第六周工作报告

丘骏鹏

2013-04-07 Sun

目录

1 本周工作

1. 实现了最小公共序列的动态规划算法，在一些小的数据上进行了简单的测试
2. 利用 Jsoup 完成了树的遍历，利用该序列进行计算，初步进行了测试
3. 阅读了一些 weka 的 tutorial，学习 weka 的用法
4. 代码重构和增加注释

2 一些实验数据

在已完成代码的基础上做了一些简单的测试，从 blog 中随机抽取了 10 篇文档（一开始取了 50 篇，但是由于计算速度太慢，改成 10 篇），利用 tag 序列求最大公共序列的方法，分别计算两两之间的相似性，统计数据如下：

```
Average Time: 1.9945767823333338(s)
Max similarity: 0.29230769230769227
Min similarity: 0.01552795031055898
```

增加测试文档到 20 篇，得到的数据如下：

```
Average Time: 2.298010719252631(s)
Max similarity: 0.36898395721925137
Min similarity: 0.0
```

说明：以上的 Average Time 表示计算两篇文档的相似度平均所花的时间（注：这个时间不包括预处理的时间，只是进行最长公共字串匹配算法的运行时间），从上面两个简单测试可以看到，大概平均需要 2s 左右。下面的 Max similarity 和 Min similarity 分别表示相似度的最大值和最小值，相似度的计算方法为：

$$1 - \frac{|lcs(t_1, t_2)|}{max(|t_1|, |t_2|)}$$

t_1, t_2 分别表示两个 tag 遍历序列。

所有的文档都先经过了简单的处理，去掉了一些无用的标签，简化了树结构。在这个基础上，通过 tag 序列计算最长公共字串的方法，每两篇文档需要花费 2s 的时间。考虑最坏的情况，我们的 blog 数据有 59998 篇，如果每两篇都进行计算，需要的时间非常之多，大约可以计算为：$60000 * 60000/2 * 2/3600 = 1000000$ 小时。当然，实际情况不会这么坏，可以加入多个优化，以上的简单计算只是为了说明数据量增大之后可能会造成计算时间的巨大增加。

现阶段可以考虑的优化包括：算法，多线程以及预处理。还包括代码本身的实现，我还需要再检查一下代码本身实现是否有问题，是否有较大的提高空间。

3 进度说明

由于本周清明节放假，上周的计划中“完成聚类”的工作未完成。此外花了一些额外的时间重构已有代码和增加必要注释，因此打算下周抓紧一些时间补上这部分的工作。

总体来说上周的进度没有完成已有要求，比较慢，下一周打算加快实现的速度。