北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

# 基于TF-IDF算法
# 在微博热搜话题发现中的运用

答辩人：张子涵 张依妹 曹媛 祁晨玥 武思铭

时间：2022.04.01

# 目录

# 选题的目的与意义

# 选题目的与意义



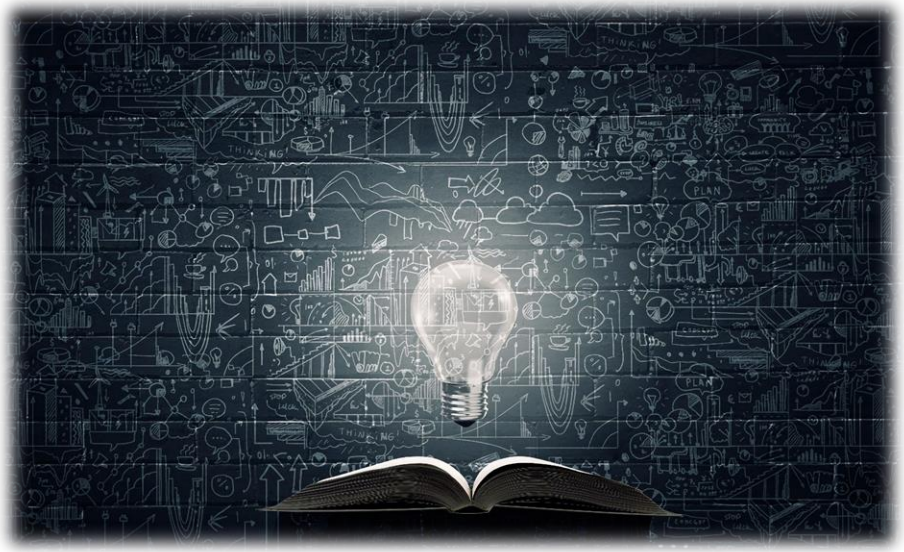
随着互联网的快速发展，网络信息交互与传播迅速且敏捷，网络中重要的信息常淹没在海量数据中发现热点话题、追踪热点话题演变以及预测话题的发展倾向，为有需求的用户及时提供有效网络舆情信息、舆情监控和竞争情报具有很大意义。

# 经典综述

# 一、研究背景

从宏观层面来看，管理部门要制定科技政策、集中优势去发展新兴领域

从微观层面来看，科研工作者依然需把握时代的脉搏，发现有潜力、有价值的研究领域，利用有限的资源完成有效果的科研工作。



# 二、研究话题发现的方法



文本挖掘方法

科学计量分析方法

# 二、研究话题发现的方法



文本挖掘方法是指：一部分的研究借助于自然语言处理技术对科技文献和专利文献中的话题、关键词或受控词进行深度挖掘分析，从而成功实现对话题发现的研究。

# 二、研究话题发现的方法



科学计量分析方法是最为常用的方法，即利用以科技文献元数据的分析与挖掘为基础的方法，主要包括各种引用关系网络、共现关系网络的分析方法。

# 三、话题发现的应用



在线新闻话题发现研究

帮助人们更加清晰全面地了解网络热点事件，也有助于相关部门积极开展对应工作，对于舆情监控、个性化推荐等具有重要意义。

# 三、话题发现的应用



基于PLSA的舆情事件子话题发现模型

成功实现了从海量网络舆情事件中提取到有效的事件种类，所生成的话题标签有着高度准确性，更精准的概括事件内容。

通过一系列的统计分析方法，话题标签可以轻松发掘事件的共同点，及时反馈出此话题热度的变化趋势。

# 三、话题发现的应用



还有研究者将新闻话题发现和情感极性结合起来进行研究

以真实的在线网络新闻为研究对象，通过一系列算法和改进，该方法可以及时的了解到在线新闻用户对热点新闻话题所持态度和立场，有助于相关部门对于问题的及时解决，并做出相应的回应和疏导，积极避免负能量、错误观点的传播，为维护社会法治安定做出贡献。

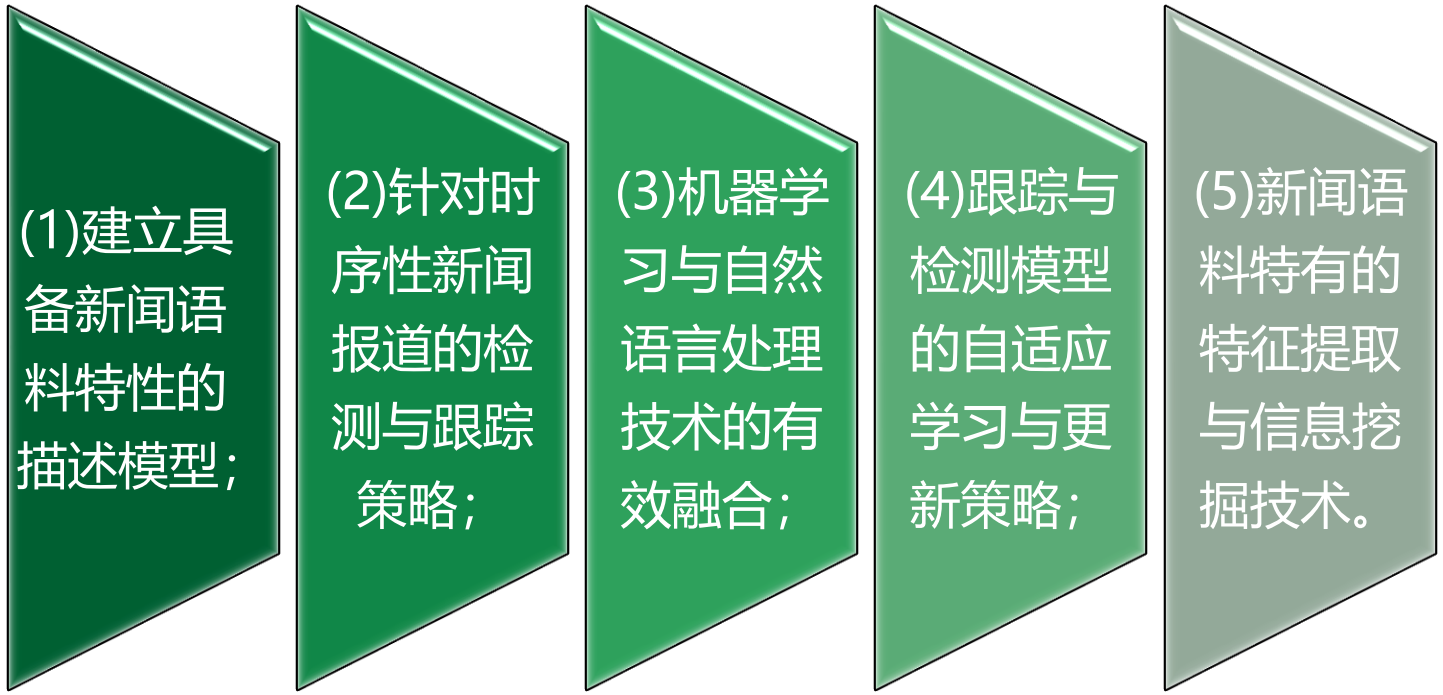
# 四、研究进展



| 1996年 | 1997年 | 2016年 | 2021年 |
| --- | --- | --- | --- |
| • 话题识别与跟踪的概念产生 | • 开始了对于这一问题的初步研究。 | • 提出了一种对文本进行隐含主题提取的话题发现方法。 | • 改进了WMD算法和传统SP算法。 |

# 五、未来发展展望

TDT领域未来的研究方向将主要集中于如下几个方面：



# 五、未来发展展望

在如今的背景下，需要解决的问题就是社交话题发现所带来的多语适用性不足和低内聚、低效率。

第一点是为了解决多语适用性不足所带来的问题，多语语义串自动发现上下文信息的计算模型的使用于改进是关键；

第二点是为解决丰富长短文本特征抽取的有效性和处理效率的问题，多语语义关键特征抽取是关键；

第三点是为解决低内聚、低效率的热点话题发现问题，基于关键特征的TOP N热点话题发现和归类算法是关键。

# 五、未来发展展望

为及时了解微博圈的动态，有必要对海量的微博信息进行组织和分析，提出有效的算法从这些信息中对话题行提取，并以简洁的形式提供给用户。

# 前沿进展

国外研究现状

卡内基-梅隆大学采用向量空间模型，在时间窗口内，利用组平均聚类的方法先把报道聚成小类，再利用最近邻聚类将小类与已有的信息合并。

实验证明，这种按照时序进行组内优先聚类的方法是非常合适的，因为它把时间相近的报道以更大的优先级聚到一起。

实验发现，组平均聚类更适合于“突发性的事件”，而最近邻聚类适合“持续性事件”，把两者混合起来能达到比较好的效果。

国外研究现状

与之相似的是IBM公司所采用的两层聚类方法，当时间窗口增大时，系统的效果提高得比较明显。

马萨诸塞大学采用了两种模型：向量空间模型和语言模型。

实验发现,采用K最近邻聚类，K取1时效果是最好的。

国外研究现状

与此同时,Dragon系统公司和TNO大学都采用的是语言模型。

其中,TNO大学在最近邻聚类中引人了聚类结果调整的策略

即当一个时间窗口内的报道处理完毕时，重新比较每一-篇文档和已有的类，把文档调整到最相近的类中，并且如果一个类在一段时间内不发生变化，这个类就固定不动了。

实验表明，结果调整的策略大大提高了系统的效果。

# 前沿进展


当前研究不足

目前的研究现状仍然以传统基于统计策略的信息检索、信息过滤、分类和聚类等技术为主，忽视了新闻语料本身具备的特点

当前的研究趋势是将多种方法进行融合，并嵌入新闻语料特性实现话题的识别与追踪

# 模型Demo演示

# 模型Demo演示



利用爬虫爬取了5天内的微博
其中每一条包括了：
是否原创、微博id、微博内容、微博位置、发布时间、点赞、评论、转发数量以及发布工具

C: > 张子涵 > 大数据 > weibo > weibo > weibo_follow.py > ...

```python
class Follow(object):
    def __init__(self, config):
        """Follow类初始化"""
        self.validate_config(config)
        self.cookie = config['cookie']
        user_id_list = config['user_id_list']
        if not isinstance(user_id_list, list):
            if not os.path.isabs(user_id_list):
                user_id_list = os.path.split(
                    os.path.realpath(__file__))[0] + os.sep + user_id_list
            user_id_list = self.get_user_list(user_id_list)
        self.user_id_list = user_id_list  # 要爬取的微博用户的user_id列表
        self.user_id = ''
        self.follow_list = []  # 存储爬取到的所有关注微博的uri和用户昵称

    def validate_config(self, config):
        """验证配置是否正确"""
        user_id_list = config['user_id_list']
        if (not isinstance(user_id_list,
                           list)) and (not user_id_list.endswith('.txt')):
            sys.exit(u'user_id_list值应为list类型或txt文件路径')
        if not isinstance(user_id_list, list):
            if not os.path.isabs(user_id_list):
                user_id_list = os.path.split(
                    os.path.realpath(__file__))[0] + os.sep + user_id_list
            if not os.path.isfile(user_id_list):
                sys.exit(u'不存在%s文件' % user_id_list)

    def deal_html(self, url):
        """处理html"""
        try:
            user_agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
            headers = {
                'User_Agent': user_agent,
                'Cookie': self.cookie,
                'Connection': 'close'
            }
            html = requests.get(url, headers=headers).content
            selector = etree.HTML(html)
            return selector
        except Exception as e:
            print('Error: ', e)
            traceback.print_exc()

    def get_page_num(self):
        """获取关注列表页数"""
        url = "https://weibo.cn/%s/follow" % self.user_id
        else:
            page_num = (int)(
                selector.xpath("//input[@name='mp']")[0].attrib['value'])
        return page_num

    def get_one_page(self, page):
        """获取第page页的user_id"""
        print(u'%s第%d页%s' % ('-' * 30, page, '-' * 30))
        url = 'https://weibo.cn/%s/follow?page=%d' % (self.user_id, page)
        selector = self.deal_html(url)
        table_list = selector.xpath('//table')
        if (page == 1 and len(table_list) == 0):
            print(u'cookie无效或提供的user_id无效')
        else:
            for t in table_list:
                im = t.xpath('.//a/@href')[-1]
                uri = im.split('uid=')[-1].split('&')[0].split('/')[-1]
                nickname = t.xpath('.//a/text()')[0]
                if {'uri': uri, 'nickname': nickname} not in self.follow_list:
                    self.follow_list.append({'uri': uri, 'nickname': nickname})
                    print(u'%s %s' % (nickname, uri))

    def get_follow_list(self):
        """获取关注用户主页地址"""
        page_num = self.get_page_num()
        print(u'用户关注页数：' + str(page_num))
        page1 = 0
        random_pages = random.randint(1, 5)
        for page in tqdm(range(1, page_num + 1), desc=u'关注列表爬取进度'):
            self.get_one_page(page)

            if page - page1 == random_pages and page < page_num:
                sleep(random.randint(6, 10))
                page1 = page
                random_pages = random.randint(1, 5)

        print(u'用户关注列表爬取完毕')

    def write_to_txt(self):
        with open('user_id_list.txt', 'ab') as f:
            for user in self.follow_list:
                f.write((user['uri'] + ' ' + user['nickname'] + '\n').encode(
                    sys.stdout.encoding))

    def get_user_list(self, file_name):
        """获取文件中的微博id信息"""
        with open(file_name, 'rb') as f:
```

# 模型Demo演示

```
126
127     def initialize_info(self, user_id):
128         """初始化爬虫信息"""
129         self.follow_list = []
130         self.user_id = user_id
131
132     def start(self):
133         """运行爬虫"""
134         try:
135             for user_id in self.user_id_list:
136                 self.initialize_info(user_id)
137                 print('*' * 100)
138                 self.get_follow_list()  # 爬取微博信息
139                 self.write_to_txt()
140                 print(u'信息抓取完毕')
141                 print('*' * 100)
142         except Exception as e:
143             print('Error: ', e)
144             traceback.print_exc()
145
146
147 def main():
148     try:
149         config_path = os.path.split(
150             os.path.realpath(__file__))[0] + os.sep + 'config_follow.json'
151         if not os.path.isfile(config_path):
152             sys.exit(u'当前路径：%s 不存在配置文件config_follow.json' %
153                      (os.path.split(os.path.realpath(__file__))[0] + os.sep))
154         with open(config_path) as f:
155             try:
156                 config = json.loads(f.read())
157             except ValueError:
158                 sys.exit(u'config.json 格式不正确，请参考 '
159                          u'https://github.com/dataabc/weiboSpider#3程序设置')
160         wb = Follow(config)
161         wb.start()  # 爬取微博信息
162
163     except Exception as e:
164         print('Error: ', e)
165         traceback.print_exc()
166
167
168 if __name__ == '__main__':
169     main()
```

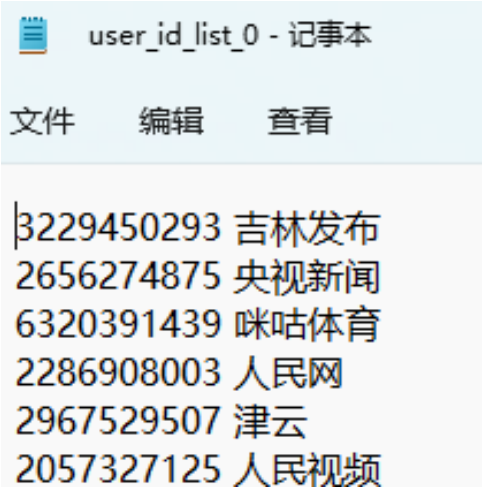

爬取微博id并且进行数据去重得出了微博id合集

# 模型Demo演示

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-


import os
import sys

from absl import app
sys.path.append(os.path.abspath(os.path.dirname(os.getcwd())))
from weibo_spider.spider import main

app.run(main)
```

利用Spyder爬取数据，
将config_follow作为参数，
根据list_0，输入cookies登陆




> 张子涵 > 大数据 > weibo > weibo > {} config_follow.json > ...

```
1   {
2       "user_id_list":  "./user_id_list_0.txt",
3       "cookie":  "SUBP=0033WrSXqPxfM725Ws9jqgMF55529P9D9WW
4   }
5
```

# 模型Demo演示



设置了从哪天开始到哪天的爬取
用csv和txt的格式导出
防止因为数据量较大，
微博发现异样而禁止爬取，
控制速度在6-8之间

```
{
    "user_id_list": "./uuid_list.txt",
    "filter": 1,
    "since_date": "2022-03-20",
    "end_date": "now",
    "random_wait_pages": [1, 7],
    "random_wait_seconds": [1, 3],
    "global_wait": [[1000, 360], [500, 200],[700,150]],
    "write_mode": ["csv", "txt"],
    "pic_download": 0,
    "video_download": 0,
    "file_download_timeout": [5, 5, 10],
    "result_dir_name": 0,
    "cookie": "SUBP=0033WrSXqPxfM725Ws9jqgMF55529P9D9WWdSr0Ec
    "mysql_config": {
        "host": "localhost",
        "port": 3306,
        "user": "root",
        "password": "123456",
        "charset": "utf8mb4"
    },
    "kafka_config": {
        "bootstrap-server": "127.0.0.1:9092",
        "weibo_topics": ["spider_weibo"],
        "user_topics": ["spider_weibo"]
    },
    "sqlite_config": "weibo.db"
}
```

# 模型Demo演示


sample_text - 记事本
文件 编辑 查看
人生 最大 运气 不是 捡钱 不是 中奖 有人 鼓励 指引 帮助 发现 更好 走向 更 高 平台 其实 限制 发展 所处 生活圈 工作 圈
气象 科普 时间 南北 两极 出现 极端 高温 南极洲 多个 考察站 本月 18 日测 创纪录 高温 康 科迪亚 南极 考察站 测 11.5 较往年 同期 平均水平 高出 约 40 美联社 报道 北极 部分 地区 中旬 气温 较往
城区 精细化 预报 2022 23 20 时 24 20 时
发布 头条 文章 阳光 中迎 世界 气象日 气温 开启 回升 路 阳光 中迎 世界 气象日 气温 开启 回升 路
短期 天气预报 泰州市 气象台 23 16 时 发布 天气预报 今夜 多云 明天 白天 晴转多云 夜里 多云转阴 小雨 东南风 级 明晨 最低气温 明天 最高 气温 18 19 25 阴有 小到中雨 11 18 26 多云到阴 12 18
上下班 预计 下午 下班 多云 偏南风 级 气温 13 明晨 上班 多云 偏南风 级 气温
泰州 未来 天 空气质量 预报 23 14 时 空气质量 指数 aqi 49 优 24 夜里 25 日有 一次 降水 过程 受 地面 倒 槽 气旋 影响 25 26 风力 较大 预计 空气污染 气象 扩散 条件 24 转较 好 25 26 日较 好 预
气象 科普 时间 323 世界 气象日 早 预警 早 行动 大 数据 解读 霜冻 预警 我国 霜冻 预警 严重 级别 低 高 分别 蓝 黄 橙 表示 2021 我国 发布 超过 7000 霜冻 预警 发布 预警 最多 地方 地处 亚热带
2022 世界 气象日 世界气象组织 秘书 长佩蒂瑞 塔拉斯 致辞 二 〇 二 二年 世界 气象日 早期 预警 必须 人人 受益 戳 中国气象局
短期 天气预报 泰州市 气象台 23 11 时 发布 天气预报 今天下午 夜里 多云 晴 南到 西南风 级 左右 今天 最高 气温 14 左右 明晨 最低气温 24 白天 晴转多云 夜里 多云转阴 小雨 18 25 阴有 小到中雨
萌虎教 疫情 防控
气象 科普 时间 323 世界 气象日 早 预警 早 行动 大 数据 解读 暴雨 预警 我国 气象 灾害 预警 信号 防御 指南 规定 暴雨 预警 信号 严重 级别 低 高 分别 蓝色 黄色 橙色 红色 四级 一般来说 南方 暴雨
上下班 预计 中午 下班 多云 偏南风 级 气温 12 下午 上班 多云 偏南风 级 气温 14
今天 323 世界 气象日 2022 世界 气象日 主题 早 预警 早 行动 气象 水文 气候 信息 助力 防灾 减灾
城区 精细化 预报 2022 23 08 时 24 08 时
短期 天气预报 泰州市 气象台 23 时 发布 天气预报 今天 多云 晴 南到 西南风 级 左右 今天 最高 气温 14 左右 明晨 最低气温 24 白天 晴转多云 夜里 多云转阴 小雨 18 25 阴有 小到中雨 11 18
雷达 风云 追踪 最新 雷达 资料 监测 受 层状 云 降水 回波 影响 今天 夜里 我市 部分 地区 小 阵雨
城区 精细化 预报 2022 22 20 时 23 20 时
短期 天气预报 泰州市 气象台 22 16 时 发布 天气预报 今天 夜里 阴 有时 小雨 明天 多云 晴 东北风 转 西南风 级 左右 明晨 最低气温 明天 最高 气温 14 15 24 白天 晴转多云 夜里 多云转阴 小雨 19
发布 头条 文章 春天 别急 春姑娘 回来 春天 别急 春姑娘 回来
泰州 未来 天 空气质量 预报 22 14 时 空气质量 指数 aqi 33 优 23 24 白天 好 24 夜里 25 日有 一次 降水 过程 受 地面 倒 槽 气旋 影响 24 25 风力 较大 预计 空气污染 气象 扩散 条件 23 24 转较 好
上下班 预计 下午 下班 阴到 多云 偏 东风 级 左右 气温 明晨 上班 多云 晴 西到 西南风 级 左右 气温
短期 天气预报 泰州市 气象台 22 11 时 发布 天气预报 今天下午 阴到 多云 夜里 阴 有时 小雨 东北风 级 今天 最高 气温 11 左右 明晨 最低气温 23 多云 晴 15 24 白天 晴转多云 夜里 多云转阴 小雨 1
疫情 防控 防疫 关键字 记牢 做实 防疫 关键 十二 字 12
上下班 预计 中午 下班 阴到 多云 东北风 级 气温 10 下午 上班 阴到 多云 东北风 级 气温 11
城区 精细化 预报 2022 22 08 时 23 08 时
短期 天气预报 泰州市 气象台 22 时 发布 天气预报 今天 阴到 多云 有时 小雨 东北风 级 今天 最高 气温 11 左右 明晨 最低气温 23 多云 晴 15 24 白天 晴转多云 夜里 多云转阴 小雨 19
院士 称新冠 影响 人类 心理 20 需 做好 长期 准备 保证 睡眠 调节 情绪 近日 中国科学院 院士 陆林 科普 栏目 院士 开讲 中 提到 新冠 肺炎 爆发 以来 数亿 人 出现 失眠 问题 新增 7000 万 抑郁症 患
城区 精细化 预报 2022 21 20 时 22 20 时
行1，列1
100%
Unix (LF)
UTF-8

# 模型Demo演示



大体流程：

```
input: doc vector;threshold
output:cluster
begin
    input doc vector
    input threshold
    first doc as first cluster and it's vector as the center of the cluster
    while(doc vectors){
        while(clusters){
            max_sim,max_cluster = simlarity(doc vector,cluster);
        }
        if(max_sim > threshold){
            max_cluster.put(doc vector);
            max_cluster.update_center()
        }
        else{
            build new cluster(doc vector);
        }
    }
end
```

计算生成话题簇

# 模型Demo演示

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

```
    1.利用tfidf vec计算cossim
'''
def tfidf_vec(self, corpus, pivot=10, slope=0.25):
    dictionary = corpora.Dictionary(corpus)  # 形成词典映射
    print(len(dictionary))
    dictionary.filter_extremes(no_below=10,no_above=0.4)
    print(dictionary.most_common(200))
    self.dict_size = len(dictionary)
    print('dictionary size:{}'.format(len(dictionary)))
    corpus = [dictionary.doc2bow(text) for text in corpus]  # 词的向量表示
    tfidf = models.TfidfModel(corpus, pivot=pivot, slope=slope)
    corpus_tfidf = tfidf[corpus]
    return corpus_tfidf

def get_max_similarity(self, cluster_cores, vector):
    max_value = 0
    max_index = -1
     print('vector:{}'.format(vector))
    for k, core in cluster_cores.items():
         print('core:{}'.format(core))
        similarity = matutils.cossim(vector, core)
        if similarity > max_value:
            max_value = similarity
            max_index = k
    return max_index, max_value

def single_pass(self, corpus_vec, corpus, theta):
    clusters = {}
    cluster_cores = {}
    cluster_text = {}
    num_topic = 0
    cnt = 0
    for vector, text in zip(corpus_vec, corpus):
        if num_topic == 0:
            clusters.setdefault(num_topic, []).append(vector)
            cluster_cores[num_topic] = vector
            cluster_text.setdefault(num_topic, []).append(text)
            num_topic += 1
        else:
            max_index, max_value = self.get_max_similarity(cluster_cores, vector)
            if max_value > theta:
                clusters[max_index].append(vector)
                text_matrix = matutils.corpus2dense(clusters[max_index], num_terms=self.dict_size,
                                                    num_docs=len(clusters[max_index])).T  # 稀疏转稠密
                core = np.mean(text_matrix, axis=0)  # 更新簇中心
                core = matutils.any2sparse(core)  # 将稠密向量core转为稀疏向量
                cluster_cores[max_index] = core
                cluster_text[max_index].append(text)
            else:  # 创建一个新簇
                clusters.setdefault(num_topic, []).append(vector)
                cluster_cores[num_topic] = vector
                cluster_text.setdefault(num_topic, []).append(text)
                num_topic += 1
        cnt += 1
        if cnt % 100 == 0:
            print('processing {}...'.format(cnt))
    return clusters, cluster_text

def fit_transform(self, corpus, raw_data, theta=0.5):
    tfidf_vec = self.tfidf_vec(corpus)  # tfidf_vec是稀疏向量
    clusters, cluster_text = self.single_pass(tfidf_vec, raw_data, theta)
    return clusters, cluster_text
```

# 模型Demo演示

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

```python
def doc2vec_single_pass(self, corpus_vec, corpus, theta):
    clusters = {}
    cluster_cores = {}
    cluster_text = {}
    num_topic = 0
    cnt = 0
    for vector, text in zip(corpus_vec, corpus):
        if num_topic == 0:
            clusters.setdefault(num_topic, []).append(vector)
            cluster_cores[num_topic] = vector
            cluster_text.setdefault(num_topic, []).append(text)
            num_topic += 1
        else:
            max_index, max_value = self.get_doc2vec_similarity(cluster_cores, vector
            if max_value > theta:
                clusters[max_index].append(vector)
                core = np.mean(clusters[max_index], axis=0)  # 更新簇中心
                cluster_cores[max_index] = core
                cluster_text[max_index].append(text)
            else:  # 创建一个新簇
                clusters.setdefault(num_topic, []).append(vector)
                cluster_cores[num_topic] = vector
                cluster_text.setdefault(num_topic, []).append(text)
                num_topic += 1
        cnt += 1
        if cnt % 100 == 0:
            print('processing {}...'.format(cnt))
    return clusters, cluster_text
```

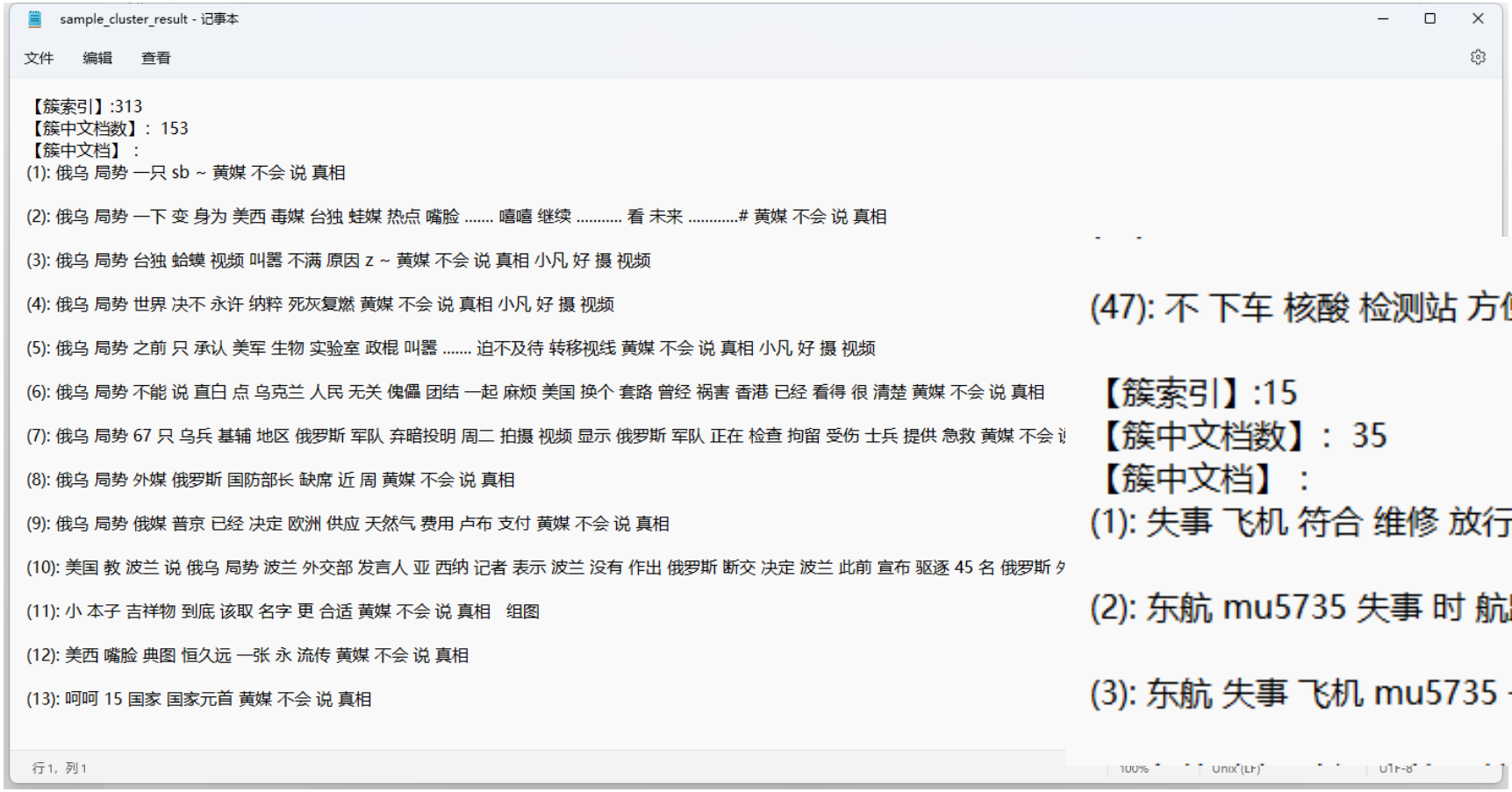
```python
if __name__ == '__main__':

    # cal_vec_type: tfidf 或者 doc2vec 二选一
    cal_vec_type = 'tfidf'
    # theta: 控制相似度的超参数，越小，类别数越少
    theta = 0.2

    base_path = os.path.join(os.getcwd())
    process_text = base_path + '/data/process_text.txt'  # 处理后的样本路径
#     doc2vec_path = base_path + '/data/doc2vec.pkl'
    cluster_result = base_path + '/data/cluster_result.txt'
#     doc_vec_path = base_path + '/data/doc_vec.vec'  # 经过doc2vec推荐的文本向量

    corpus = load_data(process_text)
    raw_text = load_samples(process_text)


    index2corpus = collections.OrderedDict()
    for index, line in enumerate(raw_text):
        index2corpus[index] = line
    text2index = list(index2corpus.keys())
    print('docs total size:{}'.format(len(text2index)))

    single_cluster = SingelPassCluster()

    if cal_vec_type == 'tfidf':

        clusters, cluster_text = single_cluster.fit_transform(corpus, text2index
```

# 模型Demo演示



```
print("..........................................................................................")
print("得到的类数量有: {} 个 ...".format(len(clusters)))
print("..........................................................................................\n")
# 按聚类语句数量对聚类结果进行降序排列
clusterTopic_list = sorted(cluster_text.items(), key=lambda x: len(x[1]), reverse=True)
with open(cluster_result, 'w', encoding='utf-8') as file_write:
    for k in clusterTopic_list:
        cluster_text = []
        for index, value in enumerate(k[1],start=1):
            cluster_text.append('(' + str(index) + '): ' + index2corpus[value])
        cluster_text = '\n'.join(cluster_text)
        file_write.write("【簇索引】:{} \n【簇中文档数】:{} \n【簇中文档】 :\n{}".format(k[0], len(k[1]), cluster_text))
        file_write.write('\n')
        file_write.flush()
```

# 模型Demo演示


sample_cluster_result - 记事本
文件 编辑 查看
【簇索引】:313
【簇中文档数】： 153
【簇中文档】：
(1): 俄乌 局势 一只 sb ~ 黄媒 不会 说 真相
(2): 俄乌 局势 一下 变 身为 美西 毒媒 台独 蛙媒 热点 嘴脸 ..... 嘻嘻 继续 ..... 看 未来 .....# 黄媒 不会 说 真相
(3): 俄乌 局势 台独 蛤蟆 视频 叫嚣 不满 原因 z ~ 黄媒 不会 说 真相 小凡 好 摄 视频
(4): 俄乌 局势 世界 决不 永许 纳粹 死灰复燃 黄媒 不会 说 真相 小凡 好 摄 视频
(5): 俄乌 局势 之前 只 承认 美军 生物 实验室 政棍 叫嚣 ..... 迫不及待 转移视线 黄媒 不会 说 真相 小凡 好 摄 视频
(6): 俄乌 局势 不能 说 直白 点 乌克兰 人民 无关 傀儡 团结 一起 麻烦 美国 换个 套路 曾经 祸害 香港 已经 看得 很 清楚 黄媒 不会 说 真相
(7): 俄乌 局势 67 只 乌兵 基辅 地区 俄罗斯 军队 弃暗投明 周二 拍摄 视频 显示 俄罗斯 军队 正在 检查 拘留 受伤 士兵 提供 急救 黄媒 不会
(8): 俄乌 局势 外媒 俄罗斯 国防部长 缺席 近 周 黄媒 不会 说 真相
(9): 俄乌 局势 俄媒 普京 已经 决定 欧洲 供应 天然气 费用 卢布 支付 黄媒 不会 说 真相
(10): 美国 教 波兰 说 俄乌 局势 波兰 外交部 发言人 亚 西纳 记者 表示 波兰 没有 作出 俄罗斯 断交 决定 波兰 此前 宣布 驱逐 45 名 俄罗斯
(11): 小 本子 吉祥物 到底 该取 名字 更 合适 黄媒 不会 说 真相 组图
(12): 美西 嘴脸 典图 恒久远 一张 永 流传 黄媒 不会 说 真相
(13): 呵呵 15 国家 国家元首 黄媒 不会 说 真相
行 1，列 1
(47): 不 下车 核酸 检测站 方便 网约车 出租车 司机
【簇索引】:15
【簇中文档数】： 35
【簇中文档】：
(1): 失事 飞机 符合 维修 放行 标准 和适航 要求 东
(2): 东航 mu5735 失事 时 航路 上 适航 民航 事故
(3): 东航 失事 飞机 mu5735 一部 黑匣子 已 找到 记

# 主要参考文献

# 主要参考文献


1. 李想, 倪丽萍, 夏千姿,等. 在线新闻话题发现技术研究及应用综述[C]// 第十四届(2019)中国管理学年会. 0.
2. Sakaki T, Okazaki M, Matsuo Y. Earthquake shakes Twitter users:real-time event detection by social sensors[C]// Proc. International World Wide Web Conference, 2010:851-860.
3. 王琦.面向网络新闻的热点话题发现与极性分析[D].大连：大连海事大学，2016
4. Chen C M. CiteSpace II: Detecting and visualizing emerging trends and transient patterns in scientific literature[J]. Journal of the American Society for Information Science and Technology, 2006, 57(3): 359-377.
5. Small H, Boyack K W, Klavans R. Identifying emerging topics in science and technology[J]. Research Policy, 2014, 43(8): 1450-1467.、

# 主要参考文献


5. Yan E. Research dynamics: Measuring the continuity and popularity of research topics[J]. Journal of Informetrics, 2014, 8(1): 98-110.
6. 殷风景, 肖卫东, 葛斌,等. 一种面向网络话题发现的增量文本聚类算法[J]. 计算机应用研究, 2011, 28(1):4.
7. 郭蓝天,李扬,慕德俊,等. 一种基于LDA主题模型的话题发现方法[J]. 西北工业大学学报,2016,34(4):697-701. DOI:10.3969/j.issn.1000-2758.2016.04.022.
8. 卢超,侯海燕,DING YING,等. 国外新兴研究话题发现研究综述[J]. 情报学报,2019,38(1):97-110. DOI:10.3772/j.issn.1000-0135.2019.01.011.
9. 郝晓玲,茅嘉惠,于秀艳. 微博热词抽取及话题发现研究[J]. 情报杂志,2015(6):109-113,157. DOI:10.3969/j.issn.1002-1965.2015.06.020.

# 主要参考文献


10. 游丹丹,陈福集. 我国网络舆情热点话题发现研究综述[J]. 现代情报,2017,37(3):165-171. DOI:10.3969/j.issn.1008-0821.2017.03.029.
11. 陈兴蜀,罗梁,王海舟,等. 基于ICE-LDA模型的中英文跨语言话题发现研究[J]. 工程科学与技术,2017,49(2):100-106. DOI:10.15961/j.jsuese.201601032.
12. 李勇. 基于两层聚类的微博热点话题发现算法研究[J]. 自动化技术与应用,2021,40(11):45-50. DOI:10.3969/j.issn.1003-7241.2021.11.010.
13. 刘雅筠. 微博话题发现技术国内外研究现状[J]. 科教导刊-电子版（中旬）,2021(2):297-298.
14. 宋莉娜, 冯旭鹏, 刘利军,等. 基于SOM聚类的微博话题发现[J]. 计算机应用研究, 2018, 35(3):5.

# 需要说明的问题

# 需要说明的问题

北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

# 感谢聆听