# 中文分词系统

**姓 名：** 陈明新

**学 号：** 2120140994

**姓 名：** 魏明伟

**学 号：** 2120141061

**姓 名：** 蒋 斌

**学 号：** 2220140499

**姓 名：** 佟安格

**学 号：** 2120141050

目录

[1.问题描述 2](#_Toc408937509)

[2.相关工作 2](#_Toc408937510)

[3.系统框架和算法设计 3](#_Toc408937511)

[3.1系统整体框架 3](#_Toc408937512)

[3.2基于HMM模型分词算法设计 4](#_Toc408937513)

[3.2.1构建HMM模型（HMM的学习问题） 4](#_Toc408937514)

[3.2.2viterbe算法（HMM的解码问题） 5](#_Toc408937515)

[3.3正向最大匹配分词算法设计 6](#_Toc408937516)

[3.4逆向最大匹配分词算法设计 7](#_Toc408937517)

[4.任务分工 9](#_Toc408937518)

[5.系统运行步骤 10](#_Toc408937519)

[5.1基于HMM模型 11](#_Toc408937520)

[5.1.1训练集训练 11](#_Toc408937521)

[5.1.2文件输入测试 14](#_Toc408937522)

[5.1.3直接输入测试 16](#_Toc408937523)

[5.2正向最大匹配 17](#_Toc408937524)

[5.2.1文件输入测试 18](#_Toc408937525)

[5.2.2直接输入测试 18](#_Toc408937526)

[5.3逆向最大匹配 19](#_Toc408937527)

[5.3.1文件输入测试 19](#_Toc408937528)

[5.3.2直接输入测试 19](#_Toc408937529)

[6.实验和分析 20](#_Toc408937530)

[6.1基于HMM模型 20](#_Toc408937531)

[6.1.1文件输入测试结果 20](#_Toc408937532)

[6.1.2直接输入测试结果 20](#_Toc408937533)

[6.1.3结果集评估 21](#_Toc408937534)

[6.2正向最大匹配 22](#_Toc408937535)

[6.2.1文件输入测试结果 22](#_Toc408937536)

[6.2.2直接输入测试结果 22](#_Toc408937537)

[6.3逆向最大匹配 23](#_Toc408937538)

[6.3.1文件输入测试结果 23](#_Toc408937539)

[6.2.2直接输入测试结果 24](#_Toc408937540)

[7.总结 25](#_Toc408937541)

# 1.问题描述

中文分词 (Chinese Word Segmentation) 指的是将一个汉字序列切分成一个一个单独的词。分词就是将连续的字序列按照一定的规范重新组合成词序列的过程。我们知道，在英文的行文中，单词之间是以空格作为自然分界符的，而中文只是字、句和段能通过明显的分界符来简单划界，唯独词没有一个形式上的分界符，虽然英文也同样存在短语的划分问题，不过在词这一层上，中文比之英文要复杂的多、困难的多。

而对于中文分词也有很多种算法，基本算法主要分为基于词典的方法、基于统计的方法和基于规则的方法。

基于词典的方法是指按照一定策略将待分析的汉字串与一个“大机器词典”中的词条进行匹配，若在词典中找到某个字符串，则匹配成功。按照扫描方向的不同包括正向匹配和逆向匹配，按照长度的不同分为最大匹配和最小匹配。在基于词典的方法中，我们采用了正向最大匹配和逆向最大匹配的方法进行分词。

而基于统计的分词方法，没有词典，主要思想是在上下文中，相邻的字同时出现的次数越多，就越可能构成一个词，因此字与字相邻出现的概率或频率能较好的反映词的可信度。主要的统计模型包括N元文法模型(N-gram)，隐马尔科夫模型(Hidden Markov Model, HMM)。在基于统计的分词方法中，我们使用了隐马尔科夫模型来进行分词。

基于规则的方法，主要思想就是通过模拟人对句子的理解，达到识别词的效果，基本思想是语义分析，句法分析，利用句法信息和语义信息对文本进行分词。自动推理，并完成对未登录词的补充是其优点。但是这种方法现在发展还不成熟，需要继续发展和研究。

因此在我们的中文分词系统中，采用了基于词典的方法：正向最大匹配和逆向最大匹配，以及基于统计的方法中的隐马尔科夫(HMM)模型。

# 2.相关工作

现如今已经有很多的开源中分分词系统，而且效果都还不错。下面介绍几种比较常见的中文分词项目。

　SCWS，Hightman开发的一套基于词频词典的机械中文分词引擎，它能将一整段的汉字基本正确的切分成词。采用的是采集的词频词典，并辅以一定的专有名称，人名，地名，数字年代等规则识别来达到基本分词，经小范围测试大概准确率在 90% ~ 95% 之间，已能基本满足一些小型搜索引擎、关键字提取等场合运用。45Kb左右的文本切词时间是0.026秒，大概是1.5MB文本/秒，支持PHP4和PHP 5。

ICTCLAS，这是最早的中文开源分词项目之一，ICTCLAS在国内973专家组组织的评测中活动获得了第一名，在第一届国际中文处理研究机构SigHan组织的评测中都获得了多项第一名。ICTCLAS3.0分词速度单机996KB/s，分词精度98.45%，API不超过200KB，各种词典数据压缩后不到3M。ICTCLAS全部采用C/C++编写，支持Linux、FreeBSD及Windows系列操作系统，支持C/C++、C#、Delphi、Java等主流的开发语言。

HTTPCWS，是一款基于HTTP协议的开源中文分词系统，目前仅支持Linux系统。HTTPCWS 使用“ICTCLAS 3.0 2009共享版中文分词算法”的API进行分词处理，得出分词结果。HTTPCWS 将取代之前的 PHPCWS 中文分词扩展。

# 3.系统框架和算法设计

## 3.1系统整体框架

系统主要包括选择分词算法和进行数据的测试。首先需要选择进行分词的算法，包括基于HMM模型的分词算法，正向最大匹配分词算法和逆向最大匹配分词算法。选择了分词算法之后即可进行分词操作。

若选择的是基于HMM模型的分词算法，需要先进行训练集的训练，得到训练的统计数据，然后再进行数据测试。数据测试包括了文件输入测试，进行大规模数据的测试，也包括了直接输入测试，测试一句话或多句话进行简单测试。

对于正向最大匹配和逆向最大匹配则不需要进行训练集的训练，因为其是基于词典的方法，需要词典，而词典我们已经事先载入程序中了。正向最大匹配和逆向最大匹配也包括了文件输入测试和直接输入测试。

系统的整体框架如图3-1所示。



图3-1 系统整体框架

## 3.2基于HMM模型分词算法设计

### 3.2.1构建HMM模型（HMM的学习问题）

#### 3.2.1.1中文分词在HMM中的五元组参数

HMM模型包含了一个五元组的参数，分别是两个状态集合和三个状态矩阵。两个状态集合分别为隐含状态S和可观察状态O；三个概率矩阵分别为初始状态概率矩阵P、转移矩阵A和发射矩阵B。 想要利用HMM进行中文分词首先应该找出HMM中的五元组对应中文分词中的哪些元素。

在我们实现的基于HMM的中文分词程序中，隐含状态S对应的是集合{B,M,E,S}，其中B表示的是词的开始单字，M表示的是位于词中间的单字，E表示的是词的结束单字，S表示的是单字成词。比如“北京市”按照BMES划分就应该为“北/B京/M市/E”，“我”则应为“我/S“；可观察状态O表示的是训练语料出现的所有汉字，我们所使用的训练语料就是老师给我们的训练集；初始概率矩阵P表示的是BMES各个状态出现的概率；转移矩阵A表示BMES各个状态之间的转换概率；发射矩阵B表示的是在BMES各个状态各汉字出现的概率。

#### 3.2.1.2中文分词的HMM参数计算

在本文实现的中文分词程序中，采用的是最大似然估计法来获得HMM的三个概率矩阵。

对于初始概率矩阵P有:

对于转移矩阵A有：

对于发射矩阵B有：

由于矩阵B可能会出现很多为0 的数据项，所以对矩阵B进行了加1平滑。故矩阵B为：

### 3.2.2viterbi算法（HMM的解码问题）

在上一节中我们已经构建了中文分词的HMM模型了，那么下一步就是如何来对一个给定的句子进行分词。这个被称为HMM的解码问题，即根据可观察状态的序列找到一个最可能的隐藏状态序列（概率最大）。在本文实现的程序中采用的是viterbi算法。

viterbi算法是一种动规算法，它的本质其实和TSP（旅行商问题）类似，都是求多个重叠子问题的一个最优解。举例来说，如果要求解问题空间大小为t的最优解，那么该最优解肯定是在问题空间大小为t-1是最优解的情况下获得的。理解viterbi算法之后要实现中文分词也就非常简单了。

利用viterbi算法对给定的句子进行分词可以按照如下几步进行：

（1）计算初始概率：delta[0][i]=P[i]\*B[i][word\_index];word\_index表示汉字的数字编码值。

（2）递归计算：

delta[t][i] = max{delta[t-1][j]\*A[j][i]|j∈[0,3]}\* B[i][word\_index];

并用path[t][i]记录下使delta[t-1][j]\*A[j][i]最大的j值。t表示的是给定句子的第t个汉字。

（3）取最大概率序列：经过上述的递归计算后，取max{delta[row-1][k]|k∈[0,3]}的k，即当前最可能的隐藏序列。

（4）回溯：通过path回溯找出隐藏序列，如何是E或者S则分成一个词语。

## 3.3正向最大匹配分词算法设计

正向最大匹配的大致思想是，给定一个需要分词的中文字符串，从左至右将待分词文本中的几个连续字符与词表中的词进行匹配，如果是一个词就记录下来，否则通过减少一个单字，继续比较，一直还剩下一个单字则终止，如果该单字串无法切分，则作为未登录处理。

我们组设计的算法中，包括三个词库，一个是中文词典Dictionary.txt，一个是姓氏词典Surname.txt，还有一个是中文数字词典Number.txt。Dictionary.txt是主要的词库词典，Surname.txt用于简单处理中文姓氏以此能够分出一些中文姓名，而Number.txt主要是用于处理中文数字的分割。

对于最大正向匹配算法来说，输入一个待分割的中文字符串，我们以8个字作为最大匹配个数，首先匹配该字符串的前8个字在Dictionary词典中是否存在，若存在则作为一个词进行分割，然后将记录分割位置的i增加继续判断和分割剩余字符串。若不存在，则将匹配个数减为7，继续匹配词典，以此类推，直到若出现匹配个数减至1都未在词典中查找到，则开始匹配Surname和Number中的词，也是相同的思路。若在Surname和Number中依旧找不到，则在存有中文符号以及阿拉伯数字数组中进行查找并分割。

通过以上思路便可以将一个中文字符串进行分词，达到一定的分词效果。我们将该算法封装成了一个函数forwardSegment()，算法的大致流程如图3-2所示。



图3-2 正向最大匹配算法流程

## 3.4逆向最大匹配分词算法设计

逆向最大匹配基本原理和正向最大匹配大致相同，只是匹配的方向相反，大致思想是，给定一个需要分词的中文字符串，从右至左将待分词文本中的几个连续字符与词表中的词进行匹配，如果是一个词就记录下来，否则通过减少一个单字，继续比较，一直还剩下一个单字则终止，如果该单字串无法切分，则作为未登录处理。

对于逆向最大匹配算法来说，输入一个待分割的中文字符串，以8个字作为最大匹配个数，从右往左开始匹配，首先匹配该字符串的最后8个字在Dictionary词典中是否存在，若存在则作为一个词进行分割，然后将记录分割位置的i减少继续判断和分割前面剩余字符串。若不存在，则将匹配个数减为7，匹配最后7个字，继续匹配词典，以此类推，直到若出现匹配个数减至1都未在词典中查找到，则开始匹配Surname和Number中的词，也是相同的思路。若在Surname和Number中依旧找不到，则在存有中文符号以及阿拉伯数字数组中进行查找并分割。

通过以上思路便可以将一个中文字符串进行分词，达到一定的分词效果。我们将该算法封装成了一个函数backwardSegment()算法大致流程如图3-3所示。



图3-3 逆向最大匹配算法流程

# 4.任务分工

此次大作业我们组的成员均非常认真地完成了各自的任务。其中陈明新同学作为组长在前期工作中查找了相关资料，主要完成了编程过程中的逆向最大匹配和正向最大匹配的编码工作，也完成了基于HMM模型中的部分编码工作，包括求解HMM模型中的三个状态矩阵的部分编码，同时也完成了大部分系统界面的编写，并对最后的代码进行了整合，此外还完成了文档的大部分内容的编写和PPT的讲解。

魏明伟同学在前期工作中查找了很多相关资料，也参与了部分基于HMM模型的编程工作，包括了求解HMM模型的三个状态矩阵和viterbi算法的部分编码工作，同时参与了部分文档的编写，另外完成了系统的测试工作以及结果的评估测试工作，此外魏明伟同学还主要完成了我们组演讲的PPT的制作。

而蒋斌同学查找了相关资料，并主要完成了基于HMM模型算法的整体框架的搭建以及HMM模型的大部分编程工作，包括求解三个状态矩阵以及使用viterbi算法进行求解隐含状态序列最优解的编码工作等，同时参与了部分文档的编写，并对程序进行了调试和优化。

佟安格同学参与了部分编码工作，其中包括了正向最大匹配和逆向最大匹配的部分编码工作，同时参与了部分系统界面的编写，并对程序进行了调试工作，此外还参与了系统的测试工作。

# 5.系统运行步骤

我们使用的是Java语言进行编程的，为方便使用，我们加入了Java中的图形组件swing组件进行用户界面的编写。打开程序，可以看到如图5-1所示的界面。

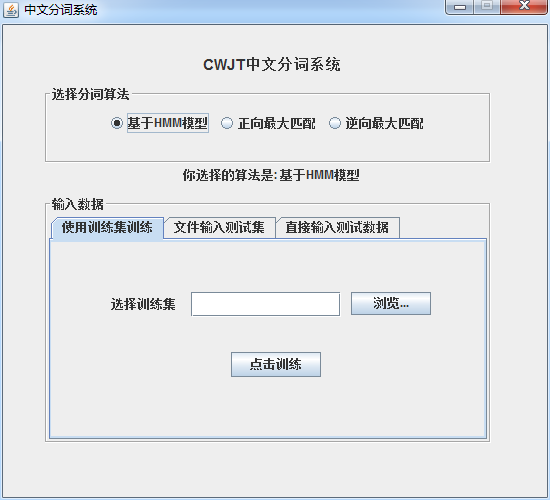


图5-1 CWJT中文分词系统界面

这就是我们组开发的CWJT中文分词系统的界面，选择分词算法部分包括了三种算法，分别是基于HMM模型，正向最大匹配和逆向最大匹配。要使用某种算法进行分词便可点选对应算法然后进行分词操作。下面分别介绍这三种不同的算法的操作步骤。

## 5.1基于HMM模型

点击选择分词算法一栏中的基于HMM模型，就如图5-1所示那样，这里就不再截图累述。

### 5.1.1训练集训练

基于HMM模型的中文分词，没有词典，需要使用训练集先进行训练。点击输入数据一栏中的“使用训练集训练”，点击浏览按钮，选择训练集文件，如图5-2所示。

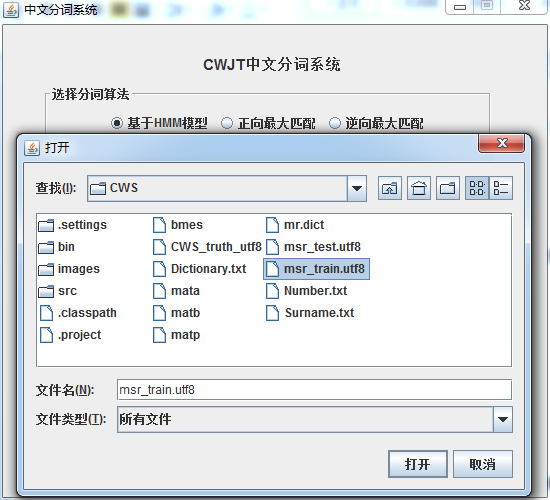


图 5-2 训练集文件选择

选择训练集文件之后，点击训练，系统就开始对训练集进行训练了。如图5-3所示显示的训练等待界面。



图5-3 训练集训练等待

训练完成之后会弹出训练成功的对话框，如图5-4所示。

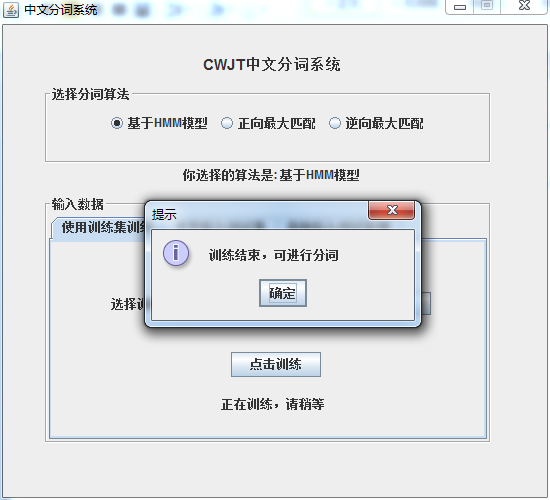


图5-4 训练集训练成功提示

训练完成之后便可进行测试集测试了。

### 5.1.2文件输入测试

文件输入测试主要是用于测试大规模数据的分词，点击输入数据一栏中的“文件输入测试集”，如图5-5所示显示的界面。



图5-5 基于HMM模型文件输入测试界面

可以看到在文件输入测试一栏中，包括了选择测试集和结果集存储目录，点击浏览按钮分别选择测试集文件和结果集存储目录，如图5-6所示。



图5-6 文件测试集输入

选择了测试集和结果集存储目录之后，点击开始分词，便进行分词，如图5-7所示是分词的等待界面。



图5-7 基于HMM模型文件输入测试分词等待界面

分词结束之后会弹出分词结束的提示，同时提示在结果集存储目录下查找结果集文件。如图5-8所示即为分词结束之后的弹出提示界面。



图5-8 基于HMM模型分词结束提示界面

### 5.1.3直接输入测试

直接输入测试数据是用于测试小数据，直接输入一句话或多句话进行测试。点击输入数据一栏中的“直接输入测试数据”，如图5-9所示显示的界面。



图5-9 基于HMM模型直接输入测试界面

可以看到，在左侧是输入测试数据的文本框，右侧是显示输出分词结果的文本框。若要输入数据进行分词测试，则在左侧文本框中输入，输入完之后点击开始分词按钮。具体测试效果会在第6部分进行介绍，这里不再累述。

## 5.2正向最大匹配

点选选择分词算法一栏中的正向最大匹配，选择正向最大匹配算法进行中文分词。如图5-10所示，为点选正向最大匹配之后的界面。

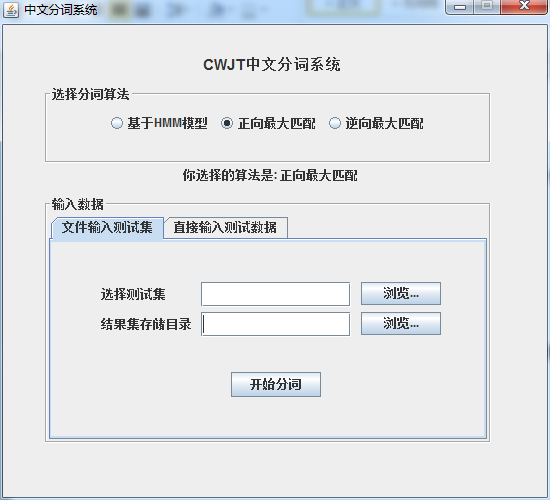


图5-10 正向最大匹配界面

在正向最大匹配算法中，没有了选择训练集进行训练的步骤，因为正向最大匹配算法是基于词典的方法。

### 5.2.1文件输入测试

操作和基于HMM模型过程一样，见5.1.2。

### 5.2.2直接输入测试

操作和基于HMM模型过程一样，见5.1.3。

## 5.3逆向最大匹配

点选选择分词算法一栏中的逆向最大匹配，选择逆向最大匹配算法进行中文分词。如图5-11所示，为点选逆向最大匹配之后的界面。

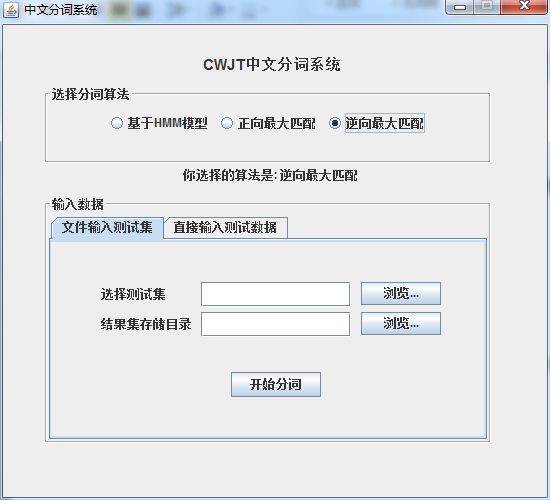


图5-11 逆向最大匹配界面

### 5.3.1文件输入测试

操作和基于HMM模型过程一样，见5.1.2。

### 5.3.2直接输入测试

操作和基于HMM模型过程一样，见5.1.3。

# 6.实验和分析

这一部分主要是介绍系统运行之后的结果，也就是中文分词的结果。同时我们使用开源评估程序对这三种不同的方法的结果进行了评估分析。

## 6.1基于HMM模型

### 6.1.1文件输入测试结果

我们测试的测试集是老师给我们的测试集msr\_test\_utf8。分词结果如图6-1所示，这里仅给出部分分词结果的截图。

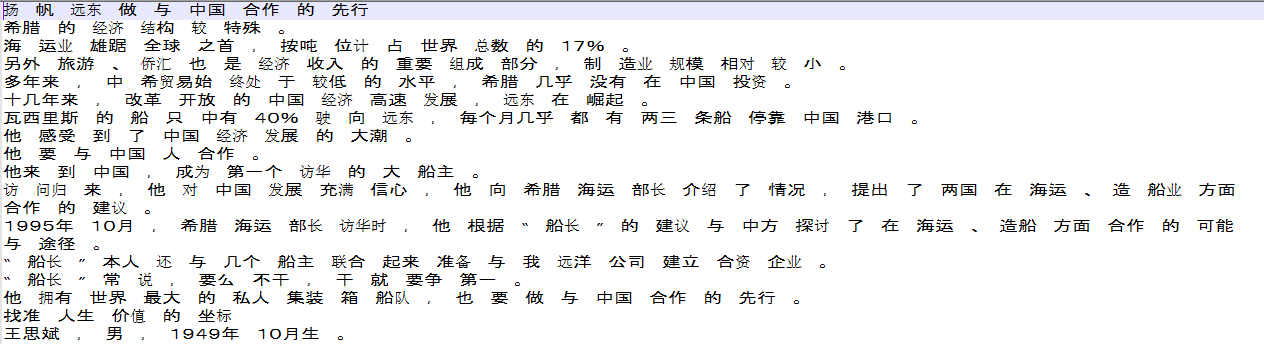


图6-1 基于HMM模型的分词结果部分截图

可以看出整体的分词效果还算可以，但是有些部分的分词结果也有不准确的地方。因为基于HMM模型的方法是一种基于统计的方法，需要训练集进行训练，而这种统计的方法很大程度上也就依赖于训练集。

### 6.1.2直接输入测试结果

我们也可以选择直接输入测试数据进行测试，主要针对小数据进行测试，对一句话或多句话进行简单测试。如图6-2所示为直接输入测试一些数据的测试结果。

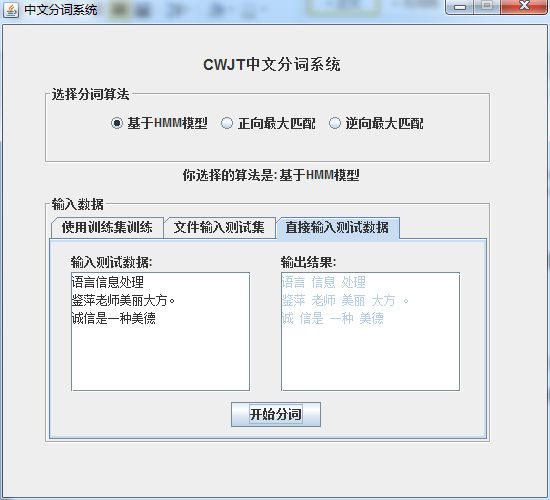


图6-2基于HMM模型直接输入测试结果

可以看出基于HMM模型的方法其实对于人名的分词可以达到一个较好的效果，而基于词典的方法不一定能够做到这种效果。

### 6.1.3结果集评估

我们采用了开源工具icwb2-data对结果集进行了评估，作为该算法的分词效果的一个评估。如图6-3为其评估结果。

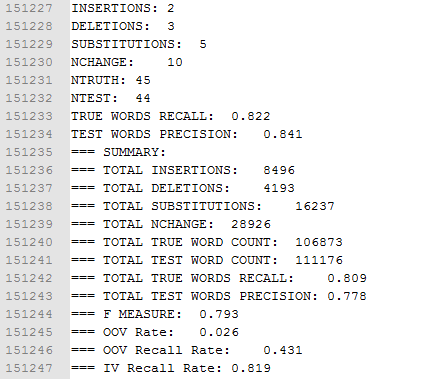


图6-3 基于HMM模型的分词评估

可以看到正确率为77.8%，召回率为80.9%，F值为79.3%。效果也只是差强人意。前面也说到，因为基于HMM模型的方法是一种基于统计的方法，需要训练集进行训练，而这种统计的方法很大程度上也就依赖于训练集。

## 6.2正向最大匹配

### 6.2.1文件输入测试结果

测试集依然是老师给我们的测试集msr\_test\_utf8。分词结果如图6-4所示，这里仅给出部分分词结果的截图。

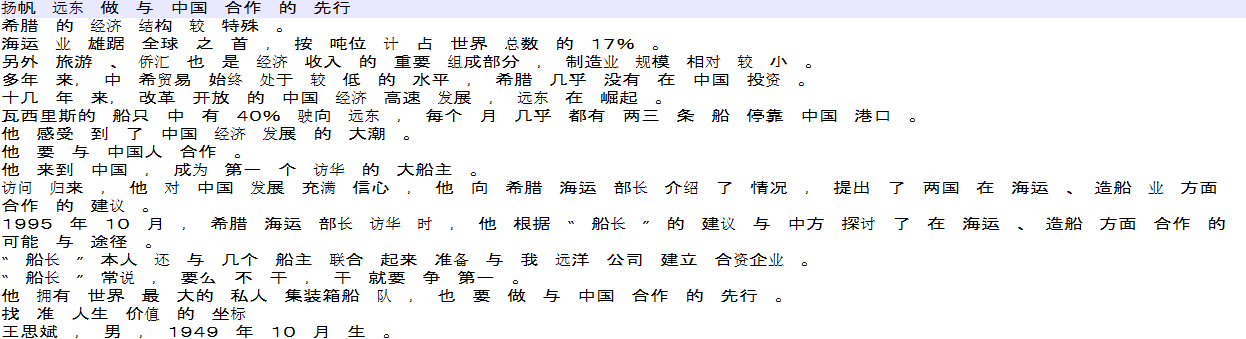


图6-4 正向最大匹配分词结果截图

可以看出整体的分词效果还算不错，但是也有的分词结果有不准确的地方。因为正向最大匹配的方法是一种基于词典的方法，这种方法很大程度上也就依赖于词典的规模大小。

### 6.2.2直接输入测试结果

我们也可以选择直接输入测试数据进行测试，主要针对小数据进行测试，对一句话或多句话进行简单测试。如图6-5所示为直接输入测试一些数据的测试结果。

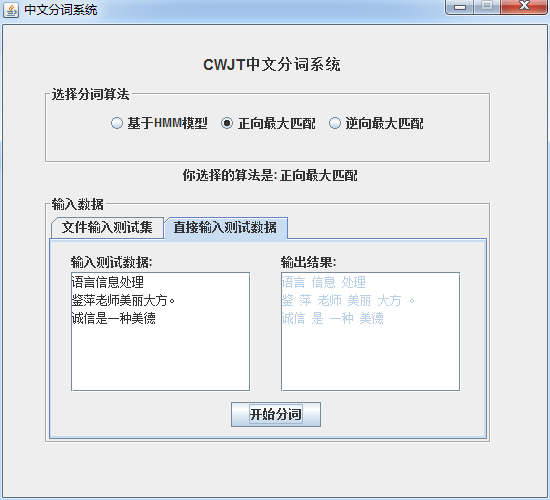


图6-6 正向最大匹配直接输入测试结果

可以看到，这里对于人名的分词效果不佳，因为正向最大匹配的方法是基于词典的方法，即使在我们的算法中加入了姓氏词典，可以处理一些人名的分词，但对人名的分词依然效果不佳，若要进行人名的分词需要使用基于统计的方法和基于规则（语义）的方法进行分析。

## 6.3逆向最大匹配

### 6.3.1文件输入测试结果

测试集依然是老师给我们的测试集msr\_test\_utf8。分词结果如图6-7所示，这里仅给出部分分词结果的截图。

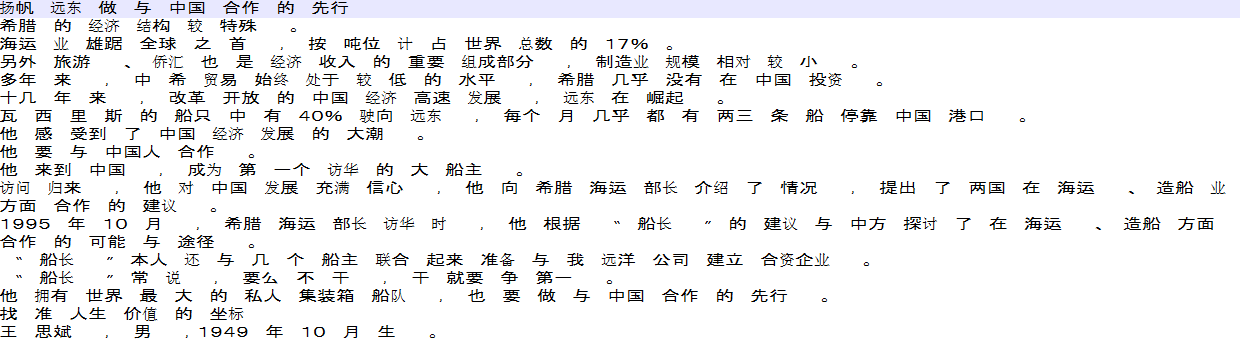


图6-7 逆向最大匹配分词结果截图

可以看出整体的分词效果还算不错，但是也有的分词结果有不准确的地方。因为逆向最大匹配的方法是一种基于词典的方法，这种方法很大程度上也就依赖于词典的规模大小。

### 6.2.2直接输入测试结果

我们也可以选择直接输入测试数据进行测试，主要针对小数据进行测试，对一句话或多句话进行简单测试。如图6-8所示为直接输入测试一些数据的测试结果。

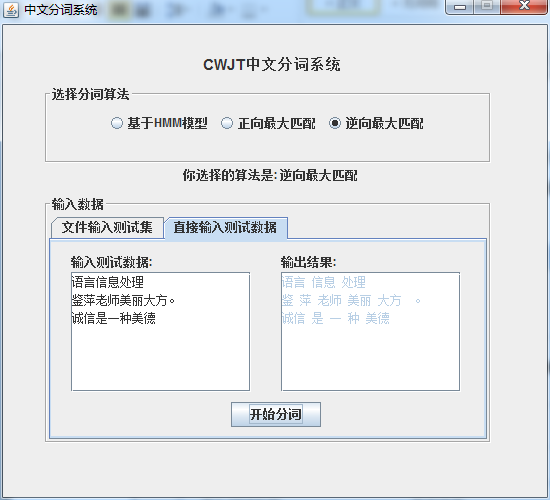


图6-8 逆向最大匹配直接输入测试结果

可以看到，这里对于人名的分词效果不佳，因为逆向最大匹配的方法和正向最大匹配的方法一样，也是基于词典的方法，对人名的分词效果不佳，若要进行人名的分词需要使用基于统计的方法和基于规则（语义）的方法进行分析。

# 7.总结

中文分词的算法主要可以分为三类，基于词典的方法、基于统计的方法和基于规则的方法。在此次大作业中，我们组采用了基于词典方法中的正向最大匹配和逆向最大匹配以及基于统计方法的隐马尔科夫(HMM)模型。正向最大匹配和逆向最大匹配的分词效果主要依赖于词典的规模，而HMM模型主要依赖于训练集的规模。对于这三种算法最后的效果和评估结果在第6部分已详细介绍，可以看到效果虽然不算太差，但也并没有很好。

如果能够将正向最大匹配和逆向最大匹配与HMM模型结合处理，效果会更佳，同时若加上语义信息的分析会得到更好的处理效果。也就是说若能结合基于词典，基于统计的方法以及基于规则（语义）的方法进行中文分词，将会达到非常好的效果。而基于规则（语义）的方法现如今还不成熟，需要进一步发展和研究。