# 动漫弹幕的文本检索和应用

## 开题报告

11811721 庄湛

2020/4/25

# 语料库选择


## 语料特点：

- 语料库总量大
- 单句精简短小
- 新词叠词较多
- 错别字词较多
- 词语较口语化
- 部分词项词频较高
- 不同语境下差异大

# 语料库选择

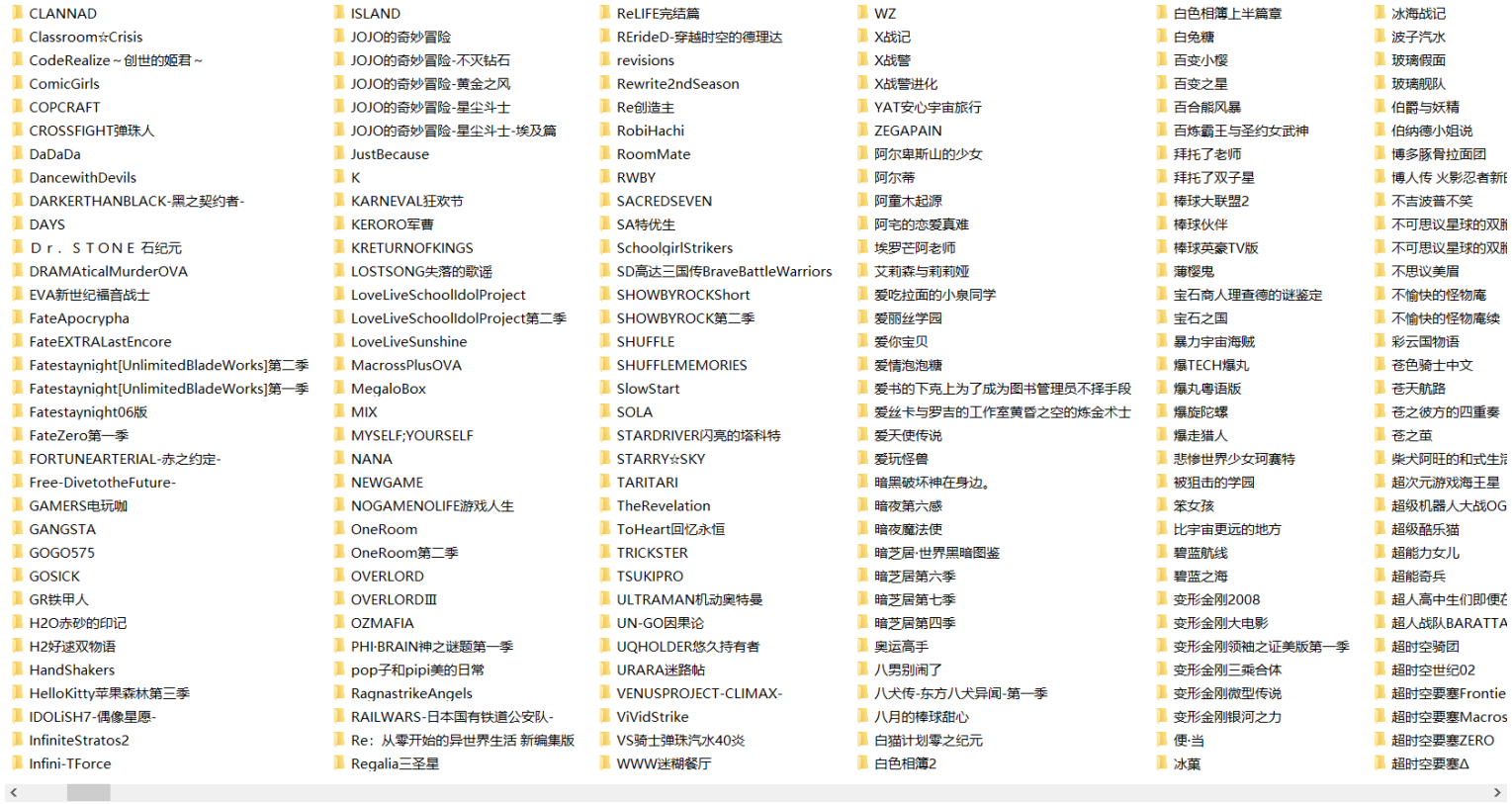
哔哩哔哩弹幕网

| api类型 | 单集弹幕数 | 访问需求 | 解决方案 |
| --- | --- | --- | --- |
| 实时弹幕 | 当日3000 | 无需登录 | 长期爬取 |
| 历史弹幕 | 累计10w+ | 需要登录，反爬虫 | cookies 池 |

两部较热门动画采用了实时弹幕爬取，约400w条数据
其余1000多部动漫采用实时弹幕爬取，约5000w条数据

https://api.bilibili.com/x/v2/dm/history?type=1&oid={}&date={}
http://comment.bilibili.com/{}.xml

# 存储格式




## 4月14日第一次爬取：

大小: 271 MB (284,595,618 字节)

占用空间: 310 MB (325,726,208 字节)

包含: 26,035 个文件，26,976 个文件夹

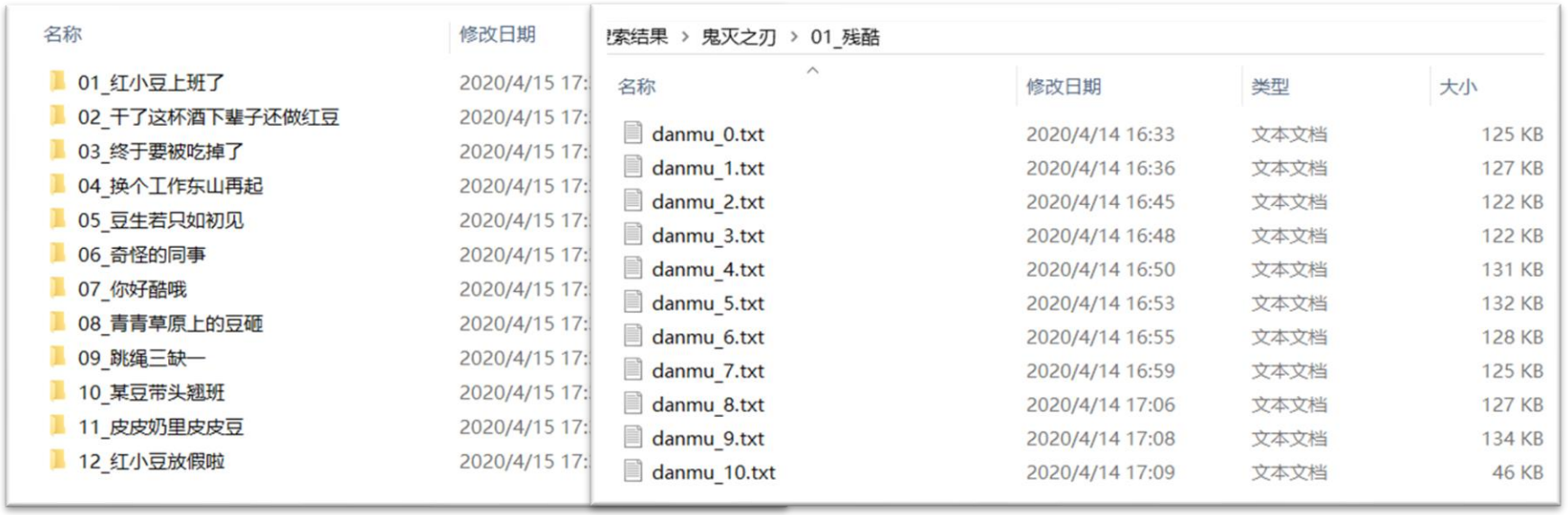
## 4月30日第二次爬取：

大小: 529 MB (555,043,668 字节)

占用空间: 611 MB (640,708,608 字节)

包含: 51,697 个文件，31,306 个文件夹

# 存储格式




动漫 – 剧集 – 弹幕文件

文件大小接近

按照时间顺序增加弹幕文件

每条弹幕占据一行

```
1960 典型的转校生都是怪物
1961 别去救它
1962 好燃
1963 好燃
1964 温馨
1965 可怕吗
1966 害怕
1967 被发现了
1968 今晚刀谁都懂了吧
1969 巴拉能量
1970 这兔头肯定香
1971 战场原啊
1972 王水洗头
1973 橘势大好
1974 钉钉时代
1975 一袋漏糠机打米
1976 咂瓦鲁多
1977 战场原黑仪
1978 斋藤千和
1979 不要随便救陌生人啊
1980 熟练的让人心疼
1981 古神低语
1982 炖了
1983 已经不用害怕了
1984 画面与歌词严重不符
1985 强的一匹
1986 见渊进
1987 强势橘气
1988 UMB
1989 预言家
1990 互攻他不香吗
1991 BGM听着渗的慌
1992 对不起ue
1993 熟练得令人心疼
1994 八嘎呀路
```

# 项目计划

| 项目简介 | 所使用的方法或工具 | 预期完成时间 |
| --- | --- | --- |
| 网页爬虫 | Requests, RegExp | 5.1 |
| 弹幕分词 | Jieba分词, FP-Tree, KMP | 5.4 |
| 建立倒排索引 | SPIMI | 5.5 |
| 检索方法 | 向量空间模型或概率似然模型 | 5.15 |
| 实现任务一：梗百科搜索 | Tf-Idf or BM25 Top k rank<br>或查询似然检索 | 5.20 |
| 实现任务二：动漫推荐 | Tf-Idf or BM25, k-means | 5.25 |

# 分词方法

1.分词原理

（1）基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图 (DAG)；（2）采用了动态规划查找最大概率路径，找出基于词频的最大切分组合；（3）对于未登录词，采用了基于汉字成词能力的HMM模型，使用了 Viterbi 算法。

2.分词三种模式

（1）精确模式，试图将句子最精确地切开，适合文本分析；（2）全模式，把句子中所有的可以成词的词语都扫描出来，速度非常快，但是不能解决歧义；（3）搜索引擎模式，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。

# 分词方法

```
['完结 撒花',
 '看 面包 的 眼神',
 '全体 起立',
 '朋也 你 怎么 了',
 '大乔 停止 了 思考',
 '兄弟 长椅 情',
 '黄金 精神',
 '不愧 是 乔鲁诺',
 '大乔 不 知道 应该 点赞 还是 沉默',
 '大乔 又 点 了 赞',
 '比如 压路机',
 '康一',
 '大乔为 你 点赞',
 'nice',
 '好 了 忘 了 这个 设定',
 '真就速 A 呗',
 '指 爽朗 的 偷 了 你 的 行李',
 '孔乙己 收到 一张 好人 卡',
 '乔纳森 也 很 痛心',
 '父词 子嗜',
 '压路机 可以 吗',
 '呼叫 由 花子',
 '孔乙己 你长 荒木线 了',
 '民风 淳朴 意大利',
 '老汉 突然 出现',
 '大乔点 了 个 踩',
 '笑 死',
 '吼吼',
 '请 忘 了 这个 设定',
 '是 艺术',
 '再 放送',
 '传统 的 开局 先 揍 队友',
 '嘟噜 噜 噜 噜 噜 噜 噜 噜 噜 噜 噜']
```

```
print ("/".join(jieba.cut("大乔为你点赞")))
jieba.add_word("大乔", freq = 10000, tag = None)
print ("/".join(jieba.cut("大乔为你点赞")))
```

```
大乔为/你/点赞
大乔/为/你/点赞
```

```
[60]: print(shorten("23333333"))
      print(shorten("wryyyyyyyy"))
      print(shorten("ohhhhhhhhhh"))
      print(shorten("义勇啊啊啊啊啊啊"))
      print(shorten("木大木大木大木大木大"))
      print(shorten("你们有毒吧哈哈哈哈哈哈"))
      print(shorten("得得得得得得得得得得得得得得得得"))
```

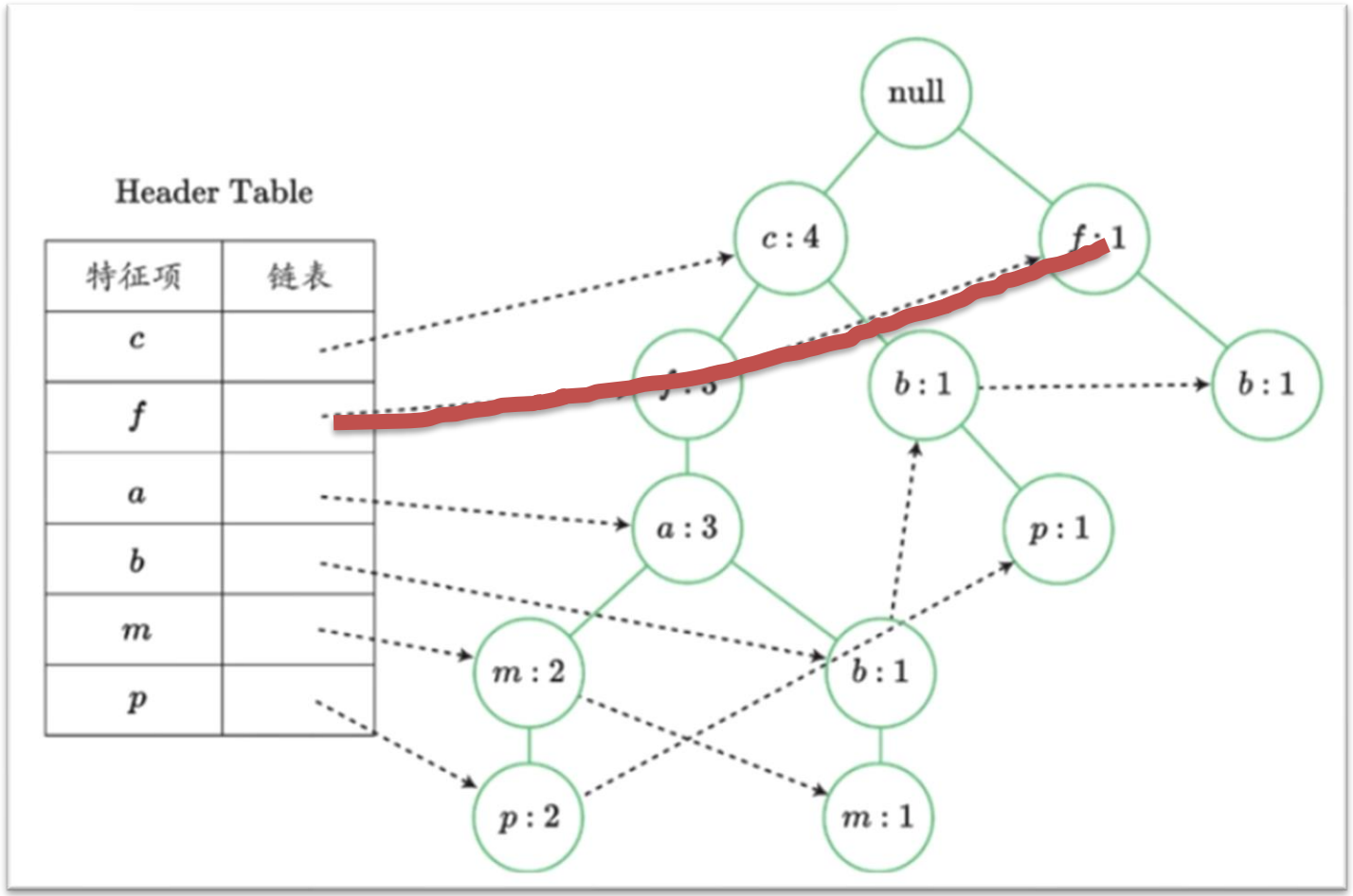
```
233
wryy
ohh
义勇啊啊
木大
你们有毒吧哈哈
得得
```

利用后缀数组，做叠词清洗

1. 叠词清洗
2. 转换为字典（弹幕，频率）
3. 收录较短高频词
4. 转换为（单字，频率）

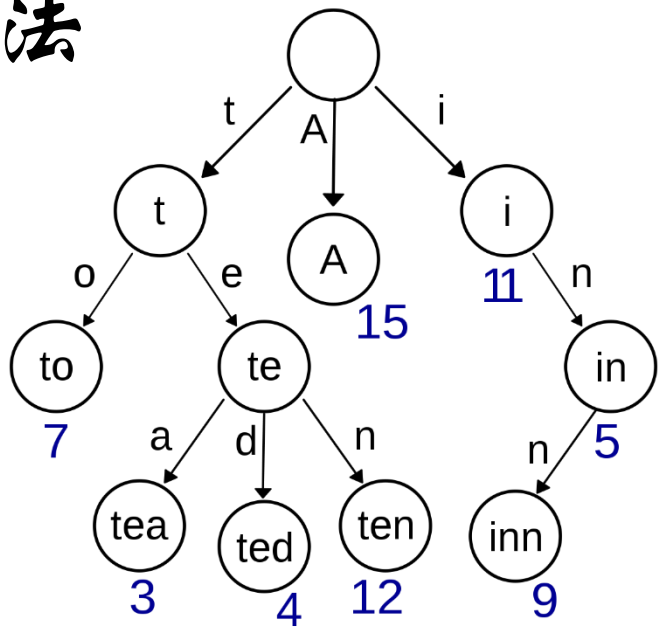
```
[('卖炭翁', 40),
 ('哈哈', 37),
 ('百鬼丸', 36),
 ('满面尘灰烟火色', 35),
 ('泪目', 32),
 ('两鬓苍苍十指黑', 26),
 ('前方高能', 26),
 ('awsl', 25),
 ('flag', 22),
 ('犹豫就会败北', 19),
 ('工具人', 17),
 ('好看吗', 17),
 ('果断就会白给', 17),
 ('追了', 16),
 ('护食', 16),
 ('恶鬼灭杀', 16),
```

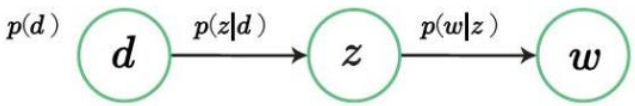
# 分词方法




Frequent Pattern Tree (FP Tree)

# 分词方法




S(大) = 305
S(乔) = 271
P(乔|大) = 202/305
P(大|乔) = 202/271

(1) S(a) > 10
(2) P(b|a) > 1.3 - 0.35ln(S(b))
(3) P(a|b) > 0.6

```
retTree = treeNode('大', headerTable['大'][0], None)
next_tree = headerTable['大'][1]
while next_tree is not None:
    for child in next_tree.children.values():
        addNode(retTree, child)
    next_tree = next_tree.nodeLink
```

```
retTree.disp()
```

```
  大   305
    乔   202
      停   9
        止   9
          了   9
            思   9
              考   9
      不   7
        知   7
          道   5
            应   1
              该   1
                点   1
                  赞   1
                    还   1
                      是   1
                        沉   1
                          默   1
            该   3
              怎   1
                么   1
                  办   1
              点   2
                赞   2
                  还   2
```

# 分词方法

```
留下jio印 留个jio印 0.3333333333333333
祝君武运昌 祝各位武运昌隆 0.2222222222222222
下jio印 留个jio印 0.375
下一季再见 期待下一季 0.2857142857142857
鸡心打开 几盆鸡心打 0.3333333333333333
祝武运昌 祝各位武运昌隆 0.25
君武运昌 祝各位武运昌隆 0.25
等下一季 期待下一季 0.3333333333333333
一季再见 期待下一季 0.16666666666666666
为什么 什么时候 0.25
再见了 下季再见 0.25
肝完了 天肝完 0.3333333333333333
个脚印 下脚印 0.3333333333333333
缘再见 下季再见 0.25
```

1. 子串去重
2. 2 gram 相似度去重
3. 导入词典
4. 去停用词，分词

Result →

```
simplify(new_words)
```

```
['女装大佬',
 '珍爱上了',
 '糟糕的台',
 '下季再见',
 '盗梦空间',
 '错误示范',
 '一天肝完',
 '撩完就跑',
 '笑死我了',
 '下次一定',
 '美女你谁',
 '剧场版见',
 '猝不及防',
 '前方高能',
 '完结撒花',
 '撑不住了',
 '不娶何撩',
 '口气肝完',
 '逻辑鬼才',
 '到此一游',
 '优秀员工',
 '证明我来',
 '虎狼之词',
 '精神小伙',
 '感谢陪伴',
 '二十六集',
 '进度条',
 '下爪印',
 '香奈乎',
 '好可爱',
 '舍不得',
 '紫藤花',
 '为什么',
 '晒太阳',
 '我来过',
```

# 倒排索引

类别 → 文档 → 词项

动漫 → 剧集 → 词项

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

$$\text{Weighted Term Frequency} = \frac{(k+1)\mathrm{tf}_{td}}{k * ((1-b) + b * (\mathrm{len}(D)/\mathrm{avg\_doclen})) + \mathrm{tf}_{td}}$$

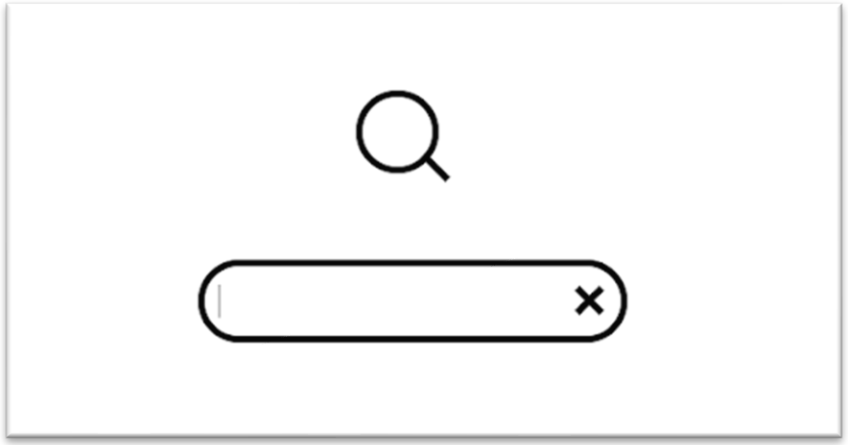
```
1960 典型的转校生都是怪物
1961 别去救它
1962 好燃
1963 好燃
1964 温馨
1965 可怕吗
1966 害怕
1967 被发现了
1968 今晚刀谁都懂了吧
1969 巴拉能量
1970 这兔头肯定香
1971 战场原啊
1972 王水洗头
1973 橘势大好
1974 钉钉时代
1975 一袋漏糠机打米
1976 咂瓦鲁多
1977 战场原黑仪
1978 斋藤千和
1979 不要随便救陌生人啊
1980 熟练的让人心疼
1981 古神低语
1982 炖了
1983 已经不用害怕了
1984 画面与歌词严重不符
1985 强的一匹
1986 见渊进
1987 强势橘气
1988 UMB
1989 预言家
1990 互攻他不香吗
1991 BGM听着渗的慌
1992 对不起ue
1993 熟练得令人心疼
1994 八嘎呀路
```

# 梗百科搜索

## 梗的意思指动画、电视剧等作品中喜闻乐见的桥段

特点：

1. 小众，代表着部分圈子文化
2. 源于一部作品，但会出现于多部作品弹幕中（玩梗）
3. 一般不符合常规语言句式，但复用性强
4. 传播效率高，有些具有时效性



火之意志 → 火影忍者疾风传

黄金精神 → JOJO的奇妙冒险

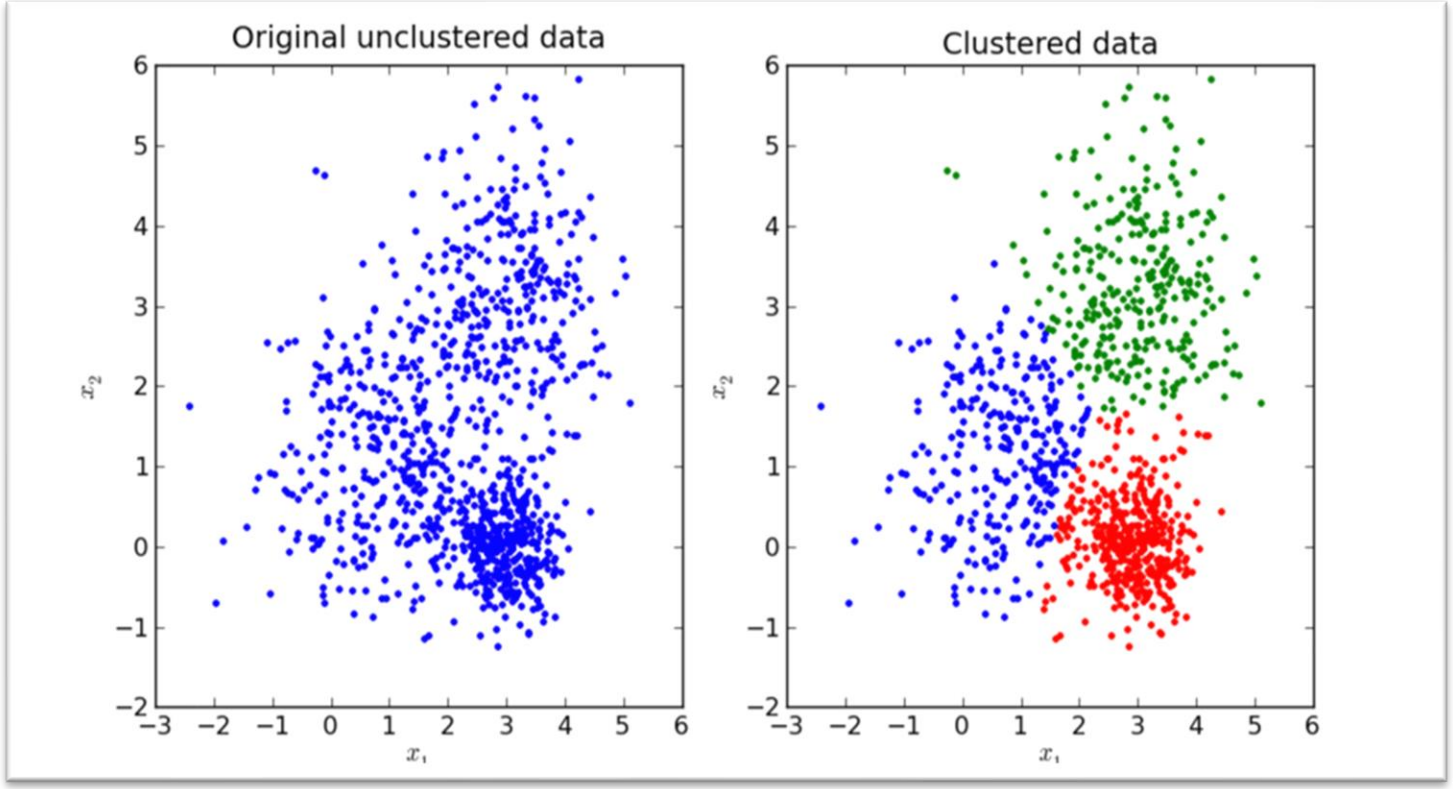
（符号化词汇）

# 动漫推荐

动漫间距离：用余弦相似度或杰卡德相似度刻画

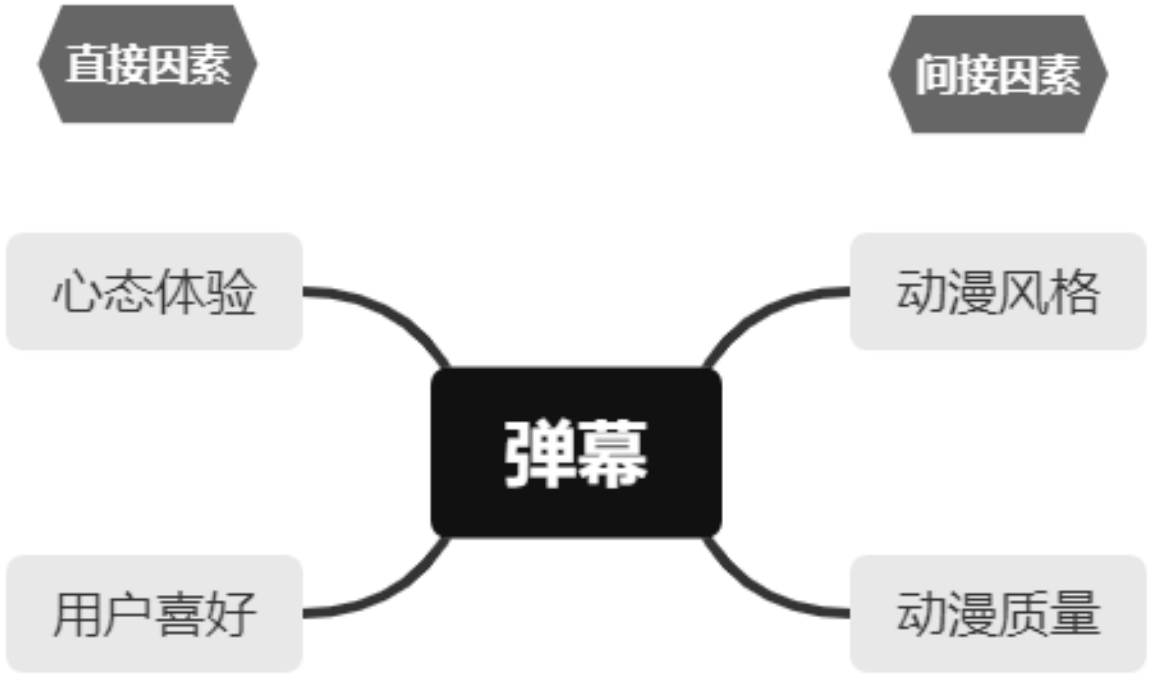
数据存储：用单个文档表示一部动漫的高维向量

聚类方法：K-means



# 项目意义

哔哩哔哩弹幕网




对于视频平台：

1. **通过对弹幕的聚类分析**，可以将动漫进行初分类。将用户近期观看的动漫或所发的弹幕进行相似度分析，平台可以刻画用户类型，并进行精准推荐。
2. **梗百科**也可以作为弹幕文化的一部分促进平台社区文化，能让新用户更快了解社区文化，增加归属感。

# Thanks