南京農業大學
NANJING AGRICULTURAL UNIVERSITY

数字人文下的汉字分词

目录

CONTENT

- 汉语分词基本知识
- 自动分词在数字人文研究中的应用背景
- 非物质文化遗产文本自动分词系统

南京農業大學
NANJING AGRICULTURAL UNIVERSITY

汉语自动分词基本知识

什么是自动分词

使用/计算机/将/字符串/自动/转换/为/词串

◆为什么要分词?

●文本分析的第一步

●中文信息处理

●英语、日语

◆分词带来的帮助

●信息检索的预处理：提高查准率

●语音合成的预处理：降低读音复杂性

●汉字识别的后处理：提高识别正确率

●语音识别的后处理：提高识别正确率

●计算机辅助词典编撰：新词、新义项获取

汉语自动分词的三个里程碑

- ◆分词规范（国标、台湾、ISO）
- ◆分词词表（体现各自的分词规范）
- ◆分词竞赛（带标语料库）
  - 搁置“什么是词”的分歧
  - 专注于分词方法，特别是机器学习方法
  - SIGHAN2003以后

分词规范

◆什么是词?

- “词是最小的能够独立活动的有意义的语言成分”—朱德熙
- “词是具有语音形态，又能表示特定意义，且能在句法上单独出现或与其他的词共同形成词组的最小的单位”—汤廷池

◆汉语的词和非词界限不清

- 词还是词素?
  - 楼、院、氧、叶、虎、云、时
- 词还是短语?
  - 鸡蛋、鸭蛋
  - 高射、高射炮、高射机关枪
  - 人造纤维、人造丝、人造革
  - 大型彩色纪录片
  - 多弹头分导重入大气层运载工具

—吕叔湘《汉语语法分析问题

分词规范

分词的前提是确定词语的边界，而语言学关于汉语词语边界的讨论尚无定论。为满足信息处理的需要，全国信息技术标准化技术委员会制定了国标GB/T13715-1992，即《信息处理用现代汉语分词规范》。该规范明确而具体地界定了汉语分词的主题内容和适用范围，并相对全面地规定了分词原则，在一定程度上有效地保证了各种汉语信息处理系统之间的兼容性。其规定汉语分词的对象包含了词和词组，并定义了分词单位的概念以指示上述对象。在国标GB/T13715-1992的基础上，一些研究机构从自然研究的需求出发，也制定了相应的规范，比如面向通用领域的南京师范大学分词规范、面向新时代人民日报语料的南京农业大学自动分词规范、面向中国古代典籍跨语言文本的南京农业大学中国古代典籍跨语言自动分词规范等。

信息处理用现代汉语分词规范

GB/T 13715-92 文档

主题内容和适用范围（规定分词原则，满足信息处理需要，规范汉语信息处理，兼容各种汉语信息处理系统；汉语信息处理各领域可根据需要加以补充和细化）

◆引用标准

◆术语（汉语信息处理、词、词语、分词单位、汉语分词）

◆概述（10项原则）

◆具体说明（分13种词类叙述细则）

- 名词、动词、形容词、代词、数词、量词、
- 副词、介词、连词、助词、语气词、叹词、象声词

分词规范的10项原则

1. 标点符号是分隔标记；

2-4. 词长一般为二至四字；结合紧密、使用稳定的二至四字词组，一律为分词单位；五字以上结合紧密、使用稳定的谚语、格言拆开后违背原意或影响后续处理，也作为分词单位；

5-9. 惯用语、略语、儿化词、非汉字符号、音译外来词都作为分词单位；

10. 同形异构的，根据上下文做不同切分。

分词单位：汉语信息处理使用的、具有确定的语义或语法功能的基本单位（词或凝固短语）

分词规范具体问题讨论

- 绿/叶，小/床
- 我们，你们，他们，人们，朋友/们，学生/们
- 五月，元月，3月，1988/年/3/月/15日
- 汉族，哈萨克/族，长江，牡丹江，乌苏里/江，忻县，正定/县，黄山，横断山，沂蒙/山
- 学院路，刘家村，永久/牌，中华/烟，牡丹/Ⅲ/型
- 说说/看，研究/研究，想/一/想，想/了/想，想/了/一/想
- 看/不/看，相信/不/相信，容易/不/容易
- 七/百/二/十/三，五/分之/三，百/分之/二
- 生/于，走/向/胜利

分词单位

根据《信息处理用现代汉语分词规范》中的定义，分词单位即“汉语信息处理使用的、具有确定的语义或语法功能的基本单位。它包括本规范的规则限定的词和词组。” 此外，《信息处理用现代汉语分词规范》 中的词为“最小的能独立运用的语言单位”，词组为“由两个或两个以上的词，按一定的语法规则组成，表达一定意义的语言单位”，比如“聂/海胜/谈/中国/航天员/的/未来/”这个现代汉语中共有7个词，而“中国”和“航天员”为两个词，而“中国航天员”则为一个词组，同样的在“古者/富/贵/而/名/摩/灭/，/不/可/胜/记/，/唯/倜傥/非/常/之/人/称/焉”这个古汉语例子中共有21个词，其中“胜”和“记”为两个词，而“胜记”则为一个词组。

中文自动分词的三大难题

- ◆未登录词：自动分词主要是根据底表来进行的，真实文本中存在大量的未见于底表的词语，它对自动分词正确率的影响最大。
- ◆分词歧义：根据底表，一个串可以切开也可以不切开（组合性歧义），或者可以切在这里也可以切在那里（交集型歧义），但从上下文来看，至少有一种切法是不正确的。
- ◆分词不一致：上下文相同或相似情况下，一个串在分词语料库中有多种切法，也许几种切法都有道理，但应该保持一致。

未登录词

汉语自动分词需要依据底表来辅助模型构建或评价分词结果。然而，真实文本中存在大量的未见于底表的词语，对自动分词的准确率造成很大影响。未登录词不可能被穷尽，且语言的变化和发展始终会带来新的未登录词（如网络流行词语），比如“奥利给、内卷、杠精”等。因此未登录词的切分是汉语分词需要解决的重要问题，未登录词的切分效果也是衡量汉语分词性能的一个重要指标。

未登录词

◆未登录词：out-of-vocabulary（OOV）

- 小猪佩奇身上纹，掌声送给社会人。
- 好嗨呦！感觉人生已经达到了巅峰！
- 我们都是佛系青年，偶尔会转发个锦鲤，上网只相信官宣

◆未登录词不可能被穷尽

切分歧义

根据底表，一个待分词汉字串可能会具有多种分词切分形式，从构成分词歧义。分词歧义一般可以归纳为两类。一类是组合型歧义，即待分词汉字串（一般为两个汉字）既可以切开也可以不切开，如“从马上跳下来”中的“马上”；另一类是交集型歧义，即待分词汉字串（至少三个汉字）有多个切分位置，如“使用户满意”中的“使用户”。一般情况下，可以根据上下文消解分词歧义 。

分词的一致性问题

上下文相同或相似情况下，存在同类分词歧义的待分词汉字串应该始终保持切分方式的一致。对于人工标注分词语料或机器自动分词结果来说，分词一致性都是衡量分词质量的重要指标。人工标注分词语料库构建时可以通过多组多轮交叉验证的形式保证一致性；机器自动分词则应在模型构建时充分考虑分词一致性的问题。在所构建的语料库中分词不一致的现象会出现，比如“中国科学院”存在“中国科学院/”和“中国/科学院/”这种分词形式。

分词的一致性问题

◆个例的不一致
- 发展中国家/，发展中/国家/，发展/中/国家/

◆类型的不一致
- 教育部/， 林业/部/
- 露出（合/总：94.4%）揭开（合/总：87.5%）
- 翻开（合/总：34.8%）离去（合/总：28.6%）

◆跟交集型歧义和组合型歧义都有纠缠：
- 他/把/花鸟画/成/了 一/团浓墨
- 希望/尽快/将/这/幅/作品/画成
- 希望/尽快/将/这/幅/作品/画/成

◆训练语料中本身存在不一致

古汉语语料库的分词策略

- 先秦汉语语料的特点
  - 资源
    - 无训练语料
    - 无分词底表
  - 语言特点
    - 与现汉的用字、用词、语法差异大
    - 单字词比例高
- 两种策略
  - 人工制作训练语料或分词底表，采用机器学习的方式来加工（先学后用）
  - 机器辅助制作词表和训练语料（人机交互）
- 综合：先人机交互，再机器学习

分词的评测

- SIGHAN(Special Interest Group of HAN)
  - 正确率、召回率、F1值
  - OOV的召回率
- 更深入地
  - 交集型歧义消解能力
  - 组合型歧义消解能力
  - 评测语料的分词一致性如何
  - 按句子计算

南京農業大學
NANJING AGRICULTURAL UNIVERSITY

自动分词在数字人文研究中的应用背景

汉语自动分词是数字人文研究的重要前提，是深度挖掘经典文献和深入研究传统文化的必要根基。作为最小表意单元，汉语词汇间并不具有天然分隔。面对海量现代与古代汉语文本，完全依赖手工分词不仅工作量巨大，还难以保证分词结果的一致性与规范性。因此，需要借助现代计算机信息技术，自动化完成汉语词汇切分。

基于规则的自动匹配分词

汉语自动分词主要可以分为基于规则的自动匹配分词和基于概率统计分词两类方法。

◆前者通常通过人工标注或引入外部词典信息构建领域词汇表，融合停用词表获得分词底表。例如，黄建年 通过N元语法（N-gram）和词典分词技术在农业古籍上实现了自动分词，徐润华和陈小荷 构建了注疏词表，采用最大匹配分词算法对《左传》进行了分词。但是，由于基于规则匹配的方法通常无法识别未登录词，因此当前大多采用基于概率统计的机器学习或深度学习方法进行汉语自动分词。虽然目前已经出现了诸如NLPIR 、Jieba 、HanLP 、NLTK 等中文分词工具可供直接使用，但由于上述软件内置的分词模型大多基于通用语料训练，因此难以在依赖语言学、历史学、文献学等领域数据和知识的数字人文研究中使用。

基于概率统计分词

采用统计学习模型的方法能够根据语料库先验概率与条件概率分布自动判断词汇边界。其分词过程大致如下：首先，由相关专业标注人员进行手工词汇切分，经校验后形成精标数据集。其次，基于人工标注语料构建训练集，制定不同人工特征模板。最后，采用序列标注方法训练得到最佳分词模型，并完成对全部未标注语料的自动分词。

基于概率统计分词

◆深度学习模型无需人工选择待分词文本特征，神经网络架构能够自行在大规模标记数据集上提取丰富的语义与关联特征。当前较为主流的研究方法是采用Word2Vec词嵌入工具对文本进行向量化表示，继而基于RNN、Bi-LSTM、Transformer等深度神经网络架构完成自动分词。

◆基于Transformer架构的预训练语言模型具有更强的语义表征能力，尤其是面向数字人文领域相关任务继续训练的模型可以更加充分的学习到特定领域文本的句法与文法规则。采用小样本标注语料进行微调，即可利用BERT、RoBERTa、SikuBERT等进行全文分词。

基于概率统计分词

◆基于深度学习的分词方法在相同数据集上往往能够取得超越传统机器学习模型的分词表现。这一方面得益于更加复杂且深度的神经网络结构能够学习到更多显式和隐式的词法与句法特征，另一方面用于支撑模型训练的大规模标注数据集包含较为全面的自然语言知识，使得模型在开放测试中具有较强的泛化能力。尤其是近几年火热的预训练语言模型，在预训练阶段以无监督的方式充分学习海量真实文本的语言学特征与词汇、句子关联，面向下游任务仅需通过迁移学习与领域微调的方式即可取得优异的分词性能。

南京農業大學
NANJING AGRICULTURAL UNIVERSITY

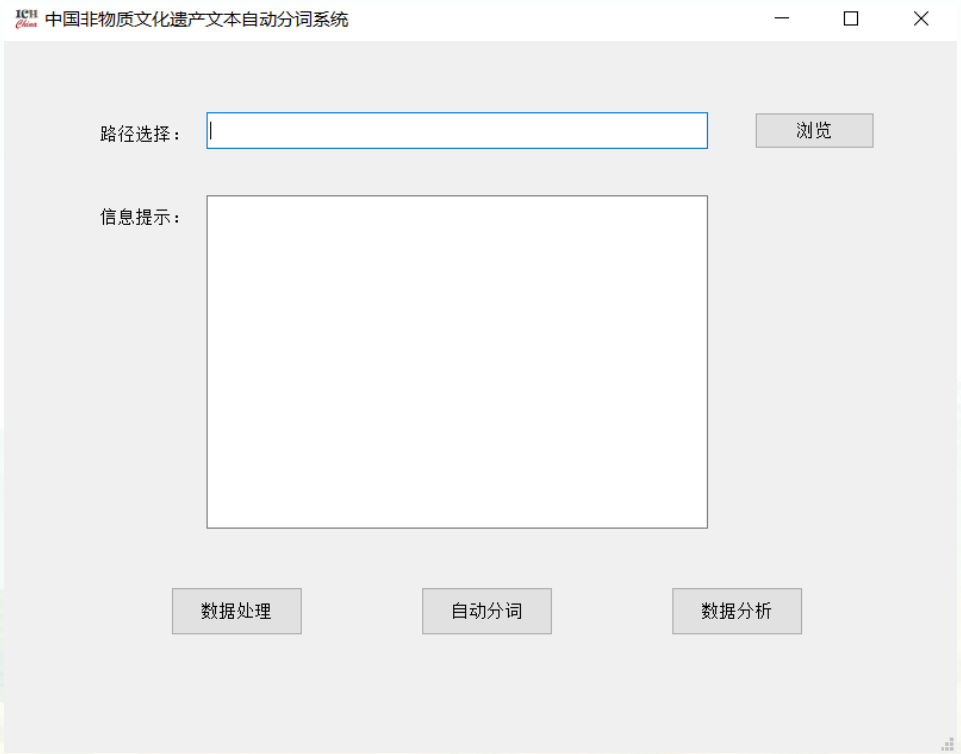
非物质文化遗产文本自动分词系统

非物质文化遗产简介

非物质文化遗产是一个国家和民族历史文化的重要标志。随着联合国教科文组织发布了《保护非物质文化遗产公约》，中国政府不断加强非物质文化遗产保护工作，这是对中华民族优秀传统文化的珍视。因此，基于数字人文理念，通过数据科学与人工智能技术对国家级非物质文化遗产项目申报文本进行分词（本章后续简称为非遗文本），实现对非遗知识的多维度组织、管理、分析、挖掘与呈现，对加强非物质文化遗产的保护，促进非遗的可持续发展具有重要的意义和价值。

系统简介

该系统基于Python编程语言开发，能够以可视化、交互式的方式，实现数据预处理、自动分词和数据分析。自动分词系统的主页面如下图所示。



系统简介

该系统基于Python编程语言开发，能够以可视化、交互式的方式，实现数据预处理、自动分词和数据分析。自动分词系统的主页面如下图所示。

系统简介

◆以项目源码（主文件：ICHAutoWordSegGUI.py）或可执行文件（双击ICHAutoWordSegGUI.exe）的形式启动系统后，点击右上“浏览”按钮，在路径选择弹窗中选取本地data文件夹，系统即可自动读取该文件夹内全部文本，并在信息提示框内呈现部分非遗文本。

系统简介

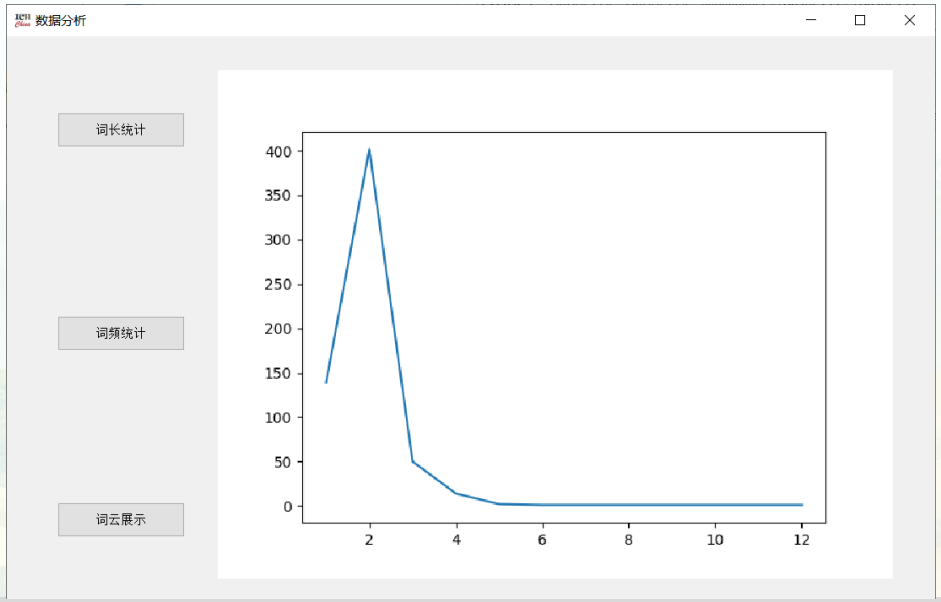
◆下一步，点击左下“数据处理”按钮，系统即可自动将语料库中的原始文本转换为待分词模型自动标注的文本格式，并在信息提示框呈现部分文本示例。

系统简介

◆下一步，点击“自动分词”按钮，即可进入自动分词模块。在该模块，左侧文本框显示的是分词前语料库内的全部文本。点击下方“开始分词”按钮，系统便会调用内置训练完成的分词模型自动切分词汇，并在右侧文本框输出全部分词后文本。词汇与词汇之间通过“/”符号间隔。

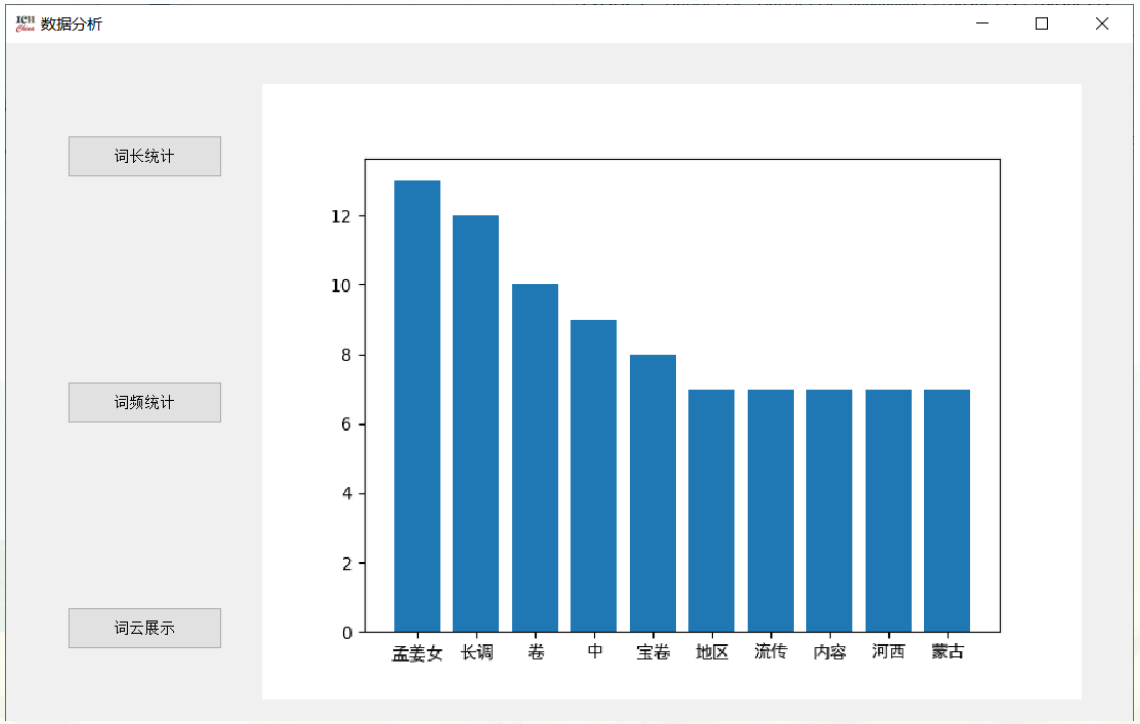
系统简介

◆返回上一界面，点击右下“数据分析”按钮，即可进入数据分析与可视化模块，该模块集成了“词长统计”、“词频统计”、“词云展示”三种数据分析与呈现功能。点击“词长统计”，即可在右侧生成分词后语料库中词汇长度分布的折线图。用户可以直观的对比语料库中不同长度词汇的数量，洞察长度分布规律。



系统简介

◆点击“词频统计”按钮，系统会在右侧呈现语料库中的高频词汇分布图。在本次演示中，分词后的语料库中“孟姜女”、“长调”、“卷”等词汇出现的次数较高。



系统简介

◆点击“词云展示”按钮，系统会自动统计语料库中词汇类别与出现频次，并生成词云。在图像中，词汇的大小与其在语料库中出现的频次呈正相关。通过词云呈现，可以快速获取导入语料库的关键词与核心主题，分析语料库的词汇特征。

数据处理模块

数据预处理是进行自动分词的前提。数据处理模块可以读取用户存储在本地的待分词非遗文本语料库，并将其自动转换为机器学习模型可识别的文本格式，用于后续自动分词与可视化分析功能的实现。

```python
import os

def read_dir(path):
    fileArray=[]
    for root,dirs,files in os.walk(path):
        for fn in files:
            eachpath=str(root+'\\'+fn)
            fileArray.append(eachpath)
    return fileArray

```

参数path为文件夹根目录，返回的列表fileArray中存储了根目录文件夹中全部语料的路径。

数据处理模块

定义读取函数reader和写入函数output

```python
# 读取本地.txt文件
def reader(path):
    with open(path, 'r', encoding='utf-8') as f:
        lines = f.readlines()
        return lines
# 保存.txt文件到本地
def output(path,tokens):
    with open(path, 'w', encoding='utf-8') as f:
        for temp in tokens:
            for token in temp:
                f.write(token + '\n')
f.write('\n')
```

在读取函数reader中，参数path为本地.txt文件的路径，在写入函数output中，path为.txt文件保存路径，tokens为预处理后的列表。

数据处理模块

调用上述函数，完成原始文本的数据预处理

```
# 进行数据预处理
all_tokens = []  # 存储处理后文本
dir_path = r'data'  # 语料库路径
# 遍历读取文件夹
fileArray = read_dir(dir_path)
for path in fileArray:
    # 读取每一个.txt文件
    lines = reader(path)
    # 数据清洗
    for para in lines:
        para = para.replace('\u3000', '').replace('\t', '').replace(' ', '').strip()
        if para:
            # 转化成待标注token格式
            all_tokens.append(list(para))
# 在根目录下保存token格式文本
output(r'test.txt', all_tokens)
```

运行上述代码后，即可在代码所在目录下查看程序自动生成的test.txt文件。完整代码可在系统源代码的ICHAutoWordSegGUI.py文件中查看。

自动分词模块

对于经数据处理模块自动加工后的待分词非遗文本，自动分词模块可以调用已经训练完成的分词模型，自动对其进行分词，并同时输出分词前后的文本。鉴于部分用户的计算机存在性能限制，本项目分词模型采用CRF机器学习模型训练，从而适用于任意型号的运行windows系统的计算机。

自动分词模块

定义自动分词函数auto_tag

```python
# 自动分词
def auto_tag():
    # 切换到CRF模型所在路径
    os.chdir(r'crf')
    print(os.getcwd())
    # 执行自动标注指令
    os.system('crf_test -m model ../test.txt > ../output.txt')
    # 切换回当前工作目录
    os.chdir(r'../')
    print(os.getcwd())
    # 读取自动标注后的token格式文件
    lines = reader(r'output.txt')
    tokens = get_tokens(lines)
# 以'/'分割的形式整理分词后文本
    all_line = []  # 存储格式整理后文本
    for temp in tokens:
        line = ''
        for token in temp:
            char = token.split('\t')[0]
            ind = token.split('\t'[1].split('-')[0]
if ind == 'B' or ind == 'M':
                line += char
            elif ind == 'E' or ind == 'S':
                line += char + '/'
        all_line.append(line)
```

自动分词模块

将tokens按段落存入列表

```
def get_tokens（lines）:
    tokens = []
    temp = []
    for line in lines:
        line = line.strip（
        if line:
            temp.append（line）
        else:
            if temp:
                tokens.append（temp）
                temp = []
    if temp:
       tokens.append（temp）
    return tokens
```

自动分词模块

执行该代码块后，会在程序所在目录下生成output.txt文件，该文本文件存储了调用自动分词模型自动标注后的token形式语料。

```
15  孟→B-tag
16  姜→M-tag
17  女→E-tag
18  传→B-tag
19  说→E-tag
20  的→S-tag
21  滥→B-tag
22  觞→E-tag
23  ，→S-tag
24  源→B-tag
25  于→E-tag
26  齐→B-tag
27  国→E-tag
28  对→S-tag
29  莒→B-tag
30  国→E-tag
31  发→B-tag
32  动→E-tag
33  战→B-tag
34  争→E-tag
35  。→S-tag
```

自动分词模块

在该输出文件中，每一行均由“字符+分词标记”组成，每种分词标记的含义见下表。对于图中示例的token格式文本，经程序处理后，最终在系统前端页面展示的分词后文本格式为“孟姜女/传说/的/滥觞/，/源于/齐国/对/莒国/发动/战争/。/”。

BMES分词标记

序号	标记	含义
1	B-tag	词汇起始字
2	M-tag	词汇中间字
3	E-tag	词汇末尾字
4	S-tag	单字词

数据分析模块：词长统计

数据分析模块能够从词汇长度分布、词汇频次统计、以及分词结果的词云呈现三个维度，对分词后语料库的词汇特征进行统计分析与可视化呈现。

◆首先，定义函数count_freq（）用于统计并排序不同长度词汇出现的频次

数据分析模块：词长统计

```
1 from collections import Counter
2
3 def count_freq(d_list, n=0, rev=True):
4     c_dict = dict(Counter(d_list))
5     d_count = sorted(c_dict.items(),
  key=lambda x: x[n], reverse=rev)
6     return d_count
7
```

其中，参数d_list为列表形式，存储待统计的全部数据；参数n可选值为0或1，分别对应按照词长或频次进行排序，默认按照词长排序；参数rev为布尔变量，默认取值为True，表示按数值从大到小排序，当取值为False时则相反。函数返回的列表d_count中存储了统计并排序后的词长与对应频次，其数据结构为“[（词长1，频次1），（词长2，频次2），…，（词长n，频次n）]”。

数据分析模块：词长统计

◆其次，分词后文本中存在大量诸如“的”、“了”、“呀”等无实意的词汇，各种中英文标点符号也同样没有实际含义，即通常所说的停用词。因此，定义函数del_stopwords用于删去文本中的停用词

数据分析模块：词长统计

```python
def del_stopwords(word_list):
    # 读取停用词表
    stopwords_path = r'stopwords.txt'
    stopwords = reader(stopwords_path)
    stopword_list = [each.strip() for each in stopwords if each.strip()]
    # 去停用词
    word_list_no_stop = []  # 存储删去停用词的分词后语料
    for word in word_list:
        if word not in stopword_list:
            word_list_no_stop.append(word)
    return word_list_no_stop
```

输入变量word_list为存储了分词后文本中全部词汇的列表（不去重），其数据结构为“[词汇1,词汇2,…,词汇n]”。停用词表stopwords.txt可在系统源码的根目录中查看。运行代码，输出变量word_list_no_stop为删去了停用词表中停用词的word_list，数据结构与word_list相同。

数据分析模块：词长统计

◆最后，编写函数count_len_freq，并调用工具包绘制词长分布统计图，即可实现对分词后语料的词长统计与可视化呈现

数据分析模块：词频统计

```python
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
font = FontProperties(fname=r"Fonts\simhei.ttf")
# 统计词频分布
def count_word_freq(word_list_no_stop):
    # 统计词频并按照词频降序排列
    word_count = count_freq(word_list_no_stop, 1)
    # 指定图像x、y轴数据
    x = [each[0] for each in word_count[:10]]
    y = [each[1] for each in word_count[:10]]
    # 绘制条形图
    plt.bar(range(len(y)), y)
    # 设置x轴坐标轴
    plt.xticks(range(len(y)), x, fontproperties=font)
    # 展示图像
    plt.show()
    # 保存图像
    plt.savefig(r'bar.png')
    # 关闭画布
    plt.close()
```

其中，输入变量word_list_no_stop与词长统计的变量相同。由于matplotlib工具默认不支持显示中文字体，因此需要通过上述代码块的第2、3行代码指定中文字体。运行程序后，会在屏幕上呈现词频统计图，并在该程序所在目录下保存图像，命名为bar.png。

数据分析模块：词云展示

一幅词云图像通常由大小及颜色不同的词汇构成。其中，出现频次越大的词汇字号越大。定义函数word_cloud，并调用第三方工具包wordcloud用于生成分词后文本的词云图像并保存

数据分析模块：词云展示

```python
from wordcloud import WordCloud
# 生成词云图像
def word_cloud(word_list_no_stop):
    # 统计词频并按照词频降序排列
    word_count = count_freq(word_list_no_stop, 1)
    # 将统计结果转换为字典形式
    freq_dict = {}
    for each in word_count:
        freq_dict[each[0]] = int(each[1])
    # 指定词云图像参数
    wc = WordCloud(font_path=r"Fonts\simhei.ttf", background_color="white", max_words=1000, scale=1, width=800, height=600)
    # 生成词云
    wc.generate_from_frequencies(freq_dict)
    # 保存图像
    wc.to_file(r'wordcloud.png')
```

其中，输入变量word_list_no_stop同样与词长统计的变量相同。此处对WordCloud类中的各项参数进行简要说明：font_path为字体文件路径，background_color为图片背景颜色，max_words为词云中显示的最大词汇个数，scale为画布放大比例，width为图像宽度，height为图像高度。运行代码块后，即可在程序所在目录下查看自动保存的词云图像，其文件名为“wordcloud.png”。

自动分词模型构建：数据集构建

首先，从中国非物质文化遗产网（http://www.ihchina.cn/）获取国家级非物质文化遗产名录的申报正文内容。

京剧

项目序号：172　　项目编号：Ⅳ-28

公布时间：2006(第一批)　　类别：传统戏剧

所属地区：北京市　　类型：新增项目

申报地区或单位：北京市　　保护单位：北京京剧院

申报地区或单位：北京市

京剧又称平剧、京戏，是中国影响最大的戏曲剧种，分布地以北京为中心，遍及全国。清代乾隆五十五年起，原在南方演出的三庆、四喜、春台、和春四大徽班陆续进入北京，他们与来自湖北的汉调艺人合作，同时接受了昆曲、秦腔的部分剧目、曲调和表演方法，又吸收了一些地方民间曲调，通过不断的交流、融合，最终形成京剧。

在文学、表演、音乐、舞台美术等各个方面，京剧都有一套规范化的艺术表现程式。京剧的唱腔属板式变化体，以二簧、西皮为主要声腔。四平调、反四平调、汉调等都从属于二簧，南梆子、娃娃调则从属于西皮。二簧旋律平稳，节奏舒缓，唱腔浑厚凝重；西皮旋律起伏较大，节奏紧凑，唱腔明快流畅。京剧伴奏分文场和武场两大类，文场使用胡琴（京胡）、京二胡、月琴、弦子、笛子、唢呐等，而以胡琴为主奏乐器；武场以鼓板为主，小锣、大锣次之。京剧的脚色分为生、旦、净、丑、杂、武、流等行当，后三行现已不再立专行。各行当内部还有更细的划分，如旦行就有青衣、花旦、刀马旦、武旦、老旦之分。其划分依据除人物的自然属性外，更主要的是人物的性格特征和创作者对人物的褒贬态度。各行当都有一套表演程式，唱念做打的技艺各具特色。

京剧以历史故事为主要演出内容，传统剧目约有一千三百多个，常演的在三四百个以上，其中《宇宙锋》、《玉堂春》、《长坂坡》、《群英会》、《打渔杀家》、《空城计》、《贵妃醉酒》、《三岔口》、《野猪林》、《二进宫》、《拾玉镯》、《挑华车》、《四进士》、《搜孤救孤》、《霸王别姬》、《四郎探母》等剧家喻户晓，为广大观众所熟知。新中国成立后，京剧改编、移植、创作了一些新的历史剧和现代题材作品，重要的有《将相和》、《穆桂英挂帅》、《杨门女将》、《海瑞罢官》、《曹操与杨修》、《沙家浜》、《红灯记》、《智取威虎山》、《黛诺》、《骆驼祥子》等。

京剧有“京派”和“海派”之分，不同时期出现过许多优秀的演员，如清末的程长庚、余三胜、张二奎、梅巧玲、谭鑫培、孙菊仙、汪桂芬、刘鸿声、田桂凤、余紫云、陈德霖、王瑶卿等，民国年间的余叔岩、言菊朋、高庆奎、马连良、杨宝森、梅兰芳、程砚秋、荀慧生、尚小云、周信芳、金少山等。

京剧流播全国，影响甚广，有“国剧”之称。它走遍世界各地，成为介绍、传播中国传统文化的重要手段。以梅兰芳命名的京剧表演体系已经被视为东方戏剧表演体系的代表，与斯坦尼斯拉夫斯基及布莱希特表演体系并称为世界三大表演体系。京剧是中国民族传统文化的重要表现形式，其中的多种艺术元素被用作中国传统文化的象征符号。但近年来随着社会的变迁，京剧艺术与当代人的审美距离逐渐加大，观众锐减，上演剧目萎缩，如何实现京剧的保护和振兴已成为一个亟待解决的课题。

自动分词模型构建：数据集构建

将上述内容以文本文件的形式保存在本地，通过人工标注的形式，对全部正文文本中的句子进行手工分词标注，从而构建可供机器学习模型训练的标注后数据集。

117_PN__<ICH-TITLE>京剧<ICH-TITLE/>/n
117_DI__申报/gnzy 地区/n 或/c 单位/n ： /wp <ICH-PLACE>北京市/<ICH-PLACE/>ns ,/wd ,/wd
<ICH-TITLE>京剧<ICH-TITLE/>/n 又/d 称/v <ICH-TERM>平剧<ICH-TERM/>/n 、/wn <ICH-TERM>京戏/n<ICH-TERM/>
，/wd 是/vshi 中国/ns 影响/v 最/d 大/a 的/ude1 戏曲/n 剧种/n ，/wd 分布/vi 地/ude2 以/p
<ICH-PLACE>北京<ICH-PLACE/>/ns 为/p 中心/n ，/wd 遍及/v 全国/n 。/wj 清代/t 乾隆/t 五十五/m 年/qt 起/f
，/wd 原/d 在/p 南方/s 演出/v 的/ude1 <ICH-TERM>三庆<ICH-TERM/>/n 、/wn <ICH-TERM>四喜/<ICH-TERM/>ng
、/wn <ICH-TERM>春台/n<ICH-TERM/> 、/wn <ICH-TERM>和/cc春/<ICH-TERM/>tg 四大/n 徽/b 班/n 陆续/d 进入/v
<ICH-PLACE>北京<ICH-PLACE/>/ns ，/wd 他们/rr 与/p 来自/v <ICH-PLACE>湖北/<ICH-PLACE/>ns 的/ude1 汉/tg
调/v 艺人/n 合作/vn ，/wd 同时/c 接受/v 了/ule <ICH-TERM>昆曲/<ICH-TERM/>n 、/wn
<ICH-TERM>秦腔<ICH-TERM/>/n 的/ude1 部分/n 剧目/n 、/wn 曲调/n 和/cc 表演/vn 方法/n ，/wd 又/d 吸收/v
了/ule 一些/mq 地方/n 民间/n 曲调/n ，/wd 通过/p 不断/d 的/ude1 交流/vn 、/wn 融合/v ，/wd 最终/d
形成/v 京剧/n 。/wj ,/wd 在/p 文学/n 、/wn 表演/vn 、/wn 音乐/n 、/wn 舞台/n 美术/n 等/udeng 各个/rz
方面/n ，/wd 京剧/n 都/d 有/vyou 一/m 套/q 规范化/vi 的/ude1 艺术/n 表现/vn 程式/n 。/wj 京剧/n 的/ude1
唱腔/n 属/v 板/n 式/k 变化/v 体/ng ，/wd 以/p <ICH-TERM>二簧<ICH-TERM/>/n 、/wn
<ICH-TERM>西皮<ICH-TERM/>/n 为/p 主要/b 声腔/n 。/wj <ICH-TITLE>四平调/<ICH-TITLE/>n、/wn
<ICH-TITLE>反/vi四平调<ICH-TITLE/>/n、/wn <ICH-TITLE>汉/tg调<ICH-TITLE/>/v 等/udeng 都/d 从/p 属于/v
<ICH-TERM>二簧<ICH-TERM/>/n ，/wd <ICH-TERM>南/b梆子<ICH-TERM/>/n 、/wn <ICH-TERM>娃娃/n调<ICH-TERM/>/v
则/c 从/p 属于/v 西皮/n 。/wj 二簧/n 旋律/n 平稳/a ，/wd 节奏/n 舒缓/z ，/wd 唱腔/n 浑厚/a 凝重/z ： /wf
西皮/n 旋律/n 起伏/vi 较/d 大/a ，/wd 节奏/n 紧凑/a ，/wd 唱腔/n 明快/a 流畅/a 。/wj
<ICH-TITLE>京剧/<ICH-TITLE/>n 伴奏/vn 分文/n 场/qv 和/cc 武场/n 两/m 大/a 类/n ，/wd 文场/n 使用/v
<ICH-INST>胡琴/<ICH-INST/>n （/wkz <ICH-INST>京胡<ICH-INST/>/n ）/wky 、/wn
<ICH-INST>京/b二胡/n<ICH-INST/> 、/wn <ICH-INST>月琴/<ICH-INST/>n 、/wn <ICH-INST>弦子/<ICH-INST/>nr
、/wn <ICH-INST>笛子/<ICH-INST/>n 、/wn <ICH-INST>唢呐/<ICH-INST/>n 等/udeng ，/wd 而/cc 以/p
<ICH-INST>胡琴<ICH-INST/>/n 为主/vi 奏/v 乐器/n ： /wf 武场/n 以/p <ICH-INST>鼓板/<ICH-INST/>n 为主/vi
，/wd <ICH-INST>小锣<ICH-INST/>/n 、/wn <ICH-INST>大/a锣<ICH-INST/>/n 次之/vi 。/wj 京剧/n 的/ude1 脚/n
色/ng 分为/v <ICH-TERM>生<ICH-TERM/>/v 、/w<ICH-TERM>n旦<ICH-TERM/>/ng 、/wn <ICH-TERM>净<ICH-TERM/>/a
、/wn <ICH-TERM>丑/a<ICH-TERM/> 、/w<ICH-TERM>n杂/a<ICH-TERM/> 、/wn <ICH-TERM>武<ICH-TERM/>/ag 、/wn
<ICH-TERM>流<ICH-TERM/>/v 等/udeng 行当/n ，/wd 后/f 三/m 行/q 现/tg 已/d 不再/d 立/v 专/d 行/vi 。/wj

自动分词模型构建：数据集构建

对于标注好的数据集，通过编写的Python代码make_dataset.py，即可实现将标注数据集转换为机器学习模型可识别的token格式数据。以下是make_dataset.py脚本中部分核心代码的解释。

自动分词模型构建：数据集构建

```python
# 将序列写入txt文件，text为待写入文本/string，path为待写入文件路径/string
def wry_tags(sentence):

    # 用于存储结果
    temp = []
    for word in sentence:
        text = word.strip()
        # print(text)
        if text:
            try:
                if len(text) == 1:
                    # 若为句号问号则换行
if text == "。" or text[0] == "? ":
                        temp.append(text + '\t' + 'S' + '-' + 'tag' + '\n\n')
                    else:
                        temp.append(text + '\t' + 'S' + '-' + 'tag' + '\n')
                elif len(text) > 2:
                    temp.append(text[0] + '\t' + 'B' + '-' + 'tag' + '\n')
                    for i in range(1, len(text) - 1):
                        temp.append(text[i] + '\t' + 'I' + '-' + 'tag' + '\n')
                    temp.append(text[-1] + '\t' + 'E' + '-' + 'tag' + '\n')
                else:
                    temp.append(text[0] + '\t' + 'B' + '-' + 'tag' + '\n')
                    temp.append(text[-1] + '\t' + 'E' + '-' + 'tag' + '\n')
            except Exception as e:
                print(e)
                raise
    return temp
```

运行代码，即可完成训练集与测试集的构建。为了消除数据集分布不平衡带来的随机误差，

上述代码采用了十折交叉的方式，按照9：1的比例分别生成了10组不同的训练集与测试集。

自动分词模型构建：模型训练

首先，在MakeDataset文件夹根目录下，编写如下代码，以十折交叉的方式训练非遗文本自动分词模型。

待训练完成后，将自动在代码所在根目录下生成模型文件。

自动分词模型构建：模型训练

```
import os
os.system('crf_learn template ./train/train0.txt model0')
os.system('crf_learn template ./train/train1.txt model1')
os.system('crf_learn template ./train/train2.txt model2')
os.system('crf_learn template ./train/train3.txt model3')
os.system('crf_learn template ./train/train4.txt model4')
os.system('crf_learn template ./train/train5.txt model5')
os.system('crf_learn template ./train/train6.txt model6')
os.system('crf_learn template ./train/train7.txt model7')
os.system('crf_learn template ./train/train8.txt model8')
os.system('crf_learn template ./train/train9.txt model9')
```

其中，os.system方法的作用是调用控制台命令行，执行相关指令。

template文件是CRF模型的特征模板，此处使用的特征模板如右图所示。

```
# Unigram
U00:%x[-2,0]
U01:%x[-1,0]
U02:%x[0,0]
U03:%x[1,0]
U04:%x[2,0]
U05:%x[-1,0]/%x[0,0]
U06:%x[0,0]/%x[1,0]

# Bigram
B
```

自动分词模型构建：自动标注

通过调用上一步骤生成的model文件，即可完成对test文本的自动标注。

自动分词模型构建：自动标注

```
import os
os.system('crf_test -m model0 ./test/test0.txt >output0.txt')
os.system('crf_test -m model1 ./test/test1.txt >output1.txt')
os.system('crf_test -m model2 ./test/test2.txt >output2.txt')
os.system('crf_test -m model3 ./test/test3.txt >output3.txt')
os.system('crf_test -m model4 ./test/test4.txt >output4.txt')
os.system('crf_test -m model5 ./test/test5.txt >output5.txt')
os.system('crf_test -m model6 ./test/test6.txt >output6.txt')
os.system('crf_test -m model7 ./test/test7.txt >output7.txt')
os.system('crf_test -m model8 ./test/test8.txt >output8.txt')
os.system('crf_test -m model9 ./test/test9.txt >output9.txt')
```

运行上述代码后，系统会调用训练好的分词模型，对test文件进行自动分词标注，并生成output文件。

右图是output文件的示例。

```
申 B-tag B-tag
报 E-tag E-tag
地 B-tag B-tag
区 E-tag E-tag
或 S-tag S-tag
单 B-tag B-tag
位 E-tag E-tag
： S-tag S-tag
河 B-tag B-tag
北 I-tag I-tag
省 E-tag E-tag
魏 B-tag B-tag
县 E-tag E-tag
， S-tag S-tag
， S-tag S-tag
我 B-tag B-tag
国 E-tag E-tag
传 B-tag B-tag
统 E-tag E-tag
纺 B-tag B-tag
织 E-tag E-tag
技 B-tag B-tag
艺 E-tag E-tag
历 B-tag B-tag
史 E-tag E-tag
悠 B-tag B-tag
久 E-tag E-tag
```

自动分词模型构建：性能测评

为了检测自动分词模型的性能如何，需要对比人工标注的标签与机器自动标注的标签，从而计算精确率（Precision）、召回率（Recall）、调和平均值（F-score）三个评价指标，编写代码以调用测评工具conlleval.py对自动标注结果进行评价指标计算。

自动分词模型构建：性能测评

```python
import os
os.system('python conlleval.py output0.txt > eval0.txt')
os.system('python conlleval.py output1.txt > eval1.txt')
os.system('python conlleval.py output2.txt > eval2.txt')
os.system('python conlleval.py output3.txt > eval3.txt')
os.system('python conlleval.py output4.txt > eval4.txt')
os.system('python conlleval.py output5.txt > eval5.txt')
os.system('python conlleval.py output6.txt > eval6.txt')
os.system('python conlleval.py output7.txt > eval7.txt')
os.system('python conlleval.py output8.txt > eval8.txt')
os.system('python conlleval.py output9.txt > eval9.txt')
```

运行上述代码后，系统自动在程序所在根目录生成评价文件eval0.txt——eval9.txt，分别对应十折交叉验证的output文件。

eval文件的示例如右图所示。

```
processed 151370 tokens with 97403 phrases;
found: 97480 phrases;
correct: 92574.
accuracy:  95.52%;
precision:  94.97%;
recall:  95.04%;
FB1:  95.00
tag:
precision:  94.97%;
recall:  95.04%;
FB1:  95.00  97480
```