订阅DeepL Pro以编辑此演示文稿。  
访问www.DeepL.com/Pro，了解更多信息。

物理学杂志。会议系列

**论文 - 开放存取**

基于Django的MVC框架的网络日志优化存储和分析系统的研究

引用这篇文章。Huan Tian *et al* 2021 *J. Phys:Conf.Ser.* **1769** 012065

[在线](https://doi.org/10.1088/1742-6596/1769/1/012065)查看[文章的](https://doi.org/10.1088/1742-6596/1769/1/012065)更新和改进。



此内容是在08/06/2021 15:08从IP地址122.14.229.212下载的。

**网络优化存储和分析系统的研究**

**基于Django的MVC框架的日志**

# 田欢，赵静\*，沈佳琪

兰州职业技术学院电子与信息工程系，甘肃省兰州市，中国

通讯作者的电子邮件。120839127@qq.com

**摘要。**关联规则分析算法在Web日志分析中得到了广泛的应用，但现有的关联规则分析算法在Web日志量比较大的情况下会明显降低分析和挖掘性能。本文提出了一种改进的聚类算法，首先对具有相同兴趣和爱好的用户进行聚类，然后对同一类别的用户进行关联规则挖掘，从而减少数据的分散性。基于Django的MVC框架，它优化了网络日志的存储和储存。在分析部分，用户可以通过前端配置关联规则挖掘的支持度和置信度，同时通过Hive查询挖掘结果，并在数据传输过程中使用加密算法，保证数据安全。

**Keywords.**HDFS；网络日志挖掘；聚类；FP-Growth算法

## 1.绪论

网络日志挖掘系统设计的总体要求是安全、高效和易于使用。使用优化的分布式文件存储架构来保存日志数据，并使用加密算法来保证日志传输过程中的数据安全。同时，利用分布式计算工具提取Web日志数据中的有用特征，结合改进的聚类算法对日志数据进行分类，再通过关联规则挖掘算法找到符合属性要求的规则序列。用户可以对系统创建的任务进行管理，设置关联规则挖掘算法的参数，及时获得挖掘任务的结果。

## 2.K-means聚类算法的优化

对于海量网络日志数据的关联规则的挖掘，主要使用的算法是Apriori和FP-Growth算法。它们在算法的执行效率上都有各自的优势和劣势[1]。Apriori算法需要不断访问数据库，数据库的开销显然是不可接受的。FP-Growth算法在构建FP-Tree时需要存储在内存中，但由海量Web日志数据构建的FP-Tree会消耗大部分的内存，这将严重影响集群的性能[2]。因此，本文提出一种基于改进的K-means聚类算法的FP-Growth关联规则挖掘算法。首先，利用提出的聚类优化算法，减少数据的分散性，将具有相同爱好的用户归为一类，然后对类数据进行相同的挖掘关联规则[3]。

本作品的内容可根据CreativeCommonsAttribution 3.0许可的条款使用。对本作品的任何进一步传播都必须注明作者、作品名称、期刊引文和DOI。

由IOP出版有限公司授权出版

### 2.1.K-means聚类算法的分析

与其他聚类算法相比，K-means算法具有执行过程简单、收敛速度快、易于实现的优点。为了衡量K-means聚类算法的性能，通常用平方误差之和（SSE）来解释。平方误差之和的具体计算方法如公式（1）所示。

𝑆𝑆𝑆𝑆𝑆𝑆 = ∑𝑘𝑘𝑖=1∑𝑥∈𝑐𝑐𝑖 𝑑𝑑𝑑𝑑𝑑𝑑𝑑𝑑(𝑥, 𝑐𝑐𝑖) （1）

其中，𝐶𝑖代表第i个簇中的粒子，x代表第i个簇中的任何数据点，所以公式∑𝑥∈𝑐𝑐𝑖 𝑑𝑑𝑑（𝑥。𝑐𝑐𝑖）表示第i个聚类中的所有数据点到聚类中的粒子点的距离之和，K-means距离计算方法采用欧氏距离计算方法。因此，SSE代表所有数据点与该点所属簇的质量点之间的距离之和。如果SSE值较大，说明各聚类的聚类效果不是很好，数据点不是很密集；如果该值较小，说明各聚类之间的聚类效果较好。这里的计算只有在初始质量点确定后才能开始，所以它只是一个局部最优解，因为K-means算法没有明确的方法来确认初始质量点。如果初始质量点选得不好，会造成SSE过大。Kim等人根据最大和最小距离的思想来确定K的初始值。首先，计算每个点与每个群集点之间的最小距离集合，然后选择距离最大的点作为新的群集点。这种方法可以避免因聚类点的选择而导致聚类效果过于接近，但这种方法不能解决异常点的影响和新聚类点迭代计算的消耗问题。

### 2.2.K-means算法的优化

在分析K-means算法的基础上，针对K-means算法的缺陷，结合网络日志本身的数据特点，本章提出了一些改进的解决方案，主要包括以下内容。

* 网络日志数据的预处理。在大量的Web日志数据中，并非所有的数据记录都属于正常的用户访问。根据请求状态码不在正常范围内、请求方式不是GET、请求资源类型不是页面请求等情况，可以剔除部分异常数据。过滤掉这些不符合请求和状态码错误的日志，可以减少Web日志数据中的异常点，可以避免一些极端属性数据对样本距离的计算产生严重影响。
* 优化初始聚类的数量。初始聚类中心的选择会严重影响最终的聚类效果，所以随机选择k个聚类中心并不理想。本文的Web日志分析是基于大数据平台的，所以聚类的数量可以基于分布式计算来确定。首先，由于SSE的计算方法只考虑了局部最优性，没有考虑每个簇中粒子的差异。因此，本章提出了全局最优解的计算方法来确定最优初始簇数。该函数的具体定义如下。



𝑖

𝑂𝑂𝑂𝑂𝑂(𝐾𝐾) = 𝑘𝑘 𝑛-𝑘𝑘） (2)

∑𝑖=1,𝑗𝑗=1𝑠𝑖𝑠𝑠(𝑐𝑐𝑖,𝑐𝑐𝑗𝑗)

𝑘𝑘

* 在公式（2）中，k代表聚类的数量，n代表所有数据的数量，sim代表两个数据点之间的距离。距离的具体计算方法将在下一节解释。公式中∑𝑘𝑘𝑖=1,𝑗𝑗1𝑑𝑑𝑑𝑑𝑠𝑠(𝑐𝑐𝑖, 𝑐𝑗)表示每个簇之间质量点的平方误差之和。该值越大，就越远

每个聚类之间的距离越大，数据聚集越明显，聚类效果越好。该公式表示组内的平方误差之和，代表每个聚类的收敛程度。该值越小，每个簇的聚类效果越好。因此，在全局损失函数的算法基础上，按照全局损失函数的定义，本文通过寻找波动最大的k值来确定聚类族的数量。最大波动的值可以通过寻找变化率突然变大的转折点来确定，因为如果变化率趋于平缓，就意味着继续增加聚类的数量是没有意义的。当每个点当它是一个集群时，全局损失函数为零。

* 迭代过程的优化。在K-means算法中，下一次迭代的聚类点是由聚类中所有数据的平均点决定的。这样形成的聚类中心很可能不在数据真正的高密度区域，这就导致了最终的聚类。类结果会有一定的偏差，迭代成本也会变高。本文在二元K-means算法的基础上，结合最大距离思想，提出一种优化的二元聚类算法。首先找出群中平方误差总和最大的聚类，计算出与聚类粒子距离最大的K点，并根据这K点对聚类进行二元聚类，然后得到K个二元聚类集，选择 平方误差和最小的划分取代原有聚类。不断重复上述步骤，直到获得的聚类数量等于初始集K。

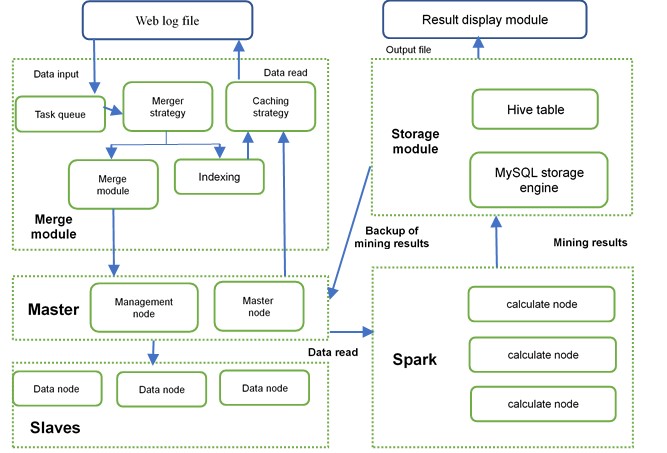
## 3.网络日志存储和分析系统界面设计

作为一个完整的网络日志挖掘系统，该系统在功能上主要实现了以下接口。

* 用户界面，主要用于验证用户的合法性。它主要分为普通用户和管理员用户。不同级别的用户有不同的权限。
* 日志存储接口，该接口主要用于用户上传需要分析的Web日志数据。通过这个接口，可以触发后端优化的HDFS存储架构来保存数据。
* 数据下载界面，主要用于下载Web日志数据和挖掘任务的执行结果，包括分类数据和相应的关联规则挖掘结果。
* 任务创建界面，该界面用于创建挖掘任务，用户可以通过该界面选择数据批次，设置关联规则挖掘算法的参数。
* 状态查询界面，用户可以通过该界面查看任务的执行状态。如果任务执行失败，可以查看失败日志，然后重新启动任务；如果执行成功，可以下载开采结果。

## 4.系统框架设计

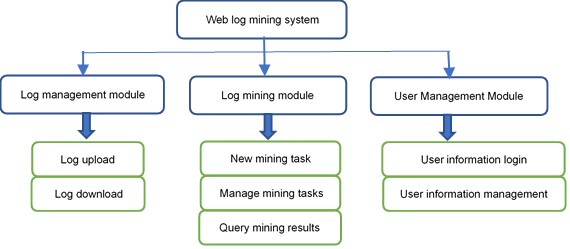
网络日志的导入主要是基于HTTP协议的。通过客户端的配置，选择需要导入的Web日志数据[4]。上传的Web日志经过文件合并模块存储在HDFS中，作为数据分析的基础。然后，在Spark集群的基础上，对Web日志进行挖掘。网络日志挖掘主要包括聚类分析和关联规则挖掘。日志挖掘的结果存储在MySQL和Hive表中，并同时持久化到HDFS。用户可以通过结果显示模块获得挖掘信息。图1是系统整体设计架构图。



**图1.**系统整体设计架构图

### 4.1.系统功能设计

图2是网络日志挖掘系统的功能结构图。该系统主要分为三个模块。日志管理模块主要负责日志的上传和下载功能；日志挖掘模块主要负责用户管理任务、查看任务状态和查询任务执行结果；用户管理模块主要负责管理用户的登录注册信息。



**图2.**系统功能结构图

### 4.2.数据库设计

当用户使用系统进行挖掘任务时，除了在HDFS中存储Web日志外，还需要在MySQL数据库中存储一些用户和任务相关的表，以显示用户有关任务执行和数据挖掘结果的相关信息。同时，还需要数据块表来存储用户相关信息，主要涉及以下四个表。

* 用户信息表。如表1所示，它是一个用户信息表，主要提供给超级用户管理系统的用户。字段user\_role是用户的角色，超级用户的值是admin，普通用户的值是工程师。超级用户可以通过添加、删除、修改和检查来修改普通用户信息。**表1.**用户信息表

|  |  |  |  |
| --- | --- | --- | --- |
| **领域名称** | **类型** | **领域约束** | **领域描述** |
| user\_id | INTEGER | 主键 | 用户ID |
| 用户名称 | VARCHAR[64] | 非空 | 用户名称 |
| 用户角色 | VARCHAR[64] | 非空 | 用户角色 |
| 创建时间 | 日期 | 非空 | 注册时间 |
| 发言 | VARCHAR[64] | 无 | 备注 |

* 文件存储表

当Web日志被导入HDFS时，需要根据文件的批号将日志文件的信息存储在MySQL中。其中，Mysql中的日志文件信息与HDFS文件的信息是一致的。如表2所示，日志存储信息中包括的主要字段，pat字段表示网络日志文件在HDFS上的位置，批号是每批日志的唯一表示。**表2.** 文件存储表

|  |  |  |  |
| --- | --- | --- | --- |
| **领域名称** | **类型** | **领域约束** | **领域描述** |
| batch\_id | INTEGER | 主键 | 批量标识 |
| batch\_name | VARCHAR[64] | 非空 | 批次名称 |
| 路 | VARCHAR[64] | 非空 | 存储路径 |
| 创建时间 | 日期 | 非空 | 储存时间 |

* 任务执行信息表

表3显示了任务执行信息表的主要字段和描述。该表主要存储执行采矿任务的用户信息。其中，user\_id和batch\_id分别与用户信息和文件存储相关。Task\_name表示任务执行的名称，主要由时间戳和上传的文件夹名称组成。status字段表示任务的执行状态（0：准备就绪，1：正在执行，2：执行成功，3：执行失败）。conf\_info字段表示用户执行任务时选择的配置信息，即挖掘日志关联规则时的支持度和置信度。

**表3.**任务执行信息表

|  |  |  |  |
| --- | --- | --- | --- |
| **领域名称** | **类型** | **领域约束** | **领域描述** |
| Task\_id | INTEGER | 主键 | 批量标识 |
| user\_id | INTEGER | 外键 | 用户ID |
| batch\_id | INTEGER | 外键 | 日志批号 |
| 任务名称 | VARCHAR[64] | 非空 | 任务名称 |
| 创造\_时间 | 日期 | 非空 | 储存时间 |
| 开始时间 | 日期 | 非空 | 开始时间 |
| 结束时间 | 日期 | 非空 | 结束时间 |
| 身份 | INTEGER | 非空 | 执行状态 |
| conf\_info | VARCHAR[64] | 非空 | 配置信息 |

* 结果信息表

如表4所示，它是任务成功执行后存储在执行结果中的相关信息表。该表主要存储Web日志聚类挖掘和关联规则挖掘的结果信息。其中，字段result\_id代表每个挖掘结果的唯一标识符，task\_id为外键，关联每个挖掘任务的具体信息。Cluster\_nuni表示这批通过改进的K-means算法挖掘的网络期刊的集群数量，每个集群的日志信息存储在HDFS上，具体路径信息存储在字段cluster\_path中。freeq\_num表7K是基于集群的FP-Growth算法挖掘出的频繁项总数，具体频繁项的信息通过路径字段freeq\_path存储在HDFS上。fp\_growth\_num表示FP\_Growth转换方法的关联规则挖掘总数，即与改进的K-means聚类算法相结合后，针对每一类兴趣相近的用户，由关联规则挖掘算法挖掘的关联规则总数，具体的关联规则信息也通过路径fp\_growth\_path存储在HDFS上。同时，将存储在HDFS中的关于Web日志信息和执行结果信息的数据导入到Hive表中，方便用户通过Hive表查询结果，并通过前端页面向用户显示相应的执行结果。**表4.**结果信息表

|  |  |  |  |
| --- | --- | --- | --- |
| **领域名称** | **类型** | **领域约束** | **领域描述** |
| result\_id | INTEGER | 主键 | 结果ID |
| Task\_id | INTEGER | 外键 | 批量标识 |
| user\_id | INTEGER | 外键 | 用户ID |
| batch\_id | INTEGER | 外键 | 日志批号 |
| 任务名称 | VARCHAR[64] | 非空 | 任务名称 |
| 创造\_时间 | 日期 | 非空 | 储存时间 |
| 集群\_num | INTEGER | 非空 | 集群的数量 |
| 集群\_路径 | VARCHAR[256] | 非空 | 集群存储路径 |
| 頻率\_num | INTEGER | 非空 | 经常性项目的数量 |
| 频率路径 | VARCHAR[256] | 非空 | 频繁出现的项目路径 |
| 增长数 | INTEGER | 非空 | 关联规则总数 |
| Fp\_growth\_path | VARCHAR[256] | 非空 | 规则存储路径 |
| conf\_info | VARCHAR[256] | 非空 | 配置信息 |

## 5.系统实施

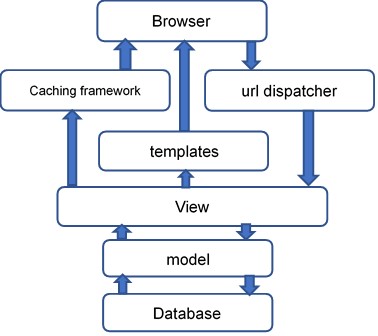
### 5.1.MVC框架的构建

Django是一个由Python编写的开源Web应用框架。图3显示了Django的整体结构。Django的整个系统框架的代码管理主要由以下文件组成。

* Urls.py。这个文件用来接收用户访问API的请求，然后根据用户的请求跳转到视图。py中的相应接口。如表5-5所示，定义了系统url和接口之间的映射关系，例如，"url(r'Alogin', views.login, name='login')"，用户通过HTML协议访问系统。页面，它将传递urls。py文件请求视图中的用户登录界面，然后返回登录结果。除了用户登录界面外，主要包括上传和下载日志的界面，创建任务的界面，以及请求任务执行结果信息的界面。
* View.py:用户定义的接口，也就是接收urls。py转发的用户请求，然后每个请求的具体执行逻辑都在这个文件中定义。如表5-6所示，它是用户请求执行结果信息的服务接口。首先获得用户下载请求的服务器地址，然后搜索相应的task\_id，再根据任务信息获得具体的执行结果信息。
* Models.py。与数据库操作有关，当用户请求任务状态和相应的执行结果信息时，需要连接到数据库，然后获得具体的数据。如表5-7所示，模型。py中获取任务执行信息的类定义主要是连接到数据库，然后将数据库中的每个字段与之前的

在末端的表格中显示的字段与视图中的具体逻辑相对应。在py中实现的。

* Admin.py。通过添加配置代码完成后台配置。
* Settings.py:存储Djaango的配置信息，如静态文件的位置等。



**图3.** Django结构图

*5.2.系统功能模块实现* 对于Web日志的采集，一般采用实时消息系统进行采集，如kafta和nsq消息队列[5]。根据网站的不同业务场景，采用不同的主题来采集不同的Web日志，然后通过HDFS将日志文件保存到磁盘。本文所设计的挖掘系统主要是指对离线Web日志的关联规则进行挖掘。所分析的Web日志是基于用户访问网站后留下的日志数据，而不是实时处理正在产生的日志。因此，为了挖掘和分析日志，用户需要选择导入日志数据的位置，然后开始上传Web日志。

为了防止用户上传的网络日志数据被篡改，有必要在上传过程中加强网络日志的安全性。首先，选择加密算法。常用的加密算法包括对称加密算法和非对称加密算法。但是，次不对称加密算法需要使用公钥和秘钥，加解密过程需要很长的时间。因此，选择AES对称加密算法对上传的网络日志进行加密。但是，由于对称加密算法的加密和解密过程中使用的秘钥是相同的，所以安全性相对较低。为了加强安全性，日志的MD5被用来作为辅助检查[6]。即对加密后的日志计算MD5，然后用逗号作为分隔符连接加密后的字符串，形成一个新的字符串。为了减少日志传输过程中的带宽消耗，新字符串通过gzip压缩算法进行压缩，然后通过HTTP协议上传到服务器。

当服务器收到信息时，它首先解压gzip，然后通过分离器获得加密的字符串和相应的MD5值。然后计算出MD5值。如果计算出的加密字符串的MD5值与传输的MD5值相同，说明传输的数据没有被修改过；如果MD5值不相同，说明数据在传输过程中被修改过，然后丢弃该数据。在验证了MD5值后，用相同的AES密钥对加密字符串进行解密，解密后的字符串就是上传的网络日志。后台获得解密的网络日志后，将日志输入到文件预处理模块[7]。

收到解密后的日志数据后，要对数据进行预处理。数据预处理是网络日志挖掘的一个必要过程，也是整个数据准备的核心工作[8]。数据预处理是整个挖掘过程的基础。如果数据预处理不好，将直接影响挖掘过程中产生的规则和模式，也是挖掘质量的保证。数据预处理主要包括数据清洗、用户识别、会话识别和路径补充等阶段[9]。

* 数据清理

在原始网络日志中，有许多状态代码为3XX系列和4XX系列的请求[10]。这些请求表示重定向或请求错误，还包括一些后缀为gif和jpg的网络资源请求。这对于分析用户行为是没有意义的，所以需要从原始数据中过滤掉，只需要保留状态码为2XX系列的GET请求。

* 用户识别

用户识别阶段从数据清洗后的数据中划分出不同用户的访问，即以用户IP为关键，以用户的访问项目为值，每个访问项目由访问链接和访问时间组成[11]。

* 会话识别和路径补充

会话识别是指识别用户的一个完整的浏览过程，即用户从访问网站到离开网站所访问的一系列的页面序列集合。这被称为用户的一个会话。系统为每个会话设置了30分钟的时间阈值，也就是说，一个会话的时间不会超过这个阈值[12]。由于受到网站代理服务器缓存的影响，用户的访问请求不会产生相应的日志，因此需要将这些服务器漏掉的访问请求加入到用户会话中，为Web日志挖掘提供完整的数据源。

## 6.结论

本文设计了一个基于大数据平台的可视化Web日志挖掘系统。在Django的MVC框架基础上，借助开源的Bootstrap框架，实现了一个面向用户的Web日志存储和挖掘系统。本章详细介绍了各个模块功能的内部实现细节，并展示了语言系统的整体框架和各个模块的具体信息。利用该系统，用户可以通过简单的前端操作实现本文提出的Web日志存储、聚类分析和关联规则挖掘。

# 鸣谢

感谢以下基金对本文的支持。兰州职业技术学院科研项目"'互联网+'视角下基于产教融合的高职院校创新创业团队建设研究"。

# 参考文献

[1] E. Steegmans, P. Rigole, T. Holvoet, et al. 2013 智能建筑。 一个多代理系统

## 方法。*人工智能与应用论文集》*第771-776页

1. Hani Hagras, Victor Callaghan, Martin Colley, et al. 2013 用于智能建筑在线学习、适应和控制的分层模糊遗传多Agent架构 *信息科学***150**页 33-57
2. Vladimir Marik和Pavel Vrba 2014年在罗克韦尔自动化公司为制造业开发代理，*代理链接新闻***12**页9-11
3. Wellman M.P, Birmingham W.P, and Durfee E.H. 2016 数字图书馆作为信息代理的社区 *IEEE expert* **11** (3) pp 10-11
4. Wooldridge M和Jennings N.R. 2015年智能代理。理论与实践 *The Knowledge Engineering Review* **10** (2) pp 115-152
5. Davidsson Paul和Boman Magnus 2015 通过嵌入式代理对办公大楼进行分布式监测和控制 *信息科学* **171** (4) pp 293-307
6. Kraus Sarit 2017 多Agent环境下的谈判与合作 *人工智能***94**页79-97
7. Flax, and Barry.M. 2011年智能建筑 *IEEE通信杂志* **29** (4)
8. Loveday 2012 建筑物的人工智能 *应用能源* **41** (3)
9. Wulf.opc dx[eb/ol].http://www.hpe.fzk.de/hgf/scs-workshop2002/Wulf.pdf。2012-11-20.
10. Kujuro 2013 智能建筑中的系统进化 *IEEE通信杂志***31** (10)
11. Dave Thompson 2020年的BACnet增益*工程系统***17**页108-116。