基于 HADOOP 的海量数据分析研究

马铭阳 张桂花
四川大学锦城学院计算机与软件学院 四川 成都 611731

【摘 要】时代在不断地发展，网络上面的信息的产生和传递的速度已经远远的超出了人们的以前对于信息接收数量的想象。每天都在有海量的数据产生并涌向客户，使得用户很难在其中快速找到需要自己关注的有用信息。如果将不断更新的海量数据进行采集整理、预处理、清洗、分析、检索、分析总结，最后将结果一图表等形式直观地展示给用户，用户能够在快速，简便地获取自己的需要的信息分布就十分有必要，基于以上的需求。本文通过以 NBA 球员数据网的在线数据为样本，对收集到的数据进行一系列的操作，运用爬虫对目标网站进行抓取，再模拟生成海量的数据，工具预处理和传输筛选，对得到的结果信息进行研究分析，并将结果使用可视化的方式来展示 NBA 球员的相关数据，使用户能够方便、有效、快速地了解到喜爱的球星数据。

【关键词】数据处理；海量数据；可视化；hadoop

1 概述

随着科技的不断发展，尤其是近几十年，人们通过对互联网的使用和改进，使得信息的交流传递更加的便捷，用户的信息来源也在逐渐地从传统的报刊杂志、电视、电台广播，变成在当今时代网络消息门户、博客、新浪、虎扑、微博、知乎等各种各样的网络舆情平台 [1]。获取信息的方式也从以前的定时播报、周期购买期刊变成了现在的随时订阅、随时访问。网络已经融入到了越来越多的人的生活之中，成为他们获取信息的重要渠道。但是，信息方便后去的同时，因为海量的信息在同时产生，人们很难及时地获取自己感兴趣的内容，如何将海量的信息进行处理筛选，呈现出客户需要的重要信息，就成为当今的研究热点之一 [2]。本文以 NBA 球员信息为例，介绍了海量球员数据的分析处理过程，可视化过程，并向客户展示重要球员的信息分布状态，帮助客户迅速了解喜爱的球员信息。

2 数据的相关分析技术

在海量数据的信息采集、数据处理、消费接受、数据分许的过程中涉及到以下的核心技术 [3]：

2.1 Hadoop

Hadoop 是一种分布式的文件系统，更为常用的说法是 HDFS。HDFS 具有非常高的容错性，它被设计用来部署在能够更加广泛分布的硬件上；而且它能够实现强大的高吞入量和吐出量来使用应用程序的相关数据，适合那些本身有着海浪数据并且在本地系统中不足以应对的程序。HDFS 放宽了很多的要求，可以接受用数据流的方式来访问文件系统中的数据步骤。Hadoop 全体的框架之中最重要的部分就是：HDFS 和 MapReduce。HDFS 为海量的数据提供了存储，而 MapReduce 则为海量的数据提供了计算。

2.2 Flume

Flume 是一个分布式的、强大的功能实用性、并且非常可靠地将巨大数量的不同数据来源的日志数据收集、聚合、移动到数据中心（HDFS）进行存储的系统。尤其是在面对任何来源的大数据量的流式数据事件时，都能够处理并整理到一个指定的储存位置，并且在流动数据过程中能够进行简单的数据筛选，初步过滤掉一些不相关的数据。

2.3 Kafka

Kafka 是分布的能够订阅消息和发布消息的系统，它因为高吞吐，和水平扩展而被广泛使用。它强大流动数据的监听和处理机制，使得大数据在传输的过程中不会轻易崩溃，通过多个消费者的设定，相当于功能强大的缓存，防止在数据的传输和接受中因为体量过大而发生崩溃。

2.4 Spark

Spark 使用 Scala 语言进行实现，它是一种面向对象、函数式编程语言，能够像操作本地集合对象一样轻松地操作分布式数据集，这里主要是运用到其中的 spark SQL 技术，因为 spark 本身的强大运算速度和方便兼容的特性，使得它能够很好加入到数据的处理中，并且在数据处理时，更方便地进行结果的查看和优化。方便开发人

员处理更加复杂海量的任务数据。

2.5 Hive

Hive 是基于 Hadoop 的数据仓库工具，实质就是一款基于 HDFS 计算的框架，对存储在 HDFS 中的数据进行分析和管理。能够将结构化数据映射成数据库表，并且用 HQL 进行检索查询。通过将处理过的大量数据倒入到预设的表格中，按照需要求筛选出对应结果并以文件的形式进行储存。使得用户可以在后续的使用中更加方便按需查看。

3 数据的分析处理过程

3.1 数据采集以及数据模拟

基础数据的建立是任何的处理分析步骤的前提，因此我们需要有足够多的样本来进行模拟实验，保证我们的展示结果的有效性。目前常用的获取目标网站的手段就是编写爬虫程序，还有方便快捷的 java 数据模拟。

爬虫程序的流程：部分代码

```
start_urls = [‘http://www.stat-nba.com/playerList.php’]
def parse(self, response):
    url = ‘http://www.stat-nba.com/player/’ + str(num) + ‘.html’
    print(url)
    yield scrapy.Request(url,callback = self.parse_nba)
    except Exception as e:
    continue
def parse_nba(self, response):
    items = {}
    label = response.xpath(‘//*[@id=”stat_box_avg”]/thead/tr/th/text()’).extract()
labelstr = “姓名,”
    for i in label:
    labelstr += i +’,’
    print(labelstr)
    items[‘label’] = labelstr
    datastr = “”
    datastr += str(response.xpath(‘//*[@id=”background”]/div[4]/div[2]/text()’).extract()[0]).replace(‘\n’,’’) + ‘,’
    datastr += str(response.xpath(‘//*[@id=”stat_box_avg”]/tbody/tr[1]/td[2]/a/text()’).extract()[0]) + ‘,’
```

先用代码定位到需要爬取的页面，接着查看页面元素，选出爬取数据的最小的相同标签前缀，进行定位和内容摘取，同时也要注意数据的浮动范围，只包裹我们需要的部分。

数据模拟的流程：

```
import random
import string
for  xunhuan1 in range(1,100):
    xing=’ABCDEFGHIGKLMEOPQRSTUVWXYZ’
    ming=’abcdefghijklmnopqrstuvwxyz’
    X=random.choice(xing)
    M=””.join(random.choice(ming) for i in range(2))
    print(X+M,end=’,’)
    x= random.randint(15,20)
    y=random.randint(15,20)
    print (str(x) + “-”,end=’’)
    print(str(y),end=’,’)
    team=[‘BostonCeltics’,’ MiamiHeat’,…’ Charlotte Bobcats’]
    …
print(random.randint(0,67),end=’\n’)
```

将整理的数据进行内容上的分析，选出那些有助于研究的词条内容。接着对每个成分的数据进行 random 的随机循环工作，在预设值中随机成大量的相关数据，保存后开始进行模拟生成，并输出到指定文件之中。

3.2 数据的格式处理

在我们进行下一步工作之前，要考虑到数据的相同性，在不同的地方可能会因为不同的编码方式导致数据不能够顺利地进行传输和转移，因为我们为了方便。需要提前将数据文件和之后需要的环境都换成 UTF-8 的编码方式，通过使用 iconv -f gbk -t utf8 a.txt > a.txt.utf8 等相关语句保证文件在传输的过程中都会是统一的编码方式，避免乱码排列不齐等错误，浪费时间和精力，工作效率自然也就不高。

3.3 数据的清洗

数据的清洗就是通过相关的工具将有用的数据留下，过滤掉无用的数据。因为有效的过滤可以让展示的结果获得用户的认可[4]。这里使用了 flume 的拦截和 hive 的筛选 HQL 运行数据的筛选和处理，对于预处理文件中的数据进行清洗筛选，选出球员姓名，成绩，贡献等信息，分别统计出潜力球员，得分好手，数据总结等相关的展示数据。

清洗过程：启动 kafka 消费经过 flume 正则拦截过后的大量有规律的数据。

Flume 配置拦截器过滤

```
# 拦截 1.0 整数超过预期
a1.sources.mysource.interceptors.i1.regex= [7-9][0-9]
```

(?!\.|%)

拦截 2.0 小数之前超过 75

a1.sources.mysource.interceptors.i2.regex=[7-9][5-9](?=\.?\d?%)

拦截 3.0 小数点之后大于 5

a1.sources.mysource.interceptors.i3.regex=(?<=\.)[6-9](?!%)

再通过 kafka 配置

bin/kafka-server-start.sh config/server.properties

cd /usr/lib/kafka/kafka_2.12-2.2.0

bin/kafka-topics.sh --create --topic nbat --replication-factor 1 --partitions 1 --zookeeper localhost:2181

bin/kafka-console-consumer.sh --bootstrap-server h1m1:9092 --topic nbat

同时通过 spark 操控 hive 的方式，进行表格建立，定位 kafka 得到的数据，按数量存入到对应的表格之中，

Hive 建立表格以及 spark 导入文件：

Create table if not exists nba (

…clustered by(tm) into 30 buckets

row format delimited fields terminated by ‘,’;

spark.sql(“load data local inpath ‘./shiyan.txt’ into table nba “).show

应为运算速度的关系，就在 spark 里面使用操作 hive 语句方式，快速地试验出想要结果数据，接着将对应的语句修改格式放到 hive 的目录下，用 hive -e 的操作保存为文件形式，就是对内容用函数进行分类、检索，通过 mapreduce 操作返回规定行数的要求数据，最后将获取的数据进行整合打包输出到对应组别文件之中。

3.4 数据的存储

在外部通过 Mysql 进行数据库建立，将清洗过后的数据提取并存储为需要展示的表格。

存储过程：首先根据展示的要求建立多张数据库表格并用对应的名称命名，通过 java 获取之前存储的文件并且和数据库建立连接，调用存储过程将数据存储到对应的表格字段之中，完成表格信息存储。

最后会生成与展示需求相同的数据库表格：得分表展示球员之间的数据高低；能力表展示单个球员的方位能力等等。

3.5 数据结果展示

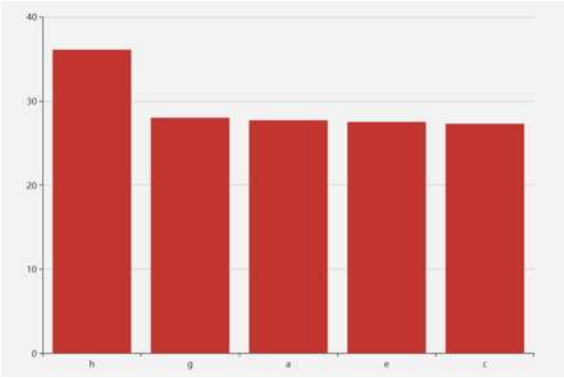
3.5.1 程序框架软件

海量数据的研究分析在本次研究中使用到的是 Spring Boot 框架，这是一款非常方便快速的框架，通过将常用的一些命令等集合成库，使得程序的搭建更加地轻松便捷，同时也更加方便我们进行需求的修改，不用受到模板化配置的困扰。

在数据的展示上，本次研究使用了 Echarts 框架，它是一款非常优秀的可视化开源库，并且能够兼容市面上常见的绝大部分浏览器，内容上有非常丰富的表格展示样式，还可以根据自己的需求个性化定制需要的数据可视化表格，非常方便我们提供更加直观、交互丰富的用户展示窗口，提升用户体验。

3.5.2 程序设计和展示

通过数据处理和研究，球员的场均数据越高，并且越平稳，证明这个球员在选取的时间内保持了较高的水准。同时，多个球员按照相同的标准来进行比并取出前五名，能够得出在对应的能力领域里，他们就是当前的佼佼者。并且通过对于一名球员的多个能力进行同时展示，能够很直观地了解到自己关注的球员的能力分布 [5]。



球员数值比较

哈登球员能力值分析

4 项目总结

本次的数据研究，通过对目标网站收集到的数据整理，按照需求来进行展示内容的选择，储存到对应的数据库，通过表格展示数据等步骤，运用 Flume 和 Kafka

的应对海量数据的过滤和缓存机制对球员数据进行传输，运用 Hive 联合 Spark 对数据进行筛选和整理，使用 mysql 来将需要展示的数据进行分类存储，最后结合 Echarts 的可视化方式将数据直观动态的展现出来。当然由于各种方面上的局限性，本次的研究只是在能力所及的范围内，运用于现有的技术体系，做出的一部分的具体实现。对于如何更好地将大量的数据在整理过后，能够以一种更加有效的方式来呈现给用户。还需要通过更多的学习和实践来实现。

【参考文献】

[1] 黄素萍 , 常加强 , 高妍 . 海量数据的分析研究 [J]. 科学技术创新 ,2020(15):60-61.
[2] 张趁香 . 基于 Hadoop 平台的海量数据分析和处理 [J]. 电脑编程技巧与维护 ,2019(01):95-97.
[3] 符添玮 . 大数据分析关键技术研究 [J]. 大众标准化 ,2020(02):125-126.
[4] 刘政宇 . 基于大数据的数据清洗技术及运用 [J]. 数字技术与应用 ,2019,37(04):92+94.
[5] 孙品一 , 周峰 . 探讨大数据时代下的数据可视化 [J]. 设计 , 2016(07):136-137.