

哈尔滨工业大学 2018 年秋季学期

计算机学院本科生 “中文信息处理”课 实验报告（一）

报 告 题 目： 中文自动分词实验

姓 名： 姜思琪

学 号： 1160300814

学 生 专 业： 计算机科学与技术学院

任 课 教 师： 刘秉权

2018 年 12 月 24 日

报 告 正 文

1. 实验内容

- 1) 使用任意分词方法实现汉语自动分词；
- 2) 给出至少 1000 个句子的分词结果（以附件形式）；
- 3) 计算出分词结果的正确率，并给出计算依据；
- 4) 用实例说明所用分词方法分别对“交叉歧义”和“组合歧义”的处理能力；

2. 实验要求和目的

- 1) 自己准备词表；
- 2) 自己准备足够规模的语料；
- 3) 编程环境、汉字编码不限。
- 4) 提交实验报告，给出详细实验过程和结果；提交源代码和可执行程序。

3. 实验环境

win10 + Eclipse

4. 程序主要算法

正向匹配(FMM)：

基本算法：

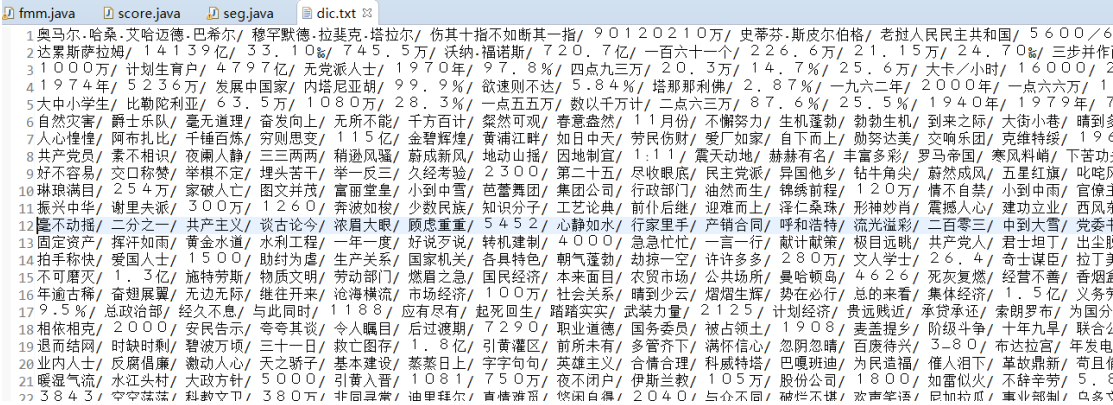
1. 设自动分词词典中最长词条所含汉字个数为 I；
2. 取被处理材料当前字符串序数中的 I 个字作为匹配字段，查找分词词典。
若词典中有这样的一个 I 字词，则匹配成功，匹配字段作为一个词被切分出来，转 6；
3. 如果词典中找不到这样的一个 I 字词，则匹配失败；
4. 匹配字段去掉最后一个汉字，I--；
5. 重复 2-4，直至切分成功为止；
6. I 重新赋初值，转 2，直到切分出所有词为止。

5. 实验过程

首先，准备语料。语料在 Untitled 2 中。是公共资源，从许多新闻中摘取出来，一共 1262 行。

1250 本报评论员
1251 做任何工作一定要从当地实际出发，这是一个老问题，也是一个在新的历史条件下，面对党的十五大提出新的历史任务，具有新的实际内容，需要以新的思维方式来研究和解决的重要问题。
1252 当地实际，不是一个点，不是一条线，也不是一个面，而是一个纵横交错的立体网络。它包含着极为丰富的内容：当地的优势，当地的劣势；当地的现状，当地的历史；当地的发展水平，
1253 要做到从当地实际出发，需要发扬深入实际、深入群众、调查研究的工作作风，需要有立足当前、着眼未来、总揽国内外全局的胸怀和眼光，需要兢兢业业、脚踏实地、埋头苦干的拼搏。
1254 在建设统一、开放、竞争、有序的大市场的新形势下，『从当地实际出发』，只是把眼光盯在当地实际上已经不够，还应站得更高，从全局的高度看『当地实际』。因为在市场经济条
1255 因而，要真正做到从当地实际出发，除了静下心来，扑下身子，认真对当地的实际作一番深入透彻的研究之外，还必须坚持改革开放，转变我们的观念，将眼光放到国内外市场上，将当
1256 江苏年初布置年中检查年底总结办实事靠制度
1257 本报南京 1 月 4 日电记者裴永泉报道：每年年末，江苏省都要研究决定第二年着重抓的 2 0 多件实事，用省政府一号文件下发，同时在报上公布，接受群众的监督，年初布置，年中检查，
1258 江苏所确定的实事，都是着眼于提高人民群众的物质生活水平和精神生活质量，与群众的生产、生活息息相关的。近几年来，江苏每年都列出基础设施建设的目标任务，环环相扣，抓出
1259 教育科技工作也是江苏办实事的重点内容之一。经过近几年的努力，全省已基本普及九年制义务教育，基本扫除青壮年文盲。
1260 1 9 9 2 年和 1 9 9 3 年，江苏把扶持贫困地区加快经济发展列入办实事计划，每年扶持 3 0 万人脱贫，同时大力发展劳动力市场，积极扩大就业，把城镇待业率控制在 2 . 5 % 左右。
1261 从这七年执行的情况看，年初决定要办的实事绝大多数都得到了较好的落实，极少数因客观原因暂时难以落实的，也都如实地向群众解释清楚，取得了群众的谅解。
1262 省委书记陈焕发表示，今后要把群众办实事作为一项制度长期坚持下去。

然后，根据语料先形成词典。根据已经划分好的词（存储在 Untitled 1 中），按行读入，用正则表达式进行剪切，将其存入 Map 表，然后根据词长度排序，再将其写入 dic.txt 文件中，形成词典（运行程序 dic.java 即可得到词典）。词典的存储形式是词之间以 / 隔开。如下图：



我的字典的存储顺序是按照词语的字数长短排列的。存在 dic.txt 文件中，字数长度由长到短。我发现，大于 7 个长度的词已经很少了，而 1~4 字的词语却有很多。所以在实现词的匹配的时候，我主要采用了二分法的思路。假设选定一段字，长度为 X，如果 X 大于 7，则从词典的最开始匹配；如果 X 等于 1，则从词典的最后开始往前比较；如果 X 在 1 和 7 之间，则每次均取字典相应范围内的中间的词的词的长度，与 X 作比较，判断 X 应落在哪一侧，如果选取的字典相应范围的中间词长度等于 X，则分别向其左边和右边挨个词对比，判断是否相同，直到词的长度不一致为止。

我将这种思路用伪代码表示：

```
boolean find(String str) {  
    if (str.length() > 7) {  
        for (文件从前向后开始对比){  
            if (相同)
```

```

        return true;
    }
} else if (str.length() == 1)
    for (文件从后向前开始对比) {
        if (相同)
            return true;
    }
else {
    int a = 0;    //字典开始
    b = dictlen - 1;    //字典结束
    int mid = (a + b) / 2;
    while (mid 位置的词长度不等于 str 的长度) {
        if (mid 位置的词长度大于 str 的长度) {
            a = mid + 1;
            mid = (a + b) / 2;
        } else if (mid 位置的词长度小于 str 的长度) {
            b = mid - 1;
            mid = (a + b) / 2;
        }
    }
    for (mid 位置的词长度等于 str 的长度且向文件开端依次
比较词) {
        if (mid 位置的词与 str 相同) {
            return true;
        }
    }
    for (mid 位置的词长度等于 str 的长度且向文件末尾端依
次比较词) {
        if (mid 位置的词与 str 相同)
            return true;
    }
}

```

```

    }
}

return false;
}

```

再编写正向匹配算法。根据正向匹配算法基本思路，在 fmm.java 中写出主要函数 String FMM(String str)，它的功能就是实现正相匹配。关键代码如下：

```

public static String FMM(String str) throws IOException {
    String s = "";
    if (str == null)
        s = "";
    else {
        int max = dict[0].length() - 1;
        if (str.length() < max)
            max = str.length();
        int start = 0, end = str.length();
        while (start < str.length()) {
            if (start + max < str.length() + 1)
                end = start + max;
            else
                end = str.length();
            while (!find(str.substring(start, end)) && end > -1 && end != start) {
                end--;
                if (end == start)
                    break;
            }
            if (end == start) {
                end = end + 1;
            }
            s = s + str.substring(start, end) + " / ";
            start = end;
        }
    }
    return s + "\n";
}

```

给它的语料在 Untitled 2.txt 中，然后依次在词典 dic.txt 里进行匹配，划分词即可。划分的结果写入 seg_FMM.txt 中。在划分词时，处理歧义。

交叉歧义：在字段 SABC 中 AB 和 BC 都是词则字段 S 称为交集型歧义切分字段 B 称为交叉段，其中 A、B、C 为字串。如脑海里因为“脑海”和“海里”都是词那么这个短语就可以分成“脑海 里”和“脑 海里”。

组合歧义：在字段 SAB 中 S、A 和 B 三者都分别成词则 AB 为组合型歧义切分字段其中 A、B 为字串。组合歧义必须根据整个句子来判断如在句子“他具有非凡的才能”中“才能”是个词，但在句子“只有他才能胜任该职位”中“才能”就不是一个词。

FMM 在处理歧义方面表现一般，交叉歧义还可以通过上下文稍加识别处理，组合歧义应该说没什么办法。

然后，对代码的准确性做测试。通过运行 score.java 就可以把刚刚我们

分好的词 (seg_FMM.txt) 与正确的语料 (Untitled 1.txt) 进行对比, 并将正确率输入到 score.txt 文件。根据查找资料可得, 有三个数值: 准确率、召回率和 F-评价。

- 准确率 Precision(P)

$$P = \frac{tp}{tp + fp}$$

- 召回率 Recall(R)

$$R = \frac{tp}{tp + fn}$$

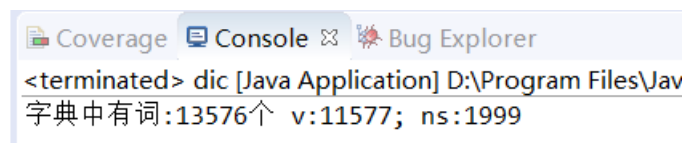
- F1度量 F1-measure(F1)

$$F1 = \frac{2PR}{P + R}$$

6. 实验结果

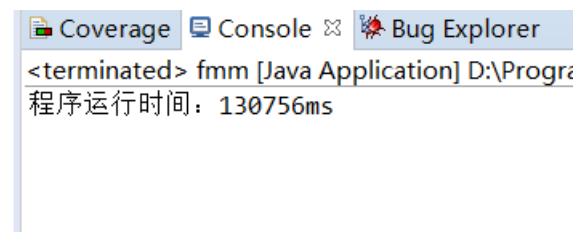
程序运行结果截图如下:

形成字典控制台输出:



```
<terminated> dic [Java Application] D:\Program Files\Java
字典中有词:13576个 v:11577; ns:1999
```

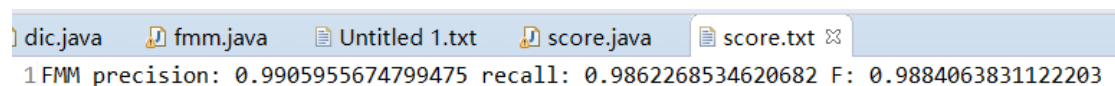
正向匹配之后控制台输出运行时间:



```
<terminated> fmm [Java Application] D:\Program Files\Java
程序运行时间: 130756ms
```

正确率计算:

分别是准确率、召回率、F-评价。



```
dic.java fmm.java Untitled 1.txt score.java score.txt
1 FMM precision: 0.9905955674799475 recall: 0.9862268534620682 F: 0.9884063831122203
```

所有程序和文件:

```
· 📁 > one
  > 📄 dic.java
  > 📄 fmm.java
  > 📄 score.java
  📄 dic.txt
  📄 score.txt
  📄 seg_FMM.txt
  📄 Untitled 1.txt
  📄 Untitled 2.txt
```

7. 实验结论和体会

通过本次实验，我体会到如何自己构造词典和给句子分词，人脑会很快的给句子断句，而机器实现起来确要复杂一些。我在本次实验中，遇到了并且思考了一些问题。

第一个要攻克的难关：大量的语料和构成的词典。在构成词典过程中，必须人为的将语料划分正确的词，才能存入词典。这将是一项繁琐庞杂的工作。

第二个想到的问题：现在只是 1200 句话，运行很快，当成千上万句话时，构成词典将花费大量时间，而在数以万计的词中匹配，将花费几小时的时间。如何提高效率是我想到的第二个问题。