MapReduce 课程设计：金庸的江湖

——分析小说中人物关系

朱庭纬，崔子寒，吴昌容

2019 年 7 月 16 日

目录

摘要

在常规的文本分析任务中，除了常规的语义分析，在数据层面上的统计和分析越来越重要。通过数据统计和分析，可以获得更丰富的知识。在本次大实验中，我们需要对金庸的武侠小说进行分析，通过数据预处理，构建关系图，运行 PageRank 算法和标签传播算法，我们可以判断哪些人物是主要人物。另外通过可视化的方法，可以直观的看出所有人物之间的关系。这个任务流程可以用于社交网络的构建和分析，具有比较好的实用性。

Abstract

In conventional text analysis tasks, in addition to conventional semantic analysis, statistics and analysis at the data level are becoming more and more important. More knowledge can be obtained through data statistics and analysis. In this experiment, we need to analyze Jin Yong's martial arts novels, through data preprocessing, constructing relational graphs, running PageRank algorithm and label propagation algorithm, we can judge which characters are the main characters. In addition, through the visual method, you can visually see the relationship between all characters. This task flow can be used for the construction and analysis of social networks, and has good practicability.

1 实验要求

1.1 数据预处理：分词

从原始的小说文本中，抽取与人物互动相关的数据，屏蔽掉与人物关系无关的文本内容，为后面基于人物共现的分析做准备。

输入：1. 金庸全本武侠小说文集（未分词）；2. 金庸武侠小说人名列表。

输出：分词后，仅保留人名的武侠小说全集。

1.2 特征提取：单词同现

在人物同现分析中，如果两个人在原文的同一段落中出现，则认为两个人发生了一次同现关系。需要对人物之间的同现关系次数进行统计，同现关系次数越多，则说明两人关系越密切。

输入：分词后仅保留人名的武侠小说全集。

输出：在金庸的所有武侠小说中，人物之间的同现次数。

1.3 归一化

获得人物之间的同现次数后，根据同现关系生成人物之间的关系图。在关系图中，人物是顶点，人物之间的互动关系是边。边的权值是两个人之间的共现概率。

输入：人物共现次数矩阵。

输出：权重归一化后的人物关系图。

1.4 PageRank

获得权重归一化化后的人物关系图后，可以进行数据分析任务。PageRan 是典型的数据分析任务，通过 PageRank 可以确定金庸武侠江湖中的“主角”是那些。

输入：权重归一化后的人物关系图。

输出：人物的 PageRank 值。

1.5 标签传播

标签传播（Label Propagation）是一种半监督的图分析算法，他能为图上的顶点打上标签，进行图顶点的聚类分析。通过在人物关系图上进行标签传播，可以将属于同一本书的人物聚为一类。

输入：权重归一化后的人物关系图。

输出：人物的标签信息。

1.6 输出整理

对于 PageRank 的结果，可以使用 MapReduce 任务按照 PageRank 值进行全局排序。

对于标签传播的结果，可以使用 MapReduce 任务将属于同一标签的任务输出到一起，便于结果查看。

2 小组分工

表 1: 小组分工

姓名	学号	任务
朱庭纬	161220186	数据预处理，特征提取，归一化，报告撰写
崔子寒	161220026	PageRank，结果整理，报告撰写
吴昌容	161220134	标签传播任务，结果整理，报告撰写

3 实验环境

表 2: 实验环境

操作系统	Ubuntu 18.04 LTS
hadoop 版本	2.7.1
jdk 版本	1.7.0
构建工具	Maven

4 编译、打包、运行

4.1 编译打包

由于使用 Maven 作为构建工具，所以可以很方便的添加依赖，我们在 pom.xml 文件中添加了表 3 所示的依赖:

表 3: 项目依赖

groupId	artifactId	version
org.apache.hadoop	hadoop-mapreduce-client-core	2.6.0
org.apache.hadoop	hadoop-test	1.2.1
org.apache.hadoop	hadoop-common	2.7.1
org.ansj	ansj_seg	5.1.6

其中，groupId 为 org.apache.hadoop 的三个依赖是 hadoop MapReduce API 的有关依赖。ansj_seg 是分词库的依赖，这里使用的是官网上最新的 5.1.6 版本。
为了在打包时将第三方的依赖和代码一起打包，使用 Maven 插件 org.apache.maven.plugins: maven-shade-plugin，通过设置目标为 shade，可以将依赖连同代码一起打包。打包的命令为：**mvn package -Dhttps.protocols=TLSv1.2**。由于我们使用的 jdk 版本是 1.7，而 1.7 默认的 https 协议是 TLSv1.1，现在 maven 仓库已经不再支持这个版本的协议，所以需要加上参数，否则在下载依赖时会出错。

4.2 运行方式

任务代码由三名成员合作编写，每人负责一个模块，然后将模块组合起来。运行时既可以通过运行主类 Main，依次运行所有任务，也可以分别运行某一个子任务。具体的运行方式如表 4 所示 (假设 jar 包名称为 jianghu.jar)：

表 4: 运行方式

指令	解释
hadoop jar jianghu.jar cn.nju.st13.Main arg0 arg1 arg2	arg0：小说文件夹路径；arg1：人名文件路径；arg2：输出结果路径
hadoop jar jianghu.jar cn.nju.st13.preprocess.Preprocess arg0 arg1 arg2 arg3	arg0：小说文件夹路径；arg1：人名文件路径；arg2：人物关系图输出路径；arg3：可选参数”-1”,”-2”,”-3”, 代表三种不同的方法
hadoop jar jianghu.jar cn.nju.st13.pagerank.PageRankDriver arg0 arg1 可选参数	arg0：人物关系图路径；arg1：排序处理后的 PageRank 结果路径；可选参数：-max_iter=n(n 为最大迭代次数，默认 15)，-retain_process=false/true(是否保留中间结果，默认 false)
hadoop jar jianghu.jar cn.nju.st13.labelprop.LabelProp arg0 arg1	arg0：人物关系图路径；arg1：整理后的标签传播算法结果路径。

5 算法介绍

5.1 任务 1

5.2 任务 2

5.3 任务 3

Algorithm 1 AdaBoost(x, y, n)

Input: 属性集 x，标签集 y, 基分类器数 n

Output: 含有 n 个基分类器的 AdaBoost 集成分类器

$models$:=[ ]
$weight[i] := \frac{1}{m}$ i = 1,2...m
$\alpha[i] := 0$,i = 1,2...n
for $i := 1$ *to* n: **do**
    从 x, y 中按照权重 $weight$ 自助采样，得到和 x, y 规模相同的数据集 d_i
    使用 d_i 训练决策树 $tree_i$
    用 $tree_i$ 对 x 进行预测，计算错误率 $error$
    if $error > 0.5$ **then**
        break
    end if
    $alpha[i] := \frac{1}{2} ln(\frac{1-error}{error})$
    将 $tree_i$ 添加至 $trees$[ ]，对于每一个样本的权重进行调整：
    如果预测正确，weight $\times exp(-\alpha[i])$
    如果预测错误，weight $\times exp(\alpha[i])$
    权重标准化
end for
return $trees$

5.4 任务 4

5.5 任务 5

6 优化

7 运行效果