司法大数据自动化标注与分析说明文档

1. 使用方法

网页示意图：

1.1 启动网页

1. 在PyCharm中打开该项目，导入相关包
2. 在app.py文件中启动main函数
3. 在浏览器地址栏输入 http://127.0.0.1:5000/ 即可进入网页

1.2 选择爬虫

在网页中输入开始日期、结束日期与爬取数量（默认为10），点击爬取案件按钮即可爬虫，此时将会跳转到爬虫展示页，如图：

爬虫结果展示

爬取文书存放在项目的static/Downloads目录下，文书名称如下：

王某涉挪用资金罪刑事一审案件刑事判决书

陈学德涉非法经营罪刑事一审案件刑事判决书

陈春飞涉非法经营罪刑事一审案件刑事判决书

李振环涉帮助信息网络犯罪活动罪刑事一审案件刑事判决书

刘国扬故意伤害刑事一审刑事判决书

李刚危险驾驶罪一审刑事判决书

李某某故意伤害罪、故意伤害罪刑事一审刑事判决书

罗楚国帮助信息网络犯罪活动罪刑事一审刑事判决书

莫汉前非法捕捞水产品罪刑事一审刑事判决书

陈洪学交通肇事刑事一审刑事判决书

若未显示名称，则爬取失败

1.3 分词标注与保存

1. 手动输入文本或者上传案例文件（txt格式）
2. 点击开始分词按钮
3. 在分词结果中勾选标注
4. 点击保存案件与标注按钮即可保存案件文本与标注信息

PS：由于案例文本、标注的清空手动较为麻烦，故增加了一键清空文本与标注的功能按钮

2. 具体实现

以flask框架为基础，综合爬虫、HTML、CSS、JavaScript、jieba分词、BootStrap、Jinja2等技能实现该项目

2.1 项目结构

```
Program:.
|
|  app.py ·····  项目入口，连接前后端，启动网页
|  jiebaAnalysis.py ·····  jieba分词
|  wenshu_spider.py ·····  爬虫源代码
|
|
├─static ·····  存放js, css, txt等文件
|  |  dict.txt ·····  jieba分词自定义词典
|  |  司法大数据自动化标注与分析.css ·····  网页除时间选择器的style
|  |  司法大数据自动化标注与分析.js ·····  网页除时间选择器的js源码
|  |
|  ├─bootstrap ·····  用于构造时间选择器组件
|  |
|  |
|  ├─Downloads ····· 文书爬取结果文件夹
|  |      刘国扬故意伤害刑事一审刑事判决书.txt
|  |      ......
|  |
|  ├─jquery ····· 用于实现时间选择器功能
```

```
|   |
|   |
|   └─js ···· 用于实现时间选择器功能
|
|
└─templates ···· 网页HTML源文件
        司法大数据自动化标注与分析.html ···· 主体网页
        文书爬取.html ···· 爬虫结果展示网页
```

2.2 前端实现

前端页面主要包括以下几个重要组件：

时间选择器、各种按钮、文本域、单选框、复选框，其中文本域实现简单，此处不再赘述。

2.2.1 时间选择器

时间选择器的实现利用了BootStrap框架，css和js代码直接copy了过来，放在了文件中，此处不再展示，下面为html代码

```
<div class="container">
    <form action="/文书爬取" method="post" class="form-horizontal"
 role="form">
        <div class="row">
            <label for="dtp_input1" class="col-md-1 control-label">开始日期
</label>
            <div class="input-group date form_date col-md-3" id="begindate"
data-date="" data-date-format="yyyy.MM.dd" data-link-field="dtp_input1"
data-link-format="yyyy-mm-dd">
                <input class="form-control" name="beginDate" size="16"
type="text" id="beginText" value="" readonly>
                <span class="input-group-addon"><span class="glyphicon
glyphicon-remove"></span></span>
                <span class="input-group-addon"><span class="glyphicon
glyphicon-calendar"></span></span>
            </div>
            <input type="hidden" id="dtp_input1" value="" />

            <label for="dtp_input2" class="col-md-1 control-label">结束日期
</label>
            <div class="input-group date form_date col-md-3" id="enddate"
data-date="" data-date-format="yyyy.MM.dd" data-link-field="dtp_input2"
data-link-format="yyyy-mm-dd">
                <input class="form-control" name="endDate" size="16"
type="text" id="endText" value="" readonly>
                <span class="input-group-addon"><span class="glyphicon
glyphicon-remove"></span></span>
                <span class="input-group-addon"><span class="glyphicon
glyphicon-calendar"></span></span>
            </div>
            <input type="hidden" id="dtp_input2" value="" />

            <label for="num" class="col-md-1 control-label">爬取数量</label>
            <div class="input-group col-md-2">
            <input type="text" id="num" name="num" class="text"
placeholder="默认为10">
            </div>
```

```
            <div class="col-md-1 control-label">
                <input type="submit" id="spiderButton" value="爬取案件" />
            </div>
        </div>
    </form>
</div>
```

其中增添了结束日期不能早于开始日期的设定，并且将语言设置为中文，实现如下：

```
$('.form_date').datetimepicker({
        language:  'zh-CN', //设置语言为中文
        weekStart: 1,
        todayBtn:  1,
        autoclose: 1,
        todayHighlight: 1,
        startView: 2,
        minView: 2,
        forceParse: 0
    }).on('changeDate', function (e) {
        var BeginTime = $("#beginText").val();
        $("#enddate").datetimepicker("setStartDate", BeginTime);    //设置结束时间只能从开始时间选择起
    });;
```

2.2.2 各种按钮

按钮的重点在于功能的实现，以下介绍了实现一些按钮功能的js代码，另外一些按钮需要后端支持

```
/**
 * 上传文件并在textarea中显示
 */
window.onload = function() {
    /**
     * 上传函数
     * @param fileInput DOM对象
     * @param callback 回调函数
     */
    var getFileContent = function (fileInput, callback) {
        if (fileInput.files && fileInput.files.length > 0 && fileInput.files[0].size > 0) {
            //下面这一句相当于JQuery的：var file =$("#upload").prop('files')[0];
            var file = fileInput.files[0];
            if (window.FileReader) {
                var reader = new FileReader();
                reader.onloadend = function (evt) {
                    if (evt.target.readyState === FileReader.DONE) {
                        callback(evt.target.result);
                    }
                };
                // 包含中文内容用utf-8编码
                reader.readAsText(file, 'utf-8');
            }
        }
    };

```

```
    /**
     * upload内容变化时载入内容
     */
    document.getElementById('uploadButton').onchange = function () {
        var textArea = document.getElementById('textArea');
        getFileContent(this, function (str) {
            textArea.value = str;
        });
    };
};

/**
 * 清空案例文本
 */
function clearContent(){
    var textArea = document.getElementById('textArea'); // 文本域文档元素：案件文本
    textArea.value = "";
}

/**
 * 清空标注
 */
function clearMark(){
    var tab__contentList = document.getElementsByClassName("tab__content");//长度为7
    for(var i = 0; i < tab__contentList.length; i++){
        var checkboxes = tab__contentList[i].getElementsByTagName("input");
        for(var j = 0; j < checkboxes.length; j++){
            checkboxes[j].checked = false;
        }
    }
}

/**
 * 保存案件文本和标注
 */
function saveFile(){
    saveTxt();
    saveJSON();
}
// 保存案件文本.txt
function saveTxt(){
    var textArea = document.getElementById('textArea'); // 文本域文档元素：案件文本
    var textAreaContent = textArea.value; // 具体内容
    if(textAreaContent){ // 有内容
        var blob = new Blob([textAreaContent], {type: "text/plain;charset=utf-8"});
        saveAs(blob, "案件文本.txt");
        //textArea.value="";// 清空输入框
    }else{ // 无内容
        window.alert("你尚未输入案件文本，请重新输入");
    }
}
// 保存标注.json
function saveJSON(){
```

```
    var tab__contentList = document.getElementsByClassName("tab__content");//长度为7
    var markList = new Array(tab__contentList.length);
    for(var i = 0; i < tab__contentList.length; i++){
        var checkboxes = tab__contentList[i].getElementsByTagName("input");
        markList[i] = "";
        for(var j = 0; j < checkboxes.length; j++){
            if(checkboxes[j].checked){
                var label = checkboxes[j].parentNode;
                if(markList[i].length != 0) {
                    markList[i] += ("、" + label.innerText);
                }else {
                    markList[i] = label.innerText;
                }
            }
        }
    }

    var nameList = ["当事人", "性别", "民族", "出生地", "文化水平", "案由", "相关法院"];
    var textContents = [];
    for (var i = 0; i < 7; i++){
        var textJSONString = '"' + nameList[i] + '":' + '"' + markList[i] + '"';
        textContents.push(textJSONString);
    }
    var textListJSONString = "{" + textContents.join(",") + "}"; // JSON字符串
    var blob = new Blob([textListJSONString], {type: "text/plain;charset=utf-8"});
    saveAs(blob, "标注.json");
}
```

2.2.3 单选框

单选框实现不难，但要通过多个单选框实现选项卡有些困难，重点在于CSS代码

```
<div class="tab-wrap" id="jiebas">

            <!-- active tab on page load gets checked attribute -->
            <input type="radio" id="party" name="Mark" class="tab" checked>
            <label for="party">当事人</label>

            <input type="radio" id="sex" name="Mark" class="tab">
            <label for="sex">性别</label>

            <input type="radio" id="nationality" name="Mark" class="tab">
            <label for="nationality">民族</label>

            <input type="radio" id="birthPlace" name="Mark" class="tab">
            <label for="birthPlace">出生地</label>

            <input type="radio" id="education" name="Mark" class="tab">
            <label for="education">文化水平</label>

            <input type="radio" id="cause" name="Mark" class="tab">
            <label for="cause">案由</label>
```

```
            <input type="radio" id="court" name="Mark" class="tab">
            <label for="court">相关法院</label>
</div>
```

CSS代码：

```
.tab-wrap {
    transition: 0.3s box-shadow ease;
    border-radius: 6px;
    max-width: 100%;
    display: flex;
    flex-wrap: wrap;
    position: relative;
    list-style: none;
    background-color: #fff;
    margin: 40px 0;
    box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px rgba(0, 0, 0, 0.24);
}
.tab-wrap:hover {
    box-shadow: 0 12px 23px rgba(0, 0, 0, 0.23), 0 10px 10px rgba(0, 0, 0, 0.19);
}

.tab__content {
    padding: 10px 25px;
    background-color: transparent;
    position: absolute;
    width: 100%;
    z-index: -1;
    opacity: 0;
    left: 0;
    transform: translateY(-3px);
    border-radius: 6px;
}
.tab {
    display: none;
}
.tab:checked:nth-of-type(1) ~ .tab__content:nth-of-type(1) {
    opacity: 1;
    transition: 0.5s opacity ease-in, 0.8s transform ease;
    position: relative;
    top: 0;
    z-index: 100;
    transform: translateY(0px);
    text-shadow: 0 0 0;
}
.tab:checked:nth-of-type(2) ~ .tab__content:nth-of-type(2) {
    opacity: 1;
    transition: 0.5s opacity ease-in, 0.8s transform ease;
    position: relative;
    top: 0;
    z-index: 100;
    transform: translateY(0px);
    text-shadow: 0 0 0;
}
```

```css
.tab:checked:nth-of-type(3) ~ .tab__content:nth-of-type(3) {
    opacity: 1;
    transition: 0.5s opacity ease-in, 0.8s transform ease;
    position: relative;
    top: 0;
    z-index: 100;
    transform: translateY(0px);
    text-shadow: 0 0 0;
}
.tab:checked:nth-of-type(4) ~ .tab__content:nth-of-type(4) {
    opacity: 1;
    transition: 0.5s opacity ease-in, 0.8s transform ease;
    position: relative;
    top: 0;
    z-index: 100;
    transform: translateY(0px);
    text-shadow: 0 0 0;
}
.tab:checked:nth-of-type(5) ~ .tab__content:nth-of-type(5) {
     opacity: 1;
     transition: 0.5s opacity ease-in, 0.8s transform ease;
     position: relative;
     top: 0;
     z-index: 100;
     transform: translateY(0px);
     text-shadow: 0 0 0;
}
.tab:checked:nth-of-type(6) ~ .tab__content:nth-of-type(6) {
      opacity: 1;
      transition: 0.5s opacity ease-in, 0.8s transform ease;
      position: relative;
      top: 0;
      z-index: 100;
      transform: translateY(0px);
      text-shadow: 0 0 0;
}
.tab:checked:nth-of-type(7) ~ .tab__content:nth-of-type(7) {
    opacity: 1;
    transition: 0.5s opacity ease-in, 0.8s transform ease;
    position: relative;
    top: 0;
    z-index: 100;
    transform: translateY(0px);
    text-shadow: 0 0 0;
}
.tab:first-of-type:not(:last-of-type) + label {
    border-top-right-radius: 0;
    border-bottom-right-radius: 0;
}
.tab:not(:first-of-type):not(:last-of-type) + label {
    border-radius: 0;
}
.tab:last-of-type:not(:first-of-type) + label {
    border-top-left-radius: 0;
    border-bottom-left-radius: 0;
}
.tab:checked + label {
    background-color: #fff;
```

```
    box-shadow: 0 -1px 0 #fff inset;
    cursor: default;
}
.tab:checked + label:hover {
    box-shadow: 0 -1px 0 #fff inset;
    background-color: #fff;
}
.tab + label {
    box-shadow: 0 -1px 0 #eee inset;
    border-radius: 6px 6px 0 0;
    cursor: pointer;
    display: block;
    text-decoration: none;
    color: #333;
    flex-grow: 3;
    text-align: center;
    background-color: #f2f2f2;
    user-select: none;
    text-align: center;
    transition: 0.3s background-color ease, 0.3s box-shadow ease;
    height: 50px;
    box-sizing: border-box;
    padding: 15px;
}
.tab + label:hover {
    background-color: #f9f9f9;
    box-shadow: 0 1px 0 #f4f4f4 inset;
}

```

2.2.4 复选框

由于复选框的内容需要分词结果，故需要通过jinja2来返回渲染后的模板来实现动态加载复选框

html代码：

```
<div class="tab__content">
    <div>
        <h3>不涉及地点名词</h3>
        {% for jieba1 in jieba1s %}
        <label class="checkbox-inline1">
            <input type="checkbox">{{ jieba1 }}
        </label>
        {% endfor %}
    </div>

    <div>
        <h3>涉及地点名词</h3>
        {% for jieba2 in jieba2s %}
        <label class="checkbox-inline1">
            <input type="checkbox">{{ jieba2 }}
        </label>
        {% endfor %}
    </div>

```

```
    <div>
        <h3>案由</h3>
        {% for jieba3 in jieba3s %}
        <label class="checkbox-inline1">
            <input type="checkbox">{{ jieba3 }}
        </label>
        {% endfor %}
    </div>

    <div>
        <h3>形容词</h3>
        {% for jieba4 in jieba4s %}
        <label class="checkbox-inline1">
            <input type="checkbox">{{ jieba4 }}
        </label>
        {% endfor %}
    </div>
</div>

```

2.3 爬虫实现

由于裁判文书网爬取困难，最终我们选择爬取威科先行网站，通过分析发现对url='https://law.wkinfo.com.cn/csi/search'，发送携带检索参数的POST请求即可获取带有文书内容的Response

2.3.1 反爬

利用谷歌开发者工具分析发现，此网站反爬很低，只要发送一次请求即可获取文书信息。可以说，只要你买了网站的数据库就可以爬取很多文本。由于我们未购买数据库，只是试用，每天可访问的文书数量有限，同时为了反爬，我们随机使用小组成员账号来爬取文书。

Cookie:

该网站的Cookie有下面特点：

1. 对于同一用户，只有一项Cookie即Hm_lpvt_fecce484974a74c6d10f421b6d3bd395不稳定，是不断变化的，研究发现其值为当前时间戳，通过time.time()即可获取
2. 对于同一用户，其余Cookie相对稳定，长期不变，故直接登录组员的几个账号来获取Cookie，放入列表中，使用时通过random.choice()随机选取

User-Agent：

由于只需要一次请求，故一个User-Agent足以

Sleep:

由于只需要一次请求，故不需要sleep

2.3.2 请求

发送POST请求，重点在于headers的Cookie和表单数据的构建，Cookie上面已经介绍。表单数据即检索条件，我们只需要设置爬取时间和数量即可，实现如下

```
def spider(beginDate, endDate, num):
    """
    :param beginDate: "2020.01.20"
    :param endDate: "2021.12.25"
```

```
    :param num:
    :return:
    """
    dateList = [] #表单中的时间参数
    date = "judgmentDate:["+beginDate+' TO '+endDate+']'
    dateList.append(date)

    #小组成员Cookie
    Cookies = ['autologin=true; username=2034885420@qq.com; userConfig=
{"moduleList":
[],"userStaffType":0,"isIpUser":0,"parentId":0,"sourceSiteUrl":null,"sourceS
iteName":null,"clientSource":"自主注
册","trial":true,"expire":false,"caseView":false,"advancedlSearchExpires":"2
","proximateLatestExpires":"48","proximateSearch":false}; check=valid;
TY_SESSION_ID=827ba7ea-6794-4eaf-abf1-8f66eb227b74; connect.sid=s:ZXBCNi-
URL3-e59bSiqzeRJNCckhEXEX.zuDa4MclAW4qWi4yAyCr88dNkXS19xsGOxBGpudIEYk;
userInfo=
{"id":"1000290499","username":"2034885420@qq.com","password":"0x02000000511e
c6b5b749cd46de1cf27e985d76cd0b5f608d2d7f8172158330542a1718a781d215aaee94b1de
4b5377df41109118","userType":"normal","email":"2034885420@qq.com","userLang"
:"cn","userPageSize":25,"isSend":true,"sendLang":"cn","recieveEmails":
[],"groupName":"law","libraryCode":"law,taa,hr,HKBold","licences":1,"telepho
ne":"18355442634","conf":""}; loginin=true;
loginId=e6029baaa3b9484eb96ecf84287938f6;
acw_tc=78e2b62616426938197051079e9f34ff3064942aaf94eb0cf973020c11;
Hm_lpvt_fecce484974a74c6d10f421b6d3bd395=',
                'check=valid; TY_SESSION_ID=346b2eee-b5d7-4e77-8e10-
fd5498f42ede; autologin=true; username=632101734@qq.com;
acw_tc=78cebd9816427533357544001e2808a8cd92fc1f6388c9ef189f60038f;
connect.sid=s:Z3qGUVaiBnAsqnk4OKZY5ELvtrB3GHCv.L4+emsCG+duBmRk5BwwAHOM2VyyE5
yjj+/7BXr5flHg; userInfo=
{"id":"1000290723","username":"632101734@qq.com","password":"0x0200000030923
68a749aa28c92e2afa813943ebb1d1d136f1baa5e9a3533b889a410991b300166b3365384146
50dec260dcf5280","userType":"normal","email":"632101734@qq.com","userLang":"
cn","userPageSize":25,"isSend":true,"sendLang":"cn","recieveEmails":
[],"groupName":"law","libraryCode":"law,taa,hr,HKBold","licences":1,"telepho
ne":"18169538061","conf":""}; loginin=true;
loginId=29891ca248749449df3198c9dba1df4; userConfig={"moduleList":
[],"userStaffType":0,"isIpUser":0,"parentId":0,"sourceSiteUrl":null,"sourceS
iteName":null,"clientSource":"自主注
册","trial":true,"expire":false,"caseView":false,"advancedlSearchExpires":"2
","proximateLatestExpires":"48","proximateSearch":false};
Hm_lvt_fecce484974a74c6d10f421b6d3bd395=1642745554,1642747324,1642753336,164
2753375; Hm_lpvt_fecce484974a74c6d10f421b6d3bd395='
            ]

    cookie = random.choice(Cookies)

    headers = {
        'Accept': 'application/json, text/plain, */*',
        'Accept-Encoding': 'gzip, deflate, br',
        'Accept-Language': 'zh-CN,zh;q=0.9',
        'Appversion': '2021.03.17.1',
        'Cache-Control': 'no-cache',
        'Connection': 'keep-alive',
        'Content-Type': 'application/json;charset=utf-8',
        'Cookie': (cookie+str(int(time.time()))).encode('utf-8'),
        'Host': 'law.wkinfo.com.cn',
```

```
        'Identification': '_2a66579079e311ecbc3ba7fea9a4723d',
        'Referer': 'https://law.wkinfo.com.cn/judgment-documents/list?mode=advanced',
        'sec-ch-ua-mobile': '?0',
        'Sec-Fetch-Dest': 'empty',
        'Sec-Fetch-Mode': 'cors',
        'Sec-Fetch-Site': 'same-origin',
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36 Edg/97.0.1072.62',
        'X-Tingyun-Id': 'tN6Win9ZeY4;r=96286256',
    }

    params = {
        "indexId": "law.case",
        "query": {
            "queryString": "typeOfCase:刑事 AND typeOfDecision:((001))",
            "filterQueries": [],
            "filterDates": dateList
        },
        "searchScope": {"treeNodeIds": []},
        "relatedIndexQueries": [],
        "sortOrderList": [{"sortKey": "judgmentDate", "sortDirection": "DESC"}],
        "pageInfo": {"limit": int(num), "offset": 0},
        "otherOptions": {
            "requireLanguage": "cn",
            "relatedIndexEnabled": False,
            "groupEnabled": False,
            "smartEnabled": True,
            "buy": False,
            "summaryLengthLimit": 100,
            "advanced": True,
            "synonymEnabled": True,
            "isHideBigLib": 0,
            "relatedIndexFetchRows": 5,
            "proximateCourtID": "",
            "module": ""
        },
        "chargingInfo": {"useBalance": True}}

    url = 'https://law.wkinfo.com.cn/csi/search'
    result = requests.session().post(url, headers=headers, json=params)

    if result.status_code == 200:
        content = result.content
        return htmlParser(content) #返回标题列表，用于前端展示爬虫结果
    else:
        print(result.content.decode('utf-8'))
        return []
```

2.3.3 解析与下载

对请求的结果通过json.loads()转化为字典，获取文书列表documentList，遍历文书列表，对每个document进行正则分析等来提取文书内容，并保存至static/Downloads目录下

```
def htmlParser(content):
```

```
    titleList = []
    contentDict = json.loads(content,strict=False)
    documentList = contentDict['documentList']
    for i in range(len(documentList)):
        titleList.append(wenshuDownload(documentList[i]))
    return titleList #返回标题列表

def wenshuDownload(document):
    title = document['title'] #文书标题
    additionalFields = document['additionalFields'] #文书内容
    res = title + '\r' #待保存的结果
    for key,value in additionalFields.items(): #正则分析
        if key == "judgmentDate":
            continue
        if key == "instancecode":
            continue
        if key == "referenceLevel":
            continue
        if key == "updateContentStatus":
            continue
        if key == "product":
            continue
        if value == None:
            continue
        if value=='':
            continue
        r = re.compile(r'\s+')
        value = r.sub("\r", value) #正则表达式替换
        res+=value
    path = r'static/Downloads/'+title+'.txt' #保存路径
    with open(path,'w',encoding='utf-8') as file:
        file.write(res)
    return title #返回标题
```

2.4 分词实现

通过分析多份法律文书，最终决定分为4类词性：{"不涉及地点名词": nr, "涉及地点名词": ns, "案由": r, "形容词": adj}，同时添加自定义词典以增强分词的通用性。

2.4.1 方法分析

2.4.1.1 out方法

获取前端界面得到文书的内容，然后仅需调用该方法即可得到分词结果。

传出处理后的分词结果，形式为字典，其中value值为一个列表，其中的元素为某一类型的词语的集合，每个value值对应的key值为该类型词语的名称

```
"""传出结果，形式为字典，key为value中词语类型，value为列表"""
def out(content):
    jieba.load_userdict("static/dict.txt")
    # 补充由于结构较为复杂难以分出的案由
    content = content.replace(" ", "").replace("\n", "")
    words = pseg.lcut(content, use_paddle=True)
    wordsList = []
    # 初步分词结果
    flagList = []
```

```
    # 初步分词结果对应的词性
    length = 0
    for word, flag in words:
        length += 1
        wordsList.append(str(word))
        flagList.append(str(flag))
    nr = get_about_nr(wordsList, flagList, length)
    ns = get_about_ns(wordsList, flagList, length)
    # v = get_v(wordsList, flagList, length)
    r = get_r(wordsList, flagList, length)
    adj = get_adj(wordsList, flagList, length)
    result = {"不涉及地点名词": nr, "涉及地点名词": ns, "案由": r, "形容词": adj}
    return result
```

2.4.1.2 get_about_nr方法

得到不包含地点的名词（考虑到名词数量过多）如图所示

```
#在引入自定义词典后，wordsList为分词的初始结果，flagList为分词得到的词性的初始结果，
length为列表长度
def get_about_nr(wordsList, flagList, length):
```

后续方法与该方法传入参数相同

2.4.1.3 get_about_ns方法

得到包含地点的名词

```
def get_about_ns(wordsList, flagList, length):
```

2.4.1.4 get_r方法

得到案由

```
def get_r(wordsList, flagList, length):
```

2.4.1.5 get_adj方法

得到形词

```
def get_adj(wordsList, flagList, length):
```

2.4.1.6 getStrFromTxt & showRes方法

由于本地测试时，传入为txt格式的文本文件，该方法将从txt文件中获得文本，返回形式为字符串，主要目的是为处理由于编码不同产生的部分问题，该方法仅供本地测试使用
将本地分词结果进行输出，便于本地测试

2.4.2 分词过程

2.1 直接使用jieba分词结果的部分

人名部分

2.2 结合jieba分词结果及对部分字符串的解析部分

文化程度部分；民族部分；性别部分

2.3 结合jieba分词时构建的自定义词典和对字符串的解析

地名；相关法院名称；案由
地名采用向后匹配策略，由于部分地名无法识别（例如西藏自治区那曲市），故添加至自定义词典
相关法院采用添加前继位名词后匹配地点名词的方式
案由构成的分析，首先进行字符串解析，由于形式较为复杂，且不存在普适性，故全部添加至自定义词典并保存为re词性（dict.txt文件）

2.5 前后连接

利用flask框架实现前后段连接

2.5.1 初始

项目启动后，直接在浏览器输入http://127.0.0.1:5000/来发出GET请求即可进入初始界面，路由实现如下

```
@app.route('/', methods = ['GET','POST'])
def jiebas():
    if request.method == 'GET':
        return render_template('司法大数据自动化标注与分析.html', text='')
```

2.5.2 爬虫

通过初始页面的爬虫按钮来提交爬取日期和数量，发出POST请求，app.py调用wenshu_spider.py来爬取文书，并返回Jinja2渲染文书标题后的爬虫展示页面

```
@app.route('/文书爬取', methods = ['GET','POST'])
def crawl():
    if request.method == 'GET':
        return render_template('司法大数据自动化标注与分析.html', text='')  # 返回模板
    else:
        #获取参数值
        beginDate = request.form.get('beginDate')
        endDate = request.form.get('endDate')
        num = request.form.get('num')
        #设置参数默认值
        if beginDate == "":
            beginDate = "2002.12.25"
        if endDate == "":
            endDate = "2021.12.25"
        if num == "":
            num = "10"
        titleList = wenshu_spider.spider(beginDate,endDate,num)
        return render_template('文书爬取.html', titles=titleList) #Jinja2渲染
```

2.5.3 分词

通过初始页面的分词按钮来提交案件文本，发出POST请求，app.py调用jiebaAnalysis.py来进行分词，并返回Jinja2渲染分词结果后的页面

```python
@app.route('/', methods = ['GET','POST'])
def jiebas():
    if request.method == 'GET':
        return render_template('司法大数据自动化标注与分析.html', text='')  # 返回模板
    else:
        text = request.form.get('textArea') #获取案件文本
        result = jiebaAnalysis.out(text)
        jieba1s = result.get('不涉及地点名词')
        jieba2s = result.get('涉及地点名词')
        jieba3s = result.get('案由')
        jieba4s = result.get('形容词')
        return render_template('司法大数据自动化标注与分析.html', text=text, jieba1s=jieba1s,jieba2s=jieba2s,jieba3s=jieba3s,jieba4s=jieba4s) #Jinja2渲染

```

```python
def spider(beginDate, endDate, num):
    """
    :param beginDate: "2020.01.20"
    :param endDate: "2021.12.25"
    :param num:
    :return:
    """
    dateList = [] #表单中的时间参数
    date = "judgmentDate:["+beginDate+' TO '+endDate+']'
    dateList.append(date)

    #小组成员Cookie
    Cookies = ['autologin=true; username=2034885420@qq.com; userConfig=
{"moduleList":
[],"userStaffType":0,"isIpUser":0,"parentId":0,"sourceSiteUrl":null,"sourceS
iteName":null,"clientSource":"自主注
册","trial":true,"expire":false,"caseView":false,"advancedlSearchExpires":"2
","proximateLatestExpires":"48","proximateSearch":false}; check=valid;
TY_SESSION_ID=827ba7ea-6794-4eaf-abf1-8f66eb227b74; connect.sid=s:ZXBCNi-
URL3-e59bSiqzeRJNCckhEXEX.zuDa4MclAW4qWi4yAyCr88dNkXS19xsGOxBGpudIEYk;
userInfo=
{"id":"1000290499","username":"2034885420@qq.com","password":"0x020000000511e
c6b5b749cd46de1cf27e985d76cd0b5f608d2d7f8172158330542a1718a781d215aaee94b1de
4b5377df41109118","userType":"normal","email":"2034885420@qq.com","userLang"
:"cn","userPageSize":25,"isSend":true,"sendLang":"cn","recieveEmails":
[],"groupName":"law","libraryCode":"law,taa,hr,HKBold","licences":1,"telepho
ne":"18355442634","conf":""}; loginin=true;
loginId=e6029baaa3b9484eb96ecf84287938f6;
acw_tc=78e2b6261642693819705107 9e9f34ff3064942aaf94eb0cf973020c11;
Hm_lpvt_fecce484974a74c6d10f421b6d3bd395=',
```

```
                'check=valid; TY_SESSION_ID=346b2eee-b5d7-4e77-8e10-fd5498f42ede; autologin=true; username=632101734@qq.com; acw_tc=78cebd9816427533357544001e2808a8cd92fc1f6388c9ef189f60038f; connect.sid=s:Z3qGUVaiBnAsqnk4OKZY5ELvtrB3GHCv.L4+emsCG+duBmRk5BwwAHOM2VyyE5yjj+/7BXr5flHg; userInfo=%7B%22id%22%3A%221000290723%22%2C%22username%22%3A%22632101734%40qq.com%22%2C%22password%22%3A%220x0200000030923168a749aa28c92e2afa813943ebb1d1d136f1baa5e9a3533b889a410991b300166b33653841465Odec260dcf5280%22%2C%22userType%22%3A%22normal%22%2C%22email%22%3A%22632101734%40qq.com%22%2C%22userLang%22%3A%22cn%22%2C%22userPageSize%22%3A25%2C%22isSend%22%3Atrue%2C%22sendLang%22%3A%22cn%22%2C%22recieveEmails%22%3A%5B%5D%2C%22groupName%22%3A%22law%22%2C%22libraryCode%22%3A%22law%2Ctaa%2Chr%2CHKBold%22%2C%22licences%22%3A1%2C%22telephone%22%3A%2218169538061%22%2C%22conf%22%3A%22%22%7D; loginin=true; loginId=298901ca248749449df3198c9dba1df4; userConfig=%7B%22moduleList%22%3A%5B%5D%2C%22userStaffType%22%3A0%2C%22isIpUser%22%3A0%2C%22parentId%22%3A0%2C%22sourceSiteUrl%22%3Anull%2C%22sourceSiteName%22%3Anull%2C%22clientSource%22%3A%22%E8%87%AA%E4%B8%BB%E6%B3%A8%E5%86%8C%22%2C%22trial%22%3Atrue%2C%22expire%22%3Afalse%2C%22caseView%22%3Afalse%2C%22advancedlSearchExpires%22%3A%222%22%2C%22proximateLatestExpires%22%3A%2248%22%2C%22proximateSearch%22%3Afalse%7D; Hm_lvt_fecce484974a74c6d10f421b6d3bd395=1642745554,1642747324,1642753336,1642753375; Hm_lpvt_fecce484974a74c6d10f421b6d3bd395='
            ]

    cookie = random.choice(Cookies)

    headers = {
        'Accept': 'application/json, text/plain, */*',
        'Accept-Encoding': 'gzip, deflate, br',
        'Accept-Language': 'zh-CN,zh;q=0.9',
        'Appversion': '2021.03.17.1',
        'Cache-Control': 'no-cache',
        'Connection': 'keep-alive',
        'Content-Type': 'application/json;charset=utf-8',
        'Cookie': (cookie+str(int(time.time()))).encode('utf-8'),
        'Host': 'law.wkinfo.com.cn',
        'Identification': '_2a66579079e311ecbc3ba7fea9a4723d',
        'Referer': 'https://law.wkinfo.com.cn/judgment-documents/list?mode=advanced',
        'sec-ch-ua-mobile': '?0',
        'Sec-Fetch-Dest': 'empty',
        'Sec-Fetch-Mode': 'cors',
        'Sec-Fetch-Site': 'same-origin',
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36 Edg/97.0.1072.62',
        'X-Tingyun-Id': 'tN6Win9ZeY4;r=96286256',
    }

    params = {
        "indexId": "law.case",
        "query": {
            "queryString": "typeOfCase:刑事 AND typeOfDecision:((001))",
            "filterQueries": [],
            "filterDates": dateList
        },
        "searchScope": {"treeNodeIds": []},
        "relatedIndexQueries": [],
        "sortOrderList": [{"sortKey": "judgmentDate", "sortDirection": "DESC"}],
        "pageInfo": {"limit": int(num), "offset": 0},
        "otherOptions": {
```

```
            "requireLanguage": "cn",
            "relatedIndexEnabled": False,
            "groupEnabled": False,
            "smartEnabled": True,
            "buy": False,
            "summaryLengthLimit": 100,
            "advanced": True,
            "synonymEnabled": True,
            "isHideBigLib": 0,
            "relatedIndexFetchRows": 5,
            "proximateCourtID": "",
            "module": ""
        },
        "chargingInfo": {"useBalance": True}}

    url = 'https://law.wkinfo.com.cn/csi/search'
    result = requests.session().post(url, headers=headers, json=params)

    if result.status_code == 200:
        content = result.content
        return htmlParser(content) #返回标题列表，用于前端展示爬虫结果
    else:
        print(result.content.decode('utf-8'))
        return []
```

3. 不足

1. 由于我们未购买威科先行数据库，故爬取数量有限，每天只允许爬取几百份案例
2. 对于单文本多犯人的案例，最好多次分词标注，一次性标注的话json文件较乱
3. 虽然已经扩展了分词词典，但可能仍存在难以识别的地名、人名

4. 分工

组长：

于欣博 201250165

前后端结合

组员：

马子睿 201250163

分词

刘云辉 201250166

爬虫、前端、前后端结合