Computer Science and Application 计算机科学与应用, 2021, 11(6), 1783-1790
Published Online June 2021 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2021.116184

Hans汉斯

基于矩阵分解的Flink实时推荐策略

谢荣臻，陈源东，白巧娈，罗金炎

闽江学院数学与数据科学学院，福建 福州
Email: 261915575@qq.com

收稿日期：2021年5月26日；录用日期：2021年6月21日；发布日期：2021年6月28日

摘　要

虽然互联网快速进步发展，但也带来了大量的网络数据流，随之而来的是数据的综合存储，数据的综合计算和数据分析等诸多问题，各种业务系统的复杂多样化，数据分析的实效性要求也变得越来越高，先前常用的离线分析很多已经不适用于当今的生产需要，如今对数据的推荐系统在实时性方面有了更高的需求。基于矩阵分解的推荐算法作为目前较为流行的推荐算法，不论从预测的准确度还是预测的精确度都要明显地优于其它的算法。但传统的矩阵分解方法在处理大规模数据时存在计算速度慢和计算资源不足的问题。Flink大数据框架作为当前热门的流数据处理框架，在迭代计算与流数据处理上有明显的优势。本文将矩阵分解方法与Flink处理相结合，在原有的矩阵分解推荐算法的基础上，提出一种基于Flink的矩阵分解算法的优化模型，解决了矩阵分解在大数据环境下的瓶颈。

关键词

Flink，大数据，实时计算，流处理

Flink Real-Time Recommendation Strategy Based on Matrix Decomposition

Rongzhen Xie, Yuandong Chen, Qiaoluan Bai, Jinyan Luo

School of Mathematics and Data Science, Minjiang University, Fuzhou Fujian
Email: 261915575@qq.com

Received: May 26th, 2021; accepted: Jun. 21st, 2021; published: Jun. 28th, 2021

Abstract

Although progress and rapid development of the Internet also brought a lot of network data flow, the following is the comprehensive storage of data, data comprehensive calculation and data analysis and many other problems. With the complexity and diversification of various business

文章引用: 谢荣臻, 陈源东, 白巧娈, 罗金炎. 基于矩阵分解的 Flink 实时推荐策略[J]. 计算机科学与应用, 2021, 11(6): 1783-1790. DOI: 10.12677/csa.2021.116184

systems, the requirements for the effectiveness of data analysis have become increasingly high. In the past, most offline analysis commonly used is no longer applicable to today's production needs. Now the data recommendation system is requested to have a higher demand in real time. As a popular recommendation algorithm at present, the recommendation algorithm based on matrix decomposition is obviously superior to other algorithms in terms of accuracy and accuracy of prediction. However, the traditional matrix decomposition method has the problems of slow computation speed and insufficient computation resources when dealing with large-scale data. As a popular streaming data processing framework, Flink big data framework has obvious advantages in iterative computation and streaming data processing. In this paper, matrix decomposition method is combined with Flink processing. On the basis of the original matrix decomposition recommendation algorithm, an optimization model of matrix decomposition algorithm based on Flink is proposed to solve the bottleneck of matrix decomposition in the big data environment.

Keywords

Flink, Big Data, Real-Time Computation, Stream Processing

1. 引言

“大数据”技术近几年来快速发展，它不仅是一个企业发展趋势，也是一个改变了人类日常生活的重大技术创新。大数据对整个行业以及用户的整合重要性也日益突出，进行高度智能化商业决策，已成为企业脱颖而出的重要关键，越来越多的中国企业已经开始针对企业大数据技术进行企业战略整合布局，重新正确定义自己的企业核心战略竞争力。

在高速发展的信息社会里，科技发达，信息广泛流通，人们之间的信息交流越来越密切，大数据与人们的生活日益紧密地联系在一起，大数据技术就是这个高科技信息时代的必然产物。生活的许多方面都可以涉及应用到企业大数据，这些企业大数据的应用属性都是直接呈现了企业大数据不断变化增长的信息复杂性，所以，大数据的应用分析方法在企业大数据分析领域已经显得尤为重要。简单地说，它是作为决定最终企业信息分析是否有价值的一个决定性考量因素[1]。

2. Flink 的选择

Apache Flink 是大数据时代下的一个的产物——基于流式数据计算模型的实时计算平台。与 Flink 同样优秀的实时计算框架还有 Apache Flink Streaming 和 Apache Storm。在众多优秀的实时计算框架的选择上，阿里巴巴研究员蒋晓伟曾对该问题做出解答：“我们觉得在流处理方面 Flink 在功能，延迟，一致性和性能上综合来看是目前社区最优秀的，所以我们决定采用它来实现流和批的一体化方案。”

Flink 是把批当作一种有限的流，而 Flink Streaming 却与之相反，Flink Streaming 是把流转化成一个个小的批来处理，因此，在流数据处理上，需要的延迟越低，额外的开销就越大，这就导致了 Flink Streaming 很难做到秒级甚至压秒级的延迟。因此如果要用一套引擎来解决流和批处理，那就必须以流处理为基础。

在流式计算上，Flink 的流式处理引擎和 Storm 很相似，然而 Storm 只能处理流数据，没有批处理的能力。在语义处理上，Storm 支持“At Most Once”和“At Least Once”，而 Flink 在此基础上还支持“Exactly Once”。同时，Flink 还提供了很多高级的 API，而 Storm 需要使用者自己去实现这些功能。

因此，在综合推荐系统的实时性、数据的精确性以及计算引擎的易用性等方面而言，Flink 是一个更加合适的选择。

3. 矩阵分解

基于系统矩阵模型分解的综合推荐分析算法，其从本质上来说是一种基于矩阵模型的用户协同特征过滤综合推荐分析算法，它是通过分析提取用户评分系统矩阵分解中的各种影响推荐因素，从而构成特征向量，分解并得出推荐用户间的特征向量和推荐物品间的特征向量，由得出用户和推荐物品间的向量高度交互相关性从而产生综合推荐。该评分方法一方面可以实时得到了评分用户的各个偏好和每件评分物品的各个特性，另一方面可以降低了原有的评分指标矩阵的各个维度。假设用户的物品评分矩阵为 R，矩阵中有 m 个用户和 n 个物品，每个元素表示用户对物品的评分。评分矩阵 R 中有很多缺失值，每位用户只对其中很少一部分的物品有评分[2]，因此，R 是一个非常大的稀疏矩阵。此时需要做的工作是补全这些缺失值。对于评分矩阵的补全，有一个最重要的原则是补全后的值对原矩阵的扰动最小，一般判断的标准是补全后的矩阵和原来矩阵的特征值相差最小。

最早使用的矩阵分解方法是奇异值分解。奇异值分解是线性代数中一种重要的矩阵分解，奇异值分解则是特征分解在任意矩阵上的推广。在信号处理、统计学等领域有重要应用。对于评分矩阵 R，首先，补全 R 中的缺失值，得到 R'；然后，使用奇异值分解方法将 R'；分解为如下形式：

$$R' = U^{\mathrm{T}} \cdot S \cdot V \tag{1}$$

其中，$U \in R^{m \times k}$，$V \in R^{k \times n}$ 是两个正交矩阵，$S \in R^{k \times k}$ 是对角矩阵。

对角线上每个元素的值就是该评分矩阵特征值的平方。对于矩阵 S，保留最大的前 f 个奇异值组成对角矩阵 S_f [3]，就是舍弃了权重较低的奇异值和所对应的特征向量，即去除了一部分噪音，并且保留 f 个奇异值在 U 和 V 中对应的行和列，得到 U_f 和 V_f [4]。将这三个矩阵再次相乘，可得到降维后的评分矩阵 R'_f：

$$R'_f(u,i) = U_f^{\mathrm{T}} \cdot S_f \cdot V_f \tag{2}$$

其中，$R'_f(u,i)$ 是用户 u 对物品 i 的预测评分。

奇异值分解模型的推荐效果优于传统的基于邻域的推荐算法，该模型也是早期推荐系统中矩阵分解算法较为常用的方法[5]。但是该方法存在存储空间的耗费特别大和时间复杂度较高的缺点，以致于很难应用在实际的应用中。

奇异值分解方法的不足有目共睹[6]，为了解决以上问题，许多研究人员提出了各种应对方案。比如，张宇等人提出了基于 MapReduce 的矩阵分解算法，解决了大批量的评分矩阵在多节点上的资源共享问题[7]。但是，该算法使用的 MapReduce 框架，在迭代计算的过程中，节点间的通信和 I/O 操作影响了算法的执行效率[8]；杨洋等人提出了一种基于矩阵分解与用户近邻模型的推荐算法，该算法实现了两种推荐模型的融合，一定程度上提高了推荐系统的推荐准确度[9]。但是，其忽略了物品或用户间的相似性对推荐系统推荐准确度的影响。

综上，本文针对矩阵分解算法展开研究。在原有的矩阵分解模型基础上，做进一步优化，然后，将优化后的矩阵分解模型与最近邻模型融合，从而提出一种基于 Flink 的矩阵分解和 K 近邻融合的推荐算法[10]。

4. 问题分析

在上文对这种矩阵向量分解特征算法的详细介绍中，本文发现在矩阵求解物品用户和其他物品的两个特征向量时，矩阵分解 p 和 q 可能会同时丢失一些物品用户和其他物品的特征信息，比如物品用户间

的特征相似性和用户物品间的特征相似性。在精确求解不同用户或其他物品间的相似度时，采用皮尔逊相关系数和采用评分概率矩阵方法求得的分析结果可能存在一定的概率差异。因此，本文首先引入了一种新的训练目标计算函数，在该目标函数中分别引入了与用户或其他物品间的绝对相似性，保证了对模型进行训练前后相似度的相对一致性。

5. 算法优化

5.1. 具体步骤

5.1.1. 相似度计算

采用皮尔逊相关系数，计算其与虚拟用户或其他虚拟物品间的相似度。计算公式基本列表内容如下：

$$sim(i,j)=\frac{\sum_{c\in I_{ij}}\left(R_{i,C}-R_i\right)\left(R_{j,C}-R_j\right)}{\sqrt{\sum_{c\in I_i}\left(R_{i,C}-R_i\right)^2}\sqrt{\sum_{c\in I_j}\left(R_{j,C}-R_j\right)^2}} \tag{3}$$

皮尔逊相关系数的变化范围为−1 到 1。系数的值为 1 意味着 I_i 和 I_j 可以很好的由直线方程来描述，所有的数据点都很好的落在一条直线上，且两集合呈正相关。系数的值为−1 意味着所有的数据点都落在直线上，两集合呈负相关。若系数的值为 0，那么意味着两个集合之间没有线性关系。

5.1.2. 设计目标函数

为了确保持设计模型运动训练前后练习用户间或训练物品间的相似度计算一致性，本文在分析设计模型目标计算函数时，融合了上式两种计算方法得到的相似度计算结果，目标计算函数定义如下：

$$\begin{aligned} f(U,M)=&\sum_{(i,j)\in I}\left(r_{ij}-u_i m_j^{\mathrm{T}}\right)^2+\sum_{i\in I_j}\left(u_{ki}-\frac{\sum_{u_p\in KNN(u_i)}sim_{u_iu_p}u_{kp}}{\sum_{u_p\in KNN(u_i)}sim_{u_iu_p}}\right)^2 \\ &+\sum_{j\in I_i}\left(m_{kj}-\frac{\sum_{m_q\in KNN(m_j)}sim_{m_im_p}m_{kp}}{\sum_{m_q\in KNN(m_j)}sim_{m_im_p}}\right)^2+\lambda\left(p_u^2+q_i^2\right) \end{aligned} \tag{4}$$

上式中，$(i,j)\in I$ 表示一个训练数据集中的用户评分项，I_j 表示用户集合，I_i 表示物品集合，$KNN(u_i)$ 表示用户 u_i 的前 K 个最近邻居集合，$KNN(m_j)$ 表示物品 m_j 的前 K 个最近邻居集合，$sim_{u_iu_p}$ 表示两个用户间的相似度，$sim_{m_im_p}$ 表示两个物品间的相似度，k 表示某一个物品特征或者的隐含相关因子。

本文在矩阵分解模型的基础上增加了物品间的相似度和用户间的相似度计算，使模型既可以处理用户对物品的评分信息，又考虑了用户或物品间的相似度，有助于推荐准确度的提高[11]。本文在最近邻模型中选取邻居数目时，只选取了前 K 个邻居，一方面是为了减少计算时间，另一方面，大部分用户或者物品的信息包含在前 K 个邻居中，其余邻居对于相似度影响很小。

5.1.3. 求解目标函数

我们通过 Flink 在迭代式计算方面的优势求解特征矩阵 u_i 和 m_j。本文采用交替最小二乘法(ALS)，利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。即，首先固定 u_i 求解 m_j，然后固定 m_j 求解 u_i，依次迭代。具体过程如下：

$$\frac{1}{2}\frac{\partial f}{\partial u_{ki}}=0,\forall i,k\Rightarrow\sum_{j\in I_i}\left(u_i^{\mathrm{T}}m_j-r_{ij}\right)^2+\left(u_{ki}-\frac{\sum_{u_p\in KNN(u_i)}sim_{u_iu_p}u_{kp}}{\sum_{u_p\in KNN(u_i)}sim_{u_iu_p}}\right)+\lambda n_{ui}n_{ki}=0,\forall i,k \tag{5}$$

$$\Rightarrow u_i = \left(M_{I_I} M_{I_I}^{\mathrm{T}} + (\lambda n_{ui} + 1)E\right)^{-1} \times \left(M_{I_i} R^{\mathrm{T}}(i, I_i) + \frac{\sum_{u_p \in KNN(u_i)} sim_{u_i u_p} u_{kp}}{\sum_{u_p \in KNN(u_i)} sim_{u_i u_p}}\right), \forall i \tag{6}$$

以上，迭代计算求得 u_i，同理，对于 m_j：

$$m_j = \left(M_{I_I} M_{I_I}^{\mathrm{T}} + (\lambda n_{ui} + 1)E\right)^{-1} \times \left(M_{I_i} R^{\mathrm{T}}(j, I_j) + \frac{\sum_{m_q \in KNN(m_j)} sim_{m_j m_q} m_{kp}}{\sum_{m_q \in KNN(m_j)} sim_{m_j m_q}}\right), \forall i \tag{7}$$

该算法的主要过程是：通过交替最小二乘方法不断迭代，寻求最优解。若算法达到迭代的次数或者计算结果收敛，可得到用户特征向量 u_i 和物品特征向量 m_j，最终使用特征向量产生推荐结果。

在上述特征算法的分析设计中，本文首先考虑到全面性并考虑到对用户或其他物品的特征相似性，然后将其算法融入应用目标特征函数求解计算中的特征向量，具有偏导性函数的特征求解算法过程复杂，计算量大的缺点。因此，本文通过计算相关用户或其他物品的前后 k 邻和近邻的前后相似性，将相关用户或其他物品的前后相似性进行融合后得到一个损失矩阵函数，在损失矩阵进行分解后，可大大减少相关用户或其他物品相关信息的大量丢失，提高用户推荐管理系统的用户推荐结果准确度。

6. 实证分析

6.1. 数据来源及说明

该数据取自于：https://grouplens.org/datasets/movielens/。

这个数据集(ratings.csv 和 movies.csv)涵盖 9742 部电影。这些数据由 610 名用户在 1996 年 3 月 29 日至 2018 年 9 月 24 日期间创建。该数据集于 2018 年 9 月 26 日生成。用户是随机选择的。所有选定的用户都对至少 20 部电影进行了评分。不包括人口统计信息。每个用户都由一个 id 表示，不提供其他信息。

6.2. 实验涉及

针对本文提出的矩阵分解算法的研究，设计实验对文中提出的矩阵分解优化模型的性能进行验证。本文主要对算法的预测评分准确度进行验证，即计算矩阵分解优化模型、传统的矩阵分解。

本文除了使用 Flink 计算框架之外，Flink 相关的一些组件也有所应用，比如 Flink SQL，在推荐列表的查询中，比较 Flink SQL 查询方案与传统的 MySQL 查询方案，验证查询速度的提升，从而得出在 Flink 框架下一种更加高效的查询方案，应用在推荐系统中有助于提升系统的实时性。

6.3. 测评结果

1) 基于 Flink 的实时推荐系统框架与传统推荐系统框架的模型训练时间对比实验，具体结果如图 1 和图 2 所示。

图 1 可以看出，对于不同的算法类型，基于 Flink 的实时推荐系统框架相比于传统推荐系统框架，其性能都有所提升。因此，在基于 Flink 的实时推荐系统框架中，其性能的提升具有很高的幅度。

图 2 是矩阵分解算法在不同的迭代次数下，Flink 和 Hadoop 两种框架下的模型训练时间对比情况，可以看出，随着迭代次数的增加，Hadoop 框架下的模型训练时间增加明显，而 Flink 框架下随着迭代次数的增大，模型的训练时间相对稳定，其原因在于，HadoopMapReduce 在每次计算之后，需要重新访问 HDFS，增加了数据 I/O 的时间，从而也极大的降低了计算的性能[12]。因此，Flink 是一种极其适合在具有多次迭代计算的数据挖掘的场景中应用，这得益于 Flink 是基于内存计算的，即在计算过程中，所得到的中间计算结果不落地，直接存储在内存中，这样减少了数据的 I/O 操作，从而很大的提升了计算速度，提高了系统的性能[13]。

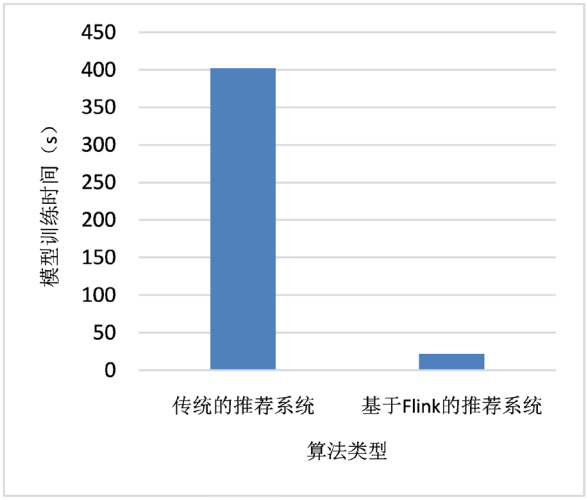

Figure 1. Comparison of model training time
图 1. 模型训练时间对比

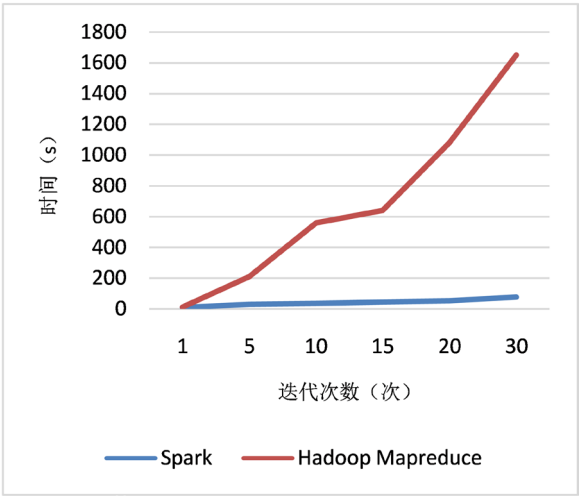

Figure 2. Comparison of training time of matrix decomposition model in different iteration times
图 2. 矩阵分解模型在不同迭代次数下模型训练时间对比

2) 基于 Flink 的实时推荐系统框架与传统推荐系统框架的性能提升对比实验，具体数据如图 3 所示。

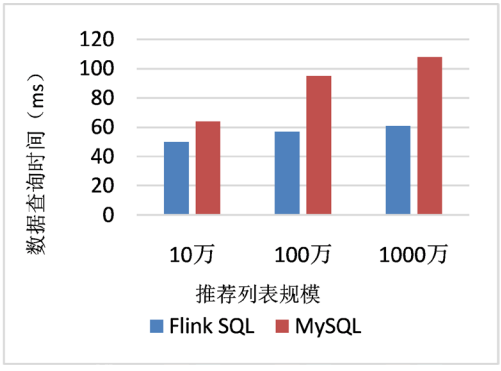

Figure 3. Comparison of Flink SQL and MySQL data query time
图 3. Flink SQL 与 MySQL 数据查询时间对比

由图 3 可以看出，采用 Flink SQL 查询推荐的方式，相比传统的 MySQL 查询方式，极大的提升了查询的速度，也因此极大的提升了推荐系统的实时性。并且可以发现，随着推荐列表规模的不断增大，传统的查询方式的查询时间增加明显，Flink SQL 则增加缓慢，这说明了 Flink SQL 在面对大量数据的查询时仍然具有优异的表现。Flink SQL 具有大量数量快速查询的能力，其主要得益于 Flink SQL 的表数据在内存中存储采用的是内存列存储的方式，这种存储方式不管是在空间的占用量还是在读取吞吐率上都有优异的表现[14]。因此，在实时推荐系统中应用 Flink SQL 查询技术，极大的提高了推荐系统的实时性。

3) 在 Flink 环境下，分别对传统的矩阵分解的推荐算法以及优化后的矩阵分解这两种的评分预测正确度值的 RMSE 值进行对比实验，验证本文中提出的矩阵分解优化算法的性能。实验结果如图 4 所示。

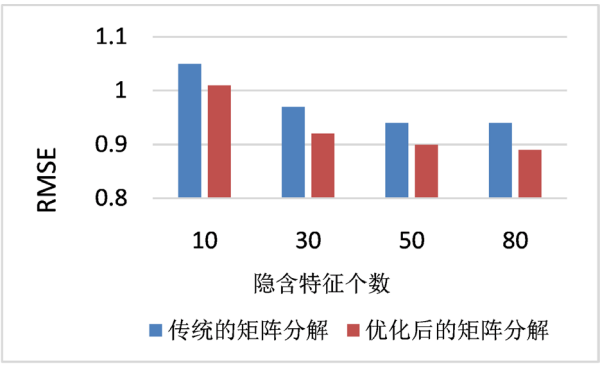

Figure 4. RMSE values of different implicit eigenvalues
图 4. 不同隐含特征值个数 RMSE 值

对于以上两种推荐模型，从图 4 的信息可以优化后的矩阵分解推荐算法在传统的矩阵分解模型的基础上很大程度的提高了推荐的准确度，说明了本文设计的优化模型的在算法的准确度上有所提升，其关键就在于引入了用户和物品的相似度计算；其次，随着隐含特征个数的增加，传统矩阵分解模型的 RMSE 值下降速度变缓，这也从侧面说明，传统的矩阵分解模型每次计算出的结果都是包含了显著特征的用户特征向量。

7. 结论

本章首先简要介绍了各种矩阵中的分解函数模型的基本原理，分析了各种矩阵中的分解函数模型可能存在的不足。针对系统矩阵信息分解系统模型信息丢失影响用户或其他物品信息相似性的复杂问题，本文提出了一种基于 Flink 的矩阵分解优化模型。该综合模型在之前原有的一个矩阵数组分解综合模型的研究基础上，引入了对用户和一个物品的前身和 k 个最近最远邻居的数据相似性综合计算，即是将融合了最多附近邻居的模型引入到矩阵数组分解的综合模型中，增强了推荐系统的推荐准确度。前 k 个最近邻居的选取，在考虑了相似度的同时，减少了计算特征矩阵的时间，有助于提高系统的实时性。

最后设计实验，对文中提出的矩阵分解优化模型的性能以及本文设计与实现的实时推荐系统的性能进行验证。实验结果验证了基于 Flink 平台搭建的实时推荐系统在性能上比传统的 Hadoop 框架好；并且，本文设计 Flink SQL 查询技术的使用，极大地提高了推荐系统的实时性；同时，本文提出的矩阵分解优化模型在推荐准确度、计算速率方面与当前传统的推荐算法相比要有优势。

基金项目

闽江学院校长基金项目(项目编号：103952018234)。

参考文献

[1] 张延彬. 基于移动通信行业的大数据服务研究[J]. 电信工程技术与标准化, 2016, 29(2): 44-47.

[2] 古来, 黄俊, 张若凡, 等. 结合多信息的概率矩阵分解模型[J]. 软件导刊, 2018, 17(9): 67-71.

[3] 翁小兰, 王志坚. 协同过滤推荐算法研究进展[J]. 计算机工程与应用, 2018, 54(1): 25-31.

[4] 孟利民, 赵维, 应颂翔. 评分预测问题中个性化推荐模型的研究[J]. 浙江工业大学学报, 2016, 180(2): 119-123.

[5] 王圣涛, 郝龙飞, 贾洁民. 一种基于 NSGA-II 的协同过滤推荐算法[J]. 电子产品世界, 2016(2): 57-60.

[6] 冯洋. 基于改进的奇异值分解的红外弱小目标检测[J]. 激光技术, 2016, 40(3): 335-338.

[7] 张宇, 程久军. 基于 MapReduce 的矩阵分解推荐算法研究[J]. 计算机科学, 2013(1): 19-21.

[8] 王振军, 黄瑞章. 基于 Spark 的矩阵分解与最近邻融合的推荐算法[J]. 计算机系统应用, 2017, 26(4): 124-129.

[9] 谢人强, 陈震. 基于共同评分项和权重计算的推荐算法研究[J]. 计算机技术与发展, 2016, 26(9): 69-72.

[10] 李昆仑, 郭昌隆, 关立伟. 一种融合近邻用户影响力的矩阵分解推荐算法[J]. 小型微型计算机系统, 2018, 39(1): 37-41.

[11] 任彩霞. 一种改进的缓解推荐系统物品冷启动的方法[J]. 软件, 2016(8): 11-15.

[12] Yazidi, A.E., Azizi, M.S., Benlachmi, Y., *et al.* (2021) Apache Hadoop-MapReduce on YARN Framework Latency. *Procedia Computer Science*, **184**, 803-808. https://doi.org/10.1016/j.procs.2021.03.100

[13] 包维宁, 任钦正, 李瑞明, 等. 一种基于 Flink 的日志流式处理方法及系统[P]. CN111177193A. 2020.

[14] 杰诚, 郑少明, 郑乐乐, 等. 一种基于 Flink SQL 的数据处理方法, 装置, 存储介质[P]. CN111026779A. 2020.