**中国科学技术大学**

**工程硕士研究生学位论文**

**开题报告**

**论文题目：基于Spark的协同过滤推荐系统研究与实现**

**学 生 姓 名： 张宁**

**学 校 导 师： 戴礼荣**

**企 业 导 师： 程宁**

**工 程 领 域： 软件工程**

**领 域 代 码：** 430113

**研 究 方 向： 移动软件设计**

**所 在 院 系： 软件学院**

**实 习 单 位：平安科技(深圳)有限公司**

**中国科学技术大学研究生院**

**填表日期： 2017 年 03月 02 日**

**说 明**

1. **工程硕士学位论文的开题报告是保证论文质量的一个重要环节，为了加强对工程硕士研究生培养的过程管理，规范其学位论文的开题报告，特制此表。**
2. **工程硕士学位论文开题报告，应该在工程硕士学位授予点或培养单位组织的学术报告会上报告，听取意见，论证后再填写此表。**
3. **此表一式两份经导师和培养单位负责人签字后，交培养单位研究生教学管理办公室存档。**
4. **工程硕士研究生在申请学位论文答辩时，必须提交该学位论文开题报告。**

**一、简况**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **研究生简况** | **学 号** | | **SA15226444** | | | | | **姓 名** | **张宁** | | **姓名拼音** | **ZHANGNING** |
| **性 别** | | **男** | | | **身份证号** | **370305199009130710** | | | | **出生年月** | **1990.09.13** |
| **工程领域** | | **软件工程** | | | | | | **研究方向** | | **软件系统设计** | |
| **入学时间** | | **2015.09.01** | | | | **录取方式** | | **统考** | | **培养方式** | **脱产** |
| **本科毕业**  **时间** | | | **2013.06** | | | **本科毕业**  **学校** | | **枣庄学院** | **本科专业** | | **电子信息工程** |
| **论文类型与性质** | **名称** | **中文** | | | **基于Spark的协同过滤推荐系统研究与实现** | | | | | | | |
| **英文** | | | **Research and implementation of collaborative filtering recommendation system based on Spark** | | | | | | | |
| **类别** | **1．技术攻关研究 2．工程项目策划**  **√3．工程设计或技术改造 4．新工艺、新材料、新设备、新产品的研制与开发** | | | | | | | | | | |
| **形式** | **√1．工程设计**  **2．研究论文** | | | | | | | | | | |
| **性质** | **√1．****应用技术研究**  **√2．技术开发** | | | | | | | | | | |
| **论文内容和意义** | **摘 要** | 当前我国已经进入信息化时代，信息内容也更加丰富。然而信息技术也带来了一些负面问题，例如“信息过载”现象的出现严重干扰了人们的正常生活。大量的信息也给人们的生活带来了一定的困扰。无论是用户要想从海量数据中摆脱出来，还是商家想要引领消费者进行消费，都离不开个性化推荐系统。为了更好的解决“信息过载”问题，推荐系统的概念应运而生，并且已经得到了越来越多的关注，相关的专家学者们对其研究也更加深入。因此，如何解决大数据时代的实时准确推荐是一个非常重要的课题。  本课题研究的内容包括以下几个方面：（1）针对现有协同过滤算法存在的问题，提出改进的项目相似度计算算法，用以更好的进行推荐。（2） 基于多种推荐引擎的混合推荐。（3）引入spark大数据计算框架，设计推荐系统相关模块，尽可能做到快速实时推荐。（4）通过实验验证算法的有效性。 | | | | | | | | | | |
| **主题词** | **主题词数量不多于三个，主题词之间空一格（英文用“/ ”分隔）** | | | | | | | | | | |
| **中文** | | | **推荐系统 Spark 计算框架 信息过载** | | | | | | | |
| **英文** | | | **Recommended system/ Spark/ computing framework/ information overload** | | | | | | | |

**二、选题依据**

|  |
| --- |
| 1. **本课题研究意义及同类研究工作国内外研究现状及分析（引用的参考文献需在文中标注）。**   **1.1选题的研究意义**  当前我国已经进入信息化时代，信息内容也更加丰富。然而信息技术也带来了一些负面问题，例如“信息过载”现象的出现严重干扰了人们的正常生活。大量的信息也给人们的生活带来了一定的困扰。无论是用户要想从海量数据中摆脱出来，还是商家想要引领消费者进行消费，都离不开个性化推荐系统。为了更好的解决“信息过载”问题，推荐系统的概念应运而生，并且已经得到了越来越多的关注，相关的专家学者们对其研究也更加深入。因此，如何解决大数据时代的实时准确推荐是一个非常重要的课题。  推荐系统的出现则在一定程度上弥补捜索引擎的缺陷[1]。推荐引擎用于处理信息过载的情况，推荐系统通过分析日志数据、用户历史行为、用户属性从而创建用户兴趣偏好模型，依靠推荐算法预测用户对商品权重偏好并进行排序。运样在用户没法准确定位需求描述的同时，推荐系统向用户推送满足需求与兴趣的推荐列表。因此二者为互补的工具，当用户有明确需求时采用捜索引擎满足要求，而在用户没有明确自身需求或者以新成员身份进入到一个新的环境中时采用推荐系统帮助他们获取到感兴趣的内容。因此推荐系统在电子商务平台上获取广泛的成功以及巨大的市场空间。然而如何填充用户商品稀疏矩阵、如何推送高质量推荐列表、快速地响应推荐请求、实时生成推荐列表、海量数据分析都成为必须面对和解决的难题[19]。  随着购物网站规模的不断发展，用户和项目的数量不断增多，这也就导致了用户项目评分阵的稀疏性问题越来越无法避免，协同过滤算法中的数据稀疏性过于严重，推荐的结果更加不准确，从而影响用户对购物网站的信任，造成用户的流失。并且在协同过滤的推荐算法中，往往不能很好的展现用户的爱好迁移的问题，因此也会对推荐系统的性能造成一定的影响。所以，缓解数据稀疏性问题和适应用户喜好化问题已经成为了网络购物系统发展的严峻考验之一。  在新一代推荐系统中为了解决如上难题以及保证系统的可扩展性，需要存储不断增长的数据集以及快速并行化处理、加速建模阶段。常用的技术主要包括[4]：大数据存储技术以及大数据处理技术。推荐算法主要特点在于多个并行操作之间重用数据工作集，属于非循环应用模型的应用，因此引入SPARK计算框架[21]。SPARK是由UC Berkeley推出的大数据处理框架，它提出分布式内存抽象，称为弹性分布式数据集RDD，它允许在计算过程中将中间工作集缓存在内存中，而在后续操作中重用该工作集，运样可以极大加速处理过程。由于大部分数据挖掘算法设及到数据集迭代计算，因此SPARK框架更符合推荐算法的需求，本课题利用改进后的协同过滤算法和用SPARK大数据处理技术实更快、更准确的推荐平台。  **1.2国内外概况和发展趋势**  **1.2.1 spark发展现状**  Spark发展成立于2012年，以利用当时的五年WordPress经验。 近四年半之后，Spark与广泛的客户合作，从个人和中小企业到跨国公司和政府机构。在2015年底之前，Apache Spark已经成为Hadoop生态系统中最为活跃的项目。作为2009年加州大学伯克利分校AMPLAB研究项目的一部分，Spark与Apache Hadoop并仍在积极发展。 2015年有三个版本，从1.3到1.5，分别包含DataFrames API，SparkR和Project Tungsten等功能。版本1.6在2016年初发布，包括新的Dataset API和数据科学功能的扩展。 Spark 2.0于2016年7月发布，这是一个主要版本，有很多新功能和增强功能，值得拥有自己的一部分。  目前，在Spark核心框架基础上，其主要提供了四个范畴的计算框架其中包括Spark Streaming,Graphx,MLbase,SparkSQL等，其中Spark Streaming是Spark核心API的一个扩展，它对实时流式数据的处理具有可扩展性、高吞吐量、可容错性等特点。Spark Streaming 可以从各种数据源中获取数据并进行复杂计算后推送到文件系统、数据库等存储介质保存。Spark Streaming接收实时的输入数据流，然后将这些数据切分为批数据供Spark引擎处理，Spark引擎将数据生成最终的结果数据。Graphx主要并行图计算，Spark Graphx性能良好，又具有非常丰富的API,并可以基于海量数据进行复杂的图计算从而得到了较为广泛的应用。MLbase 作为Spark生态系统中的一部分，由AMPLab实验室的Mlbase团队开发[8]，是Apache Spark中的核心机器学习库，专门提供机器学习相关的丰富的API。从而让机器学习的门槛更低，从而使那些不了解机器学习的开发人员也能通过MLbase来处理自己的数据。Spark SQL的引入专门用于处理结构化数据比如（DB,JSON），Spark SQL 不仅是SQL,它对各种数据源做了抽象，能够对各种数据源进行ETL操作，并提供丰富的对复杂数据读写的API,使用非常方便，并对多种数据源都提供了很好的支持。其中DataFrame API作为一种具备高伸缩性的数据操作方式，相比RDD API 效率更高。此外Spark SQL还提供了Catalyst Optimizer查询优化器，可以自动优化程序，使得程序更加高效。  Spark引入的弹性分布式数据集RDD，是一种高度受限的共享内存模型，是基于内存的。 这使得其计算效率和速度相比Hadoop 的 MapReduce （基于磁盘，每次执行时都要从磁盘读取数据，计算完毕后都要把数据存放到磁盘上）提升很多。  Apache Spark是围绕速度、易用性和复杂分析构建等方面的大数据处理框架，适用于多种不同的情形的分布式应用场景，包括批处理，流处理，交互式查询等等，并大大减轻了原先需要对各种平台分别管理的负担。统一的RDD抽象和操作，基于统一的技术堆栈，使得Spark称为大数据通用计算平台。在大数据时代逐渐普及的背景下得到了越来越广泛的应用。    1.**2.2 推荐系统研究现状**  推荐系统的研究至今已有约20多年的历史，其最早期的研究源自于美国明尼苏大学GroupLens[3]研究小组对一个叫MovieLens的电影推荐系统的研究。研究内容主要是对网站用户打分情况进行分析，从而得到用户可能感兴趣的电影，从而进行推荐。早期的算法主要是基于内容的推荐，通过对推荐物品本身进行特征提取。该方式会导致如果用户没有对某类产品评价，该类产品就永远不会被推荐。随着时间推移，旧的推荐系统问题逐渐暴露，新的推荐算法应运而生。  协同过滤推荐（Collaborative Filtering，简称CF）起源于1992年。从算法理论方面来说，最早期的个性化推荐算法是GroupLens小组提出的协同过滤算法(即基于用户的协同过滤推荐算法)，至于为止该算法在推荐领域仍占有一席之地。基于该算法，人们陆续提出了基于物品的协同过滤推荐算法，基于模型的协同过滤推荐算法等等。经过近20多年的发展，当前推荐系统衍生出多种多样的推荐方式[5]。一、基于上下文的推荐，即推荐系统会根据用户的访问时间、所在城市、天气、季节等因素研究给用户生成TopN推荐，提高推荐效果[14]。二、基于用户画像的推荐，所谓用户画像，就是想用户的所有特征信息标签化，是对用户相关联数据可视化的一种展现形式。多角度抽象用户信息，从而实现个性化推荐系统等。三、基于协同过滤的推荐系统，其中有分为两大类，其一是基于内存的协同过滤推荐算法，包括基于用户的协同过滤推荐算法和基于项目的协同过滤推荐算法等等，其二是基于模型的协同过滤推荐算法，即利用数学建模的手段，评估你的意图，包括基于贝叶斯的协同过滤推荐算法、基于隐语义的协同过滤推荐算法和基于线性回归的协同过滤推荐算法等等[18]。四、基于内容的推荐、即对数据内容进行分析并得到物品的结构化描述信息，分析用户在过去时间中评分活评论的商品，利用用户画像构建预测模型，从而生成推荐策略。五、基于机器学习的推荐系统、根据数据的多样性选取合适的算法策略，自动的归纳逻辑和规则，并根据归纳总结的结果(模型)来对未来数据进行推荐。除此之外，还有基于关联规则的推荐、基于效用的推荐、基于知识的推荐等等。但是，上述每种单一方式的推荐系统都有各自的缺点，比如基于内容的推荐有严重的矩阵稀疏问题，基于关联规则的推荐较为耗时只能做离线处理等等。  由于各种推荐方法都有优缺点，因此目前主流推荐系统都是以上各个策略的混合，研究和应用最为广泛的是内容推荐和协同过滤推荐的组合。尽管从理论上有多种组合推荐的方式，但并非对于某一具体问题一定有效，组合推荐中最重要的原则是通过各种策略的组合，使得各自的推荐技术能够相互弥补，从而实现精准高效的推荐[11]。  **1.2.3传统协同过滤算法的分类**  根据协同过滤算法在实现过程中的策略不同，即对用户项目评分矩阵的处理方法不同，可以将其分为memory-based协同过滤算法和model-based协同推荐算法[6]。具体分类如图2-1所示    图1-1 协同过滤推荐算法的分类  1.2.4数据稀疏性问题及研究现状  1) 数据稀疏性问题  数据稀疏性问题协同过滤算法中面临的最为严峻考验之一，也是致使推荐效果严重下降的原因所在。所谓的数据稀疏性问题是指，用户已经评分的项目相对于用户和项目总量来说很少，在用户项目评分矩阵中有很多的空值[14]。由于矩阵过于稀疏，许多用户活着项目之间交集很小，严重影响到相似度的计算准确度甚至出现无法计算，从而导致最近邻居集合的准确性难以保证，最终导致推荐的精准度急剧下降。数据稀疏度就是评分矩阵中的空值的数量和矩阵中总大小的比值，其计算公式如下：    在现实的网上购物系统中，数据稀疏性问题往往是十分严重的的。随着新用户和新物品的不断加入，而用户已经评分的项目往往是十分有限的，就会出现越来越严重的稀疏性问题。据相关研究表示用户的评分数量通常会很少，稀疏度往往能到达97%，严重影响推荐系统的性能，所以减少数据稀疏性问题是现在面临的严峻课题之一[9]。  2) 数据稀疏性问题研究现状  （一）矩阵填充技术  目前，改善数据稀疏性问题最常见的方法就是对用户项目评分阵进行填充[7]。最简单的方式就是将所有没有尽兴评分的的项目直接指定为一个默认值或者设为该项目或者用户的平均值。然而，用户的兴趣并不是一样的，简单的进行填充忽略了用户之之间的差异，丧失了用户的个性化，从而影响预测结果，所以这种填充方法也就不能从真正的解决数据稀疏性带来的问题。  （二）融入信任网络的协同过滤算法  常见的改善数据稀疏性问题的方式还有融入信任网络的协同过滤算法。由于在现实生活中，用户往往存在着一定的社会关系网络，如同学，朋友，亲人等。融入信任网络的协同过滤算法，在计算过程中不单纯的依靠余弦相似度、Jaccard相关系数[8]等方式获取用户之间的相似度，还要融入用户的社会关系网络[4]。近年来，这种算法不断的涌现，其中，Yang等人通过对用户建立信任网络，并对信任网络融入到一个图模型中，并利用随机游走算法对信任关系进行传递，从而获取用户彼此之间的信任关系，并基于这种信任关系与传统相似度融合对未评分的项目进行估分预测。  （三）数据降维技术  还有一种解决数据稀疏性问题的方法就是通过一定数据降维技术，如奇异矩阵分解，对用户评分阵进行降维。通过降低用户评分阵的维度，对那些不重要的信息或者噪音进行删除。将用户、项目转换到一个低维的空间上，然后计算彼此之间的相似度关系，从而对使推荐结果得到提升[2]。目前比较流行的降维技术有奇异值分解（Singular Value Decomposition，SVD）、主成分分析（Principle Component Analysis, PCA）等。但是，在SVD技术分解的过程，两个低维矩阵的维数k越大，保存的信息越多，但是那样就失去的降维的意义；但是如果设置的维数k过小，就会造成信息缺失的现象。所以在实际的降维处理，维数k需要进行权衡。通过不断地进行实验，来找到最合适的k值[11]。  **1.3选题的先进性和实用性**  1)基于Spark环境设计并实现了一个高性能的数据仓库，该仓库使得各个推荐引擎可以高效的读取数据，从而加快了离线计算和在线计算的速度。  2)设计一个混合推荐模型以统一各推荐引擎的结果，并能够根据用户的选择自动调整各引擎的权重。从而实现更加个性化的推荐。  3)推荐系统采用Scala语言，充分利用了Spark的优势，能够分布式运行，从而系统扩展性更好，稳定性更强。  **1.4主要参考文献（列出作者、论文名称、期刊名称、出版年月）**   1. Liu J G, Zhou T, Wang B H. Research progress of personalized recommendation system[J]. Progress in Natural Science, 2009, 19(1): 1-15. 2. Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. ACM, 2001: 285-295. 3. 高建煌.个性化推荐系统技术与应用[D].中国科学技术大学,2010. 4. Ekstrand M D, Riedl J T, Konstan J A. Collaborative filtering recommender systems[J]. Foundations and Trends in Human-Computer Interaction, 2011, 4(2): 81-173. 5. Goldberg D, Nichols D, Oki B M, et al. Using collaborative filtering to weave an information tapestry[J]. Communications of the ACM, 1992, 35(12): 61-70. 6. Gong S J, Ye H W, Tan H S. Combining memory-based and model-based collaborative filtering in recommender system[C]//Circuits, Communications and Systems, 2009. PACCS'09. Pacific-Asia Conference on. IEEE, 2009: 690-693. 7. 面向稀疏性数据的协同过滤算法研究与实现[D]. 北京邮电大学,2013 8. Ghazarian S, Nematbakhsh M A. Enhancing memory-based collaborative filtering for group recommender systems[J]. Expert Systems with Applications, 2015, 42(7): 3801-3812. 9. Chang N, Terano T. Improving the Performance of User-Based Collaborative Filtering by Mining Latent Attributes of Neighborhood[C]// International Conference on Mathematics and Computers in Sciences and in Industry. IEEE, 2015:272-276. 10. Koychev I, Schwab I. Adaptation to drifting user’s interests[C]//Proceedings of ECML2000 Workshop: Machine Learning in New Information Age. 2000: 39-46. 11. Bellogín A, Castells P, Cantador I. Neighbor selection and weighting in user-based collaborative filtering: a performance prediction approach[J]. ACM Transactions on the Web (TWEB), 2014, 8(2): 12. 12. Li C, Zhao P, Wu J, et al. An Item-Based Collaborative Filtering Framework Based on Preferences of Global Users[C]//Proceedings of the 4th International Conference on Computer Engineering and Networks. Springer International Publishing, 2015: 1113-1120. 13. Hu W H, Yang F, Feng Z W. Item based collaborative filtering recommendation algorithm based on MapReduce[J]. Multimedia Communication and Computng Application, 2015: 95-100. 14. 蔡雄峰. 一种缓解协同过滤算法的数据稀疏性的方法[D].东南大学,2015. 15. 刘晓琳. 基于协同过滤及关联规则的混合推荐算法研究[D]. 西安电子科技大学, 2014. 16. De Campos L M, Fernández-Luna J M, Huete J F, et al. Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks[J]. International Journal of Approximate Reasoning, 2010, 51(7): 785-799. 17. Ma Z, Yang Y, Wang F, et al. The SOM Based Improved K-Means Clustering Collaborative Filtering Algorithm in TV Recommendation System[C]//Advanced Cloud and Big Data (CBD), 2014 Second International Conference on. IEEE, 2014: 288-295. 18. 李文栋. 基于Spark的大数据挖掘技术的研究与实现[D].山东大学,2015. 19. 查九. 基于协同过滤技术的个性化推荐算法研究[D].上海大学,2015. 20. 戴春娥. 基于GPU和分布式CPU的协同过滤推荐算法加速技术研究[D].华侨大学,2015. 21. 杨志伟. 基于Spark平台推荐系统研究[D].中国科学技术大学,2015. |

**三、课题主要内容及具体方案**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1. 课题内容**  本课题研究的基于大数据处理技术的协同过滤推荐系统，一方面针对数据稀疏性的问题改进现有的协同过滤算法，增强矩阵语义。另一方面利用SPARK大数据技术，实时处理用户数据，能够做到实时推荐。主要研究内容包括：  （1）针对现有协同过滤算法存在的问题，提出改进的项目相似度计算算法，用以更好的进行推荐。  （3）结合时间因素，提出混合的协同过滤推荐算法。  （4）引入SPARK大数据计算框架，设计推荐系统相关模块，在大数据时做到实时推荐。  （5）通过实验验证算法的有效性。  **2、系统需求分析(应用软件工程专业描述工具描述)**  **2.1 概述**  近几年来，个性化推荐技术已经被成功的应用于电子商务、视频音乐网站、社交、个性化阅读等多个领域，成为人们日常生活中不可或缺的一部分，国内具有代表性的有京东商城、豆瓣、百度、腾讯视频等。广泛的市场应用需求推动了推荐技术的快速发展，如今的推荐技术主要包括基于内容的推荐、基于关联规则的推荐和协同过滤推荐。其中协同过滤推荐是研究最广泛，应用最成功的一种推荐技术，但是仍然存在预测准确度不理想、冷启动的问题。本课题研究的基于大数据处理技术的协同过滤推荐系统，一方面针对数据稀疏性的问题改进现有的协同过滤算法，增强矩阵语义；另一方面利用SPARK大数据技术，实时处理用户数据，能够做到实时推荐。  **2.2功能需求**  **2.2.1 高性能数据仓库**  数据仓库是面向主题的数据集合，通常供数据分析使用。本文的数据仓库搭建在HDFS之上，底层使用Parquet文件类型，并采用Snappy压缩，上层由Spark SQL查询，本系统从仓库的主题来说主要有两部分构成：一是经由原始日志计算得来的初始数据仓库，数据层次上属于原始数据层。而是将各推荐引擎离线部分计算的结果存入数据仓库，供在线计算部分实时推荐使用。数据层次上属于离线中间层。  Spark SQL 和Parquet列式文件类型具有很高的性能，因此本文实现的数据仓库效率很高，远远超过当前广泛使用的HIVE.更为重要的是它能够与Spark编程模式兼容，能够很好的和推荐引擎在线计算模块相配合。  **2.2.2 推荐引擎组**  本系统实现了三种推荐引擎，每种推荐引擎都有其适用场景，系统会根据不能的场景选择具体的推荐引擎进行推荐。  1、基于SVD的推荐  在评分系统中，如果建立一个二维评分矩阵，则这将是一个巨大的矩阵，假设用户有10000个，而物品也有10000个，那么这个矩阵的规模就是10000\*10000=1亿。对于大型电商动辄数以亿计的用户及物品而言，这个矩阵的规模更是无法衡量，更重要的是，该矩阵很可能非常稀疏。例如Netflix数据集中非零元素个数只占所有元素个数的1%左右，这种稀疏性给存储和计算带来了巨大的挑战。  奇异值分解(SVD)是一种分解技术。假设A是一个m行n列()的矩阵，则对A进行奇异分解如下：  (公式2.1)  其中∑是一个m\*n的矩阵，除了对角线的元素外，其他的值都是0，而对角线上的非零元素成为奇异值。令s=(), s中的每一个元素为一个奇异值，且按由大到小排列，则一般s中元素会减小的很快，当我们去前r个奇异值进行降维时，即得到：  (公式2.2)  其中r远远小于m和n，这样 , , 三个矩阵的计算会比基于A的运算快很多。  基于SVD的推荐就是先将评分矩阵进行SVD分解，选取适当的奇异值个数r进行低阶近似，得到，,三个矩阵，然后基于这三个矩阵进行推荐。  基于SVD的推荐本质上还是利用用户行为数据，只是在推荐前后先离线对评分矩阵进行分解，然后进行降维，从而大大减小存储的数据量和在线计算量。但它也有以下两个主要缺点：  1) 降维时r维不好确定。r太大，则降维效果不好，r太小又会损失太多信息。在实际使用过程中往往根据奇异值的衰减程度确定r值。  2）SVD本身计算量就非常大，当数据量比较大时，消耗的时间会难以忍受。不过现在Spark已经实现了SVD的并行化计算，当r值选取的较小时，Spark的计算速度还是比较快的。  2、基于人口统计学的推荐  基于人口统计学的推荐是利用人的静态描述信息进行推荐的方法。它简单的根据人口统计信息发现用户的相似度，然后给当前用户推荐与他相近的人喜欢的物品。  具体来说，该方法首先会利用用户的统计信息，如年龄，性别，住址等，然后采用特定的相异度计算策略，计算用户之间的距离。实际计算时用户属性类别往往是不同的，例如性别是二元属性，年龄是序列属性，职业是名称属性等。假设数据集含有p个混合属性，对于对象i和j，我们采用下面的公式计算它们的距离：  (公式2.3)  公式中是属性f的指示符，是含有属性f的记录i和j的相异度。  如图2-1所示，左边是用户甲、乙、丙的信息，根据公式（2.3），用户的属性均是名称类或者二值类属性，计算后得到的用户丙和用户甲的距离为0，而和乙的距离为1，因此用户丙和用户甲更加相似，故我们就能将甲喜欢的东西向用户丙推荐。这里的邻居数只有一个，在实际应用中，可以通过调整邻居数目进行覆盖率更高的推荐。    图2-1基于人口统计学的推荐  3、基于内容的推荐  基于内容的推荐是利用物品的内容信息，先算出物品的相似度，然后根据用户过去喜欢的物品，来向用户推荐与这些记录相似的物品。物品间相似度计算方法很多，这里采用Jaccard公式：  (公式2.4)  或者采用简化的余弦相似度公式：  (公式2.5)  公式（2.4）和公式（2.5）中的表示电影u与电影v的相似度，表示电影u的内容，表示电影v的内容。  如图2-2所示，电影A类型为“爱情，浪漫”，电影B类性为“恐怖，惊悚”，电影C类性为“爱情，浪漫”使用公式2.5计算的到电影A与B的相似度为0，电影A与C的相似度为1，用户甲喜欢电影，故系统会将C推荐给甲。    图 2-2基于内容的推荐  **2.2.3 结果处理模块**  本系统采用加权并行混合模型，各个推荐引擎都有自己的输出，因此结果处理模块就是将推荐引擎组的输出以一定规则统一起来，以达到准确率更高、更多样化的推荐。结果处理模块包括两种：其一是评分预测的结果处理，这一部分相对来说比较简单，主要各个引擎输出的评分以一定的权重线性组合起来即可；其二电影列表的结果处理，这一部分较为复杂。如图2-3所示，各推荐引擎结果输出后，先后经过过滤，排名，推荐解释，结果生成几个部分，其中结果生成部分比较简单，下面介绍其他三个模块。  1) 过滤模块。得到推荐引擎输出的结果后，系统并不是直接将电影列表输出给用户，而是先要进行过滤。过滤模块就是将一些不符合要求的结果过滤掉。例如用户已经看过的电影，或某些普通评价很差的电影。  2) 排名模块。过滤模块输出结果后往往还需要根据一些性能指标进行排名。如根据准确度排名，从而将更好的结果记录排在前面，供用户选择。  3) 推荐解释模块。推荐解释就是注明每一条推荐记录的由来，比如经过人口统计学引擎推荐的记录，其推荐解释可以是“年龄在20~25之间，性别为女的用户，往往喜欢看这部电影”，推荐解释非常重要，用户会了解到为什么系统会给他推荐这部电影，从而提高了接受度。    图2-3推荐结果处理模块  **3.3系统概述**  **3.3.1基于Spark的系统架构**  本系统采用Scala语言，基于Spark实现。Spark是一个分布式的大数据计算框架，其运行架构由Driver和Executor构成。对于每个推荐引擎来说，Spark SQL作为数据仓库的接口，从数据层次的角度来说负责量方面的任务：一是对于离线计算，Spark SQL将原始数据层的记录从仓库中取出，抽象成RDD，供推荐系统离线计算使用。在计算完成后，Spark SQL会将离线中间层数据写会到数据仓库中。二是在线计算Spark SQL负责将离线中间层的记录从仓库中取出，载入内存后供在线推荐使用。Spark Driver负责生成任务的DAG调度，并将任务分配由Executor执行。  ../Downloads/scala%20(1).png  图3-1 基于Spark的设计架构  **3.3.2分布式日志框架**  基于Spark平台的分布式日志框架如图3-2所示，该框架用于采集访问各个业务系统的日志信息，通过可靠的消息传送中间件作为模型训练与数据源采集之间的纽带，系统构建了基于Kafka集群的消息分发中间件，实现了日志数据的统一下发。由于日志数据中包含着各个业务系统的日志以及用户点击流的日志，在进入离线或实时推荐节点前，需要进行统一的数据清洗。本系统采用Spark Streaming技术按时间分片，对固定时间间隔内收到的数据进行统一的批处理，能达到实时处理的效果，并具有很高的吞吐量。  ../Downloads/协同过滤算法流程图.png  图3-2 基于Spark分布式日志框架  数据收集主要会经历如下阶段：日志数据采集，日志数据聚合和日志数据过滤。  1)分布式日志实时采集  实时推荐系统需要大量的隐式的用户行为作为基础数据,而且没种用户的行为的源日志信息分布在不同的业务系统中，需要构建分布式日志汇总系统将日志进行收集。以备后续流程使用。分布式日志采集如图3-3所示    图3-3 分布式日志采集模块  2)基于Kafka集群的数据传输  通过构建分布式日志实时采集模块，完成用户行为日志的采集。但是在进入日志过滤阶段之前，由于日志流并发产生且数据量大，为了保证数据的实时性以及尽量减少数据丢失，使用分布式消息订阅和发布系统-Kafka,图3-4 为基于Kafka集群的数据分发架构。    图3-4基于Kafka集群的数据传输  **3.3.3推荐系统架构设计**  整个系统由数据仓库模块、推荐引擎组、数据收集模块、评估模块、用户反馈处理模块、引擎管理模块和用户交互模块构成，其中推荐引擎组是整个系统的核心，每个引擎都有自己的推荐策略；结果处理模块对引擎组输出结果按照权重统一起来，然后将最终结果展示给用户；评估模块对推荐引擎组的每一个引擎进行各指标的评估（如准确度，多样性等），以确定各引擎的适用场景，以及最终的权重。引擎管理模块则根据评估模块的结果，动态的增加、删除推荐引擎，更改各引擎的权重。    图3-5推荐系统整体架构  **3.3.4推荐引擎组**  本系统在线计算部分与具体的推荐目标相关，需要在离线计算的基础上，以最快的速度给出推荐结果。在线计算部分计算量不是很大，数据量也不多，引入spark后，先将输入载入内存，来提高推荐速度。在线计算的结果就是推荐引擎的输出。在经过一系列处理后，就可以返回给前台页面展示。    图3-6推荐系统整体架构  **3.4拟采用的开发方法、环境、测试方案等**  **3.4.1开发方法**  系统开发拟采用面向对象的分析及设计方法，开发平台采用Scala函数式编程，服务器集群采用Spark集群。  **3.4.2 开发环境**  本系统在具有3个节点的spark集群上运行，每个节点配置玩完全相同，所需要的实验环境及使用工具如表1所示：  **表1 开发环境及使用工具**   |  |  | | --- | --- | | **项 目** | **开 发 工 具** | | 开发平台 | Macos | | 设计工具 | OmniGraffle、ProcessOn | | 开发工具 | Intellij IDEA 、Eclipse | | 开发语言 | scala、java | | 数据库 | Hbase、Redis |   实验主要的软硬件环境如表2所示：  **表2 实验的软硬件配置**   |  |  | | --- | --- | | **项 目** | **开 发 工 具** | | CPU | 8核 2.40GHZ | | 内存 | 12GB | | 硬盘 | 1TB | | 操作系统 | Centos 6.5 | | Spark版本 | spark1.6 |   **3.4.3 测试方案**  基于spark的推荐系统进行试验设计和相关测评，在MovieLens数据集下进行相关测试，主要考虑推荐系统的以下指标：   1. 各推荐引擎的离线及在线计算速度。 2. 各推荐引擎的测评数据对比。 3. 推荐引擎组合在测评上是否会由于单个引擎。     **3.5. 技术难度及特色分析**  1)设计一个混合推荐模型以同意各推荐引擎的结果，并能根据用户的选择自动调整各引擎的权重，从而实现更加个性化的推荐。  2)推荐系统整体框架设计、 采用Scala语言，充分利用spark的优势，能够分布式运行，提高系统可扩展性，稳定性，以及计算速度。  **3.6、本人主要工作描述**  1)研究并改进协同过滤混合推荐算法，并进行相关指标的测评。  2)实时的数据日志采集系统，以及对于高性能数据仓库的设计。  3)推荐引擎组选取，以及根据具体场景选择不同的推荐引擎。  4)对推荐结果处理，评测。 |

**四、工作进度的大致安排**

|  |
| --- |
| **应包括文献调研，工程设计，项目开发和调试，实验数据的分析处理，撰写论文等。**  2016.11～2017.01 文献调研，了解课题相关内容和国内外研究状况，确定论文方向和选题。  2017.01～2017.03 掌握项目各模块具体的原理以及自己相关工作的重点和难点，熟悉系统架构的每个细节，编写开题报告，完成论文开题。  2017.03～2017.05 搭建系统开发环境，学习项目各模块所涉及的的技术，熟悉系统架构的每个细节。  2017.05～2017.07 根据设计文档完成自己相关模块的工作的详细设计，完成代码框架和各个模块的代码编写。  2017.07～2017.08 各个模块代码的整合，并完成整个系统的测试。同时整理项目的相关文档，完成论文初稿。  2017.08～2017.09 修改和完善研究生毕业论文，完成论文定稿。  2017.09～2017.10 总结研究过程，制作答辩PPT，完成毕业答辩。 |

**五、预期成果**

|  |
| --- |
| **应包括软硬件产品、文档、模型、专利、论文等**   1. 完成基于SPARK大数据处理的系统过滤推荐系统 2. 撰写工程硕士学位论文 |

**六、审核意见**

|  |
| --- |
| **导师意见**  **导师签名：**  **年 月 日** |
| **培养单位负责人意见**  **培养单位负责人签名：**  **年 月 日** |