



**[作业2]B站弹幕定制化词典**

**课程名称 文本信息挖掘概论**

**学生学院 计算机学院**

**专业班 18级软件工程2班**

**学 号 3118005002**

**学生姓名 胡梓泽**

**指导老师 杨易杨**

**日期：2021年5月26日**

B站弹幕定制化词典

目录

[1.初始化词典 2](#_Toc75700986)

[2..爬虫处理步骤分析 2](#_Toc75700987)

[2.1 语料获取流程： 2](#_Toc75700988)

[2.2 数据处理目标： 2](#_Toc75700989)

[2.3 制定个人词典流程 2](#_Toc75700990)

[3. 基于个性化词典进行分词 3](#_Toc75700991)

[3.1 语料介绍 3](#_Toc75700992)

[3.2 详细处理过程 5](#_Toc75700993)

[3.2.1 Word2vec得到词向量 5](#_Toc75700994)

[3.2.2 Annoy对向量进行索引 5](#_Toc75700995)

[4. 对新词进行评估 5](#_Toc75700996)

[4.1空间索引中的K近邻 5](#_Toc75700997)

[4.2 筛选满足条件的新词 6](#_Toc75700998)

[4.2.1 TF-IDF算法步骤 6](#_Toc75700999)

[4.2.2 代码及结果分析 7](#_Toc75701000)

[5. 案例分析，例句展示 8](#_Toc75701001)

# 1.初始化词典

搜集B站汇总的一些热词，汇总成最初始的词典。

# 2..爬虫处理步骤分析

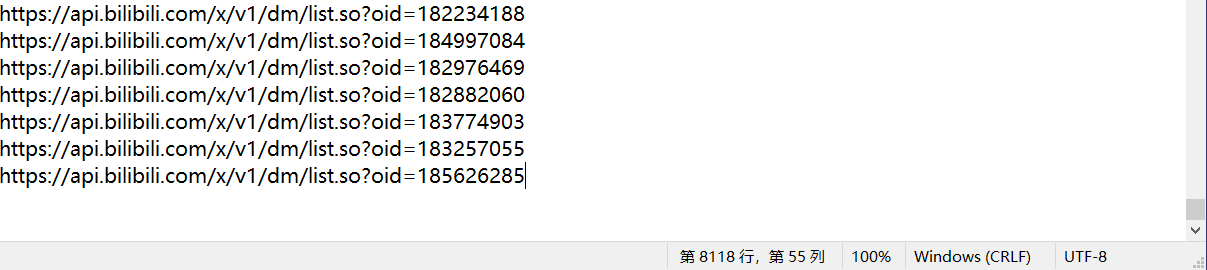
## 2.1 语料获取流程：

①哔哩哔哩弹幕网-bilibili(<https://www.bilibili.com/>)

② webdriver.Chrome访问特定视频，动态获取html信息

③提取html信息中的弹幕网页，汇总成danmuUrl.txt

如下图，10万个视频中共有8118个有弹幕信息。



## 2.2 数据处理目标：

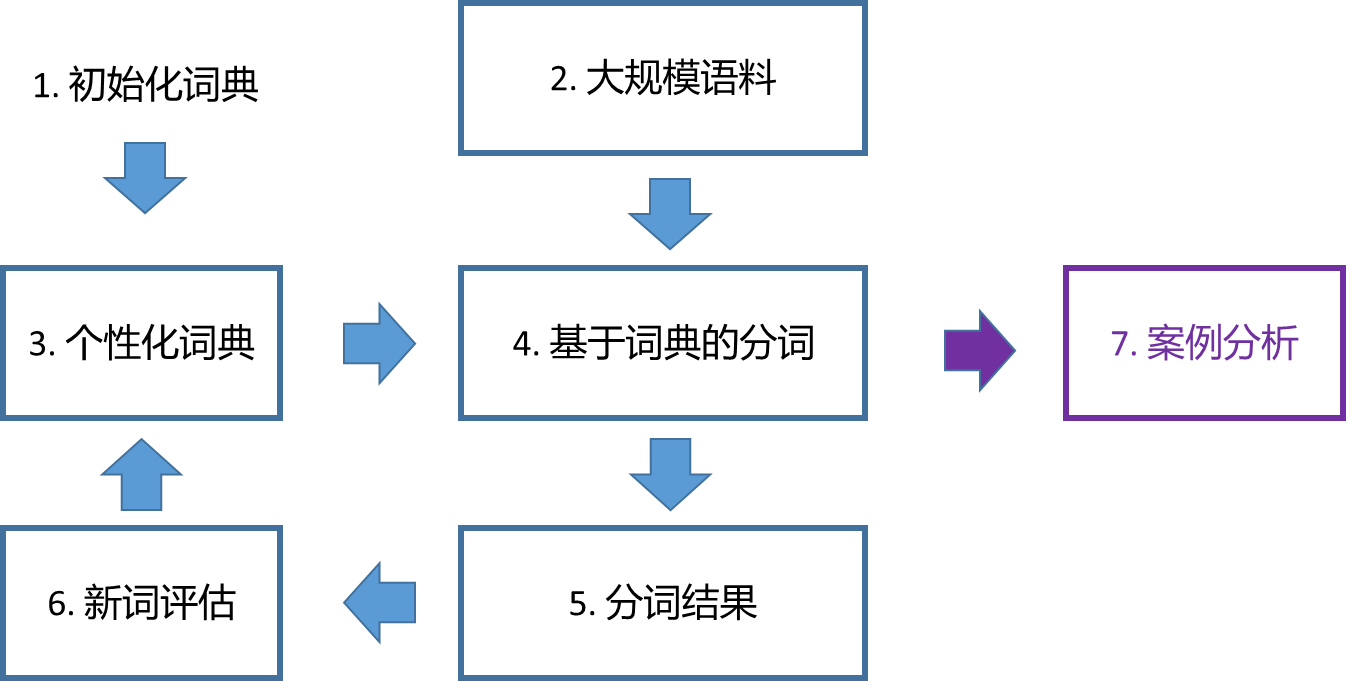
①视频号从BV1FV411d7u7开始的10万个视频中未删除且有效的弹幕

②对弹幕进行分词（去除停用词）

③得到词频、tf、idf等数据作进一步筛选

## 2.3 制定个人词典流程

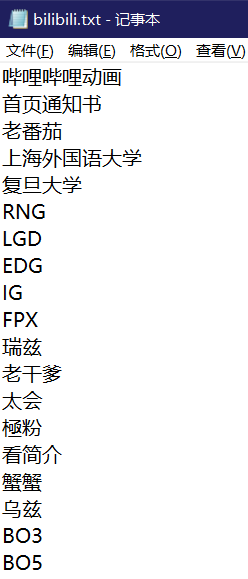
流程图如下：



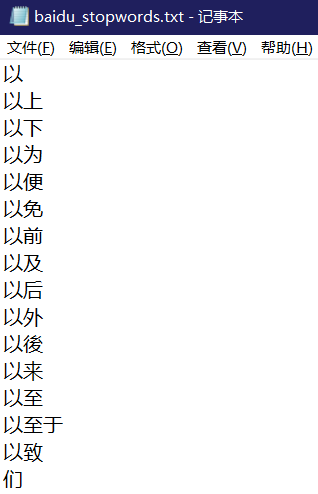
# 3. 基于个性化词典进行分词

## 3.1 语料介绍

* B站弹幕词典（持续更新） 路径：根目录下bilibili.txt



* 通用词典（不变）+ 停用词（百度常见停用词表）路径：根目录下baidu\_stopwords.txt



* 弹幕语料处理过程：

①初始语料（damu.txt）：

共28867条弹幕纪录

|  |
| --- |
| 奔涌吧，后浪！ |
| 百大等我 |
| 愿青年人都往上走，摆脱冷气，有一分热，发一分光 |
| ………… |

②对原有语句进行分词，再写入（fenci2.txt）

|  |
| --- |
| 奔涌 吧 ， 后浪 ！ |
| 百大 等 我 |
| 愿 青年人 都 往上走 ， 摆脱 冷气 ， 有 一分 热 ， 发一分光 |
| ………… |

③去掉停用词后去重，将词语写入（fenci.txt）

共有15795个词语

|  |  |  |  |
| --- | --- | --- | --- |
| 恶报 | 分析 | 会展中心 | dog |
| 问候 | QAQ | 保卫者 | 别尬 |
| …… | …… | …… | …… |

④对原有语句进行分词，去掉停用词，再写入（fenci2\_deleteStop.txt）

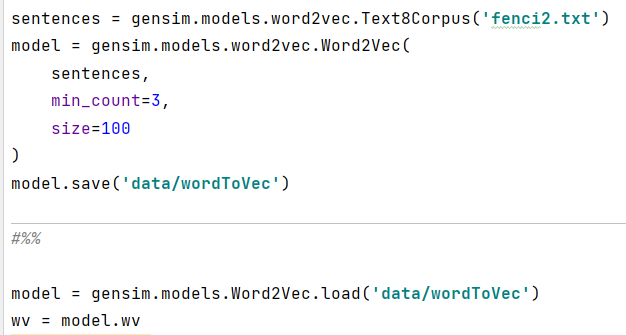
|  |
| --- |
| 奔涌 后浪 |
| 百大 |
| 愿 青年人 都 往上走 摆脱 冷气 一分 热 发一分光 |
| ………… |

## 3.2 详细处理过程

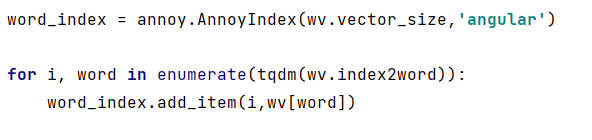
【代码请对应w2v.ipynb，使用的语料为fenci2.txt】

## 3.2.1 Word2vec得到词向量

词向量文件可重复利用，保存为data/wordToVec



## 3.2.2 Annoy对向量进行索引



得到索引后，为下面求k近邻提供变量。

# 4. 对新词进行评估

## 4.1空间索引中的K近邻

这里我们采用annoy库自带的寻找k近邻函数get\_nns\_by\_item(item, 10)

寻找当前节点的最近10个邻居，经历一次循环得到节点中比较聚集的词语，即新词集合。

相应代码如下：



## 4.2 筛选满足条件的新词

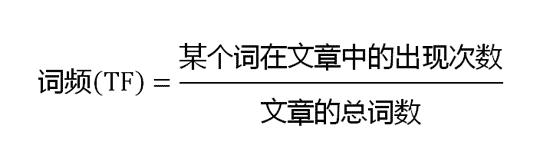
### 4.2.1 TF-IDF算法步骤

筛选新词，本作业采用的是根据**TF-IDF**的值来判断是否入选新词。计算步骤如下：

第一步，计算词频：

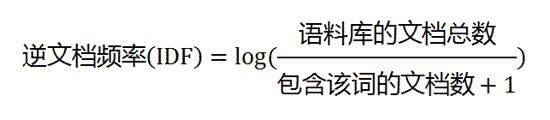


考虑到文章有长短之分，为了便于不同文章的比较，进行"词频"标准化。



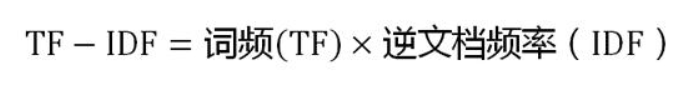
第二步，计算逆文档频率：

这时，需要一个语料库（corpus），用来模拟语言的使用环境。



如果一个词越常见，那么分母就越大，逆文档频率就越小越接近0。分母之所以要加1，是为了避免分母为0（即所有文档都不包含该词）。log表示对得到的值取对数。

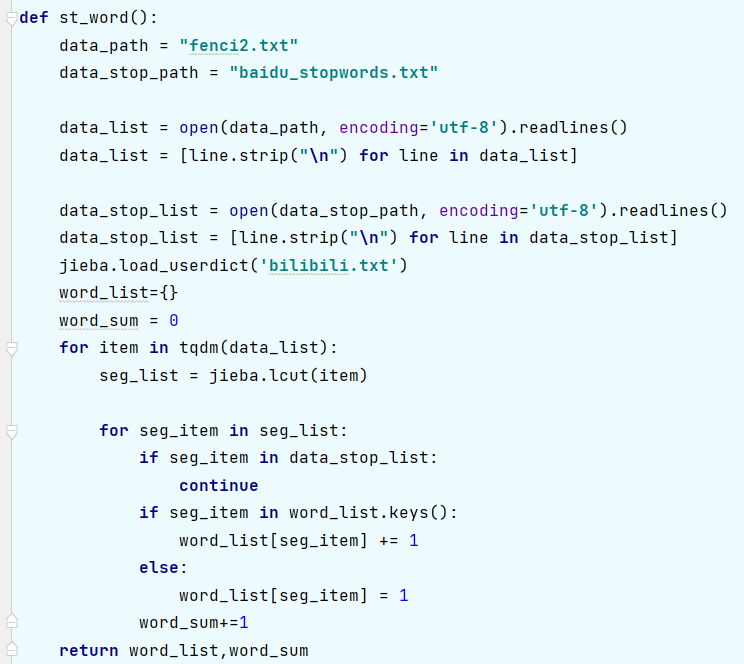
第三步，计算TF-IDF：

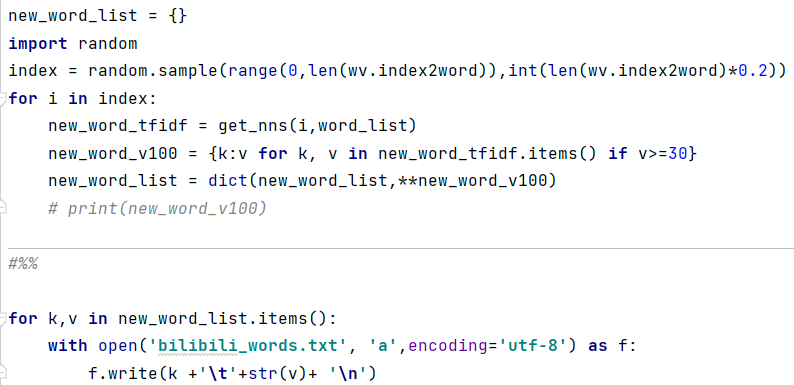


可以看到，TF-IDF与一个词在文档中的出现次数成正比，与该词在整个语言中的出现次数成反比。所以，自动提取关键词的算法就很清楚了，就是**计算出文档的每个词的TF-IDF值，然后按降序排列，取排在最前面的几个词。**

### 4.2.2 代码及结果分析

计算的代码如下：

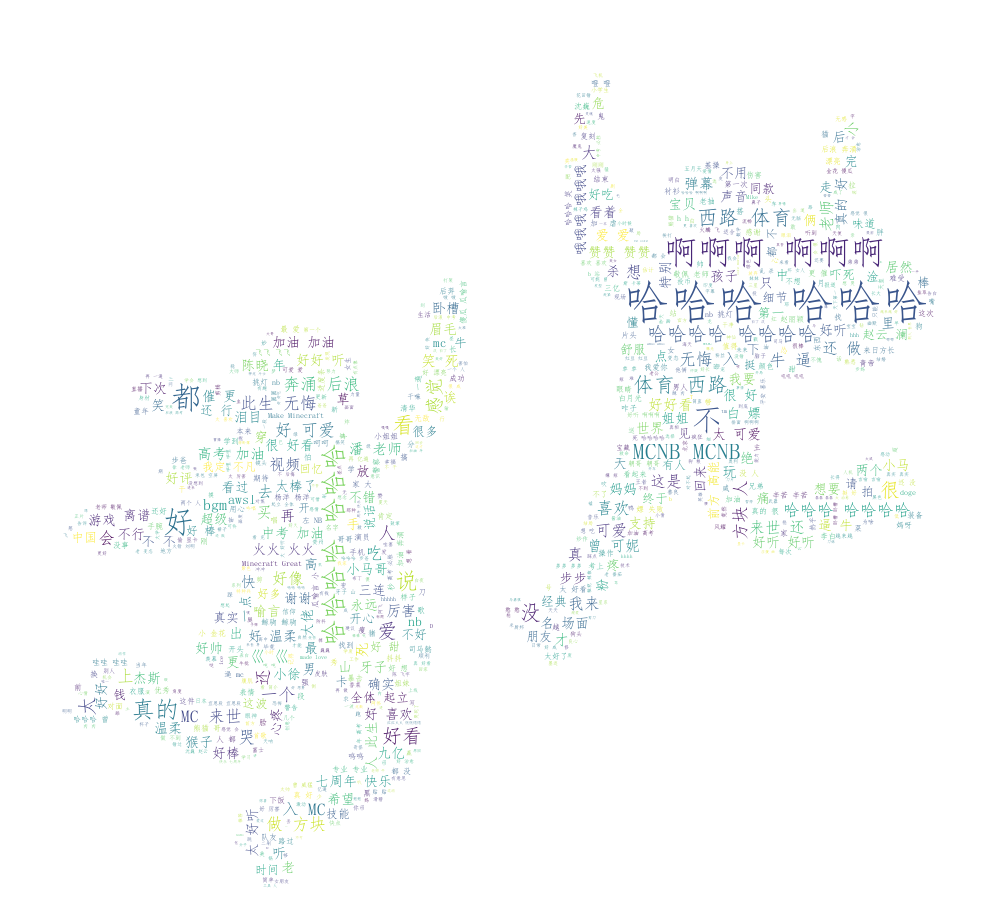




直到新词集合不变后，我们将新词以及其对应的词频写入（bilibili\_words.txt）

|  |  |
| --- | --- |
| 新词 | 词频 |
| 白 | 73 |
| 基操 | 35 |
| bgm | 70 |
| …… | …… |

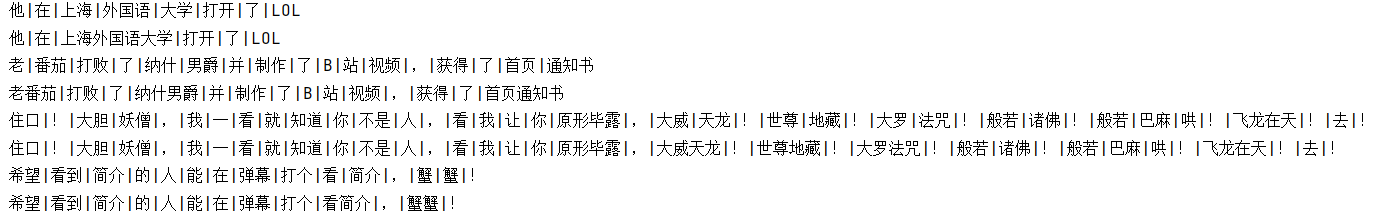
如下为词云展示：



# 5. 案例分析，例句展示

当前小节的代码见 fenci.py

|  |  |
| --- | --- |
| 序号 | 原句+不加载个人词典+加载个人词典 |
| ① | **他在上海外国语大学打开了LOL** |
| **他|在|上海|外国语|大学|打开|了|LOL** |
| **他|在|上海外国语大学|打开|了|LOL** |
| ② | **老番茄打败了纳什男爵并制作了B站视频，获得了首页通知书** |
| **老|番茄|打败|了|纳什|男爵|并|制作|了|B|站|视频|，|获得|了|首页|通知书** |
| **老番茄|打败|了|纳什男爵|并|制作|了|B站|视频|，|获得|了|首页通知书** |
| ③ | **住口！大胆妖僧，我一看就知道你不是人，看我让你原形毕露，大威天龙！世尊地藏！大罗法咒！般若诸佛！般若巴麻哄！飞龙在天！去！** |
| **住口|！|大胆|妖僧|，|我|一|看|就|知道|你|不是|人|，|看|我|让|你|原形毕露|，|大威|天龙|！|世尊|地藏|！|大罗|法咒|！|般若|诸佛|！|般若|巴麻|哄|！|飞龙在天|！|去|！** |
| **住口|！|大胆|妖僧|，|我|一|看|就|知道|你|不是|人|，|看|我|让|你|原形毕露|，|大威天龙|！|世尊地藏|！|大罗法咒|！|般若|诸佛|！|般若|巴麻|哄|！|飞龙在天|！|去|！** |
| ④ | **希望看到简介的人能在弹幕打个看简介，蟹蟹！** |
| **希望|看到|简介|的|人|能|在|弹幕|打个|看|简介|，|蟹|蟹|！** |
| **希望|看到|简介|的|人|能|在|弹幕|打个|看简介|，|蟹蟹|！** |



# 附录

[1] URLlib: <https://docs.python.org/zh-cn/3.7/library/urllib.request.html>, Accessed 2021-03-15

[2] Scapy, <https://scapy.net/>, Accessed 2021-03-15

[3]Selenium, <https://www.selenium.dev/>, Accessed 2021-03-15