COVID-19背景下的网络社会心态

1. 基本信息

> 本报告由小组三人共同完成

姓名	学号	邮箱	分工职责	
王文渊	191250140	191250140@smail.nju.edu.cn	数据筛选、LDA模型建模、建立心态词典、可视化、建立停用词表	组员
谢瀚杵	191250157	191250157@smail.nju.edu.cn	分词、建立停用词表、TF-IWF关键词提取、建立心态字典、可视化、LDA模型建模、	组员
郑伟鑫	191250206	191250206@smail.nju.edu.cn	数据获取、可视化、建立心态词典、心态分析	组长

GitHub仓库链接：https://github.com/zwx-zwx/AnalysisOFSentimentInCOVID-19

2. 摘要

为了研究新冠肺炎疫情下的大众心态，本小组使用Python作为主要研究工具，采用了爬虫来进行数据的爬取，分布拟合来筛选数据，LDA模型以及频数分布来心态聚类，TF-IWF算法提取关键词来描绘心态分布，并合理地进行了一些人工的数据筛选和归类，最终描绘出了疫情不同阶段下大众的心态分布和变化，并剖析了背后的可能原因。

3. 前言

3.1 研究背景

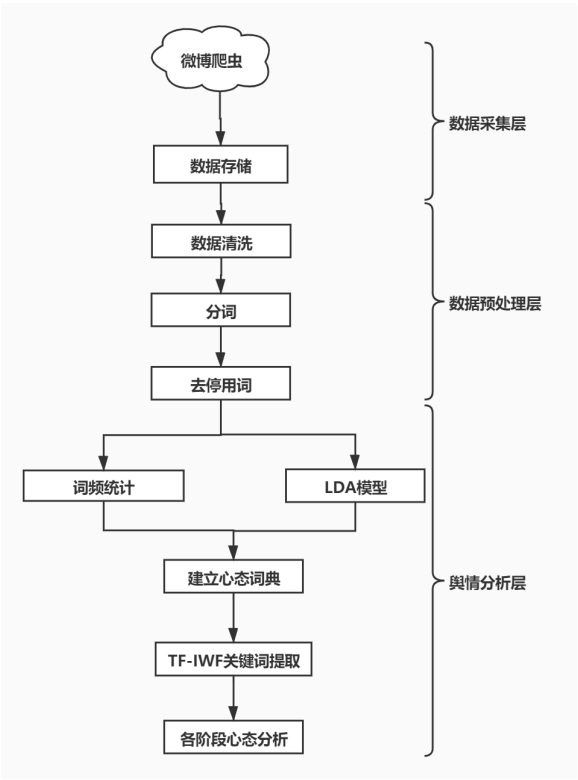
中国社会正处在深刻而快速的转型期，其中，在社会变迁层面，社会结构的快速分化，以“撕裂”的方式强化了社会团体、阶层
之间的张力，使得整体社会结构出现紧张，并投射在个体心理层面，进一步凸显出公众的社会认知、情绪、信念、意向、行动等对社会治理的重要影响。同时，随着互联网应用的不断普及，日益多元复杂的公众情绪，借助网络的力量传播和放大，对社会心态的塑形力量进一步增强，赋予了群体心理及集体行为的极化可能。当下，COVID-19正肆虐全球，对社会的生产生活造成了极大地影响。基于此，我们可以通过数据科学相关技术，准确并客观地了解这一时段社会心态的分布情况。

3.2 研究问题

在COVID-19背景下的网络社会心态的分布情况

4. 研究方法

4.1 研究框架



本小组研究框架如右图所示，主要分为三个部分，分别是数据采集层、数据预处理层和舆情分析层。其中，数据采集层主要使用python爬虫较笨获取微博主流媒体账号发布的微博的内容、发布时间等相关信息（4.2将描述数据集的详细信息），然后以json格式储存在本地。数据预处理层包含数据清洗、jieba分词、去停用词，将与疫情无关和与情绪无关的词语筛选掉。舆情分析层则尝试使用LDA模型，人工统计高频情绪词以将无关信息筛选，并同时建立心态词典。最后通过TF-IWF提取每条评论的关键词对每个阶段的网络社会心态进行分析。

4.2 数据集描述

数据来源于人民日报、光明日报以及新华网在微博平台的官方账号，包括2019年12月8日至2020年1月22日、2020年1月23日至2020年2月7日、2020年2月10日至2020年2月13日以及2020年3月10日至2020年6月中旬这四个时间段下，这三个账号发布的微博内容、发布时间、点赞数、评论数和每条微博下的热门评论。

数据集以json文件的形式保存在“DataSet”文件夹中，按照各个时间阶段分别存放在各自的json文件中。共包含约一万五千条微博以及十二万条评论信息。

人民日报，新华网以及光明日报均为官方新闻媒体，分别拥有粉丝量1亿、1亿以及2293万，作为数据源，庞大的粉丝基数一定程度上保障了分析结果的可靠性。

4.3 数据预处理

4.3.1 如何筛选疫情相关新闻

1、构建初步疫情相关字典

我们只需通过新闻标题正文中查看疫情相关词，即可筛选出相关数据。我们先选取2000条新闻数据作为一个小样本研究，先把所有标题正文分词合计统计如图

	0
病例	107
确诊	94
新增	68
无症状	55
检测	54
感染者	51
核酸	45
小区	36
疫情	35
视频	34
隔离	34
石家庄	30
河北	28
北京	27
微博	24
记者	22
工作	21
防控	21
中国	21
医学观察	20
阳性	20
金矿	20
密切接触	19
栖霞	19
累计	18
井下	18
报告	18
人员	18
新冠	18
输入	18
肺炎	17
境外	16
医院	16
发布会	16
出院	16
隆尧县	16

然后人工判断疫情相关词，构建一个相对合理的疫情相关词典，以选择疫情相关新闻。

原则有：1、较少的频数比如1,2可以忽略其中相关的词。与疫情相关但与其他新闻主题也相关的不能加入。

```
rela=['病','疫','医','确诊','新增','无症状','检测','感染','核酸','隔离','防控','阳性','阴性','密切接触',
      '新冠','肺炎','管控','重症','新型','高风险','卫健委','消毒','冠状','检测','抗体','口罩','防护'
    ,'双检','血清','防护服']
```

2、小样本检测合理性

我们还不能确定相关字典能不能帮我们选出需要的，筛去不要的。可能有如下两种筛选错误：

A：疫情相关的被筛掉了。

B：疫情无关的被加入了。

由原则易知，B错误更应该避免发生。

我们为了检测，人工打excel表，判断了100条数据的相关性，预估正确率应该在96%及以上才算达标。

	微博地址:	发布时间:	微博标题:	转发数	疫情相关 1 相关 0 无关
2	https://m.weib	Tue Jun 16 0	【北京：#严控高风险人员离京#，已出京的要及时通报当地】昨日，北京新冠肺炎疫情防控工作领导小组第	521	1
3	https://m.weib	Tue Jun 16 0	【#全球新冠肺炎累计确诊超800万#】据约翰斯·霍普金斯大学最新统计，截至北京时间6月16日6时33分，全	620	1
4	https://m.weib	Tue Jun 16 0	#炸酱面稳住#！#加油北京#！	43422	1
5	https://m.weib	Tue Jun 16 0	【#北斗三号最后一颗组网卫星发射任务推迟#】新华社消息，发射北斗三号最后一颗全球组网卫星的长征三	689	0
6	https://m.weib	Tue Jun 16 0	【第93届#奥斯卡颁奖典礼推迟#】当地时间15日，美国电影艺术与科学学院宣布，考虑到新冠疫情的影响，	267	1
7	https://m.weib	Tue Jun 16 0	【#美国新冠肺炎确诊超211万#】据美国约翰斯·霍普金斯大学疫情实时监测系统显示，截至美东时间6月15日	453	1
8	https://m.weib	Mon Jun 15	【#你好，明天#】“首都疫情形势十分严峻”，今天北京的新闻发布会如此强调。北京疫情卷土重来，确实让	2149	1
9	https://m.weib	Mon Jun 15	【扩散！#四川急寻确诊病例同航班密接者#】6月15日，四川省雅安市石棉县报告1例省外输入新冠肺炎确诊	2666	1
10	https://m.weib	Mon Jun 15	【夜读：这3个人生真相，越早明白越好】请记住：①人生不只一个方向，可以恣意生长、叱咤四方，也可	2331	0
11	https://m.weib	Mon Jun 15	【#人民微评#：失去的人生如何修复？】山东冠县#顶替他人上大学女子成绩低于分数线243分#，更显讽刺	1132	0

导入程序进行判断，输出正确率。

第一次：

前100条-90.00%（1条人工出错）即91.00%

B:4条由于病和医单个字加入的无关新闻

A:5条相关而为加入的中2条是与学生复课相关的，2条是与中风险地区通报相关的，1条中过段且较有效检测词只有“加油”“稳住”故可能考虑不加入，1条是公共卫生，可防可控。以漏加好于错加的原则，添加词。

'病','医'改为词组

可能不易加入：出院 检验 医疗机构 医院 采集 采样 患者 医生 医护人员

加入：病例 医学观察 病毒 流行病

```
rela=['病例','医学观察 ','病毒','流行病','疫','确诊','新增','无症状','检测','感染','核酸','隔离','防控','阳性','阴性','密切接触',
      '新冠','肺炎','管控','重症','轻症''新型','高风险','卫健','消毒','冠状','检测','抗体','口罩','防护'
    ,'双检','血清','防护服','中风险','公共卫生','复课','停课','居家']
```

改进后再检测99.00%：以下1条出错 #炸酱面稳住#！#加油北京#！

第二次：

用第二个人工打好1/0相关与否的新的100条新闻检测，正确率为94.00%

A：有两条是与疫情恢复相关的没有选中，可选词比如“封闭/恢复通车”，“恢复开放”

有一条是“疾控（中心”

Add疾控 恢复开放 恢复通车

B:2条人工打0，两条为爆炸事故，与医学相关，碰撞词分别为“管控”“防控”

Delete“管控”“防控”

other：一条其实较难判断是否疫情相关且与国外相关算为0/1都对

【太荒唐！#推特疯狂关闭17万个账号#】.....此次“虚假”信息运动中增加了宣传中国政府抗疫表现的...

改进后100.00%/99.00%（取决于【太荒唐！#推特疯狂关闭17万个账号#】）

复查第一次，为99.00%

第三次：

96.00%满足了达到96.00%的条件

A:2个与恢复相关，可选词：康复患者/患者/康复，复工复产

一个为“”传染病/世卫“”

Add传染病 复工 复产 康复？ 康复患者

以下这种主题与疫情无关但提及疫情的没有很好的方法，并评论可能与疫情心态相关，可以算加入即可以算1

0【特别的毕业季，送你一份#暖心毕业图鉴#】...这个毕业季因疫情而特....

改进后100.00%/99.00%

复查1,2：

99.00%

100.00%/99.00%

此疫情相关词典基本在此阶段达到了预期，可以再在检测小样本（100）满足大于95%正确性后用于其他阶段或其他数据来源。

4.3.2 如何区分重点与非重点新闻

1、原因

我们研究的是关于大众的疫情心态，数据来源是平台下的评论，如果不把非重点新闻去除，由于非重点即关注参与此人数较少，可能会使得极端言论（不能代表大众）作为数据，被误以为为大众心态。

2、研究方法

先取足够的小样本来确定区分方式，我们取了2000条新闻来研究此问题。

1.为什么以评论数来衡量：

可以得知，初始数据格式：#Index(['微博地址:', '发布时间:', '微博标题:', '点赞数:', '评论数', '转发数', '评论'])

其中可以作为判断指标的有'点赞数:', '评论数', '转发数'.

可以使用降维的方法综合一个指标，但三者相关度理应很强，故先相关系数来衡量其相似度程度。

```
# # c_l = df[['评论数','点赞数:']]
# # print(c_l.corr()) #相关系数矩阵即给出了任意两个变量之间的相关系数
```

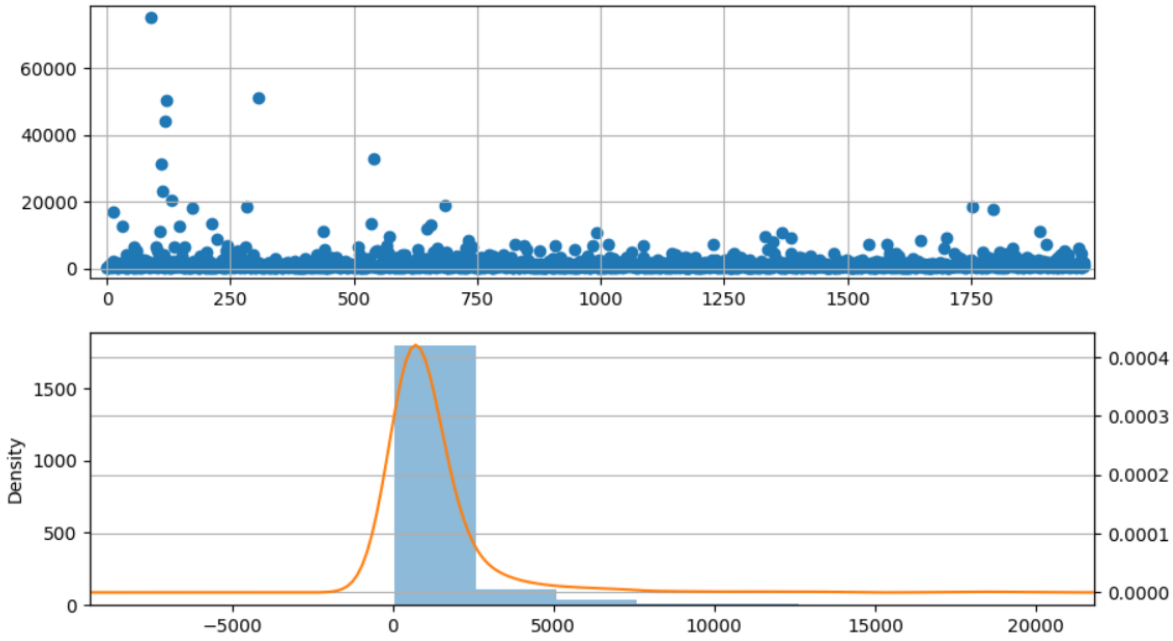
可得：

	评论数	点赞数
评论数	1.000000	0.758292
点赞数	0.758292	1.000000|

| r | > 0.8时称为高度相关，当 | r | < 0.3时称为低度相关，其它值为中度相关。0.76说明点赞与评论接高中度相关，可以考虑只用用评论数来筛选重点新闻，同理点赞数。

2.尝试拟合评论数分布：

首先我们表示出了评论数的散点图和经验分布图来初步判断和有一个直观的看法，并添加正态理想曲线。



可知，分布并不很正态，接下来考虑用更准确的判断方法。

首先采用KSTEST方法：程序参数分别是：待检验的数据，检验方法（分布），均值与标准差

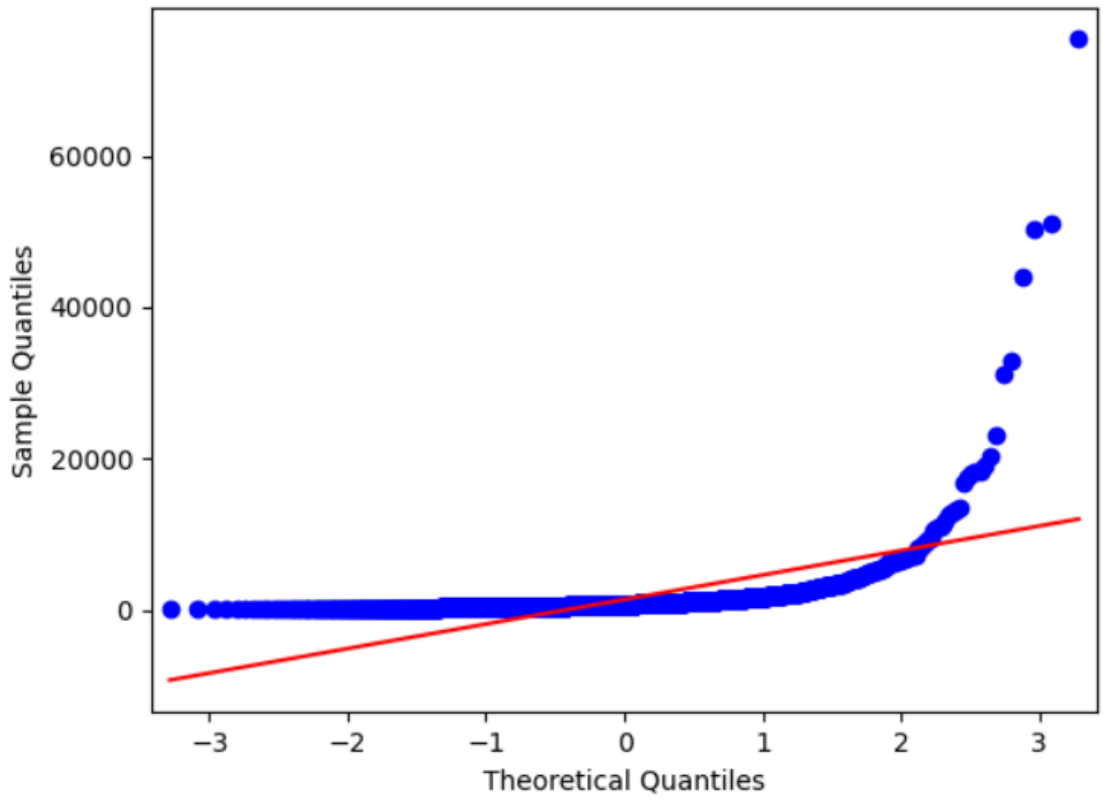
程序上会返回两个值：statistic → D值，pvalue → P值
p值大于0.05，为所输入分布
H0:样本符合 # H1:样本不符合
如何p>0.05接受H0 ,反之

其中D表示两个分布之间的最大距离，所以D越小，因为这两个分布的差距越小，分布也就越一致以下分布：t,uniform,poisson,f,lognorm,expon,chi2(卡方分布)其pvalue=0.0 p过于小，显著拒绝而对于norm

KstestResult(statistic=0.34287869783416797,pvalue=2.4449225111857205e-208) 2e-208 <0.05 也依旧拒绝H0

其次如果用偏度峰度检验正态其中已知正态性检验要求严格通常无法满足，如果峰度绝对值小于10并且偏度绝对值小于3，则说明数据虽然不是绝对正态，但基本可接受为正态分布。但计算可得12 偏度计算 218峰度计算，也难以符合

最后采用QQ图常用于直观查看数据是否正态分布。（其将实际数据累积比例作为X轴，将对应正态分布累积比例作为Y轴，作散点图。）如果散点图近似呈现为一条对角直线，则说明数据呈现出正态分布。反之则说明数据非正态。如图也显然不正态。



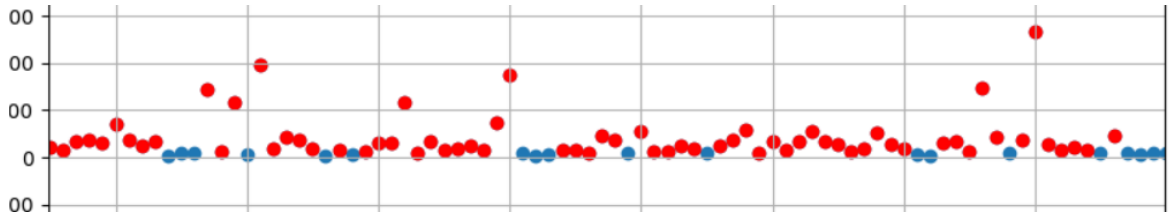
3、反思与确定阈值

2000条数据来源于同一个大v/新闻媒体的微博账号，一个账号有一定的粉丝数，大多数据有一个基础量可能是难以正态的原因。最后我们采用了定值可视化与百分比判断是否能做为阈值，首先500评论数以上个人觉得可以说是一个比较大的量了，有一定有效度代表大众，根据上直方图也可看出。

以下为一些选取阈值的参考数据和图的尝试。

均值1390.7439393939394

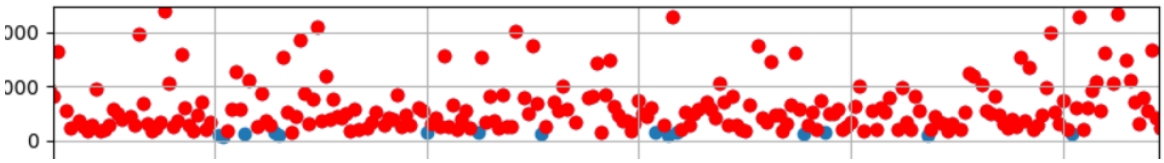
10000 3.46% 5000 10.15% 1000 53.29%



这个图是500以下去掉（蓝色）剩余有82.7%

400 88.74%

300作为分界线能保留94.13%



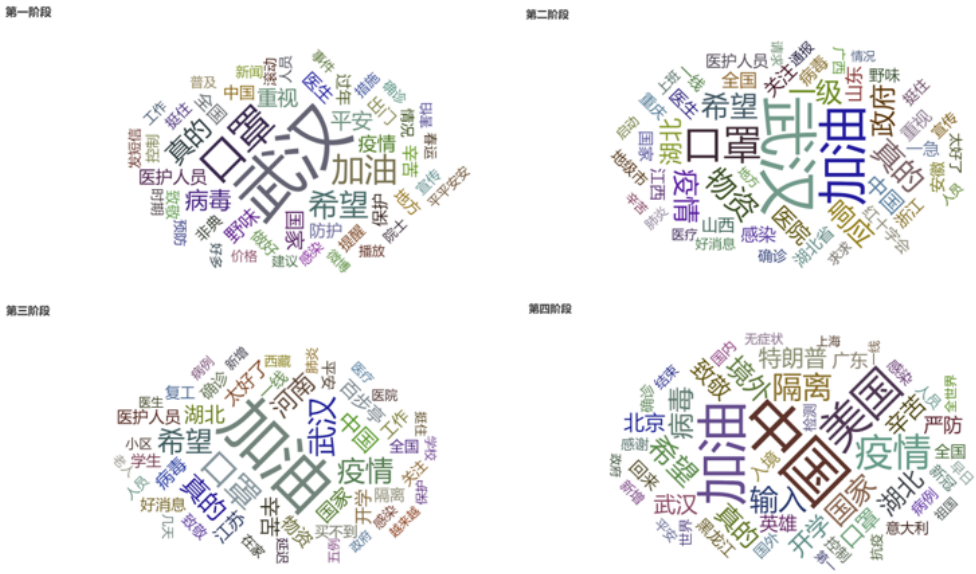
除了考虑代表大众的评论量不能太少，我们还需要考虑后续数据量的需求量。

综合两方面，最终我们觉得以400条为分界线选取作为重点最合理。

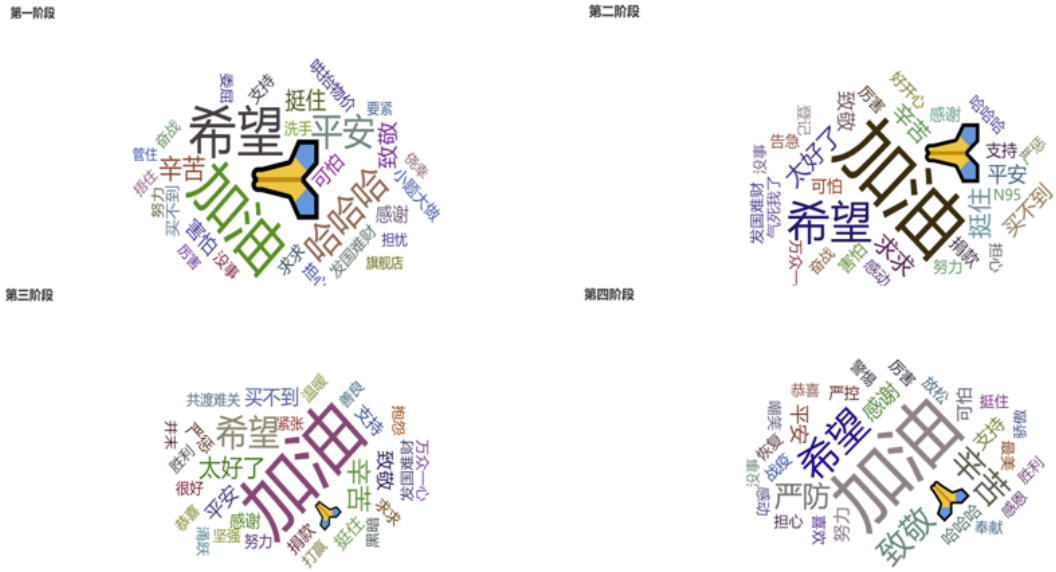
4.4 分词与筛选

得到筛选处理后的数据之后，我们首先要对文本进行分词。综合考虑代码可读性和分词的效率和准确性后，我们选择调用python的第三方库jieba库来进行分词。对于词语列表中表意无用的词我们构建了一个初始的停用词表。该停用词表是将github上包括哈工大停用词、四川大学机器智能实验室停用词、百度停用词(https://github.com/goto456/stopwords)以及最全中文停用词表(https://blog.csdn.net/shijiebei2009/article/details/39696571)进行整合得到的。在研究过程中我们发现数字和名词对于心态的描绘几乎没有作用；单个汉字可能会有一定的描绘作用，但由于大部分单字的保留会影响分词的准确性进而影响关键词的提取，因此也将其去除。另外，对于爬取的评论区数据，我们发现'🙏'表情出现的频率相当之高，将其全部去除会影响到心态分布的描绘，因此选择将其特判保留。

将文本初步分词之后，为了更进一步精确地筛选出描绘心态的词语，以及初步建立心态词典，我们统计词语出现的频数并绘制出四个阶段的高频词词云图如下：



由于jieba库词典和词性判断的局限性，导致词频统计中仍然产生了不少噪声。因此我们选择人工筛查的方式，设定频数阈值为10，对于出现频数高于10的词语进行筛选，初步分为三类：表示积极心态的、表示消极心态的、对心态描绘无用的。将四个阶段人工筛选的词进行整合去重，得到的无用词语将其加入停用词表中整合去重，得到最终版本的停用词表。之后再重复上述分词和筛选的过程，统计频数并绘制四个阶段的词云图如下，可以看出筛选的效果还是较为显著的。



4.5 LDA文本主题模型

1、目的和模型介绍

LDA由Blei, David M.、Ng, Andrew Y.、Jordan于2003年提出，是一种主题模型，它可以将文档集中每篇文档的主题以概率分布的形式给出，从而通过分析一些文档抽取出它们的主题（分布）出来后，便可以根据主题（分布）进行主题聚类或文本分类。同时，它是一种典型的词袋模型，即一篇文档是由一组词构成，词与词之间没有先后顺序的关系。此外，一篇文档可以包含多个主题，文档中每一个词都由其中的一个主题生成。

我们的想法是，每条评论的主题与每个词相关，去除非心态表现疫情无关词，主题词将由心态相关词组成，同种心态的评论将由于用到同样的心态词被聚类到同一组，这种无监督的聚类方式可以有效的分出不同的心态。

2、具体操作

我们选择LDA模型

先是每阶段通过对每一条有效评论，分词，通过已经构成的疫情心态表现不相关词的停用词去除高频不相关词，然后通过LDA模型把所有评论分为n个主题，导出主题词构成比例和主题分布，主题由几个主题词组成一个分布，选取合适的主题个数便于给每一个主题打上一个心态相关的标签。

心态列表可选项：['事不关己','困惑质疑','痛苦悲伤' ,'愤愤不平','焦虑恐慌','担忧关切','美好期盼','感动赞美','轻松愉悦','坚定信任']

例子，第一阶段，我们选取了几种不同的ntopic取值发现7种时主题最容易分类成不同心态，并且我们持续不断改进停用词表，使得心态疫情相关词为主题词。

	A	B
1	焦虑恐慌	0.515*"求求" + 0.458*"小题大做" + 0.193*"买不到" + 0.003*"可怕" + 0.003*"希望" + 0.003*"加油" + 0.003*"致敬" + 0.003*"平安" + 0.003*"辛苦" + 0.003*"挺住"
2	美好期盼	0.979*"平安" + 0.006*"希望" + 0.002*"加油" + 0.002*"致敬" + 0.002*"恐慌" + 0.002*"求求" + 0.002*"辛苦" + 0.001*"挺住" + 0.001*"感谢" + 0.001*"可怕"
3	担忧关切	0.991*"加油" + 0.002*"希望" + 0.001*"辛苦" + 0.001*"可怕" + 0.001*"恐慌" + 0.001*"小题大做" + 0.001*"挺住" + 0.001*"平安" + 0.001*"致敬" + 0.001*"求求"
4	感动赞美	0.975*"辛苦" + 0.004*"加油" + 0.003*"挺住" + 0.003*"平安" + 0.002*"致敬" + 0.002*"希望" + 0.002*"恐慌" + 0.002*"可怕" + 0.002*"感谢" + 0.002*"买不到"
5	美好期盼	0.776*"希望" + 0.117*"感谢" + 0.102*"恐慌" + 0.001*"辛苦" + 0.001*"加油" + 0.001*"平安" + 0.001*"致敬" + 0.001*"挺住" + 0.001*"可怕" + 0.001*"小题大做"
6	焦虑恐慌	0.961*"可怕" + 0.004*"平安" + 0.004*"小题大做" + 0.004*"致敬" + 0.004*"买不到" + 0.004*"希望" + 0.004*"求求" + 0.004*"辛苦" + 0.004*"加油" + 0.004*"挺住"
7	坚定信任	0.442*"致敬" + 0.356*"挺住" + 0.003*"买不到" + 0.001*"加油" + 0.001*"求求" + 0.001*"辛苦" + 0.001*"感谢" + 0.001*"希望" + 0.001*"恐慌" + 0.001*"平安"

具体打标签策略是比如第一组“求求”，“小题大做”“买不到”都指向了焦虑恐慌且三者占比在99%，可是确定这类评论主题是如此。而比如5，“希望”是美好期盼，“感谢”，“恐慌”指向其他心态，但希望占比有75%以上，并且考虑另外两个词可能是，一、有多种心态，二、是由于组成句子的必要成分而非表示心态了，故此主题下作为美好期盼的心态。

	topic
0	1781
4	184
2	135
6	100
1	77
3	64
5	31

阶段一，主题数为7，具体主题分布

其他主题数选择比如过高10左右可能出现如下全是占比10%左右构成的杂主题，并且主题分布也会差异过大，比如只有3个在80以上。

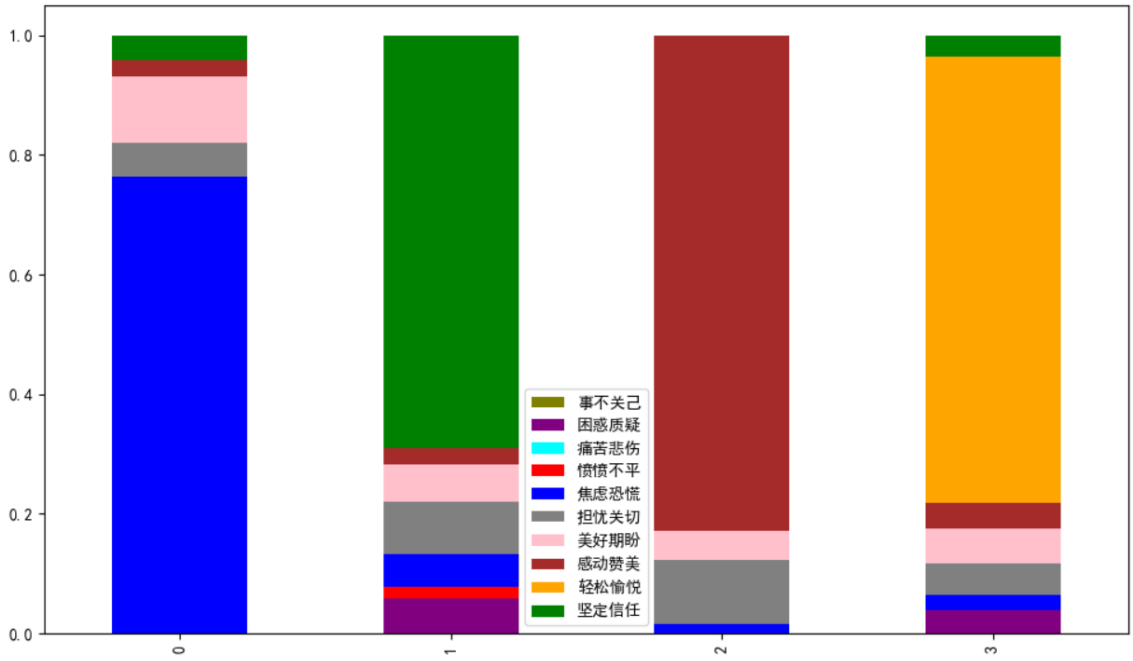
0.099*"希望" + 0.084*"小题大做" + 0.084*"平安" + 0.084*"可怕" + 0.084*"致敬" + 0.082*"加油" + 0.082*"感谢" + 0.082*"买不到" + 0.081*"辛苦" + 0.080*"挺住"（阶段一，主题数为10产生的问题）

另外第三阶段只有几天，心态个数也较少，而每一主题也会出现如下的2,4条，属于同一心态的情况。

1	感动赞美	0.668*"辛苦" + 0.322*"致敬" + 0.003*"平安" + 0.003*"加油" + 0.003*"希望" + 0.003*"买不到"
2	担忧关切	0.946*"平安" + 0.021*"希望" + 0.009*"致敬" + 0.009*"加油" + 0.008*"辛苦" + 0.008*"买不到"
3	美好期盼	0.988*"希望" + 0.003*"加油" + 0.002*"平安" + 0.002*"买不到" + 0.002*"致敬" + 0.002*"辛苦"
4	担忧关切	0.995*"加油" + 0.001*"希望" + 0.001*"买不到" + 0.001*"平安" + 0.001*"辛苦" + 0.001*"致敬"
5	焦虑恐慌	0.954*"买不到" + 0.009*"辛苦" + 0.009*"加油" + 0.009*"希望" + 0.009*"平安" + 0.009*"致敬"

值得注意的是第四阶段，出乎意料的有一些负面的，像焦虑恐慌或怀疑，经过我们的思考，和前几步的数据了解，我们认为是对国外疫情产生的，由于也与疫情心态相关，我们并未处理。

然后统计每个主题的个数，进行可视化。



依次为四个阶段，每阶段心态设为总数1。

可以看到四个阶段主流心态分别为焦虑恐慌，坚定信任，感动赞美，轻松愉悦，很符合我们的常识认可。并且担忧关切，美好期盼作为持续出现的心态。美好期盼又在第一阶段最多。

我们还可以检查主流心态的构成主题词,可以看出最多的主题都有着不同的心态词，并且打标签的心态比较合理。

Stage1焦虑恐慌：前图

Stage2坚定信任：（0.433*"努力" + 0.216*"致敬" + 0.209*"紧张" + 0.047*"加油" + 0.039*"感谢" + 0.004*"太好了" + 0.002*"买不到" + 0.002*"早日" + 0.002*"发国难财" + 0.002*"希望"）

Stage3感动赞美：（0.668*"辛苦" + 0.322*"致敬" + 0.003*"平安" + 0.003*"加油" + 0.003*"希望" + 0.003*"买不到"）

Stage4轻松愉悦：（A：0.473*"严防" + 0.203*"平安" + 0.165*"努力" + 0.073*"恭喜" + 0.046*"好不容易" + 0.023*"掉以轻心" + 0.004*"希望" + 0.001*"结束" + 0.001*"快点" + 0.001*"加油"

B：0.487*"终于" + 0.241*"严控" + 0.193*"幸福" + 0.010*"掉以轻心" + 0.009*"希望" + 0.005*"感谢" + 0.004*"致敬" + 0.004*"结束" + 0.003*"加油" + 0.003*"严惩"）

而比如针对愤愤不平，只在第二阶段出现，我们认为是疫情stage1民众先是焦虑恐慌，是一种盲目了解的状态，之后才由于新闻媒体导向，和吹哨人死亡的事把情绪导向了愤怒，并且查询评论可以知道，此阶段危及全国接近每个人身边，而口罩等资源管理可能不够完善，导致人们的愤怒不满。同理困惑质疑，而困惑质疑出现第四阶段的原因也提及了，是因为美国以及国外的原因。

但是四个阶段的心态占比波动有些太大了，从4.4的频数统计结果来看，人们的主流心态不应该发生如此剧烈的波动，因此我们合理推测LDA模型在我们的研究背景下有着一定的不足之处从而导致结果有失真实性。

3、反思不足与可以改进之处

（1）一条评论对于LDA模型来说过少了，缺少足够的文本特征，LDA主题模型比较适合处理长文本。由此带来一些问题，比如给一个未知主题打上标签时可能出现误差，不好确认心态。

（2）一条评论中出现的心态可能有多种，比如有代表多个不同心态的词，而在LDA模型中都统一归类了，那些占比较小的心态数据就损失了，应该都算入心态统计中。

（3）通过我们对评论的查阅，可以发现有些发表者的心态隐藏于语意之中，而一些符号，语气词，疑问词也是判断心态的条件，这些评论都是机器通过LDA拟合无法准确归类的。

（4）我们发现正面的词往往多集中于哪几种像是“加油”“感谢”，而负面的词五花八门，如果以统计评论相关性聚成类不容易被发掘，可能可以通过人工打标签的方式来改善。

4.6 建立心态词典

对通过cntopic库进行LDA拟合的结果的不足之处，我们决定再另外采取人工的方式建立心态词典。LDA在不同主题数量下拟合效果相对较好的主题数量大致为7-10个。因此我们结合高频词人工分类出9种心态，分别为：“事不关己”“困惑质疑”“痛苦悲伤”“愤懑不平”“焦虑恐慌”“担忧关切”“美好期盼”“感动赞美”“轻松愉悦”“坚定信任”。LDA模型分出的主题也用到了上述9个主题进行人工打标签，通过LDA模型聚类得到的结果也在一定程度上验证了上述分类的可靠性。然后对与4.4中人工筛选出的高频描绘心态的词语进行人工归类，得到心态词典。采取人工方式建立的心态词典涵盖文本库词汇范围的广度有一定的损失，并且稍微有失严谨性，但在精准度上有着计算机无法比拟的优势，因而我们选择人工聚类的心态词典来进行接下的步骤。

4.7 TF-IWF特征词提取与心态分布计算

在分词完成之后

目前，关键词自动提取技术基于统计的方法主要有有TF，TF-IDF算法等。这些算法简单快捷，且精度不俗。TF提取文本高频词作为候选关键词，TF-IDF采用文本逆频率IDF对TF值加权取权值大的作为关键词，Turney对此方法作了实验证明。计算公式如下：其中，TF部分分子ni,j表示词语ti在文本j中的频数，分母表示文本中所有词语的频数和；IDF部分D 表示语料库d的文档数，|{j:ti∈dj}|表示本语料库 d 中包含文档j中词语ti的文档数。

$$TF-IDF_{i,j} \rightarrow TF_{i,j} \times IDF_i = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{|D|}{\left|\{j : t_i \in d_j\}\right|}$$

但IDF的简单结构并不能有效地反映单词的重要程度和特征词的分布情况，使其无法很好地完成对权值调整的功能，所以TF-IDF算法的精度并不是很高，尤其是当文本集已经分类的情况下。对于微博爬取的评论区数据而言，我们在数据清洗的过程中实际上就已经对文本集进行了分类，对于不同种的心态进行聚类的过程实际上也进行了分类。因此我们选择采用改进的TF-IWF算法。IWF部分的含义是对语料库词语总数与待查文本中该词出现在语料库中的次数比求对数。这种加权方法降低了语料库中同类型文本对词语权重的影响，更加精确地表达了这个词语在待查文档中的重要程度。在TF-IWF算法中，将IDF转换为IWF，将词频比作为文本候选关键词去噪音的权值，有效的抑制了与测试文本同类语料库对所提取关键词权重的影响，修正了 TF-IDF算法的偏差。计算公式如下：其中TF部分分子ni,j表示词语ti在文本j中出现的次数，分母表示文本j中所有词语频词和，IWF部分分子表示语料库中所有词语频数之和，nti表示词语 ti在语料库中出现的总频数。

$$TF-IWF_{i,j} \rightarrow TF_{i,j} \times IWF_i = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{\sum_{i=1}^{m} nt_i}{nt_i}$$

我们起初的代码实现是针对一篇微博的评论区中所有评论计算权重提取关键词（默认每篇微博权重相同），从而将这一篇微博映射到一种心态。这种方案在实现与运算上更为简便，也能反映大众心态。但是同一评论区下低频出现的心态在这种算法实现下所占的权重就会变得很低，进而使得其不同阶段的变化不明显。我们认为这些心态在描绘大众心态中的作用不可忽视，所以为了适当凸显大众心态的全面性，即适当提高小众心态的权重，我们将每一条评论作为一个文本，针对每一条评论计算词语权重并映射到心态（默认每条评论权重相同）。采用这种方式实际上更进一步地减弱了噪声的影响，以及实现更精准的心态映射和权重计算。例如下图为第一阶段TF-IWF算法权重计算的部分结果，有提取不出关键词的以及关键词无法映射到心态的评论，这些数据在计算心态分布的过程中会直接忽略。对于关键词的权值而言，TF-IWF算法兼顾了单个评论与评论的总体，因而计算出单个评论所反映的心态在整个阶段中的权值的结果，在便于分析的前提下和合理的基础上具有较好的真实性。

```
[]
[('超负荷', 1.6983175834918467), ('运行', 1.610271953964006)]
[('再进一步', 0.6793270333967387), ('排挤', 0.6793270333967387), ('降到', 0.6793270333967387), ('最小', 0.6793270333967387),
[('吓住', 3.3966351669836934)]
[('买不到', 2.250507131305455)]
[('发国难财', 2.3174539209360687)]
[('奋战', 2.3966351669836934)]
[('要为', 3.3966351669836934)]
[('挺住', 1.8781212271058059)]
```

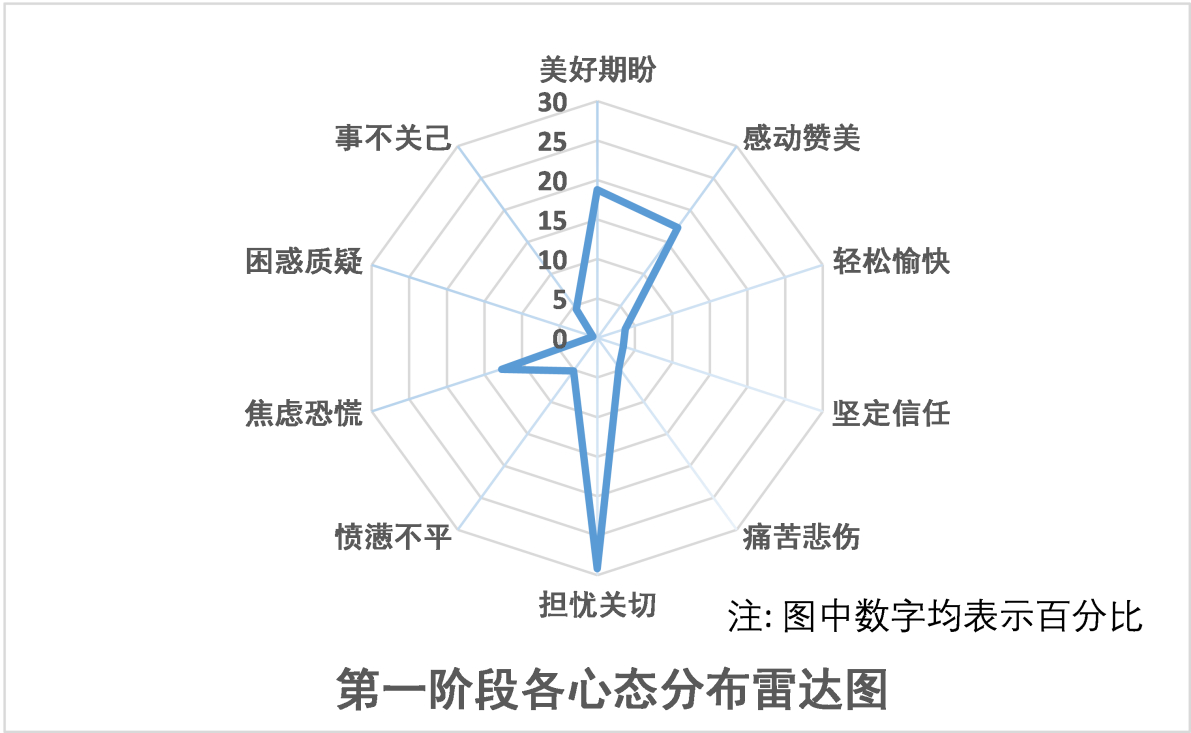
最后，我们将关键词映射到心态，关键词的权值即为心态的权值，将每条评论的权重相加，即可得到整个阶段的心态分布。

5. 案例分析

5.1 总述

以本次获得的微博大v数据为基础以及通过上述的研究方法，我们得到了在四个阶段的心态分布情况。接下来，将通过可视化的方式，进行各阶段心态分布的分析：通过雷达图我们将定性地分析各个阶段的心态分布情况，通过折线图我们将得到单独的每一心态在不同阶段间的变化情况。

5.2 第一阶段--不重视与无奈扩散阶段（2019.12.08-2020.01.22）



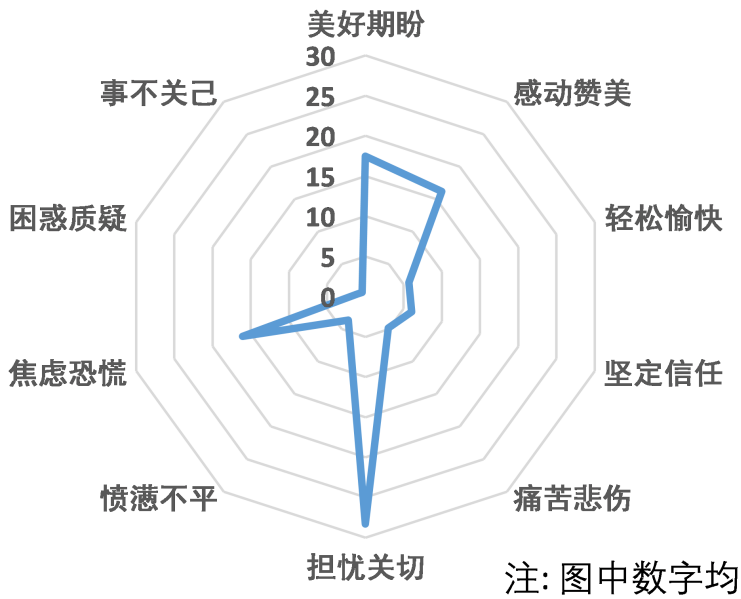
第一阶段各心态分布雷达图

从图中可知，第一阶段“担忧关切”的心态占比最高，之后依次是“美好期盼”、“感动赞美”、“焦虑恐慌”

民众总体呈现出对疫情的“担忧关切”，仍然存在“美好期盼”、“感动赞美”的积极情绪。

值得关注的是在第一阶段的负面心态中，“焦虑恐慌”的心态相对突出，这与第一阶段的事件相对应，也即“不重视与无奈扩散阶段”

5.3 第二阶段--资源缺乏阶段（2020.01.23-2020.02.07）



注: 图中数字均表示百分比

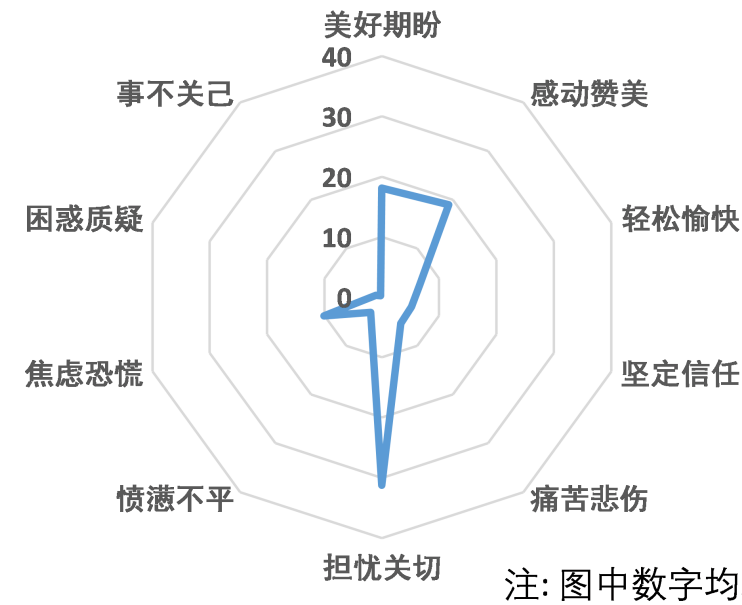
第二阶段各心态分布雷达图

从图中可知，第二阶段“担忧关切”的心态占比最高，之后依次是“美好期盼”、“感动赞美”、“焦虑恐慌”。

民众总体呈现出对疫情的“担忧关切”，仍然存在“美好期盼”、“感动赞美”的积极情绪。

值得关注的是在第二阶段的负面心态中，“焦虑恐慌”的心态更为突出，在资源缺乏阶段，民众呈现出更加“焦虑紧张”的心态

5.4 第三阶段--严格统一管控和物资匹配阶段（2020.02.10-2020.02.13）



注: 图中数字均表示百分比

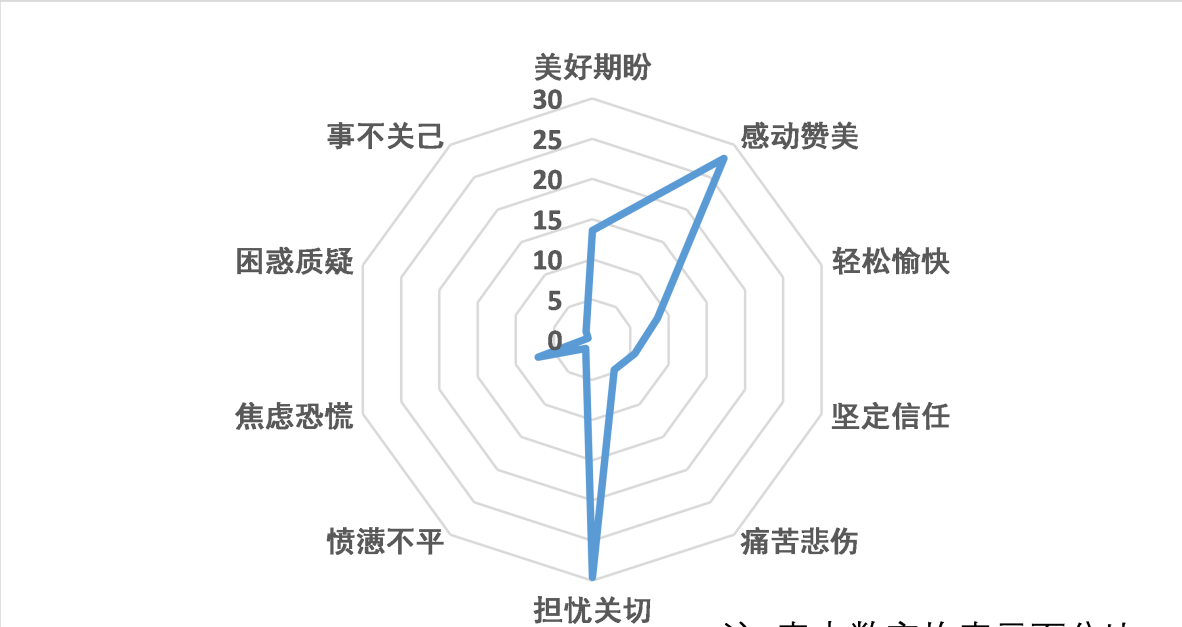
第三阶段各心态分布雷达图

从图中可知，第三阶段“担忧关切”的心态占比最高，之后依次是“美好期盼”、“感动赞美”、“焦虑恐慌”。

民众总体呈现出对疫情的“担忧关切”，仍然存在“美好期盼”、“感动赞美”的积极情绪。

值得关注的是在第三阶段的负面心态中，“焦虑恐慌”的心态仍然占据一定比重但有所减少，一定程度上体现严格统一管控对网络社会焦虑心态的安抚作用

5.5 第四阶段--有序复工阶段（2020.03.10-2020.06中旬）



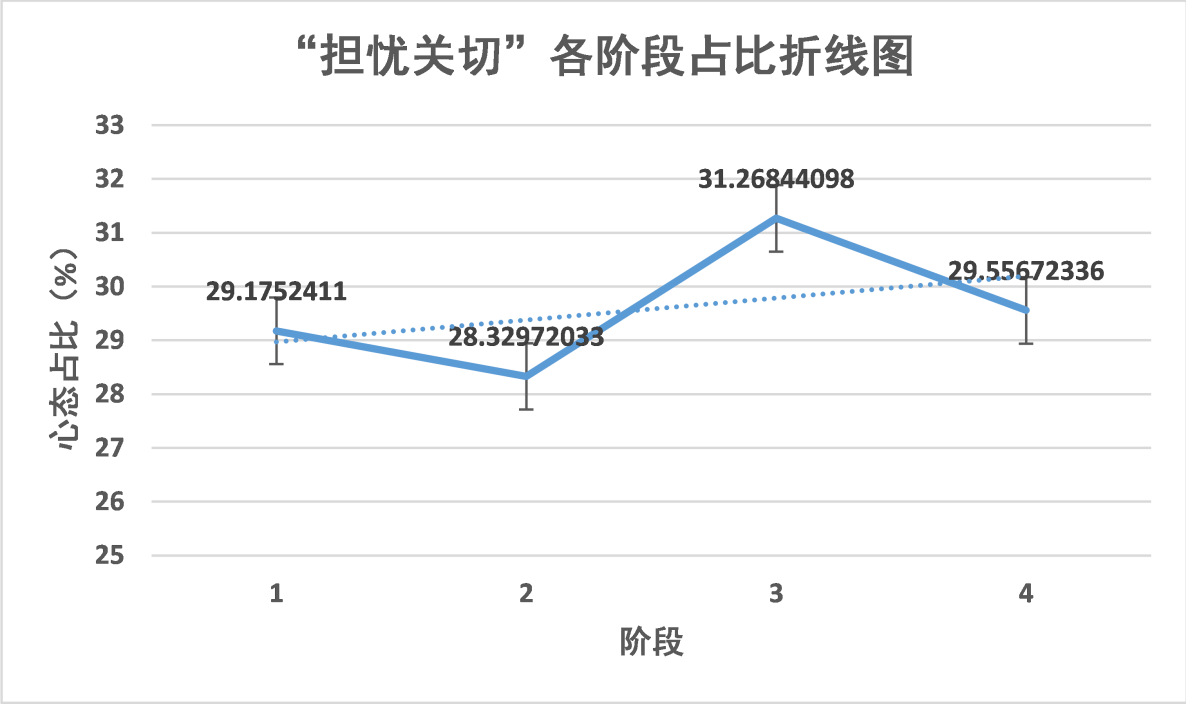
第四阶段各心态分布雷达图

从图中可知，第四阶段“担忧关切”的心态占比最高，之后依次是“感动赞美”、“美好期盼”、、“轻松愉快”。

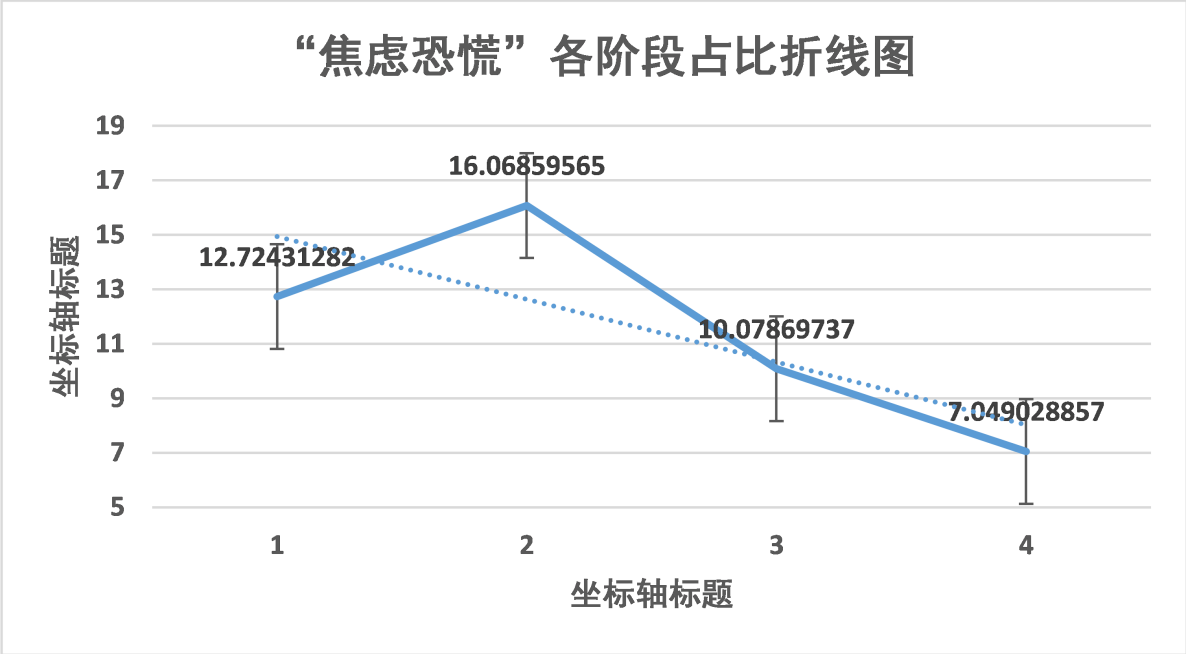
该阶段除“担忧关切”的心态以外，积极的心态占据主要比重，如“感动赞美”、“美好期待”。

值得关注的是在第四阶段的负面心态中，“焦虑恐慌”的心态仍然占据一定比重但有所减少，一定程度上体现严格统一管控对网络社会焦虑心态的安抚作用。

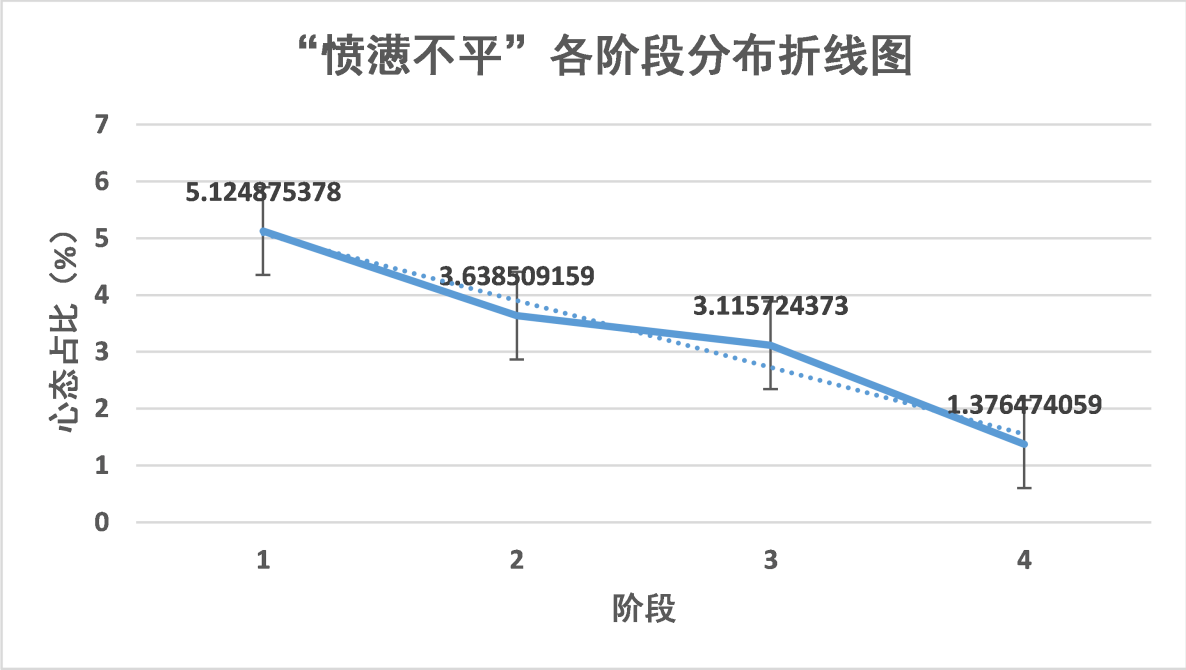
5.6 各个阶段心态变化情况



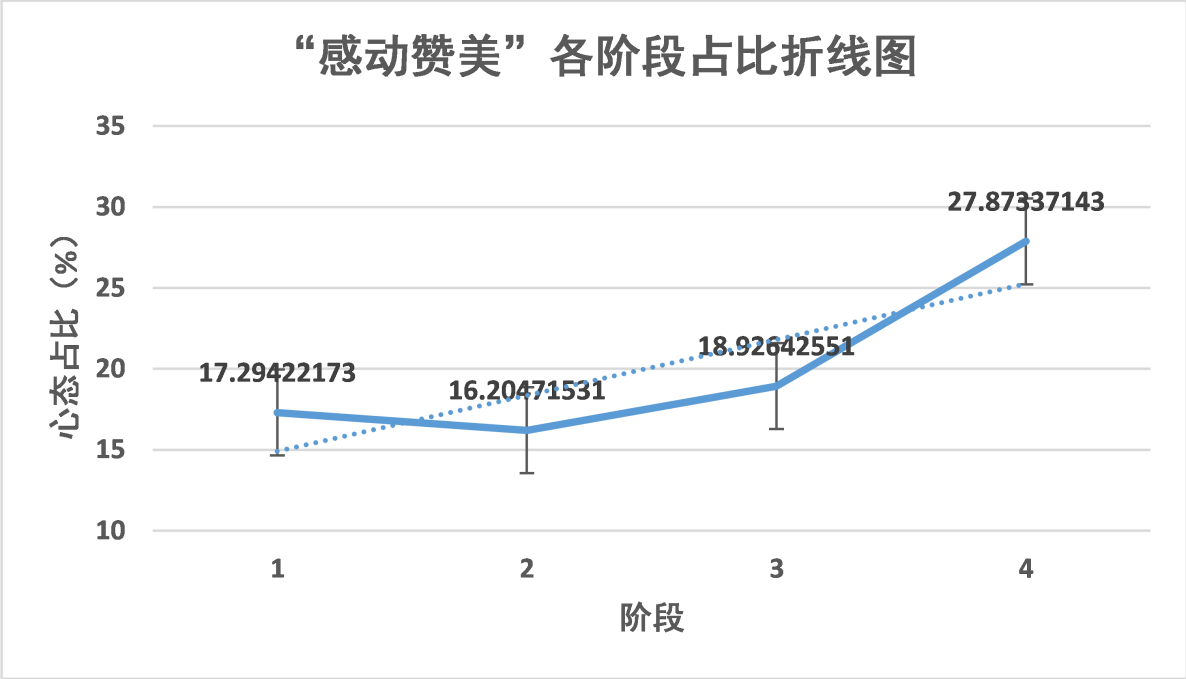
从整体来看，“担忧关切”的心态在整个疫情阶段都占据着较打比重，并呈现一定的变化，在阶段二出现与直觉相悖的下降，第三阶段民众的“担忧关切”情绪有所提高。总的来说，整个疫情阶段“担忧关切”的心态变化不大，在网络社会中占据比重较大且变化波动不大。



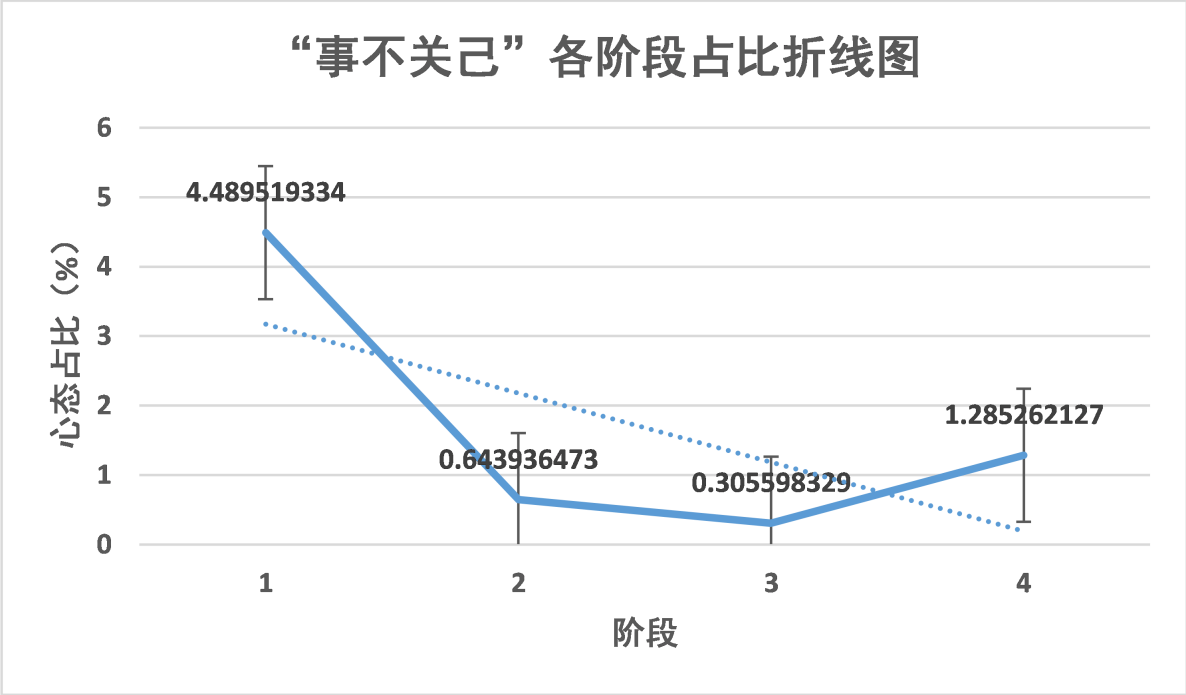
“焦虑恐慌”的心态第一阶段约占12%，在资源缺乏阶段增长至约16%，在这一阶段，武汉宣布“封城”，“吹哨人”李文亮去世等新闻加剧了网络社会的焦虑恐慌心态。到后期疫情进入严格统一管理和物资配给阶段、有序复工阶段，这种情绪逐渐减少。



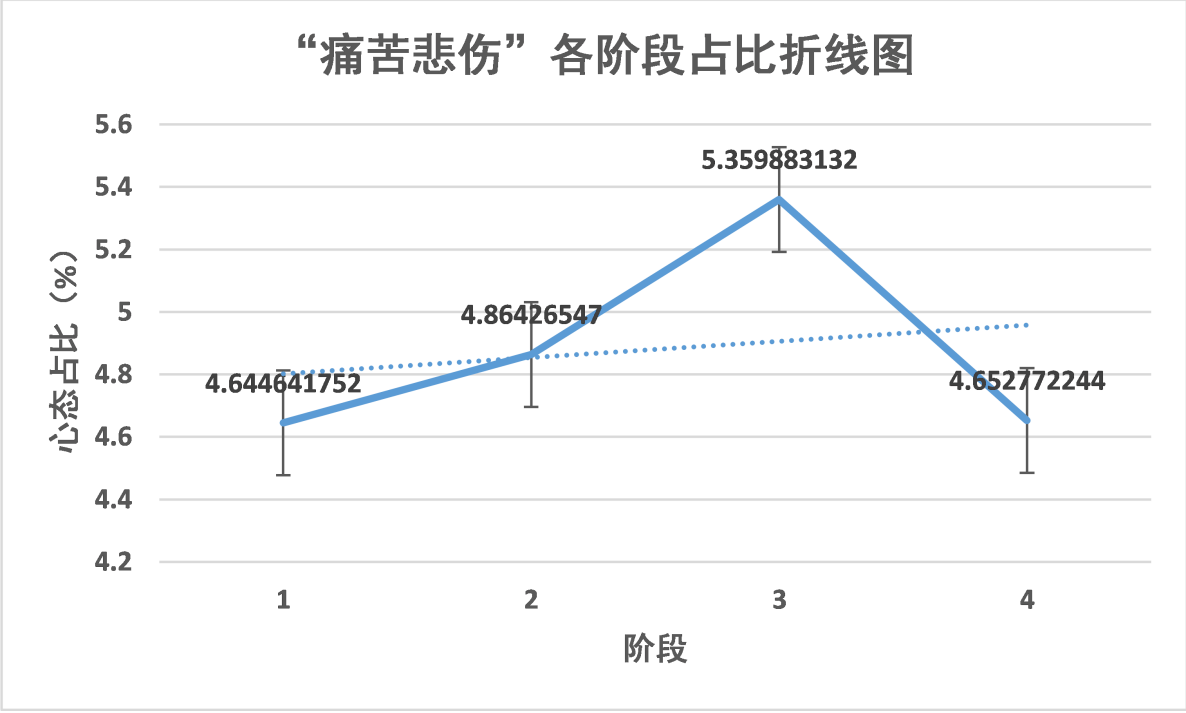
根据该图，可以看出“愤懑不平”的心态随着疫情的发展，逐渐降低。这与疫情初期湖北相关官员的系列行为以及之后湖北相关领导人事变动的新闻相对契合，也能看出政府强有力的防疫政策卓有成效，民众怨言稳步减少。



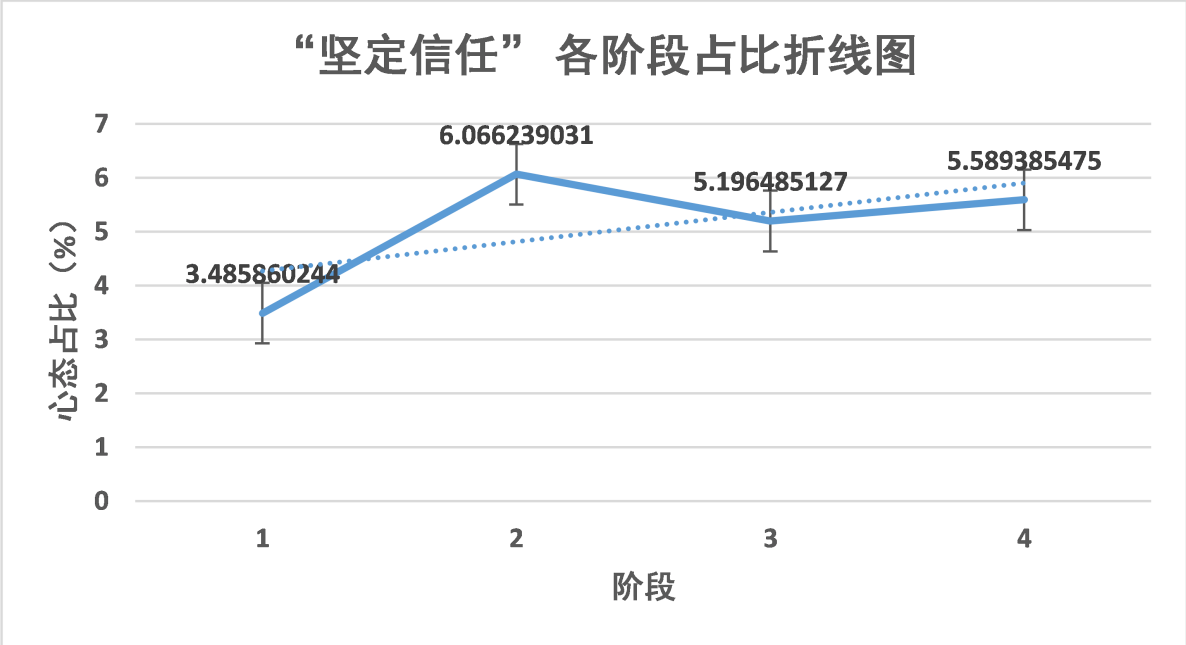
“感动赞美”的心态占比呈现出增长的趋势，这与中国在疫情中的及时有效应对相对应。在评论内容上多为：网民对政府的及时处理、“为人民服务”的理念和贯彻执行、医护人员无私无畏行为的赞美颂扬。在逐步复工的第四阶段，回顾整个上半年的抗疫过程，感动与赞美的心态达到了顶峰。



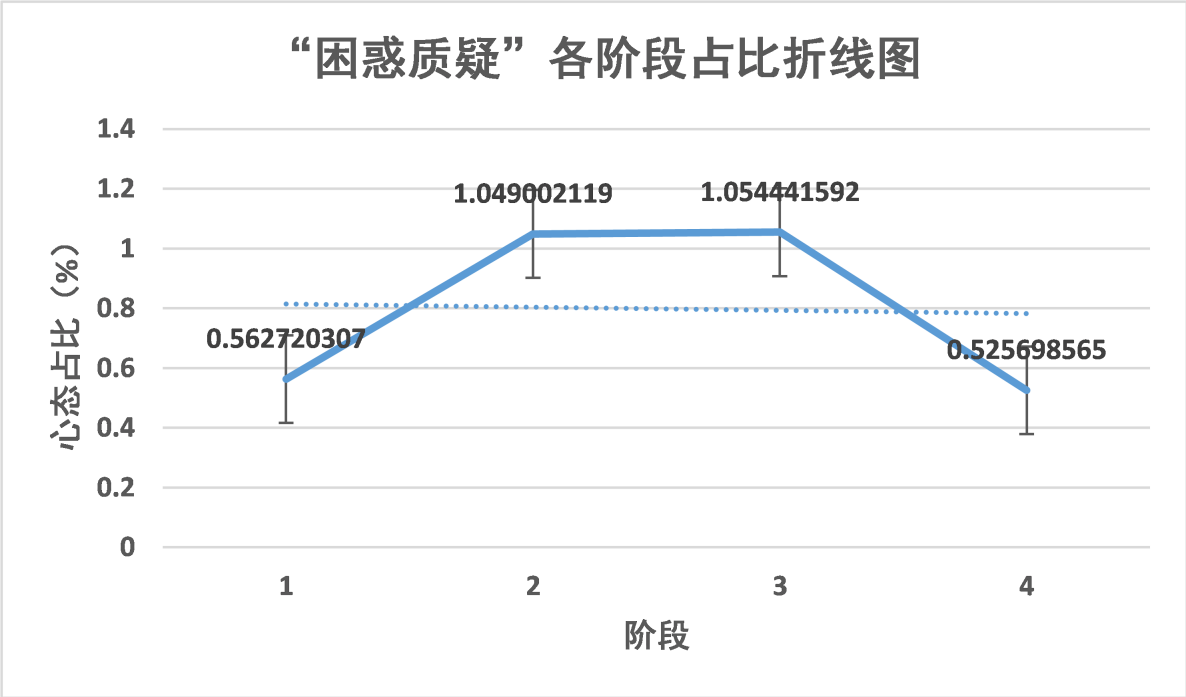
“事不关己”的心态变化也极为有趣。从阶段1约4.5%下降到约0-1%。可以见得疫情发展初期，部分人群对疫情并不在意，认为新冠肺炎是子虚乌有。而在之后的一两个阶段疫情愈演愈烈，这样的心态也随之减少。而在复工的抗疫收尾阶段这样的心态又死灰复燃，也足以警醒我们对待疫情时刻保持警惕



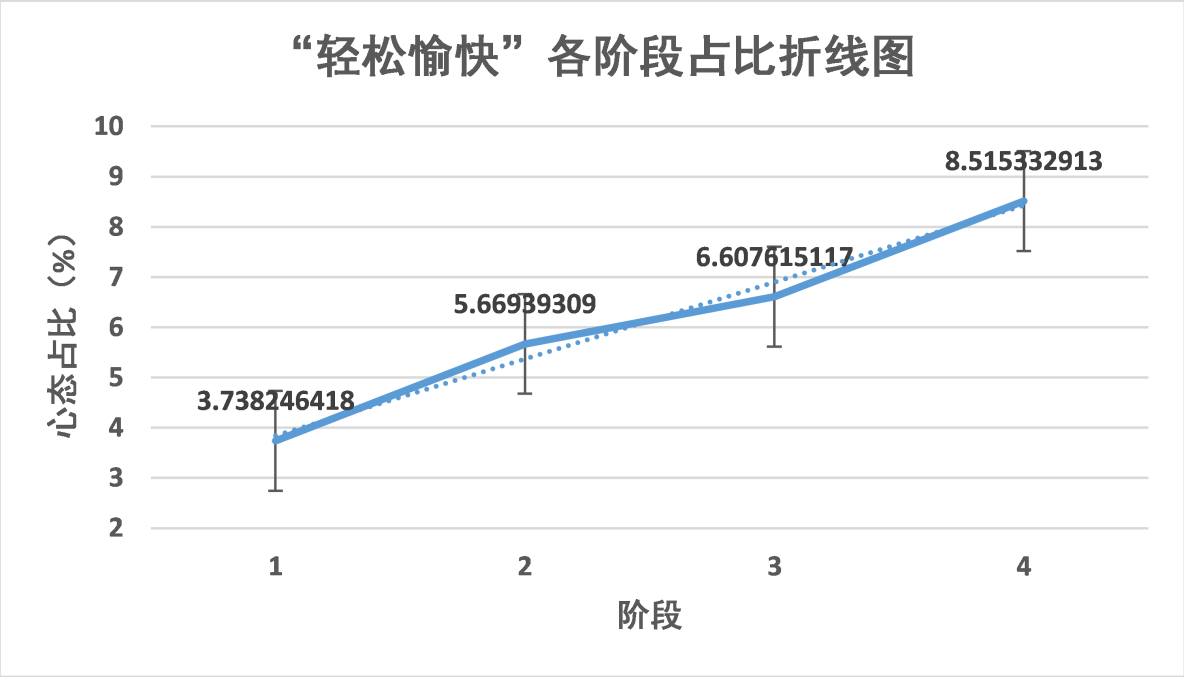
“痛苦悲伤”的心态在阶段3增加的峰值并在阶段4降低到较低水平，这一心态与疫情期间的停工停产，复工复产的新闻相关联，因为对于储蓄能力不足的家庭而言，能否恢复生产关系着整个家庭的命脉。



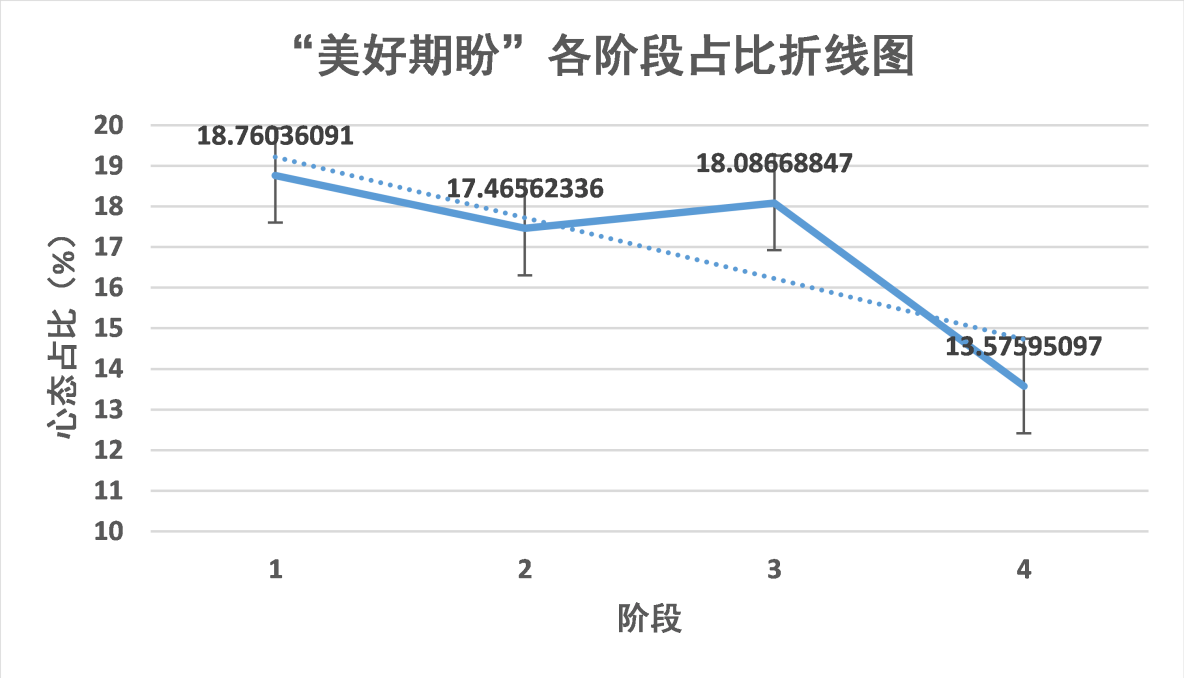
“坚定信任”的心态占比从阶段1的约3.49%增长到约6%，并在后三个阶段保持相对稳定，可见这与疫情发展初期民众对于新冠不了解与不重视和之后中国政府和人民团结一致抗疫相关。



“困惑质疑”的心态虽有变化但占据的比重较小。中国政府的疫情管控能力大家有目共睹，对于决策的质疑自然比较少，但也无法避免地会出现。



“轻松愉快”的占比随疫情发展呈现增长态势，这与我国抗疫不断取得的成果息息相关。



“美好期盼”这一心态随着时间变化呈现出减少的趋势，其中一部分可能性是疫情对经济的打击，导致民众在网络社会上呈现出“美好期盼”这一心态减少的情况。另一方面，抗击疫情已经取得了阶段性胜利，此时感动赞美和轻松愉快的心态自然会占据主流，对未来的美好期盼的占比也会随之降低。

6. 相应的分析代码解释

6.1 weibo 爬虫

此部分对应weiboSpider.py

本次使用的是基于PyQuery、Request、json库实现的weibo爬虫，使用伪装头、代理IP技术以提高爬虫访问服务器的频率，通过访问随机页数后爬虫休眠随机秒的方式实现反爬，防止服务器限制账号访问权限。

使用接口为m.weibo.cn(微博手机端)，

6.1.1 基本实现逻辑：

1） 先通过微博账号的uid访问服务器得到账号的基本信息（如微博昵称）以及其微博主页的containerid

```
# 获取微博大V账号的用户基本信息，如：微博昵称、微博地址、微博头像、关注人数、粉丝数、性别、等级等
def get_userInfo(id):
    url='https://m.weibo.cn/api/container/getIndex?type=uid&value='+id
    data=use_proxy(url,proxy_addr)
    content=json.loads(data).get('data')
    profile_image_url=content.get('userInfo').get('profile_image_url')
    description=content.get('userInfo').get('description')
    profile_url=content.get('userInfo').get('profile_url')
    verified=content.get('userInfo').get('verified')
    guanzhu=content.get('userInfo').get('follow_count')
    name=content.get('userInfo').get('screen_name')
    fensi=content.get('userInfo').get('followers_count')
    gender=content.get('userInfo').get('gender')
    urank=content.get('userInfo').get('urank')
    print("微博昵称："+name+"\n"+"微博主页地址："+profile_url+"\n"+"微博头像地址："+profile_image_url+"\n"+"是否认证："+str(verified)+"\n"+"微博说明："+description+"\n"+"关注人数："+str(guanzhu)+"\n"+"粉丝数："+str(fensi)+"\n"+"性别："+gender+"\n"+"微博等级："+str(urank)+"\n")
    return name

```

2） 再通过访问https://m.weibo.cn/api/container/getIndex?type=uid&value=[uid]&containerid=[containerid]&page=[欲访问页数]得到页面信息，使用json库得到微博标题、发布时间、点赞数、评论数以及该微博下的评论id，

```
url='https://m.weibo.cn/api/container/getIndex?type=uid&value='+id
        weibo_url='https://m.weibo.cn/api/container/getIndex?type=uid&value='+id+'&containerid='+get_containerid(url)+'&page='+str(i)
        print(weibo_url)
        try:
            data=use_proxy(weibo_url,proxy_addr)
            content=json.loads(data).get('data')
            cards=content.get('cards')
            if len(cards)>0 :
                for j in range(len(cards)):
                    print("-----正在爬取第"+str(i)+"页，第"+str(j)+"条微博------")
                    card_type=cards[j].get('card_type')
                    if(card_type==9):
                        mblog=cards[j].get('mblog')
                        attitudes_count=mblog.get('attitudes_count')
                        comments_count=mblog.get('comments_count')
                        created_at=mblog.get('created_at') # 创建时间
                        reposts_count=mblog.get('reposts_count')
                        scheme=cards[j].get('scheme')
                        ids=mblog.get('id')
                        # text=pq(mblog.get('text')).text().replace('\n','') # 文本内容
                        begin = getDate(created_at)
                        stage = getStage(created_at)

```

```
                        # 非需要的时间段，舍弃
                        if stage == 0:
                            print("stage0，前往下一页")
                            break
                        elif stage==3:
                            print("stage3 + 1")
                        elif stage==2:
                            print("stage2 + 1")
                        elif stage==1:
                            print("stage1 + 1")

                        dic = {
                            '微博地址:': scheme,
                            '发布时间:': created_at,
                            '微博标题:': get_detail(ids),
                            # '微博内容' : text,
                            '点赞数:': attitudes_count,
                            '评论数': comments_count,
                            '转发数': reposts_count,
                            '评论':get_comment(ids)
                        }
                        List.append(dic)

```

3） 访问https://m.weibo.cn/comments/hotflow?id=[评论id]&mid=[评论id]&max_id_type=0通过pyQuery解析得到评论

```
def get_comment(id):
    comments = []
    try:
        url='https://m.weibo.cn/comments/hotflow?id='+id+'&mid='+id+'&max_id_type=0'
        data = use_proxy(url, proxy_addr)
        content = json.loads(data).get('data')
        datas = content.get('data')
        for i in range(0,11):
            if(i<len(datas)):

 comments.append(pq(datas[i].get('text')).text().replace('<span class=\"Text\".*?>.*?</span>', ''))
    except Exception as e:
        print(e)
        pass
    return comments

```

4） 将得到的信息用json库写到相应文件下

```
 with open(file + str(stage) + ".json", 'w',encoding='utf-8') as fh:
                            fh.write(json.dumps(List, indent=2, ensure_ascii=False))

```

6.1.2. 实现难点

通过模拟客户端访问服务器得到需要的信息，这一过程实现较为容易。但实际操作中遇到了如下几个难点：

1) 不携带cookie访问服务器并不能访问到所需时间段的页面，请求返回为空。解决方案是携带cookie访问服务器，方可得到相应信息。

2) 直接执行爬虫程序会导致单个账号访问过于频繁，服务器反爬机制将限制账号的访问。解决方案有很多，本小组的解决方案是增加程序使爬虫在若干次访问后停止若干秒，尽可能地模拟人的访问行为。

```
delay = 3.8 + 2.2 * random.random()
time.sleep(delay)

```

3） 同时，使用代理IP避免IP被封而无法访问服务器的情况

```
    # 设置代理IP
proxy_addr=["196.168.0.1",
            "175.43.58.49:9999",
            "118.24.128.46:1080",
            "49.89.103.19:9999",
            "118.24.127.144:1080",
            "58.253.157.193:9999",
            "183.166.21.31:9999",
            "220.249.149.253:9999",
            "183.166.96.227:9999",
            "175.43.56.3:9999",
            "27.43.187.232:9999",
            "222.189.191.207:9999",
            "223.242.224.155:9999",
            "58.253.159.52:9999",
            "49.89.103.196:9999",
            "118.24.172.149:1080",
            "60.167.134.37:8888",
            "183.166.111.22:9999",
            "175.42.68.77:9999",
            "183.166.103.150:9999",
            "222.189.190.159:9999"]

```

6.2 分词

停用词表用.txt文件逐行存储，打开后逐行读入，注意.txt文件的换行符和制表符的格式问题，可能会导致乱码产生。然后先调用psg.cut方法（import jieba.posseg as psg）分词并生成词性，将名词去掉，之后在依次去掉停用词、数字和长度为1的词（除了🙏），返回的是一个列表

```
def split(parameter):
    stopword = []
    with open('data/stoplist.txt', 'r', encoding='GBK') as s:
        for line in s.readlines():
            l = line.strip()
            if l == '\\n':  # 换行符
```

```
                l = '\n'
            if l == '\\u3000':  # 制表符
                l = '\u3000'
            stopword.append(l)
    n = [x.word for x in psg.cut(parameter) if not x.flag.startswith('n')]
 # 去名词
    x = np.array(n)
    y = np.array(stopword)
    z = x[~np.in1d(x, y)]  # 去停用词
    k = [i for i in z if not i.isnumeric()]  # 去数字
    m = [i for i in k if len(i) > 1 or i == ' ']  # 去长度为1的词(除了 )
    return np.array(m).tolist()

```

6.3 LDA模型

documents为嵌套了词语列表的列表，n_topics为人为设定。

```
import os
from cntopic import Topic
topic = Topic(cwd=os.getcwd()) #构建词典dictionary
topic.create_dictionary(documents=documents) #根据documents数据，构建词典空间
topic.create_corpus(documents=documents) #构建语料(将文本转为文档-词频矩阵)
topic.train_lda_model(n_topics=10) #指定n_topics，构建LDA话题模型
```

6.4 绘制词云

词云绘制调用了pyecharts库，针对每个阶段统计词频，得到键值为词值为频数的字典取前30个降序排列生成列表，然后直接调用pyrcharts库中的方法即可，在main函数中调用该函数生成html文件。

```
def WordCloud_base() -> WordCloud:
    all = ""
    for i in range(0, len(t)):
        all = all + t[i]['评论']  # 整个文本库
    all_words = split(all)
    all_wordcount = Counter(all_words)  # 文本库中的词频统计
    words = sorted(all_wordcount.items(), key=lambda d: d[1], reverse=True)
[:30]
    c = (
        WordCloud()
            .add("", words, word_size_range=[20, 100], shape='circle')
            .set_global_opts(title_opts=opts.TitleOpts(title='第一阶段'))  #
四个阶段分别绘制
    )
    return c

```

6.5 TF-IWF 关键词提取

以下为截取的在心态分布计算中实现TF-IWF关键词提取的部分代码。先对文本库进行词频统计，然后初始化TF、IWF、TF_IWF三个字典，遍历每条评论，将评论分词后的列表中的词作为键值，分别计算TF、IWF以及TF_IWF的值并写入字典。

```
# 文本库统计
    all = ""
```

```
    for i in range(0, len(t)):
        for j in t[i]['评论'].split(','):
            all = all + j  # 整个文本库
    all_words = split(all)
    num_of_all = len(all_words)  # 文本库中词的总频数
    all_wordcount = Counter(all_words)  # 文本库中的词频统计
# 每一条评论为单位计算心态权重
    for i in range(0, len(t)):
        for j in t[i]['评论'].split(','):
            TF = {}
            IWF = {}
            TF_IWF = {}
            wordcount = Counter(split(j))  # 当前文本中的词频统计
            total = 0  # 当前文本中词的总频数
            for value in wordcount.values():  # 值相加得总频数
                total = total + value
            for key in wordcount.keys():
                tf = wordcount[key] / total  # 计算tf,即该词在文本中出现的频率
                TF[key] = tf
                num_of_word = all_wordcount[key]  # 该词在文本库中出现的总频数
                iwf = math.log10(num_of_all / (num_of_word + 1))  # 计算iwf
                IWF[key] = iwf
                TF_IWF[key] = tf * iwf

```

7. 附录

> 为需要补充到研究报告中帮助读者理解的数据、图表等,
>
> 如：数据集描述

7.1 关于数据集

json文件中的所有元素都如下例所示

```
{
    "微博地址:": "https://m.weibo.cn/status/IpWzyhtiP?mblogid=IpWzyhtiP&luicode=10000011&lfid=1076032028810631",
    "发布时间:": "Fri Jan 17 21:24:41 +0800 2020",
    "微博标题:": "【#开车进故宫奔驰车主回应#：朋友借车 说故宫邀请她参加活动】1月17日，网友@露小宝LL 发布一组“赶着周一闭馆 去故宫撒欢儿”的照片引发网友热议。涉事奔驰车车主告诉@紧急呼叫 ，涉事女子是其朋友，当天涉事女子是受故宫邀请参加活动的。#开车进故宫事件# 网页链接",
    "点赞数:": 26897,
    "评论数": 1424,
    "转发数": 560,
    "评论": [
      "溥仪：给我退票！",
      "就冲这句话，她差不多就进去了。 网页链接",
      "@故宫博物院 看见没，人家说是你们邀请的，解释解释啊",
      "车不是借的吧～ 网页链接",
      "@故宫博物院 的工作人员都是骑着自行车上班的，就是为了避免破坏故宫。",
      "记者还主动提供答案？",
      "会不会没有惩罚，没有后续，然后大家就淡忘了",
      "故宫博物院自罚三杯，好套路！",
      "当事人都说自己靠打点进去的！你特么京A8是不是也慌了？",
```

```
      "这个记者挺会采访啊！都会抢答了！",
      "邀请的啊！懂了，谁邀请的呢，参加什么活动呢，谁打点的呢"
    ]
  }
```