Part 1.前言

####（附：本次编码涵盖的所有功能均为java语言实现）

- 结对项目作业
- 结对同学高裕翔的博客
- 个人github传送门

Part 2.具体分工

本次的结对作业我们简单的拆分成:

1. 爬虫部分(高裕祥负责)
2. 词频分析部分(柯奇豪负责)
3. 附加题部分(柯奇豪负责处理数据、高裕祥负责处理爬取相关内容)
4. 单元测试(柯奇豪负责)

Part 3.PSP表

PSP3.1	Personal Software Process Stages	预估耗时（分钟）	实际耗时（分钟）
Planning	计划	---	---
· Estimate	· 估计这个任务需要多少时间	60	90
Development	开发		
· Analysis	· 需求分析 (包括学习新技术)	120	240
· Design Spec	· 生成设计文档	20	30
· Design Review	· 设计复审	10	20
· Coding Standard	· 代码规范 (为目前的开发制定合适的规范)	10	20
· Design	· 具体设计	120	140
· Coding	· 具体编码	1800	2200
· Code Review	· 代码复审	30	100
· Test	· 测试（自我测试，修改代码，提交修改）	30	120
Reporting	报告		
· Test Repor	· 测试报告	10	20

PSP3.1	Personal Software Process Stages	预估耗时（分钟）	实际耗时（分钟）
· Size Measurement	· 计算工作量	10	20
· Postmortem & Process Improvement Plan	· 事后总结, 并提出过程改进计划	30	30
	合计	2250	3030

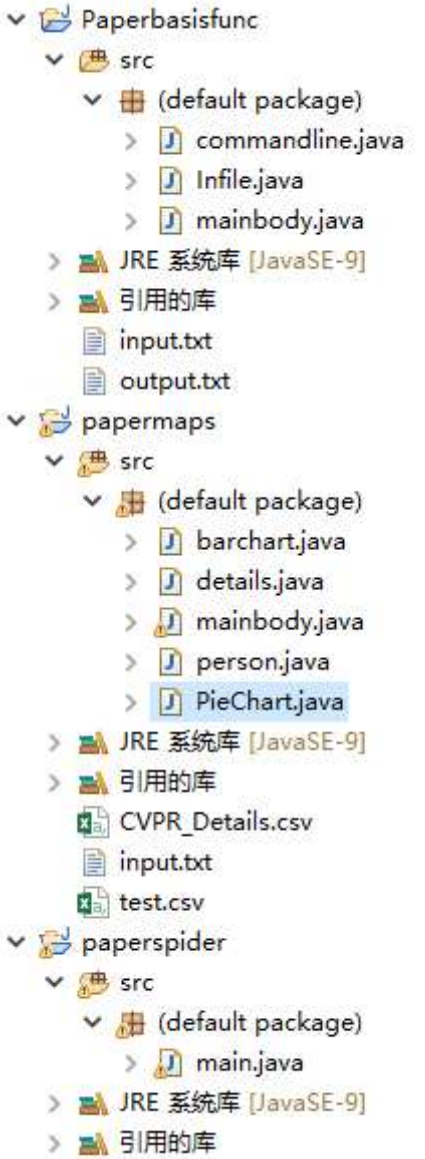
Part 4.解题思路描述与设计实现说明

1. 爬虫使用

- 使用语言：java
- 本次的爬虫我们主要利用Java爬虫工具Jsoup解析进行爬取，基本上爬虫的流程可以归纳如下：

2. 代码组织与内部实现设计（类图）

我自己在本地是建了三个java项目分别负责不同的功能模块：



- PaperBasicFuction (处理基础的词频统计功能)

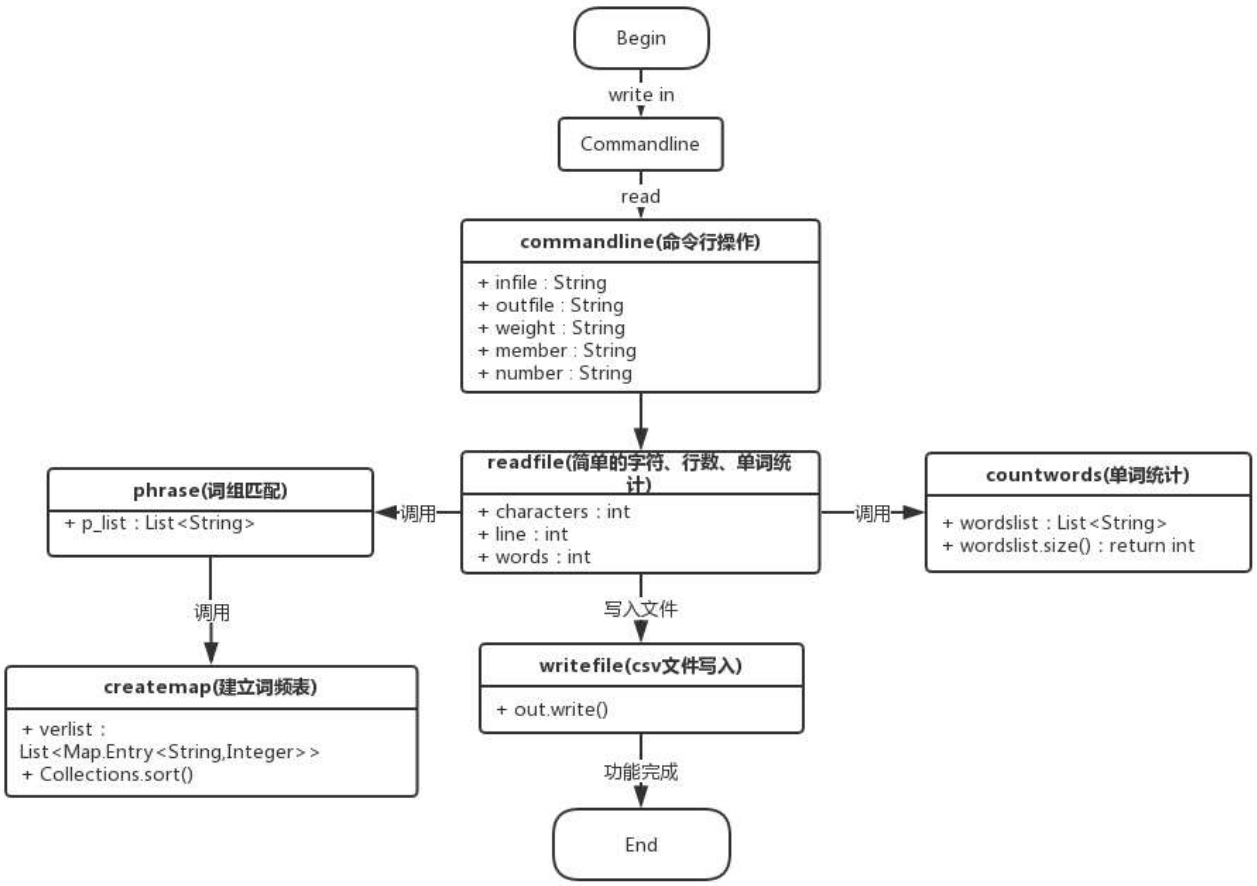
1. commandline.java (实现命令行处理功能)
2. infile。java (实现基础词频分析功能)
3. mainbody.java (构成WordCount的主体main)

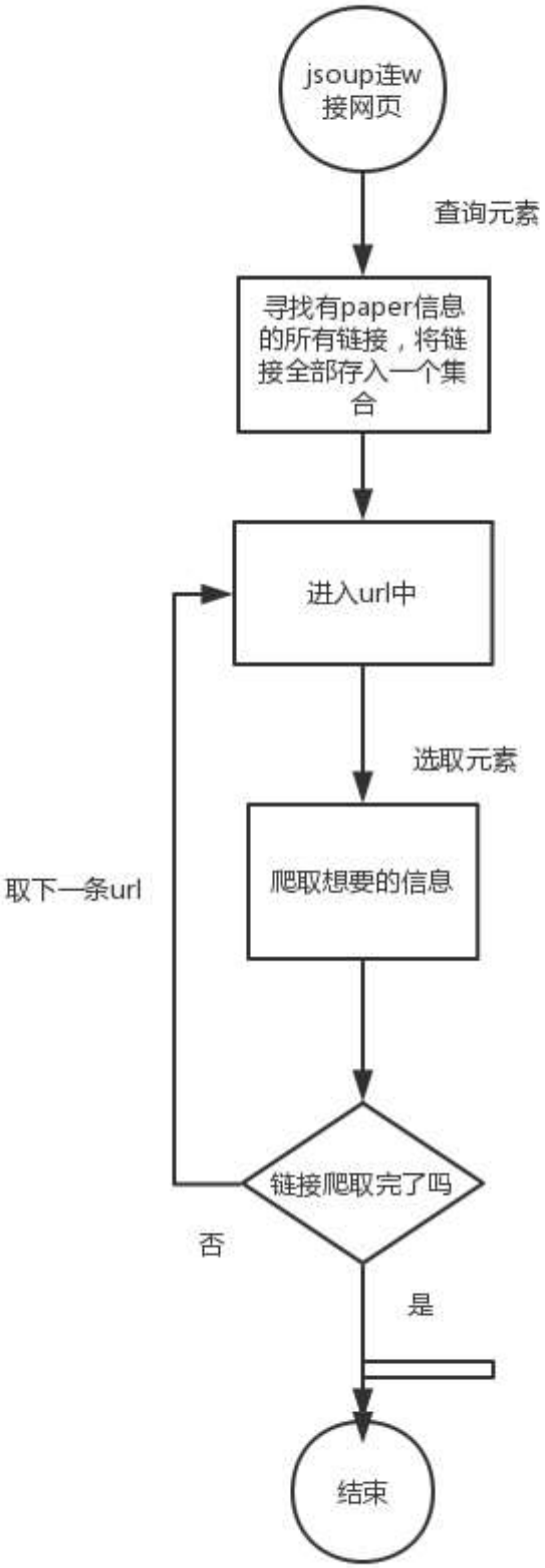
- PaperMaps (处理附加的图形化显示功能)

1. barchart.java (负责柱状图的生成)
2. Piechart.java (负责饼图的功能)
3. details.java (javabean模块进行数据处理)
4. person.java (论文作者相关性数据处理)
5. mainbody.java (构成AdditionalFunction主体main)

- PaperSpider (负责爬虫功能的实现)

3. 说明算法的关键与关键实现部分流程图(含关键代码的解释)





接下来我就简单的介绍一下我这次新补充或者改进优化的几处地方，与上回个人作业相关功能类似的地方在这里我就不再赘述

section 1.命令行功能的实现

```
options.addOption("h", "help", false,"Helps About this Command");
options.addOption("i", "input", true, "Input this file");
options.addOption("o", "output", true, "Output this file");
```

```
    options.addOption("w", "weight", true, "Weight of the word");
    options.addOption("m", "members", true, "Members of count words");
    options.addOption("n", "number", true, "Numbers of output words");

    try {
        CommandLine commandline = parser.parse(options, args);
        if(commandline.hasOption("h")) {
            HelpFormatter formatter = new HelpFormatter();
            formatter.printHelp("CommandLineParameters",options);
        }
        if(commandline.hasOption("i")) {
            infile =  commandline.getOptionValue("i");
        }
        if(commandline.hasOption("o")) {
            outfile =  commandline.getOptionValue("o");
        }
        if(commandline.hasOption("w")) {
            weight =  commandline.getOptionValue("w");
        }
        if(commandline.hasOption("o")) {
            member =  commandline.getOptionValue("m");
        }
        if(commandline.hasOption("o")) {
            number =  commandline.getOptionValue("n");
        }
    } catch (ParseException e) {
        e.printStackTrace();
    }
```

原先我的思路是使用命令行读入转String然后用if或者switch-case来逐一的判断匹配，显然这种方式有些许繁琐，为了方便使用命令行的处理，我转而借助于commons-cli来简化工作，显然操作起来就变得简单明了，方便我获取命令行参数

section 2.小细节的优化

关于标题和内容权重不同的情况，我简单的通过两句表达式做处理，不做多余的判断处理

```
w_title = w*10+(w-1)*(-1); w_content = 1;
```

依照于题目的要求，文本的格式相对的统一，所以我采用逐行获取后匹配，对应的进行相关操作

```
while((strline = buffReader.readLine())!=null){
    Matcher t_mat = t_pat.matcher(strline);
    Matcher c_mat = c_pat.matcher(strline);
    if(t_mat.find()) {
        t_str=t_mat.group(1).toLowerCase();
        t_words.append(t_str).append(" ");
        phrase(t_str,w_title,m);
        characters += t_str.length()+1;
        lines++;
    }
    if(c_mat.find()) {
        c_str=c_mat.group(1).toLowerCase();
        c_words.append(c_str).append(" ");
        phrase(c_str,w_content,m);
        characters += c_str.length();
```

```
            lines++;
        }
    }
    buffReader.close();
```

section 3.网页爬虫

```
    for(Element paper:urlLink) {                                    //循环爬取url队列
        String URL = paper.attr("href");
        if(!URL.contains("https://")){
        URL = "openaccess.thecvf.com/"+URL;
        }

        Document doc = Jsoup.connect("http://"+URL).timeout(80000).maxBodySize(0).get();
        Elements paperdatas=doc.select("#content");
        Elements title1=paperdatas.select("#papertitle");
        Elements abs=paperdatas.select("#abstract");
        Elements authors = paperdatas.select("#authors");
        Elements opway = paperdatas.select("a[href]");

            String author=authors.select("b").text();
            String title = title1.text();
            paperid=paperid+1;
            String abst=abs.text();
            String openway=opway.text();

        System.out.println("ID: "+paperid);
        System.out.println("title:"+ title );
        System.out.println("link: "+ URL);
        System.out.println("abstract: " +abst);
        System.out.println("authors: "+ author);
        System.out.println("relate: "+ openway);
        System.out.println("relate link: ");
        for(Element linklink:opway) {
            String oplink = linklink.attr("href");
            System.out.println( "http://openaccess.thecvf.com/" + oplink);
        }
    }
```

section 4.命令行处异常处理，输出err信息并进入中断

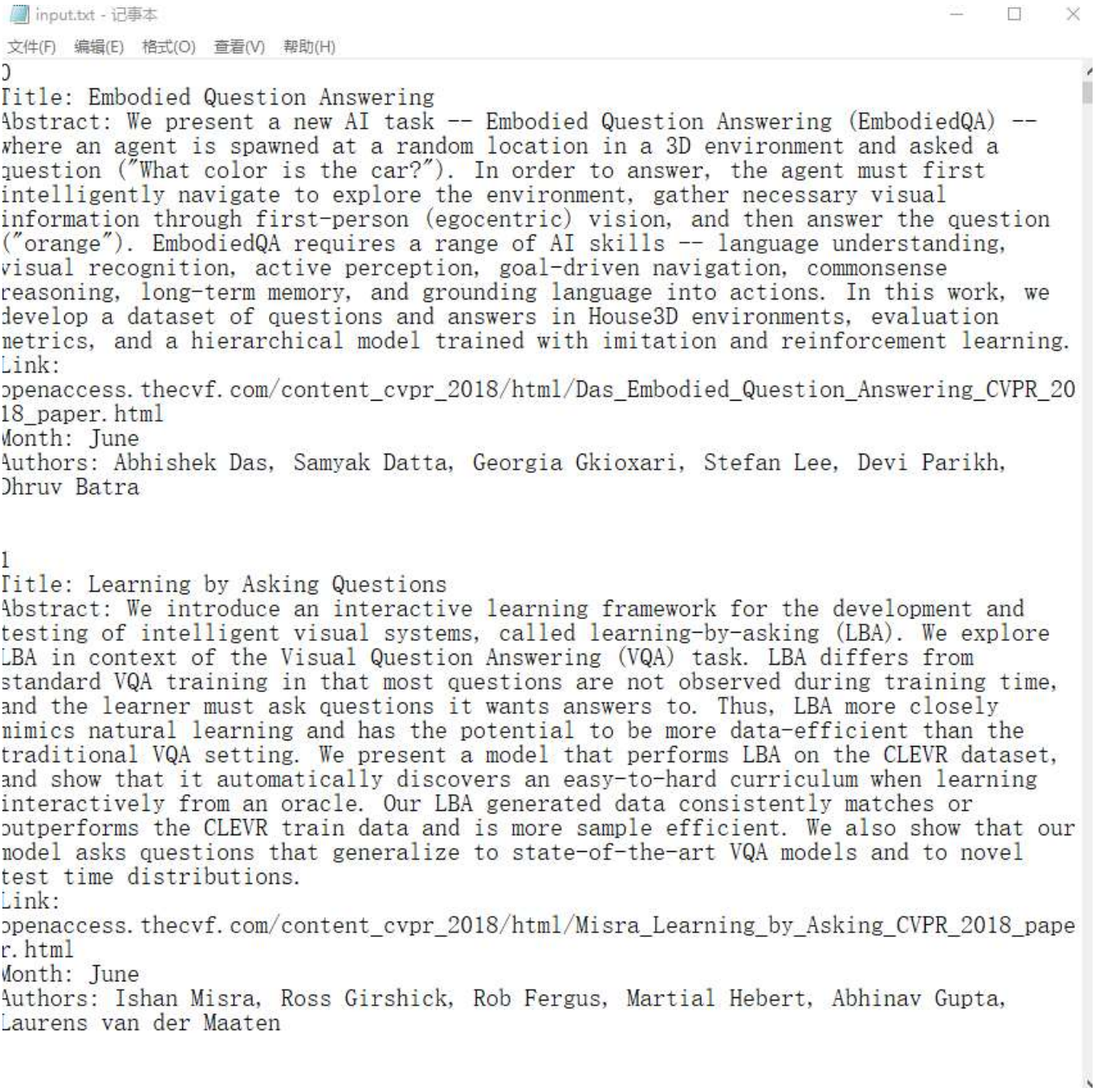
```
    if (commandline.getOptions().length > 0) {

        ...

        if(commandline.hasOption("m")) {
            member =  commandline.getOptionValue("m");
            if(Integer.parseInt(member)<2 | Integer.parseInt(member)>10) {
                System.err.println("ERROR_MESSAGE:THE PHRASE LENGTH OUT OF BOUNDS (2<=m<=10)"); //词组
    包含合法单词数的限制
                System.exit(1);
            }
        }

        ...
```

```
    if(commandline.hasOption("n")) {
        number =  commandline.getOptionValue("n");
        if(Integer.parseInt(number)>100|Integer.parseInt(number)<0) {
            System.err.println("ERROR_MESSAGE:THE OUTPUT PHRASE LENGTH OUT OF BOUNDS
(0<=n<=100)"); //输出top榜的个数限制
            System.exit(1);
        }
    }

    ...
}
else {
    System.err.println("ERROR_MESSAGE:NO ARGS");     //命令行参数的约束
    System.exit(1);
}
```

Part 5.附加题设计与展示

section 1.详情整合生成具体查询目录.csv并存储于新构建的class details中

因为官网上可以获取的相关信息有限，能够直接爬取下来的信息如下图所示



所以不够我们对其更详细信息的参考，故通过爬取所获得信息，我们将其进行整合成一份CVPR_Details.csv详细目录，截图展示如下

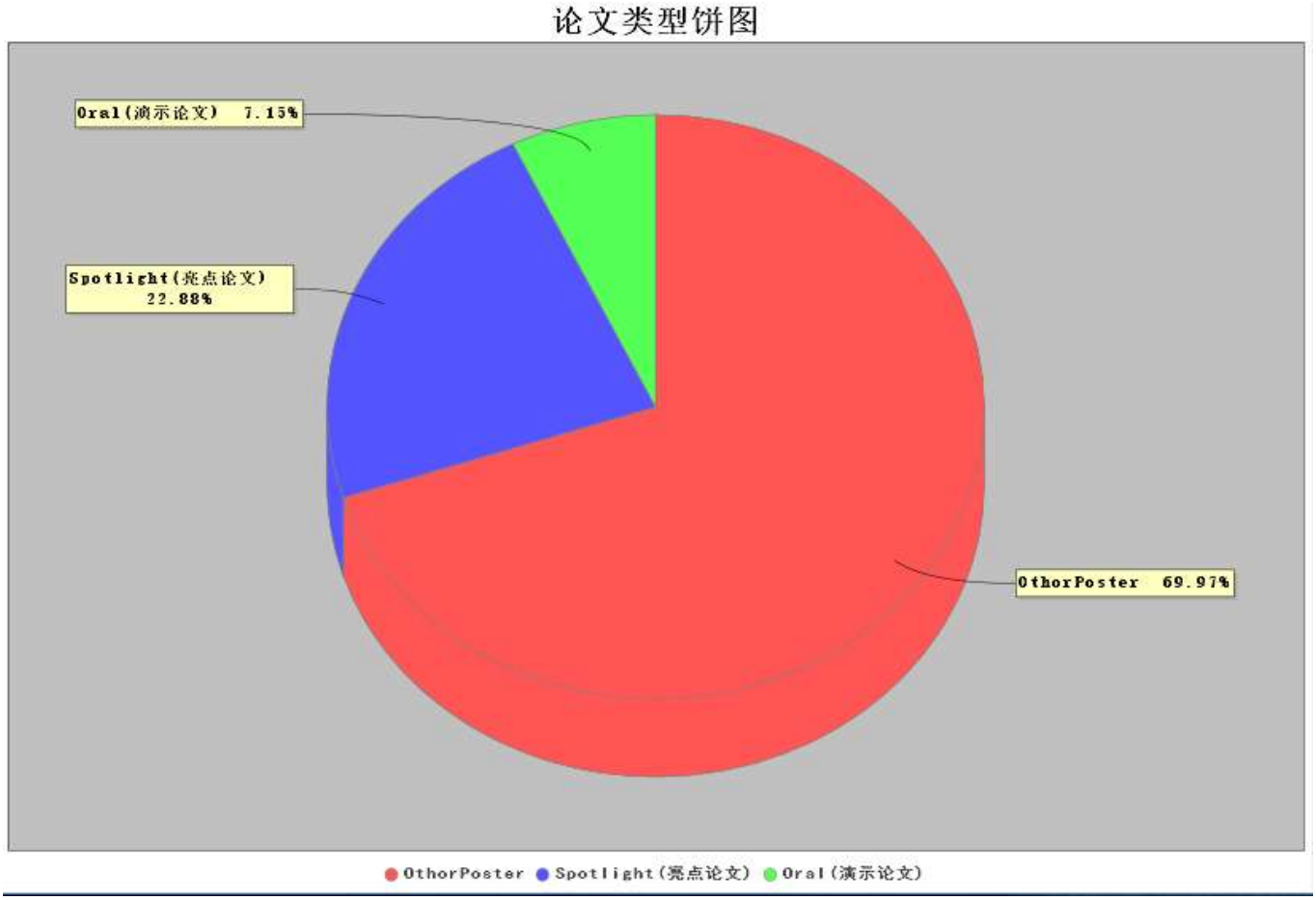
PaperID	Title	Type	Abstract	Authors	Link	Month-2018
3107	joint optim	Poster	Deep neur	Daiki Tanal	openacces	June
1029	fast video (	Spotlight	We presen	Seoung W	openacces	June
2113	end-to-er	Spotlight	Despite the	Lijie Fan, W	openacces	June
	one-shot a	Poster	Learning b	Hongtao Y	openacces	June
4172	motion seg	Poster	Many real	Xun Xu, Lo	openacces	June
170	bpgrad: to	Poster	Understan	Ziming Zha	openacces	June
2351	light field i	Spotlight	We presen	Anna Alpe	openacces	June
1815	crrn: multi-	Poster	Removing	Renjie War	openacces	June
2589	feastnet: fe	Spotlight	Convolutio	Nitika Verr	openacces	June
53	tienet: text	Spotlight	Chest X ra	Xiaosong \	openacces	June
1704	deep marc	Poster	Existing lea	Yiyi Liao, Si	openacces	June
1453	learning to	Oral	We develo	Kwang Mo	openacces	June
3289	oatm: occl	Oral	We presen	Simon Kor	openacces	June
	unsupervis	Poster	The object	Pedro O. P	openacces	June
759	multi-view	Poster	We presen	Shubham T	openacces	June
2268	detect glol	Poster	Effective in	Tiantian W	openacces	June
	multi-evid	Poster	Supervisec	Weifeng G	openacces	June
1919	teaching c	Spotlight	We study t	Oisin Mac A	openacces	June
3451	differential	Poster	In this pap	Badri Patrc	openacces	June
2624	towards ur	Poster	Unseen Ac	Yi Zhu, Yar	openacces	June
3583	towards de	Poster	From hum	Katarzyna	openacces	June
1632	triplet-cen	Poster	Most existi	Xinwei He,	openacces	June
1522	weakly-su	Poster	In this worl	Li Ding, Ch	openacces	June
4001	fast monte	Poster	Size weigh	Aditya Dha	openacces	June
	a face-to-	Poster	Neural net	Hang Chu,	openacces	June
3606	deep back	Poster	The feed fc	Muhamma	openacces	June
2833	ring loss: c	Poster	We motiva	Yutong Zh	openacces	June
3905	between-c	Poster	In this pap	Yuji Tokozu	openacces	June
	pointwise (	Poster	Deep learr	Binh-Son H	openacces	June
	matching g	Poster	We propo	Rotal Kat, I	openacces	June
1058	single imag	Poster	We presen	Xuaner Zha	openacces	June
	dimension	Poster	Many high	Wen-Yan I	openacces	June
1134	decorrelat	Poster	Batch Norr	Lei Huang,	openacces	June
1981	learning to	Poster	Imaging in	Chen Chen	openacces	June
3887	long-term	Poster	Progress to	Apratim Bh	openacces	June
3298	learning tra	Spotlight	Developing	Barret Zop	openacces	June

而这些信息我们将其存放于新构建的class details中，方便我们后续对其进行操作处理

section 2.论文类型的饼图构建

```
public PieChart(){
    DefaultPieDataset data = getDataSet();
    JFreeChart chart = ChartFactory.createPieChart3D("论文类型饼图",data,true,false,false);      //
设置百分比
    PiePlot pieplot = (PiePlot) chart.getPlot();
    DecimalFormat df = new DecimalFormat("0.00%");//获得一个DecimalFormat对象，主要是设置小数问题
    NumberFormat nf = NumberFormat.getNumberInstance();//获得一个NumberFormat对象
    StandardPieSectionLabelGenerator sp1 = new StandardPieSectionLabelGenerator("{0}  {2}", nf,
df);//获得StandardPieSectionLabelGenerator对象
    pieplot.setLabelGenerator(sp1);//设置饼图显示百分比          //没有数据的时候显示的内容
    pieplot.setNoDataMessage("无数据显示");
    pieplot.setCircular(false);
```

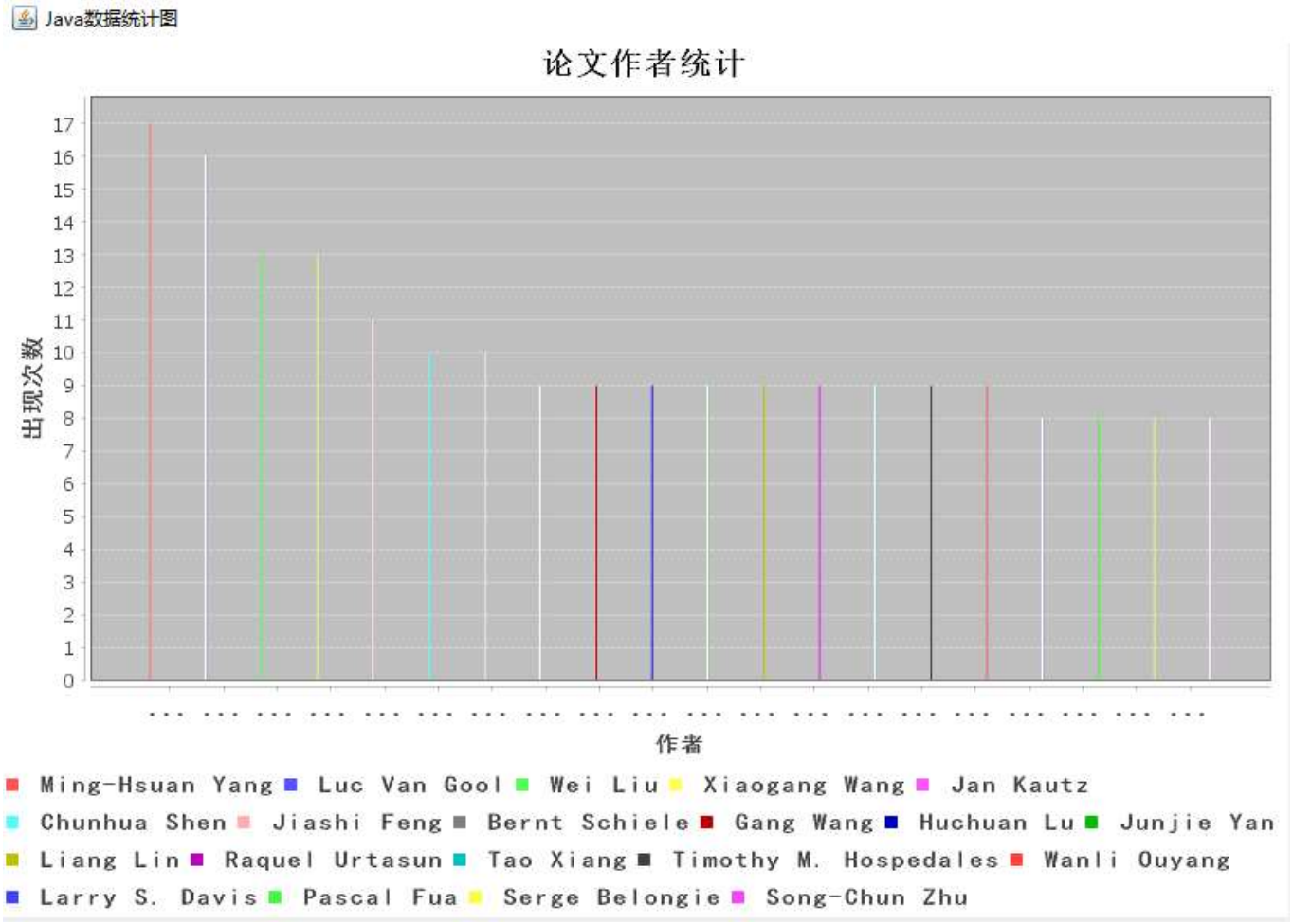
```
    pieplot.setLabelGap(0.02D);
    pieplot.setIgnoreNullValues(true);//设置不显示空值
    pieplot.setIgnoreZeroValues(true);//设置不显示负值
    frame1=new ChartPanel (chart,true);
    chart.getTitle().setFont(new Font("宋体",Font.BOLD,20));//设置标题字体
    PiePlot piePlot= (PiePlot) chart.getPlot();//获取图表区域对象
    piePlot.setLabelFont(new Font("宋体",Font.BOLD,10));//解决乱码
    chart.getLegend().setItemFont(new Font("黑体",Font.BOLD,10));
}
```

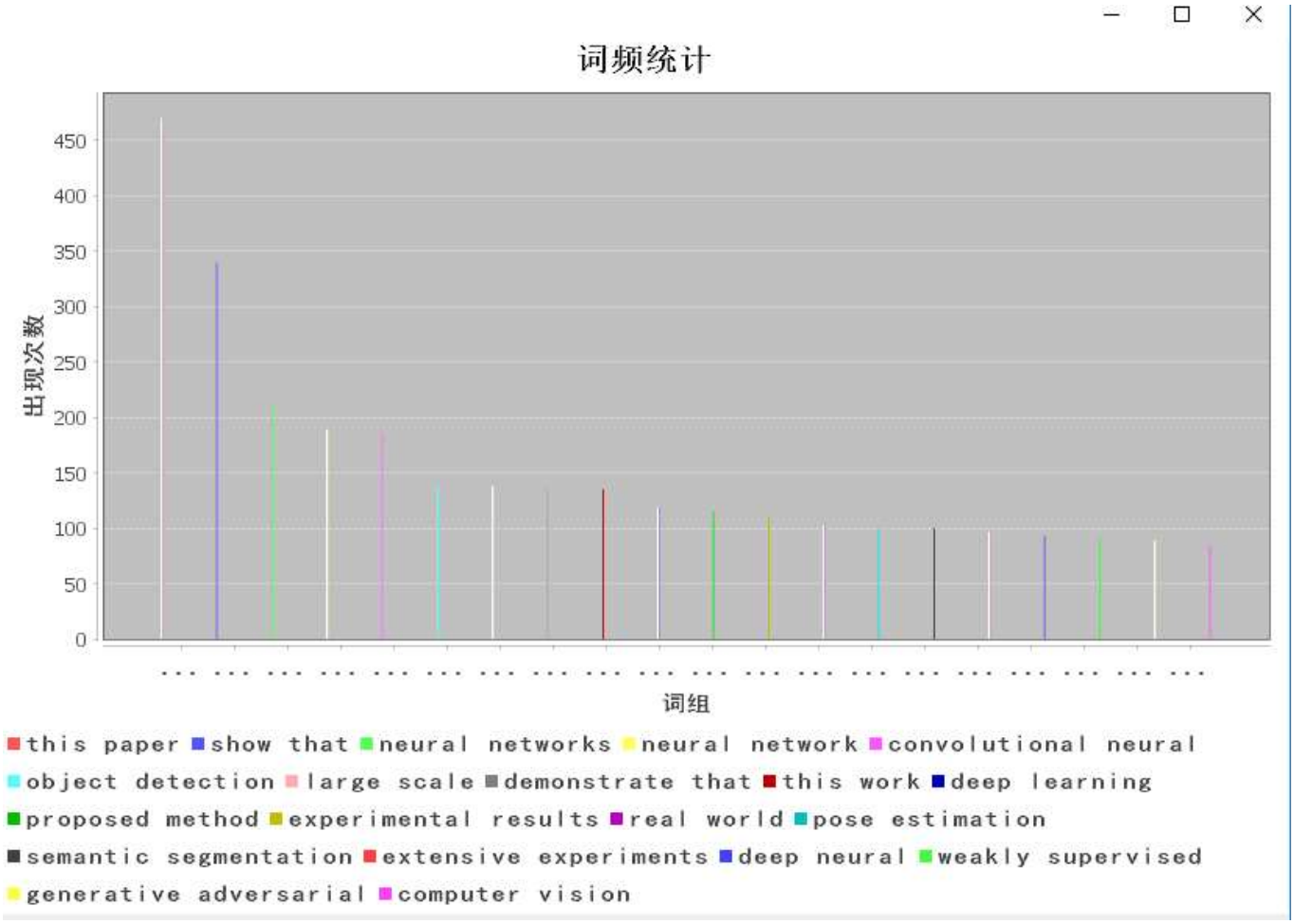


section 3.top作者论文参篇幅与top词频的直方图

```
public barchart() {
    CategoryDataset dataset = getDataSet();
    JFreeChart chart = ChartFactory.createBarChart("论文作者统计", "作者","出现次
数",dataset,PlotOrientation.VERTICAL,true,false,false);
    CategoryPlot plot = chart.getCategoryPlot();
    CategoryAxis domainAxis = plot.getDomainAxis();
    domainAxis.setLabelFont(new Font("黑体",Font.BOLD,14));
    domainAxis.setTickLabelFont(new Font("宋体",Font.BOLD,14));
    ValueAxis rangeAxis=plot.getRangeAxis();
    rangeAxis.setLabelFont(new Font("黑体",Font.BOLD,15));
    chart.getLegend().setItemFont(new Font("黑体", Font.BOLD, 15));
    chart.getTitle().setFont(new Font("宋体",Font.BOLD,20));
    frame = new ChartPanel(chart,true);
}

public CategoryDataset getDataSet() {
```

```
    DefaultCategoryDataset dataset = new DefaultCategoryDataset();
//      System.out.println(topList.size());
    for(int i=0;i<20;i++){
        dataset.addValue(topList.get(i).getValue(), topList.get(i).getKey(),
topList.get(i).getKey());
    }
    return dataset;
}
```



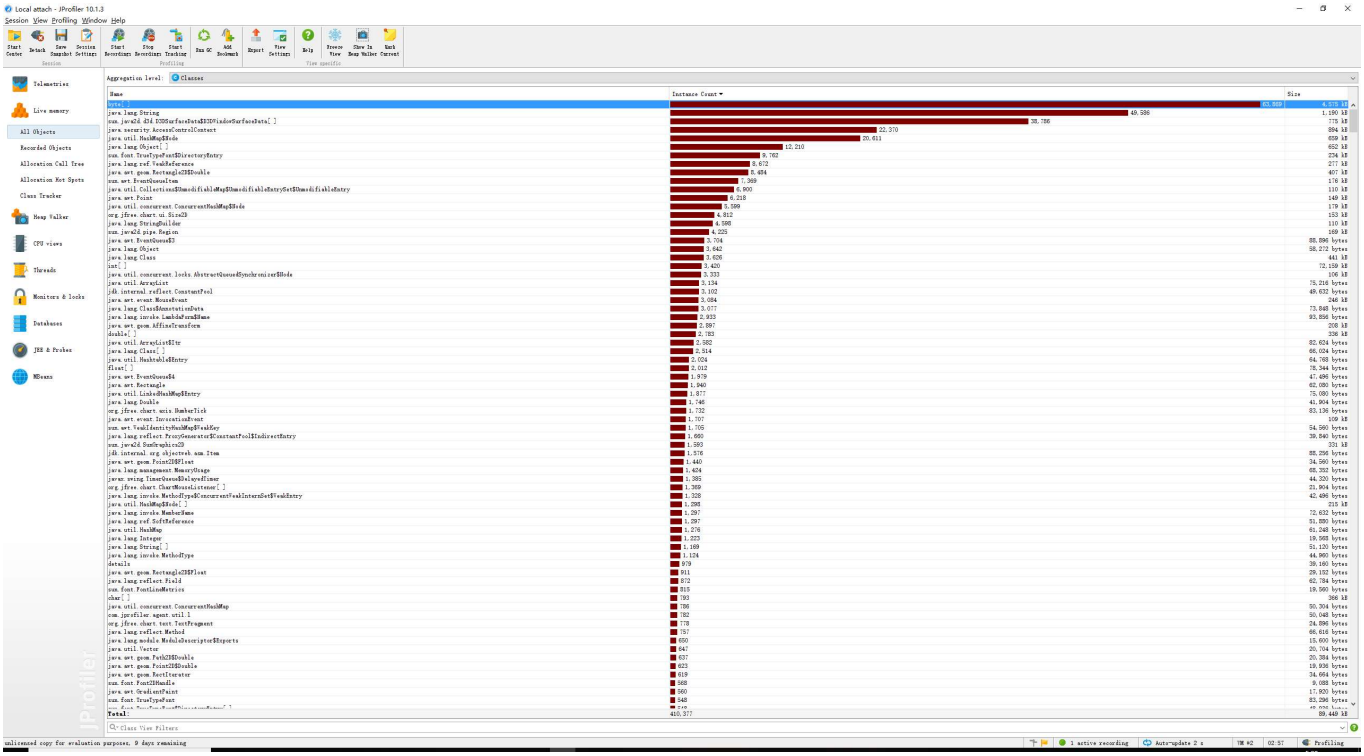


section 4.神经网络的构建(未实现)

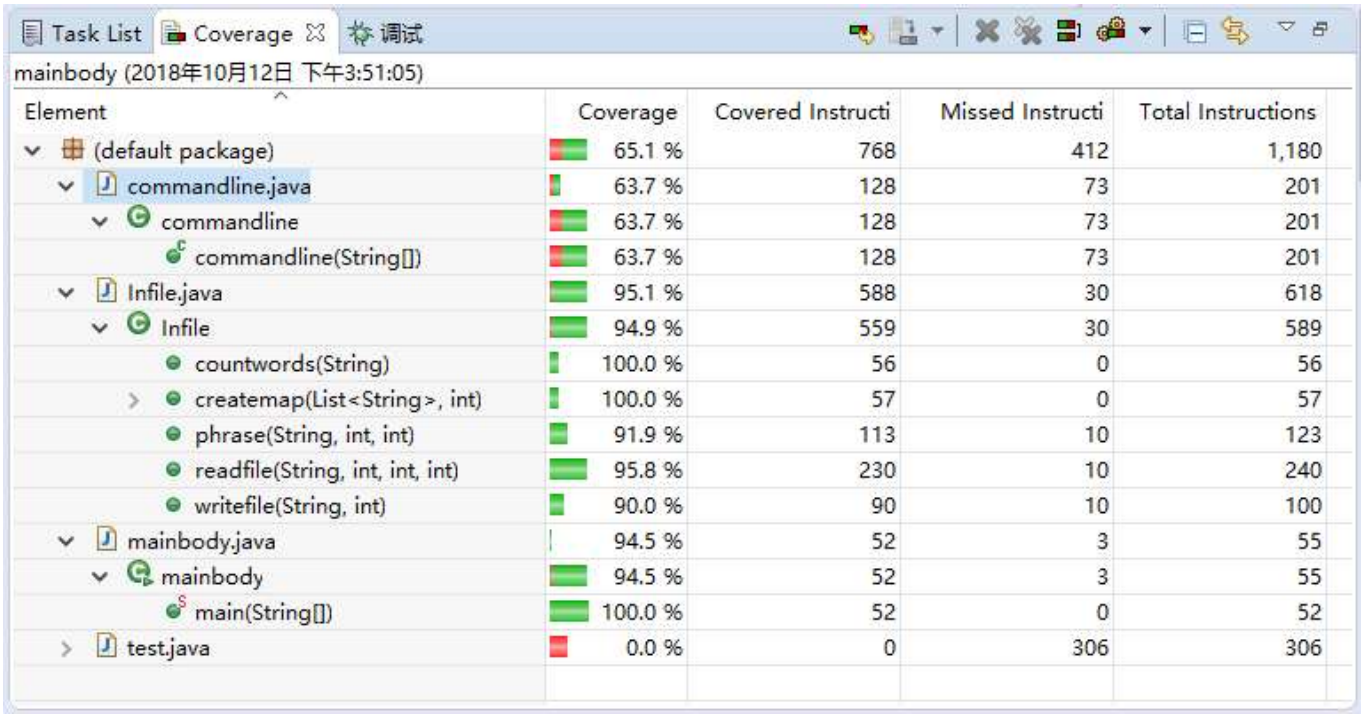
辛苦一晚上，想视图将构建神经网络功能植入代码，但很遗憾gephi-tookit没能够很好的实现，目前尚未实现。 可能是自己太过天真，想视图将所有的功能都用java语言实现，看到大家分门别类的工具使用顿时大悟，很遗憾时间不够没办法再进一步推进，其实大概也了解了Gephi工具的使用(获取“对象-目标-权重”数据后就可以生成神经网络图)。

Part 6.性能分析与改进

命令行 `-i input.txt -o output.txt -n 20 -m 2 -w 0` 下的性能测试图



整体正常运行的代码覆盖如下，主要未执行的部分代码存在与commandline.java中，因为其中存在大部分对于异常处理的操作，当正常运行过程中并不会去访问。



Task List | Coverage | 调试

mainbody (2018年10月12日 下午3:51:05)

Element	Coverage	Covered Instructi	Missed Instructi	Total Instructions
(default package)	65.1 %	768	412	1,180
commandline.java	63.7 %	128	73	201
commandline	63.7 %	128	73	201
commandline(String[])	63.7 %	128	73	201
Infile.java	95.1 %	588	30	618
Infile	94.9 %	559	30	589
countwords(String)	100.0 %	56	0	56
createmap(List<String>, int)	100.0 %	57	0	57
phrase(String, int, int)	91.9 %	113	10	123
readfile(String, int, int, int)	95.8 %	230	10	240
writefile(String, int)	90.0 %	90	10	100
mainbody.java	94.5 %	52	3	55
mainbody	94.5 %	52	3	55
main(String[])	100.0 %	52	0	52
test.java	0.0 %	0	306	306

Part 7.单元测试

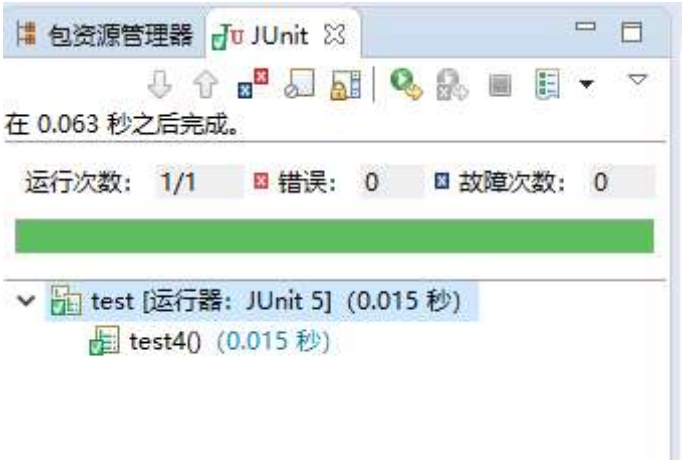
进行了四组单元测试环节，三组针对命令行的不正确输入，一组针对正常运行的代码校验

```
String args[]={"-i","test.txt","-w","1","-o","output.txt","-n","1","-m","1"};

String args[]={"-i","test.txt","-w","1","-o","output.txt","-n","150","-m","2"};

String args[]={""};

String args[]={"-i","test.txt","-w","1","-o","output.txt","-n","10","-m","2"};
```

校验的测试文本如下，应检验，代码运行正常

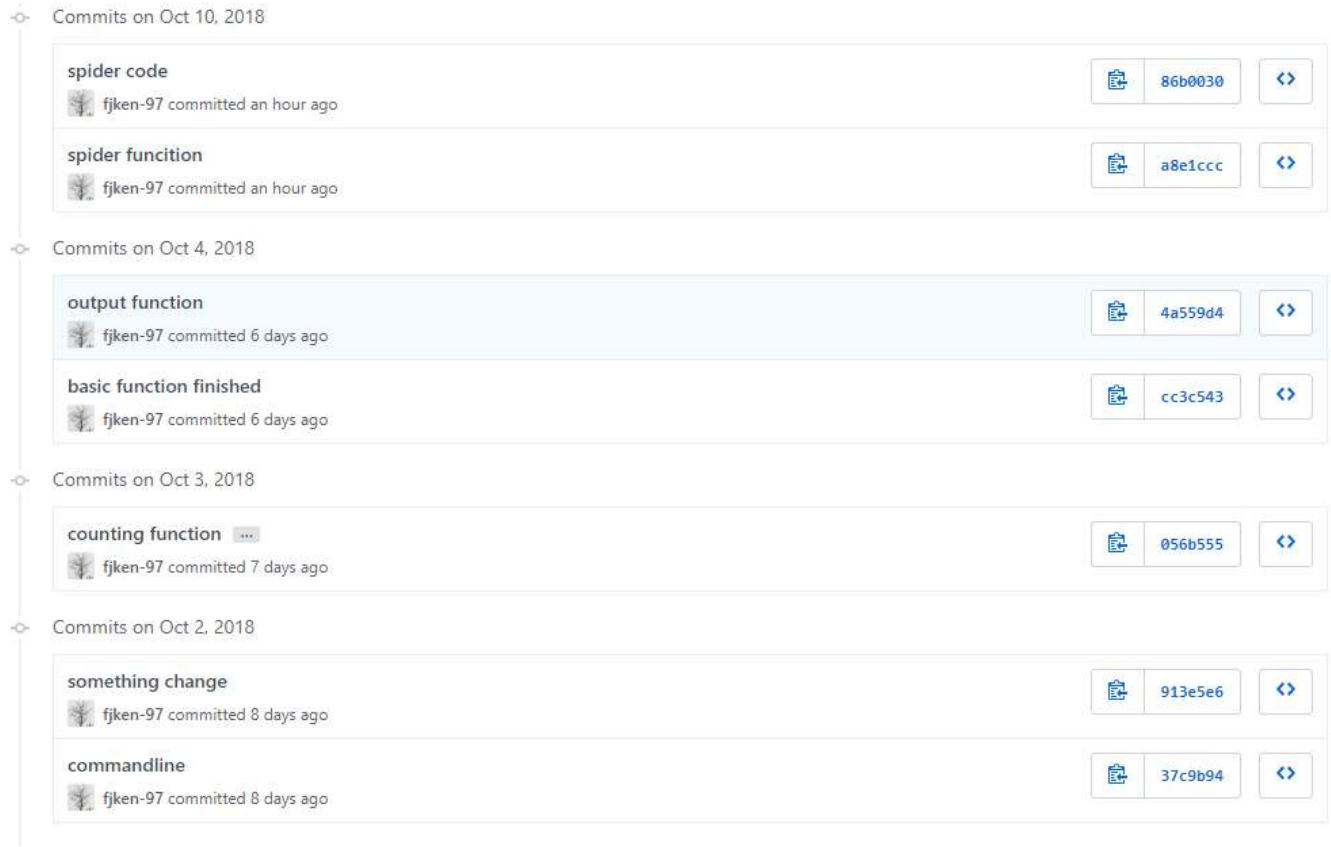
```
0
Title: E-mbodied Question
Abstract: E-mbodied Question We present a new AI task

1
Title: Embodied Question Answ-ering
Abstract: Embodied Question We present a new AI task
```

这里仅test 4可正常运行，因为之前的三个test均未通过命令行的异常处理，在输出err信息后程序均进入中断，这里贴上test 4的结果




```
<已终止> 返回 test4() [JUnit] D:\Program Files\Java\jdk-9.0.1\bin\javaw.exe (2018年10月1
Begin.....................................
characters: 133
words: 14
lines: 4
The number of output words is less than then limited(10)
<embodied question>: 11
<mbodied question>: 11
<answ ering>: 10
<question answ>: 10
End.....................................
```

Part 8.贴出Github的代码签入记录



Part 9.遇到的代码模块异常或结对困难及解决方法

问题描述

1. 命令行要求的限制问题：如何在第一时间处理掉非正常的操作
2. 两部分文本整合生成详细信息的.csv过程中，存在部分无法匹配的情形
3. 内部对于相关文本存在大量需要处理的情形

做过哪些尝试

1. 增设判断，抛出异常信息后直接exit()中断程序的运行
2. 增设文本的清洗统一化处理：对于title作于key的情形，统一进行 `.toLowerCase()` 操作，各项不合法符号化作分隔符，剩下的属于内容不匹配，即两处title仅部分内容相同的我姑且忽略(当然也可以进行模糊匹配，尽可能去配对映射)
3. 设置flag标记符作为判断依据，方便多种情况的适应性问题

是否解决

除未能实现gephi-tookit的神经网络生成外，其余均解决

有何收获

需要纠正自己的编码习惯，养成一个自定义deadline的编码原则，在代码的可读性上还需要多下功夫，总体来说，自己在接触新鲜的事物的过程中及时编码很辛苦但是内心很满足。

Part 10.评价你的队友

总的来说，我的队友在任务执行过程中相当配合，遇到不会的问题时及时的反馈与沟通，通过交流加上查找大量资料从而解决问题，关于爬虫jsoup为什么没能够爬取到全部的内容，我也是在和他一起沟通查找资料的过程中了解到：

原来在默认设置下，jsoup最大获取的响应长度正好时1M。所以这个时候需要设置connection.maxBodySize(0)来得到不限响应长度的数据。参考博客链接

但同时存在的问题是需要提高下集中式的独立编码能力，一项任务尽可能在一定时间内完成交付

Part 11.学习进度条

第N周	新增代码(行)	累计代码(行)	本周学习耗时(小时)	累计学习耗时(小时)	重要成长
5	1000+	N	36	45	学习java数据图形化处理，了解部分数据处理
6	1000+	N	18	63	学习《构建之法》，加深单元测试的应用，补缺补漏