据集,利用其中的相似性计算或者用户兴趣模型训练的办法进行过滤推荐。

在信息泛滥的互联网世界中,不仅需要过滤海量数据,而且要力求在最短时间内响应用户的需求,推荐系统必然要具备大数据处理能力。目前这一领域的框架有很多,其中 Spark 作为主流的内存计算框架,具有很强的大数据处理能力,运行于 Spark 平台<sup>[4]</sup>的推荐

收稿日期:2017-11-08

修回日期:2018-03-12

网络出版时间:2018-05-28

基金项目:国家自然科学基金(61572260)

作者简介:李 星(1991-),男,硕士研究生,研究方向为大数据分析与应用;李 涛,硕士,副教授,研究方向为信号与信息处理在网络中的应用及新型网络中的路由技术。

网络出版地址: <http://cnki.net/kcms/detail/61.1450.TP.20180525.1558.024.html>

系统有望获得极高的运行效率和处理能力。文中即是基于 Spark 分布式计算框架进行推荐系统的研究和实现。

## 1 Spark 平台

### 1.1 Spark 简介

Spark 是一个通用的大规模数据快速处理引擎<sup>[5]</sup>, 是美国加州大学伯克利分校 AMPLab 于 2010 年开发的大数据计算平台。不同于 Hadoop MapReduce<sup>[6]</sup>, Spark Job 计算的中间输出结果可以直接缓存在内存当中,不再像 MapReduce 那样频繁读写本地磁盘。基于以上特点,Spark 在分布式迭代运算方面的速度要远远优于经典的 Hadoop MapReduce 框架。

AMPLab 还基于 Spark Core 开发了大数据处理一体化的技术生态系统 BDAS(the Berkeley data analytics stack),即伯克利数据分析软件栈。除了基础的 Spark 计算框架,它还具有更高层级的计算能力<sup>[7]</sup>,主要包括 Spark Streaming 流处理框架、SparkSQL 结构化数据的查询及分析的查询引擎、GraphX 图计算框架和 MLlib 机器学习库等等。恰是因为上述子项目的存在,使得 Spark 能够提供更全面灵活的计算能力。

### 1.2 Spark 设计思想

Spark 对数据的核心抽象是弹性分布式数据集(resilient distributed datasets, RDD<sup>[8]</sup>),其实就是分布式的元素集合。有别于普通的数据集,RDD 中的数据是分区存储的,以达到数据的并行处理。因此,Spark 处理数据的过程即是通过需处理的数据创建 RDD,然后对 RDD 进行相应的 Transformation 和 Action 操作并最终得到运算结果。RDD 通常缓存在内存中,父 RDD 的输出结果可以在内存中直接作为子 RDD 的输入,迭代计算的效率因此大大提高。使用 Spark 编程无需关注底层的数据切分和存储过程以及计算过程的容错机制,只需集中于业务逻辑处理,提高了编程效率。

### 1.3 Spark 的运行架构

在 Spark 集群部署后,Master 进程和 Worker 进程分别在主节点和从节点启动运行。在 Spark 应用程序的执行过程中<sup>[9]</sup>,每个 Spark 应用都由一个 Driver 程序来负责作业的调度,即分发 Task 任务,而 Worker 负责监控本节点的资源状况以及创建 Executor 进程。在执行阶段,Driver 程序会将 Task 和 Task 所依赖的数据文件和程序 jar 包等分发给对应的 Worker 节点,同时 Executor 进程对分区 RDD 进行运算处理。

Spark 工作的流程如图 1 所示。用户首先向 Spark 集群提交应用程序,Master 进程会在一个 Worker 节点上启动 Driver 程序来进行任务管理,Driver 根据任务

情况向 Master 申请到资源(CPU、内存等),并初始化 Executor。然后 SparkContext 中的 DAGScheduler 会将任务生成有向无环图(DAG)并提交给 TaskScheduler,TaskScheduler 再根据 DAG 生成相应的 TaskSet 并分配任务给 Executor 并发执行。

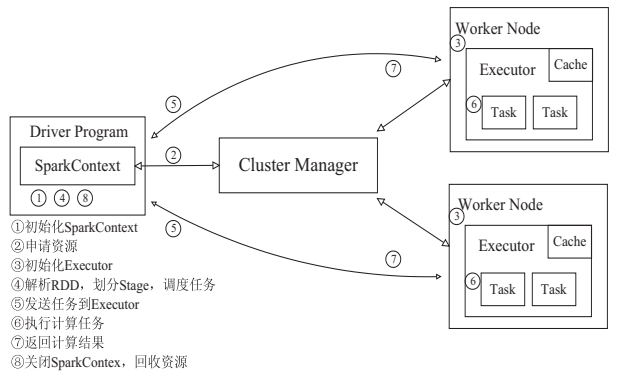


图 1 Spark 工作流程

### 1.4 环境说明

文中搭建的 Spark 集群<sup>[10]</sup>采用了实验室中的四台普通计算机,组成了一个 Master 主节点和三个 Slave 节点的运行环境,节点均采用 Centos6.5 Linux 系统,节点之间使用局域网进行网络连接,同时以上节点还运行了分布式文件存储系统 HDFS 的服务。具体情况如表 1 所示。

表 1 节点具体说明

主机名	IP 地址	作用
Spark01	192.168.1.44	NameNode, Master
Spark02	192.168.1.45	DataNode, Worker
Spark03	192.168.1.46	DataNode, Worker
Spark04	192.168.1.47	DataNode, Worker

### 1.5 软件安装

本系统 Java JDK 的安装包采用 jdk-7u79-linux-x64.tar.gz, Scala 的安装包采用 scala-sdk-2.10.4.tar.gz, Hadoop 安装包采用 hadoop-2.5.0-cdh5.3.6.tar.gz,在系统软件安装过程中最重要的一点就是要配置安全外壳协议(SSH)以便实现远程无密码登陆与管理。在完成 Hadoop 集群的配置安装后,在此基础上进行 Spark 的安装,Spark 安装包采用 Spark 1.5.1.tar.gz,软件开发环境采用 IntelliJIDEA 2017.2.1。集群所有软件在安装并配置完毕后即可启动 Spark 分布式集群进行测试。

## 2 推荐系统

### 2.1 推荐系统概述

如今推荐系统已经普遍应用于电商、新闻、地理位置等诸多领域。比如在电商领域,推荐系统实现的功能就是根据已有信息,如物品信息、用户信息、用户行为信息等,将相应的物品推荐给用户。常见的推荐任

务主要有两种;评分预测和 Top-N 推荐。对于用户 U,评分预测的任务是预测 U 对某物品可能的打分,而 Top-N 推荐则是为用户 U 推荐 N 个他可能感兴趣的物品。这两种推荐都是依据用户、物品自身的信息及用户过去的行为记录做出的预测。

## 2.2 协同过滤算法

协同过滤(collaborative filtering, CF)算法的核心思想就是利用群体的智慧来进行事物推荐。协同过滤按照参照物的不同主要分为基于用户的协同过滤(user-based CF)和基于物品的协同过滤(item-based CF)<sup>[11]</sup>。user-based CF 在推荐时首先根据行为记录找到相似用户(即人以群分),然后根据相似用户进行推荐。item-based CF 最初由 Amazon 提出并应用,基于物品协同过滤算法不是计算用户间的相似度,而是计算物品间的相似度(即物以类聚),将与目的用户偏好过的商品的相似商品作为候选推荐列表推荐给目的用户。对于大型电子商务平台而言,商品更新速率较慢并且商品的数目远小于用户数目,计算物品相似度开销远远比计算用户相似度开销小,因而 item-based CF 的稳定性比 user-based CF 的稳定性更高<sup>[12]</sup>。故该系统采用基于物品协同过滤的 Top-N 推荐。

## 2.3 基于物品协同过滤算法的基本思路

item-based CF 的基本思路和流程如下:

(1)收集用户的历史行为数据,进行初步的减噪和归一化处理。统计得出物品-用户评分矩阵,用以表示具体每件物品被哪些用户评分过,该表其实是由用户-物品评分表(如表 2 所示)转置得到,而用户-物品评分矩阵则是每位用户对物品的历史评分记录。

表 2 用户-物品评分表

用户	物品			
	$m_1$	$m_2$	$m_3$	$m_4$
$u_1$	1	1	1	0
$u_2$	1	1	0	1
$u_3$	0	1	0	0

推荐系统中主要有显式评分与隐式评分两种评分方式。其中显示评分指的是用户对物品的评价直接用具体分数来表示,比如平时比较常见的用户对某种物品的喜好程度,可以由 1 至 5 分来量化;而隐式评分则只是依照用户对于某种物品有没有购买、是否浏览过、评论过等行为,如果有相关行为则评分为 1,无则评分为 0。于是,可以得到一个  $N \times M$  矩阵  $R$ :

$$R_{ij} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1M} \\ r_{21} & r_{22} & \cdots & r_{2M} \\ \vdots & \vdots & \cdots & \vdots \\ r_{N1} & r_{N2} & \cdots & r_{NM} \end{pmatrix} \quad (1)$$

其中,  $N$  表示用户数量;  $M$  表示物品数量;  $r_{ij}$  表示用户  $u_i$  对物品  $m_j$  的评分。若用户  $u_i$  对物品  $m_j$  进行了评分,则  $r_{ij} \neq 0$ , 否则  $r_{ij} = 0$ 。由于文中采用隐反馈方式,所以  $r_{ij}$  的取值只能是 1 或 0。

(2)计算物品之间的相似度。

找到与物品最相似的  $N$  个物品集合,物品  $m$  和  $n$  的相似度可使用以下余弦相似度公式计算:

$$\text{sim}_{mn} = \frac{|N_{(m)} \cap N_{(n)}|}{\sqrt{|N_{(m)}| |N_{(n)}|}} \quad (2)$$

其中,  $N_{(m)}$  表示所有拥有物品  $m$  的用户集合;  $N_{(n)}$  表示所有拥有物品  $n$  的用户集合。使用相似度计算公式可计算出示例表 2 中物品  $m_1$ 、 $m_2$  以及  $m_1$ 、 $m_3$  的相似度:

$$\text{sim}_{m_1 m_2} = \frac{|N_{(m_1)} \cap N_{(m_2)}|}{\sqrt{|N_{(m_1)}| |N_{(m_2)}|}} = \frac{|\{u_1, u_2\} \cap \{u_1, u_2, u_3\}|}{\sqrt{|\{u_1, u_2\}| |\{u_1, u_2, u_3\}|}} = \frac{2}{\sqrt{6}} \quad (3)$$

$$\text{sim}_{m_1 m_3} = \frac{|N_{(m_1)} \cap N_{(m_3)}|}{\sqrt{|N_{(m_1)}| |N_{(m_3)}|}} = \frac{|\{u_1, u_2\} \cap \{u_1\}|}{\sqrt{|\{u_1, u_2\}| |\{u_1\}|}} = \frac{1}{\sqrt{2}} \quad (4)$$

由结果可知  $m_1$ 、 $m_2$  间有更大的相似度,因此在一位新用户购买  $m_1$  时,可优先将  $m_2$  商品推荐给他。

在示例计算过程中可发现此方法实际上会对比物品空间中的所有物品,致使其时间复杂度较高,为  $O(n^2)$ 。然而一个大型数据集中,会出现很多物品并没有共同用户交集的情况,因此在实际操作中,建立用户-物品倒排表可以优化计算,如图 2 所示。其中矩阵  $W$  中的值为式 2 中分子的值。

$m_1$	$u_1$	$u_2$						
				$u_1$	$m_1$	$m_2$	$m_3$	
$m_2$	$u_1$	$u_2$	$u_3$	$\Rightarrow u_2$	$m_1$	$m_2$	$m_4$	
$m_3$	$u_1$							
				$u_3$	$m_2$			
$m_4$		$u_2$						
					$m_1$	$m_2$	$m_3$	$m_4$
	$m_1$	0	2	1	1			
$W' = m_2$	2	0	1	1				
$m_3$	1	1	0	0				
$m_4$	1	1	0	0				

图 2 用户-物品倒排表生成过程

(3)计算出物品之间的相似度后,根据用户的历史物品表再计算出用户-物品兴趣度,以此进行物品推荐。用户  $u$  对物品  $m$  的兴趣度为:

$$\text{interest}(u, m) = \sum_{n \in N(m, k) \cap N(u)} \text{sim}_{mn} r_{un} \quad (5)$$

其中,  $\text{interest}(u, m)$  表示用户  $u$  对物品  $m$  的兴趣度;  $N(m, k)$  表示与物品  $n$  最相似的  $K$  件物品集合;

$N(u)$  表示用户  $u$  现已拥有的物品集合;  $\text{sim}_{mn}$  表示物品  $m$  和物品  $n$  之间的相似度;  $r_{un}$  表示用户  $u$  对物品  $n$  的兴趣值(若只考虑隐式反馈数据,  $r_{un}$  为 1)。

2.4 推荐系统评价指标

一个推荐系统的性能好坏可以用以下评测指标<sup>[13]</sup>来评定:

(1) 准确率(precision)。

准确率是一个非常重要的性能指标,直接关系到推荐系统服务质量的好坏。若推荐系统向用户  $u$  推荐了  $N$  件物品,推荐集合可记为  $R(u)$ ,而用户在测试集  $T$  上的喜好物品项集合记为  $T(u)$ ,则准确率计算公式如下:

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

准确率指的是向用户推荐的集合项  $R(u)$  中,有多少件物品属于正确推荐,即推荐正确项占推荐列表的比例。

(2) 召回率(recall)。

召回率指的是用户喜好的集合项  $T(u)$  中,含有多少推荐系统正确推荐的物品,即推荐正确项占真实喜好列表的比例。召回率反映了推荐系统推荐项的覆盖率,具体公式为:

Recall = 
$$\frac{\sum_u |R(u) \cap T(u)|}{\sum_u |T(u)|}$$
 (7)

(3) 覆盖率(recover)。

覆盖率指的是推荐集合项  $R(u)$  在供应商提供的物品总量中所占的比例<sup>[11]</sup>,具体公式为:

Recover = 
$$\frac{|\bigcup_{u \in U} R(u)|}{I}$$
 (8)

其中,  $U$  表示所有被推荐用户;  $I$  表示供应商提供的所有物品项。覆盖率描述了推荐系统对物品长尾的发掘能力。

(4) 均方根误差(RMSE)。

在推荐系统中,均方根误差主要是对评分预测进行评估。通常需要依照一定的方式将数据集切分为训练集和测试集,然后在训练集上训练用户的兴趣模型,即对物品的评分预测。在测试集中,假设用户-物品对为  $(u, m)$ ,用户对物品的真实评分为  $r_{um}$ ,模型训练出来获得的预测评分为  $\hat{r}_{nm}$ ,可用 RMSE 来表示最终预测的精度,公式为:

RMSE = 
$$\sqrt{\frac{\sum_{(u,m) \in T} (r_{um} - \hat{r}_{nm})^2}{|T|}}$$
 (9)

依据准确率和召回率公式定义可知,准确率和召回率往往是两个相互矛盾的指标参数<sup>[14]</sup>。系统推荐

并且用户真实喜好的物品项集与系统推荐的物品项之间的比值越大,准确率越大;系统推荐的物品项集中用户真实喜欢的物品越多,召回率也越大。如果推荐系统推荐的物品很多,虽然召回率可能提高,准确率也会有所下降;因此如果希望推荐系统更加准确,推荐物品数量一定要适当。

3 Spark 平台推荐算法并行化实现

3.1 基于物品的协同过滤推荐算法的 Spark 并行化实现流程

item-based CF 算法的核心思想是对用户推荐与其已喜欢物品相似的不同物品。其中比较关键的几个参数及公式为用户-历史评分矩阵  $R_m$ ,物品之间的相似度计算公式  $\text{sim}_{mn}$  及用户对物品兴趣度计算公式  $\text{interest}(u, m)$ 。本系统以电影推荐为背景,测试数据集选取经典的 MovieLens<sup>[15]</sup>,该数据集记录了用户评分记录。文中采用 100 万条记录作为实验数据集,将数据集文件下载并解压后,其中的 users. dat 文件记录所有用户相关信息;movies. dat 文件存储的是电影自身信息,包括电影名、类型、年份等。ratings. dat 文件记录用户-电影的评分信息及时间戳;每条评分记录的格式为 userid::itemid::rating::timestamp,前三项是需要的数据资源。基于物品的协同过滤推荐算法具体流程如图 3 所示。

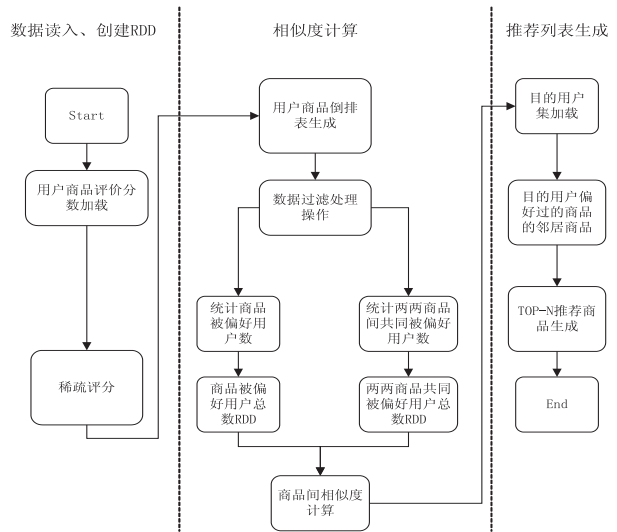


图 3 基于 Spark 的商品协同过滤算法流程

输入:userid、itemid、rating

输出:用户最感兴趣的  $N$  个物品项

(1) 计算用户对物品的喜好:采用隐反馈数据原则对每个用户给物品的评分进行预处理操作。数据处理如下:用户评分的物品记为 1,反之记为 0;

(2) 统计每个物品数好评总数;可设置阈值,低于指定数的物品不参与后续计算;

(3) 统计物品-好评键值对,对两个相关联的物品



之间的共同用户数进行统计(也可设置阈值);

(4) 计算任意两个有关联物品的相似度;

(5) 根据兴趣度向用户推荐  $N$  个最感兴趣的物品。

3.2 实验结果与分析

系统基于 Spark 平台推荐算法并行化实现,并测试计算推荐系统的相关评测指标:准确率、召回率、覆盖率等。实验中训练集占数据量 75%,测试集占 25%。根据不同的参数设置,每个实验均进行 4 次,并求得 4 次结果的平均值作为最终的测试结果<sup>[16]</sup>,如表 3 所示。

表 3 基于物品协同过滤推荐算法在不同推荐列表长度 ( $L$  值) 时的性能 %

$L$	准确率	召回率	覆盖率
20	20.74	10.23	15.37
40	19.37	9.82	14.89
60	18.36	9.57	13.75
80	17.93	9.02	12.87

从表 3 可以看出,基于物品协同过滤推荐算法中的准确率与召回率并不是和推荐列表长度呈线性关系,并且当推荐列表长度为 20 时,系统的准确率与召回率会比较高,由此可以看出,推荐列表长度的选择是基于物品协同过滤推荐系统性能的重要影响因素。而对比覆盖率可发现覆盖率随着  $L$  值的增长越来越低,是因为当推荐列表长度不断增加时,推荐算法越来越倾向于推荐比较热门的电影所致。

4 结束语

大数据时代互联网中蕴藏着海量富有价值的信息资源,如何更加快速有效地从中挖掘有用信息正是大数据应用平台需要考虑解决的问题。而用户推荐这一应用场景,正是从海量数据中挖掘有用信息的典型案例。研究表明,Spark 计算框架相比传统的 MapReduce 框架具有更强的并行计算能力。并且推荐算法中需要进行连续迭代计算,这一需求也恰好使之非常适合运行在 Spark 平台之上。文中以设计并实现一个大数据环境下的推荐系统为主线,介绍了大数据相关技术和推荐系统的相关概念,实现了基于 Spark 的 Item-CF 推荐系统,并在数据集 MovieLens 下进行了相关指标的测评。结果显示,该系统能够较好地完成推荐任务,达到了设计前的预期。下一步的工作将针对电商平台

中日益多样化的用户行为,设计多种的数据处理方式,优化推荐算法引擎,提升系统的通用性和可靠性。

参考文献:

[1] CHEN Min, MAO Shiwen, LIU Yunhao. Big data: a survey [J]. Mobile Networks and Applications, 2014, 19(2): 171-209.

[2] 项 亮. 推荐系统实践[M]. 北京:人民邮电出版社,2012.

[3] RESNICK P, LACOVOU N, SUCHAK M, et al. GroupLens: an open architecture for collaborative filtering of netnews [C]// Proceedings of the 1994 ACM conference on computer supported cooperative work. Chapel Hill, North Carolina, USA: ACM, 1994: 175-186.

[4] ZAHARIA M, CHOWDHURY M, FRANKLIN M J, et al. Spark: cluster computing with working sets [C]// Proceedings of USENIX conference on hot topics in cloud computing. [s. l.]: USENIX Association, 2010: 10.

[5] 高彦杰. Spark 大数据处理, 技术、应用与性能优化[M]. 北京:机械工业出版社,2015:215-217.

[6] SHVACHKO K, KUANG H, RADIA S, et al. The Hadoop distributed file system [C]// IEEE 26th symposium on mass storage systems and technologies. [s. l.]: IEEE, 2010: 1-10.

[7] HAN Zhijie, ZHANG Yujie. Spark: a big data processing platform based on memory computing [C]// Proceedings of international symposium on parallel architectures, algorithms and programming. Nanjing, China: IEEE, 2015: 172-176.

[8] 梁 彦. 基于分布式平台 Spark 和 YARN 的数据挖掘算法的并行化研究[D]. 广州:中山大学,2014.

[9] 郭景瞻. 图解 Spark 核心技术与案例实战[M]. 北京:中国工信出版集团,2017.

[10] 于娜娜, 王中杰. 基于 Spark 的协同过滤算法的研究[J]. 系统仿真技术, 2016, 12(1): 40-45.

[11] 王全民, 苗 雨, 何 明, 等. 基于矩阵分解的协同过滤算法的并行化研究[J]. 计算机技术与发展, 2015, 25(2): 55-59.

[12] HAN Jiawei, KAMBER M. Data mining: concepts and techniques [M]. [s. l.]: Elsevier Science Publishers, 2007.

[13] 杨志伟. 基于 Spark 平台推荐系统研究[D]. 合肥:中国科学技术大学,2015.

[14] 朱郁筱, 吕琳媛. 推荐系统评价指标综述[J]. 电子科技大学学报, 2012, 41(2): 163-175.

[15] 李现伟. 基于 Spark 的推荐系统的研究[D]. 杭州:浙江理工大学,2016.

[16] 岑凯伦, 于红岩, 杨腾霄. 大数据下基于 Spark 的电商实时推荐系统的设计与实现[J]. 现代计算机, 2016(24): 61-69.