查重程序实验报告

孙嘉玺
22920162204038

2018 年 5 月 1 日

目录

1 题目描述

给两个程序文件，判断程序是否存在抄袭，输出两个程序的相似度，根据相似都来判读是否存是抄袭。

2 算法实现

2.1 算法思想

查重算法主要借鉴最长公共字串的思想，不过需要调整一些东西。
第一就是文本的最小单位不应该是字符，而是一个单词。因为在抄袭的时候，是在抄袭关键字的级别上发生的。
另一个需要调整的就是要把文本中无效的字符除去防止干扰查重，其中一些字符就是换行符和空格，tab 键等，最终将文本删去无效字符，只留下关键字。

2.2 数据结构

存储文本的数据结构在 python 中表现为一个 list，list 中的元素是单词。存储动态规划结果的是一个 numpy array 来存储动态规划表。

2.3 程序运行环境

1. python
2. terminal
3. terminal 中使用命令 python FindCopy.py filename1 filename2

2.4 程序

本程序使用 python 编写

```
# coding: utf-8

from __future__ import division, print_function
from io import open
import numpy as np
```

```
import sys


def get_list(filename1, filename2):
    with open(filename1) as f:
        l = f.read()
        l = l.split()
        list1 = l
    with open(filename2) as f:
        l = f.read()
        l = l.split()
        list2 = l
    return list1, list2


def find_max_commen(seqs):
    dp = np.zeros(
        (len(seqs[0]), len(seqs[1]))
    )

    for i in range(0, dp.shape[0]):
        if seqs[0][i] == seqs[1][0]:
            dp[i][0] = 1

    for j in range(0, dp.shape[1]):
        if seqs[1][j] == seqs[0][0]:
            dp[0][j] = 1

    for i in range(1, dp.shape[0]):
        for j in range(1, dp.shape[1]):
            if seqs[0][i] == seqs[1][j]:
                dp[i][j] = max(dp[i-1][j-1]+1,
                               dp[i][j-1],
```

```
                                    dp[i-1][j],
                                    dp[i][j])

    return max(dp.reshape(-1))



if __name__ == "__main__":
    filename1 = sys.argv[1]
    filename2 = sys.argv[2]
    file_pair = get_list(filename1, filename2)
    commen_num = find_max_commen(file_pair)
    print("%s %s 相似度为: %.2f"
                %(filename1, filename2 ,(
                100 * commen_num / min(
                len(file_pair[0]), len(file_pair[1]))
                )), end='%\n')
```

3 实验数据 & 结果

3.1 实验数据

实验数据有两个 c++ 程序文件 file1.cpp file2.cpp
文件内容如下

- file1.cpp

```
#include <stdio.h>
#include <iostream>
#include <string>
#include <vector>

using namespace std;
```

```
int main() {
        printf("hello world");
        return 0;
}
```

- file2.cpp

```
#include <iostream>
#include <string>
#include <algorithm>
#include <vector>

using namespace std;

int main() {
        cout << "hello world" << endl;
        return ;
}
```

3.2 实验结果

程序输出结果如下

```
file1.cpp file2.cpp 相似度为： 42.11 %
```

4 实验总结

4.1 结果分析

从分析结果来看，发现程序的相似度是没有问题的，然而难点是如何设定门限来确定是否存在抄袭，这是程序难以解决的，如果真正的要判读出是否抄袭，仅仅判断相似度是不够的，可能还要需要人工校对。另一方面，程序规则是死的，如果人在了解程序逻辑之后很可能会对文本做一些手脚，来欺骗程序，这也是这个程序不足的地方。

4.2 总结

作为一个算法实验，得到这样的实验结果还可以接受，但是要放到实践中，使这个程序能够用起来，可能需要深入打磨算法，可能还需要一些机器学习的算法，学无止境，继续努力。