文章编号：2095-2163(2019)04-0216-05 中图分类号：TP311.13 文献标志码：A

众创资源分享平台的设计与实现

胡益龙

（山西大学商务学院 信息学院，太原 030031）

摘 要：本系统是一个功能较为完备的资源分享平台，实现了资源分享、资源搜索、资源推荐等功能。其中网站的主体通过经典的 JavaEE 框架构建，通过 Lucene 技术与 Solr 技术提供资源搜索服务，并且实现了以机器学习 ALS 算法为核心的资源推荐系统，使得用户可以更为方便地找到想要的资源。

关键词：资源分享；全文检索；智能推荐

Design and implementation of resource sharing platform

HU Yilong

（School of Information, Business College of Shanxi University, Taiyuan 030031, China）

【Abstract】 This system is a resource sharing platform with relatively complete functions, which realizes the functions of resource sharing, resource search and resource recommendation. The main body of the website is constructed through the classical JavaEE framework. Based on the above, resource search service is provided through Lucene technology and Solr technology, and a resource recommendation system based on machine learning ALS algorithm is also implemented. The research makes it easier for users to find the desired resources.

【Key words】 resource sharing; full-text retrieval; intelligent recommendation

0 引 言

信息过载已经成为当前互联网迅猛发展中不容忽视的一个重要问题，由此则导致用户想要精准获取资源就显得尤为困难。在此背景下，本系统设计实现了搜索与推荐两种资源相关的服务。在算法方面，常见的推荐算法主要有基于内容的推荐与基于协同过滤的推荐。而对于协同过滤推荐算法，又可将其分为 3 种，即：基于用户的协同过滤、基于物品的协同过滤、基于模型的协同过滤。其中，协同过滤的推荐算法无需考虑物品与用户标签和属性就能够做出推荐，但是在大数据量的情况下基于用户与物品的协同过滤算法将会面临稀疏矩阵与计算复杂的问题。综合上述分析后可知，本次研发系统将使用基于模型的推荐算法 ALS，为了实现分布式计算采用了大数据 lambda 架构作为整个推荐系统基础架构。这里，研究中用到的计算框架为 Spark，通过将离线计算、实时计算、机器学习三者融为一体，可以在搜索与推荐的双重服务下使用户能够更加快速地找到自己想要的资源。

1 推荐系统算法研究

1.1 特征向量与线性相似度

本系统推荐算法使用交叉线性回归（ALS）与余弦聚类。ALS 主要是基于评分矩阵（同现矩阵）来实现推荐，而评分矩阵的计算在底层则依赖于 2 个向量的线性相似度的计算。在该模型中，假设每一个用户都对应着一个特征向量，每一个资源对应着一个特征向量。模型通过计算资源的特征向量与用户特征向量的相似度来估算某一个用户对某一个资源的好感度。好感度将直接体现在用户对某一个资源的评分上，好感度越高，评分就越高。机器学习中，有很多方法可用来估算 2 个向量的相似度，本模型假设用户向量与资源向量的相似度为用户的特征向量 $\boldsymbol{U}$ 和资源的特征项向量 $\boldsymbol{R}$ 的线性相似性。也就是评分 $Pscore \approx \boldsymbol{UR} + \lambda$。基于这一假设，只要有一个用户的特征向量与一个资源的特征向量就能够预测该用户对此资源的评分，进而做出推荐。

1.2 损失函数的定义

本次研究模型的最终目标就是使模型的预测与

基金项目：山西省教育科学“十三五”规划基金项目（GH-17097）；山西大学商务学院院级教改项目（SYJ201711）；山西大学商务学院院级基金项目（2018010）。

作者简介：胡益龙（1995-），男，本科生，主要研究方向：大数据算法。

收稿日期：2019-05-10

用户真实的评分相近。对于公式 $Pscore \approx \boldsymbol{UR} + \lambda$，令 $Pscore$ 为预测分数，$Tscore$ 为真正的评分。模型希望 $|Pscore - Tscore| \approx 0$，但是绝对值函数是一个不平滑的函数，在进行模型优化时不易计算，同时也不利于 $L2$ 的正则化。通常选择平方错误函数 $(Pscore - Tscore)^2 \approx 0$ 来作为损失函数。

1.3　模型降维与矩阵分解

遵循上述原理，模型可以根据已有的数据来计算出用户的特征向量与资源的特征向量。在原始的模型中，用户特征向量的维度为资源的个数，资源的特征向量的维度为用户的个数。研究中，n 个用户 m 个资源就会构成一个 n 行 m 列的矩阵，这样就可以同时计算出 n 个用户与 m 个资源的特征向量。其中，每一行是一个用户对各个资源的评分，每一列是多个用户对一个资源的评分。当数据量十分庞大时，用户与资源向量的维度都会变得很大，整个模型的自由度会升至很高。VC 维度证明了高复杂度的模型会形成一种系统性的噪音，会干扰模型的实际准确度，如此即会出现过拟合的现象。过拟合的含义就是：在训练集上模型表现得很好，但是在实际的测试集上会出现很大的误差。

为了降低模型复杂度、减小过拟合的风险，模型采用了矩阵分解的方式降低了用户与资源的维度。具体来说，将 n 个用户 m 个资源所构成的 n 行 m 列的矩阵分解为 $n * d$ 的矩阵与 $d * m$ 的矩阵的乘积，矩阵分解示例则如图 1 所示。其中，d 为用户可以手动调节的参数，减少 d 的数值就可以降低模型的维度，从而削弱了过拟合对模型的干扰。这样一来，模型通过降维就可获得主要的隐含因子，忽略掉次要的特征。实际上，原始的 n 行 m 列的矩阵是一个稀疏矩阵，通过矩阵分解的降维处理可以将模型计算的复杂度从 $O(nm)$ 降到了 $O((m + n) * d)$。

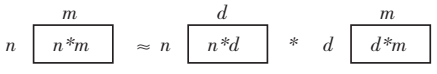

图 1　矩阵分解示例

Fig. 1　Examples of matrix decomposition

1.4　模型训练

经过损失函数的定义与矩阵分解的降维处理，下面拟进行模型的训练研究。ALS 损失函数的计算公式可表示为：

$$\min_{W,V} Error(\{\boldsymbol{W}_m\}, \{\boldsymbol{V}_n\}) \propto \sum_{m=1}^{M} \left(\sum_{(X_n, r_{nm}) \in D_m}^{n} (r_{nm} - \boldsymbol{W}_m^{\mathrm{T}} \boldsymbol{V}_n)^2 \right). \quad (1)$$

其中，m 表示资源的个数；n 表示用户的个数；$\boldsymbol{W}$ 与 $\boldsymbol{V}$ 分别表示模型训练后的资源权重的向量与用户权重的向量。

对于一组数据来说，模型在 n 号用户对 m 号资源的预测值与真实值的差值平方就是模型在一个资源向量上的损失，对于所有单一资源的误差的和就是模型的整体损失。ALS 的目标就是要根据已知的数据找到损失函数值最小的误差。为了减少运算量，一般采用随机梯度下降法(SGD)去优化模型。

分析可知，在公式(1)中有 2 个需要优化的变量 $\boldsymbol{W}$ 与 $\boldsymbol{V}$，首先让其中一个变量的值固定，这可以得到一个变量的公式，此时就能直接采用单个变量的梯度下降法计算出误差的最小值，进而计算出另一个向量的值。在此基础上，再固定刚刚计算好的向量使用线性回归计算出另一方的特征向量。如此这般轮流地进行线性回归计算，最终模型就可输出所有用户与资源的特征向量。可以通过用户的特征向量与每个资源进行线性组合，得到预测的分数，选择评分最高的 k 个资源推荐给该用户。

1.5　模型校验

在模型训练时，通常会设置多组参数，不同参数组合下的模型会有不同的准确度。通过对参数组合的优化选择，就可以得到准确率更高的模型。本系统中涉及到的参数有：模型维度、正则化参数、迭代次数。考虑到若对模型进行校验就需要准备校验数据集来评判在特定参数组合下模型的准确度，所以在划分训练集时通常会再细分为训练集与校验集。同时，为了达到训练集与校验集的独立同分布，则多会选用随机分割。至此，为了充分利用数据集、以及减少训练迭代次数，本系统直接使用了交叉验证(Cross Validation)作为模型校验的方法。事实上，本系统共将训练集随机平均分为 10 部分，每一轮训练时都将其中的 9 份作为实际的训练集，剩下的 1 份作为校验集，经过对校验集的更替，最后选出最优的参数组合。交叉验证示例如图 2 所示。

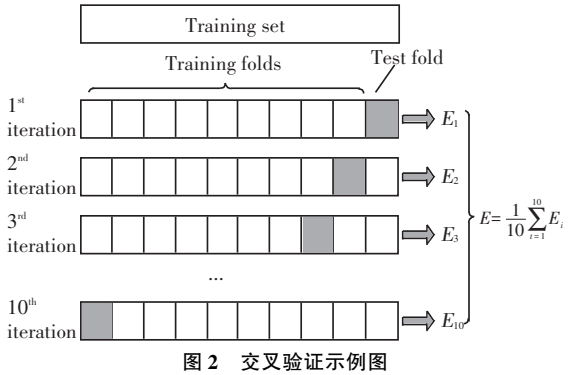

图 2　交叉验证示例图

Fig. 2　Cross validation sample diagram

1.6 余弦相似度计算

在实时推荐时,本系统将着重研究资源之间相似度的计算。在计算资源之间的相似度时用到了余弦聚类,使用向量空间中 2 个向量夹角的余弦值衡量 2 个个体间差异的大小。如果 2 个向量的余弦相似度越接近 1,就说明这 2 个向量越相似。反之,越接近 0,就越不相似。余弦相似度的数学定义可表示为:

$$\cos\theta = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}. \tag{2}$$

1.7 模型的评估

在成功构建这一推荐模型的同时,必须要保证模型的可用性与准确性,否则整个模型就几乎相当于在做随机猜测。为简单起见,本系统没有使用 *AUC* 作为模型的评价方案,而是直接计算模型的预测评分与实际标签的均方根误差 $RMSE_1$。均方差越小,就表明模型预测得越准确。此外,还要计算得出平均评分值与实际评分的 $RMSE_2$。当 $RMSE_1 - RMSE_2$ 为正数时,就证明模型有正向的预测效果。

1.8 推荐系统冷启动问题

当面对一个全新系统、全新用户或者全新资源时,由于没有历史数据的铺垫可能会出现推荐算法的冷启动问题。既然推荐算法在此时的表现欠佳,就决定了需在此处设置一个替补的策略,也就是榜单系统。新来的用户可以从榜单开始寻找自己的喜好,同时也会有更新的榜单来避免新的资源的冷启动。

2 系统设计

2.1 总体功能模块设计

系统设计包括 Web 端的设计和推荐系统的设计。与此相对应,功能模块包括用户登录、注册、创建资源、浏览资源、搜索资源、收藏资源、删除资源、离线榜单、在线榜单、离线推荐、在线推荐等。总体模块结构如图 3 所示。

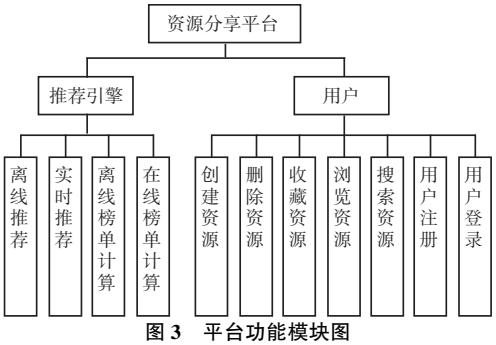

图 3 平台功能模块图

Fig. 3 Function module diagram of the platform

2.2 推荐系统架构的设计

推荐系统在功能上可解析为推荐功能与榜单计算两部分。而对数据处理而言,又分为离线与实时两大类。文中将对此展开研究分述如下。

(1)功能设计分析。服务集群由多个子集群组成,对其中各主要集群的功能设计可做出如下阐释及论述。

① Hadoop/Hive 集群。对其功能可表述为:一方面为数据存储,系统会将 MySQL 中的数据同步到集群上。另一方面提供资源调度,系统将会使用 Spark on Yarn 的调度模式,这里的 Yarn 组件来自于该集群。

② Spark 集群。该集群的功能定制为整个推荐系统的全部计算。其中,SparkML 负责推荐算法的执行,SparkStreaming 负责在线数据的计算处理,同时这两者都可以使用 SparkSQL 的 API 来给出更加便捷的实现。

③ Zookeeper 集群。主要负责分布式协调工作,并由其配合 KafKa 集群共同完成分布式消息队列的功能。KafKa 集群是一个分布式消息队列,可通过 Log4j 来消费发送自 Web 服务器的数据,分布式地存储在各个节点上,不仅具备高容错、高吞吐的特点,而且也促进了 Web 端与数据计算端的有效解耦。系统拓扑结构如图 4 所示。

(2)数据处理分析。整个数据流的处理可以分为离线与在线两个部分。其中,离线部分首先将 MySQL 中的用户评分表中的数据使用 Spark 导入 Hive 中;使用 SparkSQL 连接 Hive 读取数据,并且处理数据,计算离线榜单与推荐矩阵。进一步地,推荐矩阵中给每个用户提供了 10 条推荐资源,同时也计算出了对于某一个资源与其相似的 10 条资源,为后续的在线推荐做好准备;将计算过的数据使用 SparkSQL 写入 MySQL 中,方便在 Web 端做出展示。在线部分中,用户点击 Web 界面会产生日志信息;这些日志信息会被 KafKa 消费到自己的队列中;而 SparkStreaming 则会消费 KafKa 中的数据,将数据按照一定的时间间隔进行微批处理,在此过程中还会提取 MySQL 中的数据进行连接计算;将计算好的数据送入 MySQL 中进行 Web 端的展示。

2.3 算法参数调优

系统使用 Spark 提供的 ParamMap 进行参数设计,根据不同的维度设置不同的参数,这样模型在训练时将会考虑到所有参数的组合(笛卡尔积),最终得到一个更好的模型。本系统中设置的参数组合见表 1。

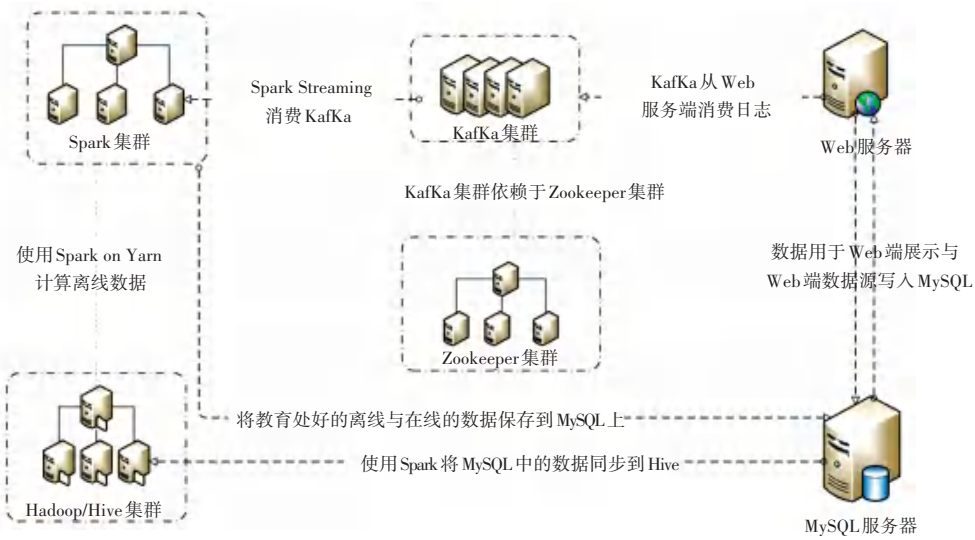

图 4　推荐系统结构图

Fig. 4　Recommendation system structure diagram

表 1　模型参数表

Tab. 1　Model parameter table

模型维度	正则化参数	迭代次数	*RMSE*
20	0.010 0	20	1.928
20	0.010 0	30	1.814
20	0.112 6	20	1.325
20	0.112 6	30	1.117
30	0.010 0	20	1.821
30	0.010 0	30	1.798
30	0.112 6	20	1.069
30	0.112 6	30	0.982

2.4　系统数据库逻辑结构设计

由于本文设计的系统选用了关系型数据库管理系统。研究推得的系统数据库的逻辑结构如图 5 所示。

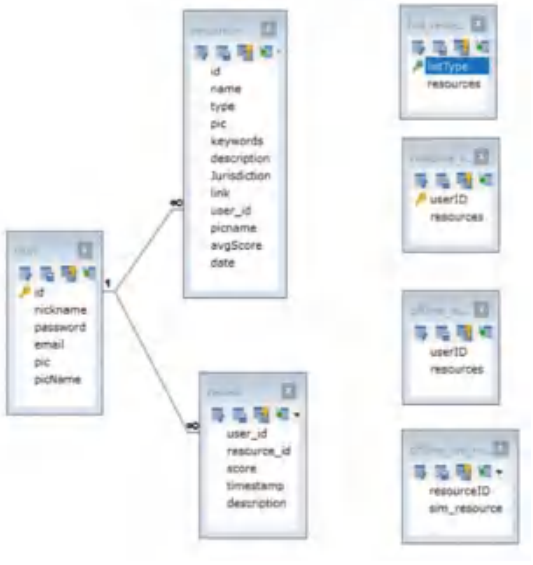

图 5　系统数据库逻辑结构图

Fig. 5　Logical structure diagram of system database

3　系统实现

3.1　资源搜索的实现

当用户在搜索框输入关键词之后，搜索引擎将会根据提前构建好的倒排索引去搜索相关的资源。资源搜索的主界面如图 6 所示。

图 6　资源搜索界面

Fig. 6　Resource search interface

3.2　离线推荐计算的实现

Spark ML 中封装了 ALS 的完整实现并且提供了非常简单的接口方便调用。ALS 算法在训练时会调取较多的迭代式计算。ALS 计算时 SparkUI 中的 DAG 图见图 7。

图 7　推荐算法 ALS 训练 DAG

Fig. 7　DAG of recommendation algorithm training

（下转第 225 页）