《数据科学与工程算法基础》实践报告

报告题目：应用多种子模函数最大化算法及其改进进行文本摘要

姓　　名：周辛娜

学　　号：10195501442

完成日期：2021.12.22

提供中英文摘要

摘要 [中文]：

本次实验采用六种方法实现了文本摘要，分别是课本上原始爬山算法(采用集合覆盖函数为子模函数的爬山算法)、通过 lazy 改进的课本上的爬山算法、modified-greedy algorithm 算法、通过 lazy 改进的 modified-greedy algorithm 算法、基于 Monotone Submodular Objectives 的爬山算法、通过 lazy 改进的基于 Monotone Submodular Objectives 的爬山算法。本次实验爬取了正点财经 10 页的内容，每一页有 40 篇文章，每一篇文章都围绕着一个主题展开。本次项目采用多种子模函数最大化算法及其改进提取了 100 篇摘要，每一篇摘要都对应着每页的 4 篇文章，每一页对应 10 篇摘要，10 页刚好 100 篇。前面两种方法均对爬取的语料库进行了文档摘要，并比较最后结果的可读性以及算法的运行时间。重点是关注后面四种方法。针对后面四种方法，先选取最优的参数，然后再产生 100 篇摘要，比较四种方法的覆盖率以及运行效率。

Abstract [英语]：

In this experiment, six methods were used to achieve text summarization. They are the original hill-climbing algorithm in the textbook (which uses the set covering function as a submodular function), the hill-climbing algorithm in the textbook improved by lazy, the Modified-greedy algorithm and the Modified-greedy improved by lazy A mountain climbing algorithm based on Monotone Submodular Objectives and improved by lazy. This experiment took 10 pages of zhengdian Finance, each page has 40 articles, each article around a theme. In this project, 100 abstracts were extracted by using the multi-seed modulus function maximization algorithm and its improvement. Each abstract corresponds to 4 articles on each page, and each page corresponds to 10 abstracts, which is exactly 100 articles on 10 pages. The first two methods summarize the crawling corpus and compare the readability of the final results and the running time of the algorithm. It is important to focus on the last four methods. For the last four methods, the optimal parameters were selected first, and then 100 abstracts were generated to compare the coverage and operation efficiency of the four methods.

一、 项目概述（阐明该项目的科学和应用价值，以及相关工作进展并描述项目的主要内容）

文本摘要可以被抽象为子模函数优化问题，每次选择摘要句的时候，根据子模函数求得句子的权重分数，从而选择权重分数最大的句子放到摘要集合中。那么采用何种子模函数以及如何优化爬山算法以加快算法性能是非常重要的问题，对实际的场景有着深刻的应用价值。

二、 问题描述（问题定义）

1) 抽取式文本摘要问题定义及课本给出的爬山算法(采用集合覆盖函数为子模函数的爬山算法)：

给定用户定义的 n 个关键词组成的集合 W 和 m 个候选句子的集合 S，其中 Si 表示该句子中包含的关键词集合，抽取式文本摘要问题是要选择 k 个句子，使得这 k 个句子能够包含最多数量的关键词。课本给出的爬山算法比较简单，选用的子模函数就是集合覆盖函数。

2) modified-greedy-algorithm 的提出：

子模函数选用集合覆盖函数仅仅考虑的是覆盖率，但是在实际对生成的摘要质量进行评价时，还要考虑到摘要句子的多样性，那么这时就会使用的 MMR 算法，希望提取的不同句子描述的是不同的方面。基于 MMR 的 submodular 函数：

$$MMR \stackrel{def}{=} Arg \max_{D_i \in R \setminus S} [\lambda(Sim_1(D_i, Q)) - (1-\lambda)(\max_{D_j \in S} Sim_2(D_i, D_j))]$$

但这个函数不是一个单调函数，因此爬山算法无法在常数级的量级上逼近最优解，每个迭代并不能保证选择的句子在当前约束下是最优的。故需采用 modified-greedy-algorithm 算法[1]。后面通过 lazy 去改进算法的性能。

3) 基于 Monotone Submodular Objectives 算法的提出：

基于 MMR 的 submodular 函数是不单调的，便希望能找到本身就是单调的 submodular 函数，这样就能过通过爬山算法，以常数项因子逼近最优解。因此，需要将 submodular 函数设计成单调函数。故将原本基于 MMR 的 submodular 函数中的冗余度函数设计成奖励多样性函数，使得其和相关性函数共同组成一个单调函数[2]。后面同样通过 lazy 去改进算法的性能。

4) 采用 lazy-greedy-algorithm 改进以上三种爬山算法：

如果采用原始的爬山算法，效率非常地下，所以采用 lazy 进行改进，可以在得到同样的结果下大大提升算法的运行效率[3]。

三、 方法（问题解决步骤和实现细节）

1）从 web 中爬取文本数据：

```
data = []
#爬取10个页面
for i in range(1, 11):
    url = 'http://www.zdcj.net/reportlist/newreport_{}.html'.format(i)
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
    MyPage = requests.get(url, headers=headers)
    MyPage.encoding="utf-8"
    dom = etree.HTML(MyPage.text)
    listnews = dom.xpath('//a[@title][@target="_blank"]/@href')
    count=0
    i=0
    while i<len(listnews):
        count=count+1
        news=listnews[i]
        news1=listnews[i+1]
        news2=listnews[i+2]
        news3=listnews[i+3]
        #第一个
        headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        MyPage = requests.get(news, headers=headers)
        dom = etree.HTML(MyPage.text.encode("latin1").decode("GBK"))
        items = dom.xpath('//div[@class="v_c_box"]/p/text()')
        items = "".join(items)
        headers1 = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        #第二个
        MyPage1 = requests.get(news1, headers=headers1)
        dom1 = etree.HTML(MyPage1.text.encode("latin1").decode("GBK"))
        items1 = dom1.xpath('//div[@class="v_c_box"]/p/text()')
        items1 = "".join(items1)
        #第三个
        headers2 = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        MyPage2 = requests.get(news2, headers=headers2)
        dom2 = etree.HTML(MyPage2.text.encode("latin1").decode("GBK"))
        items2 = dom2.xpath('//div[@class="v_c_box"]/p/text()')
        items2 = "".join(items2)
        #第四个
        headers3 = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        MyPage3 = requests.get(news3, headers=headers3)
        dom3 = etree.HTML(MyPage3.text.encode("latin1").decode("GBK"))
        items3 = dom3.xpath('//div[@class="v_c_box"]/p/text()')
        items3 = "".join(items3)
        items=items+items1
        if items:
            data.append(items)
        i=i+4
#爬取的内容放到data.csv
df = pd.DataFrame({'content': data}, columns=['content'])
df.to_csv('./data.csv')
```

可以看到，news、news1、news2、news3 分别代表 4 篇文章，它们一起构成一个 items，后面会用不同的方法从这合在一起的 4 篇文章中去生成一篇摘要，一共爬取了 10 页，每一页有 40 篇文章，4 篇文章生成一篇摘要，那么一共可以生成 100 篇摘要。

2）对爬取到的文本进行预处理：

1. 由于文本中有很多如\r，\n，\t 等的字符，需要去除

```
txt['content'] = txt['content'].str.replace('\r','')
txt['content'] = txt['content'].str.replace('\n','')
txt['content'] = txt['content'].str.replace('\t','')
txt['content']=txt['content'].str.replace('  ','')
txt['content']=txt['content'].str.strip()
txt = txt['content'].str.replace(' ','')
```

2. 使用 jieba 库分词

```
import jieba
txt['content'] = txt['content'].apply(lambda x:' '.join(jieba.cut(x)))
```

3. 去除停用词

读入停用词

```
file = open("stop.txt",encoding='utf-8')
stopword = file.readlines()
stopwords = [word.strip() for word in stopword]
tfidf1 = TfidfVectorizer(stop_words=stopwords, ngram_range=(1,1)).fit(txt['content'])
tfidf2 = TfidfVectorizer(stop_words=stopwords, ngram_range=(1,1))
```

4. 提取文档特征，提取关键词等

通过单词计数向量、TF-IDF 方法来提取文档特征，在 sklearn 里面如下导入：

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer

3）采用集合覆盖函数为子模函数的爬山算法实现：

1. 爬取语料库，提取关键词，得到关键词集合 W
2. 将文档里的所有包含关键词的句子汇总成集合 S
3. 在每次迭代中选择使得集合覆盖函数 f(A)边际增幅最大的候选元素，通过不停的迭代，从一个候选结果向另一个候选结果移动直到终止条件满足。

伪代码如下：

```
#输入：循环次数k
#输出：集合S

for i in range(k):
    选择集合u = argmax[f(S+{s})-f(S)]
    S=S+{U}
return S
```

具体实现：

```
#原始的爬山算法：
import time
k=10
start =time.perf_counter()
list_selected1=[]
origin_cover_list_number1=copy.deepcopy(cover_list_number1)
current_selected=np.argmax(cover_list_number1)
while len(list_selected1) <k:
    for j in range(0, len(sentence)):
        for i in range(0,origin_cover_list_number1[j]):
            if true_false_list1[cover_list[j][i]]==True:
                cover_list_number1[j]=cover_list_number1[j]-1

    current_selected=np.argmax(cover_list_number1)
    cover_list_number1[current_selected]=-1
    list_selected1.append(current_selected)
    for i in range(0,origin_cover_list_number1[current_selected]):
        true_false_list1[cover_list[current_selected][i]]=True
end = time.perf_counter()
es1=end-start
```

列表 W 代表关键词集合，包含了 100 个关键词。

列表 true_false_list 与关键词集合即列表 W 的长度相同，用来记录某一个关键词是否已经在被选择的句子中了，初始化均为 false。

列表 sentence 代表候选句子。

列表 cover_list_number 代表某一候选句子包含的关键词的个数。

二维数组 cover_list[i][j]代表第 i+1 个候选句子的第 j+1 个词在关键词集合 W 中的位置，例如 cover_list[0][1]=3 代表句子 1 的第二个关键词位置是关键词集合中的第 3 个词。current_selected 代表现在选择第几个句子 。

每次遍历查找使得增加的关键词最多的句子，如果在 true_false_list 中已经有了这个关键词，那么这个句子中包含的关键词个数就要减 1，因为是要看新增的关键词个数，为此还要维护一个 origin_cover_list_number，选出该句子后，将 cover_list_number 中的对应位置置为 0，这样下一次 current_selected 就不会选择它了，这样不断迭代直到选出了 k 个句子。

4）modified-greedy-algorithm 的算法实现：

这里的子模函数分为两部分：相关性-冗余性

基于 MMR 的 submodular 函数：

$$MMR \stackrel{def}{=} Arg \max_{D_i \in R \setminus S} [\lambda(Sim_1(D_i, Q)) - (1-\lambda)(\max_{D_j \in S} Sim_2(D_i, D_j))]$$

第一部分代表的是候选句子和原文的相似度，第二部分代表的是候选句子与当前已有的摘要集合的冗余性，这里相似度采用余弦相似度。

算法伪代码如下：

1: $G \leftarrow \emptyset$
2: $U \leftarrow V$
3: **while** $U \neq \emptyset$ **do**
4: $\quad k \leftarrow \arg\max_{\ell \in U} \frac{f(G \cup \{\ell\}) - f(G)}{(c_\ell)^r}$
5: $\quad G \leftarrow G \cup \{k\}$ **if** $\sum_{i \in G} c_i + c_k \leq B$ **and** $f(G \cup \{k\}) - f(G) \geq 0$
6: $\quad U \leftarrow U \setminus \{k\}$
7: **end while**
8: $v^* \leftarrow \arg\max_{v \in V, c_v \leq \mathcal{B}} f(\{v\})$
9: return $G_f = \arg\max_{S \in \{\{v^*\}, G\}} f(S)$

modified-greedy-algorithm 在计算候选摘要句边际增幅时，因为每个句子的 cost 都不一定相同引入 scaling factor r 进行 rescaling。scaling factor 能够平衡一些在选择时难以取舍的例子。由于本次实验没有验证集无法通过交叉验证的方式确定最优 r，这里就通过运行效率来确定 r。

伪代码第 8 行，跳出最外层循环后，获取边际增幅最大且 cost 满足约束条件的原始文章中的句子，取该句子对应的 submodular 函数值与摘要集合的函数值的最大值。这一步能够保证在 r=1 时得到常数项的逼近因子，克服了由于该子模函数不是一个单调函数而带来的无法收敛的问题。

5）基于 Monotone Submodular Objectives 算法实现：

这里子模函数分为两部分：相关性+冗余性

$$\mathcal{F}(S) = \mathcal{L}(S) + \lambda \mathcal{R}(S)$$

$$\mathcal{L}(S)=\sum_{i\in V}\min\{\mathcal{C}_i(S),\alpha\,\mathcal{C}_i(V)\}\qquad \mathcal{C}_i(S)=\sum_{j\in S}w_{i,j}\qquad \mathcal{R}(S)=\sum_{i=1}^{K}\sqrt{\sum_{j\in P_i\cap S}r_j}.$$

其中，S 表示候选摘要集，L(S)度量覆盖率，R(S)用于奖励多样性，λ>=0，通过奖励多样性的方式而非冗余惩罚的方式保证了该目标函数是单调的，从而用爬山算法可以得到最优解。其中，wi,j 表示句子 i 和句子 j 之间的相似度，Ci(S)表示原文中待选的句子 i 与候选摘要集 S 的相似度，αCi(V)是 Ci(S)能够取到的最大值，一旦 L(S)=αCi(V)，说明即使将句子 i 添加到 S 中，覆盖率也不会有明显提升，从而继续挑选 V 中剩余的句子。

ri 表示对句子 i 的奖励，评估了句子 i 对于摘要的重要性。

在具体实现中，相似度的度量同论文中采用的相似度相同，为余弦相似度，即：

$$w_{i,j}=\frac{\sum_{w\subset s_i}\mathrm{tf}_{w,i}\times\mathrm{tf}_{w,j}\times\mathrm{idf}_w^2}{\sqrt{\sum_{w\in s_i}\mathrm{tf}_{w,s_i}^2\mathrm{idf}_w^2}\sqrt{\sum_{w\in s_j}\mathrm{tf}_{w,j}^2\mathrm{idf}_w^2}};$$

ri 为一个候选句子与文档其他句子相似度的矩阵

$$r_i=\frac{1}{N}\sum_j w_{i,j}$$

那么目标函数中的相关性部分就变为了：

$$\mathcal{L}_1(S)=\sum_{i\in V}\min\left\{\sum_{j\in S}w_{i,j},\alpha\sum_{k\in V}w_{i,k}\right\}$$

冗余部分变为了：

$$\mathcal{R}_1(S)=\sum_{k=1}^{K}\sqrt{\sum_{j\in S\cap P_k}\frac{1}{N}\sum_{i\in V}w_{i,j}}.$$

该算法的整体逻辑同 modified-greedy-algorithm，仅仅是最大化的目标函数发生了改变。

6）lazy greedy algorithm 实现

算法伪代码如下：

Function:LazyForward$(\mathcal{G}=(\mathcal{V},\mathcal{E}),R,c,B,type)$
$\mathcal{A}\leftarrow\emptyset$; **foreach** $s\in\mathcal{V}$ **do** $\delta_s\leftarrow+\infty$;
while $\exists s\in\mathcal{V}\setminus\mathcal{A}: c(\mathcal{A}\cup\{s\})\le B$ **do**
  foreach $s\in\mathcal{V}\setminus\mathcal{A}$ **do** $cur_s\leftarrow$ **false**;
  while ***true*** **do**
    if *type=UC* **then** $s^*\leftarrow\underset{s\in\mathcal{V}\setminus\mathcal{A},c(\mathcal{A}\cup\{s\})\le B}{\operatorname{argmax}}\ \delta_s$;
    if *type=CB* **then** $s^*\leftarrow\underset{s\in\mathcal{V}\setminus\mathcal{A},c(\mathcal{A}\cup\{s\})\le B}{\operatorname{argmax}}\ \frac{\delta_s}{c(s)}$;
    if cur_s **then** $\mathcal{A}\leftarrow\mathcal{A}\cup s^*$; **break** ;
    else $\delta_s\leftarrow R(\mathcal{A}\cup\{s\})-R(\mathcal{A})$; $cur_s\leftarrow$ **true**;
return $\mathcal{A}$;

原始的爬山算法每次迭代都要计算所有句子的边际增幅，而 lazy greedy algorithm 每个迭代不一定会计算所有句子的边际增幅。

进行每一次迭代时，都将上一轮迭代得到的句子的权重分数进行排序，选出分数最大值对应的句子，之后，优先更新分数第二大的句子的边际增幅，以此类推。

由于目标函数是单调的，所以分数第二大的句子仍然可能是候选句子中分数最大的句子，从而避免了每次都要计算所有句子的边际增幅，当然最坏情况下，也是要计算所

有句子的边际增幅，但这种情况毕竟少数，这样一来，算法的效率就能够大大提高了。

lazy greedy algorithm 可以用来加速上述三种方法，所以就另外产生了三种改进方法。

以上是六种方法实现的步骤，具体代码见本文附录。

7）性能指标

比较最后结果的可读性、覆盖率以及算法的运行时间。运行时间是利用 python 中的 datetime 模块的 datetime.now()函数得到当前和结束时间，两者作差便得到运行时间。

四、 实验结果（验证提出方法的有效性和高效性）

1. 首先对四篇文章采用 6 种不同的方法生成一篇摘要，来比较生成摘要的时间及可读性

1）运行时间：

采用集合覆盖函数为子模函数的爬山算法：

```
运行时间
0.004040100000565872
```

经 lazy 加速后的采用集合覆盖函数为子模函数的爬山算法：

```
运行时间
0.0019093999999313382
```

可以看到，经 lazy 加速后算法效率提高了。

2）可读性:

由于 lazy 仅仅是加速了算法，并没有改变最后的结果，所以这里仅仅显示生成的 3 个摘要。

以集合覆盖函数为子模函数的爬山算法：

我们认为，此举彰显了中航工业集团对军民用直升机业务长期发展的信心，从中直股份业绩可以看出直升机业务正处于高景气周期，随着集团资产整合的加速推进，直升机研制及部分总装资产未来有望注入上市公司体内，增厚业绩同时完善直升机资产完整性，打造我国直升机龙头上市公司。首次覆盖给予“买入”评级，合理估值区间12.60~14.50元预测公司2019-2021FY的营业收入分别为225.3、238.0、273.9亿元，同比增速分别为3.5%、5.6%和15.1%；净利润分别为19.7、22.1和25.3亿元，同比增速分别为-11.8%、12.2%和14.0%；BPS分别为5.54、6.30和7.18元/股；对应PB分别为1.65、1.45和1.27倍。（3）昌飞集团：我国直升机科研生产基地，具备研制和批产多品种、多系列、多型号直升机和航空零部件生产的能力，主要产品有直8、直10、直11等军用直升机，AC310、AC311、AC313等民用直升机。目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间。自2017年政府加强外废进口管制以来，我国外废进口量大幅下降。直升机需求不断扩大，军民融合市场空间广阔军改后陆军航空兵扩编和海军陆战队建设，我国军用直升机需求不断扩大。自建速生林、采购外废制浆造纸提升原材料渠道自主可控程度，降低原材料价格波动对成本的冲击。预计到2020-2021年，废纸名义缺口量每年可能超过1000万吨。投资要点。据东方网10月1日消息，在我国国庆70周年阅兵表演中直20首次公开亮相。

modified-greedy-algorithm：

。风险提示。（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权。首次覆盖太阳纸业并给予公司“买入”评级。交易完成后，中航科工合计持有中直股份50.80%股份。此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持。根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间。我国直升机制造业的主力军，自主研发助推新品交付与民品替代中直主营航空产品直升机，现有核心产品包括直8、直9、直11、AC311、AC312、AC313等型号直升机及零部件，近年来已逐步完成主要产品型号的更新换代。预计到2020-2021年，废纸名义缺口量每年可能超过1000万吨。市场担忧减持对公司股价造成较大波动，预计该担忧部分影响了公司的股价表现。公司发布公告，EQT（LeaderHoldingLimited，简称“卖方”）将转让其持有的泽星投资100%股权给SonataCompanyLimited（简称“买方”），总对价为5.57亿美元。

基于 Monotone Submodular Objectives 算法

。风险提示。风险提示：。点评：。事件：。首次覆盖太阳纸业并给予公司“买入”评级。（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权。交易完成后，中航科工合计持有中直股份50.80%股份。此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持。目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间。直升机资产加速整合，公司中长期受益。EQT股权转让完成，新战略股东有望带来新变化。我们预计，公司特殊用户、民用直升机两方向将持续不断推进。根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间。在民机销售方面，各型直升机销售稳中有进，各主打产品逐步站稳市场，努力推进“一带一路市场的开拓”。投资要点。风险提示：公司重大资产重组进度低于预期，军工订单低于预期。盈利预测及估值。老挝和广西项目是太阳打开南方市场的战略重镇。

可以看到，经过 lazy 加速后的爬山算效率提升了，以集合覆盖函数为子模函数的爬山算法、modified-greedy-algorithm 和基于 Monotone Submodular Objectives 算法的可读性一

般。仅仅使用集合的覆盖函数作为子模函数去完成文本摘要事实上还有很多问题没有考虑，所以接下来主要是比较后面四种方法(modified-greedy algorithm 算法、通过 lazy 改进的 modified-greedy algorithm 算法、基于 Monotone Submodular Objectives 的爬山算法、通过 lazy 改进的基于 Monotone Submodular Objectives 的爬山算法)。

2. 在不同的超参数取值 r 下，由运行时间和覆盖率决定最终产生 100 篇摘要的 r 值

1）运行时间

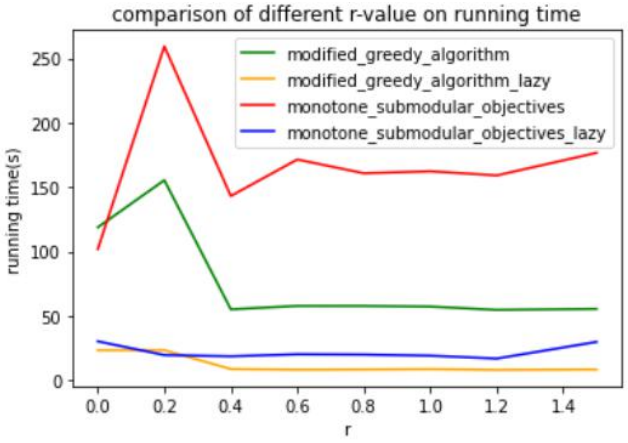


图 1

超参数取值 r 从 0 变化到 1.5 过程中，可以清楚看到，modified_greedy_algorithm 运行时间是要比基于 Monotone Submodular Objectives 算法要低的，并且经过 lazy 改进后的两种算法运行时间都比较低，可以说通过 lazy 使得算法效率改进了若干倍，同时当 r 取值为 0.4-1.4 之间时，四种算法的运行效率基本都处于稳定状态。

2）覆盖率

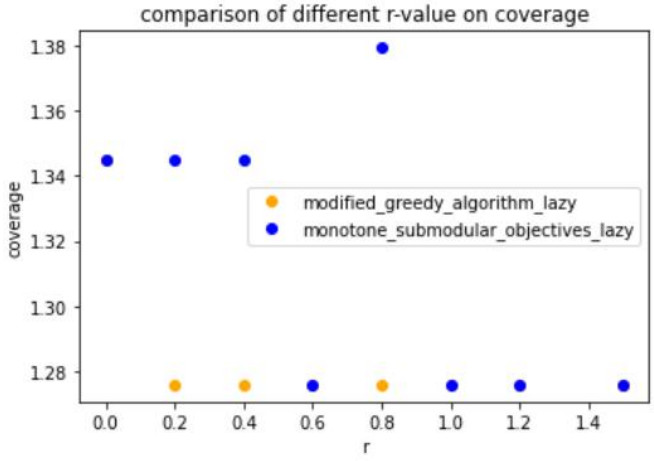


图 2

超参数取值 r 从 0 变化到 1.5 过程中，可以看到，基于 Monotone Submodular Objectives 算法在覆盖率的表现上基本都要比 modified-greedy-algorithm 好。

综上 1) 2)，r=0.8 的时候表现整体最优，故确定 scaling factor r 为 0.8。

3. 选定 r=0.8 后，用后面四种方法对 400 篇文章生成 100 篇摘要，进而比较随着文章长度的变化，算法运行时间以及覆盖率的变化

1）运行时间：

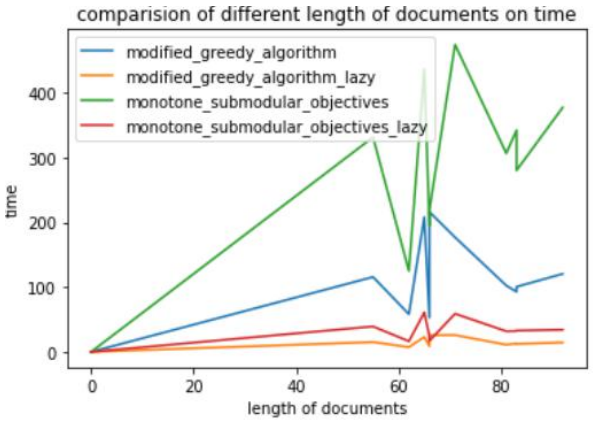


图 3

可以看到，随着文章的长度逐渐增加，没有经过 lazy 改进的两种算法的波动幅度都很大，并且波动曲线也比较相似，都是在 length of documents 约为 60 达到极小点，然后又猛增，在 length of documents 约为 65 时达到极大点，又下降，尽管经过 lazy 改进的算法也是在这个区间有所波动，但是都比较平稳，同时不管有没有通过 lazy 改进，基于 Monotone Submodular Objectives 算法的运行时间都是要高于 modified-greedy-algorithm 的。

3）覆盖率

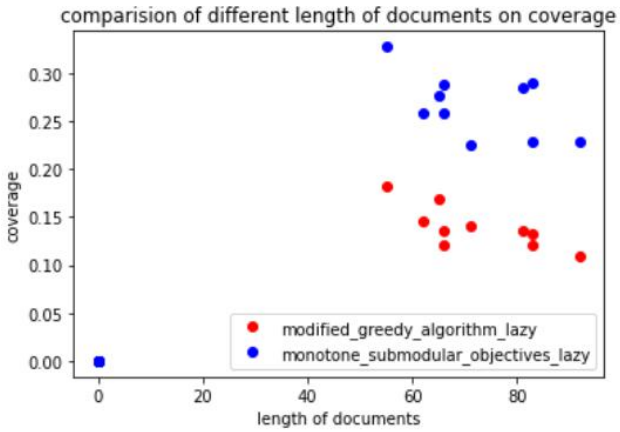


图 4

可以看到，随着文章的长度逐渐增加，经过 lazy 改进的基于 Monotone Submodular Objectives 算法的覆盖率都是要高于 modified-greedy-algorithm 的，而且甚至高达 2 倍。

综合考虑以上 2 个评价指标，经过 lazy 改进的基于 Monotone Submodular Objectives 算法表现最优，不仅有着较高的覆盖率而且运行效率也较高。同时也可以看到 lazy 改进的强大作用。

五、 结论（对使用的方法可能存在的不足进行分析，以及未来可能的研究方向进行讨论）

在本次实验爬取了正点财经 10 页的内容，采用多种子模函数最大化算法及其改进提取了 100 篇摘要，共六种方法实现文本摘要。在算法效率与覆盖率上经过 lazy 改进的基于 Monotone Submodular Objectives 算法表现最好。事实上，关于新的子模函数以及优化算法还有很多，那么针对什么样的应用究竟选择何种子模函数和何种优化方法就显得非常有意义了。此外，关于文本摘要问题也可以采用其他方法，如常见的 TextRank 算法，那么它同子模优化的方法孰优孰劣也是一个较好的研究方向。

[参考文献]

[1] Hui Lin, Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT), 2010.

[2] Hui Lin, Jeff Bilmes. A Class of Submodular Functions for Document Summarization. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 510-520, Jun. 2011.

[3] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, Natalie Glance, Cost-effective Outbreak Detection in Networks. Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining. 420-429, August 2007.

附录：

导入需要的库

```
import pandas as pd
import numpy as np
import requests
from lxml import etree
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.cluster import KMeans
import time
import os
import warnings
warnings.filterwarnings('ignore')
```

爬取正点财经的研究报告界面

```
data = []
#爬取10个页面
for i in range(1, 11):
    url = 'http://www.zdcj.net/reportlist/newreport_{}.html'.format(i)
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
    MyPage = requests.get(url, headers=headers)
    MyPage.encoding="utf-8"
    dom = etree.HTML(MyPage.text)
    listnews = dom.xpath('//a[@title][@target="_blank"]/@href')
    count=0
    i=0
    while i<len(listnews):
        count=count+1
        news=listnews[i]
        news1=listnews[i+1]
        news2=listnews[i+2]
        news3=listnews[i+3]
        #第一个
        headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        MyPage = requests.get(news, headers=headers)
        dom = etree.HTML(MyPage.text.encode("latin1").decode("GBK"))
        items = dom.xpath('//div[@class="v_c_box"]/p/text()')
        items = "".join(items)
        headers1 = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        #第二个
        MyPage1 = requests.get(news1, headers=headers1)
        dom1 = etree.HTML(MyPage1.text.encode("latin1").decode("GBK"))
        items1 = dom1.xpath('//div[@class="v_c_box"]/p/text()')
        items1 = "".join(items1)
        #第三个
        headers2 = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        MyPage2 = requests.get(news2, headers=headers2)
        dom2 = etree.HTML(MyPage2.text.encode("latin1").decode("GBK"))
```

```
        items2 = dom2.xpath('//div[@class="v_c_box"]/p/text()')
        items2 = "".join(items2)
        #第四个
        headers3 = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64)"}
        MyPage3 = requests.get(news3, headers=headers3)
        dom3 = etree.HTML(MyPage3.text.encode("latin1").decode("GBK"))
        items3 = dom3.xpath('//div[@class="v_c_box"]/p/text()')
        items3 = "".join(items3)
        items=items+items1
        if items:
            data.append(items)
        i=i+4
#爬取的内容放到data.csv
df = pd.DataFrame({'content': data}, columns=['content'])
df.to_csv('./data.csv')
```

读取数据

```
txt=pd.read_csv('data.csv')
```

进行数据预处理

```
txt['content'] = txt['content'].str.replace('\r','')
txt['content'] = txt['content'].str.replace('\n','')
txt['content'] = txt['content'].str.replace('\t','')
txt['content']=txt['content'].str.replace('  ','')
txt['content']=txt['content'].str.strip()
txt = txt['content'].str.replace(' ','')
```

```
#得到句子
sent=txt.str.split('。|\s')
sent[0]=list(set(sent[0]))
```

查看前四篇文章的所有句子，接下来就要从这四篇文章采用6种不同的方法生成一篇摘要

```
sent[0]
```

```
['',
 '直升机需求不断扩大，军民融合市场空间广阔军改后陆军航空兵扩编和海军陆战队建设，我国军用直升机需求不断扩大',
 '根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间',
 '公司作为直升机行业龙头，将受益于整个行业的向好发展',
 '我国直升机制造业的主力军，自主研发助推新品交付与民品替代中直主营航空产品直升机，现有核心产品包括直8、直9、直11、AC311、AC312、AC313等型号直升机及零部件，近年来已逐步完成主要产品型号的更新换代',
 'EQT股权转让完成，新战略股东有望带来新变化',
 '由于其投资产品已到退出期限，本次股权转让退出实属正常的退出安排',
 '股权转让落地解除减持预期对股价的压制',
 '首次覆盖太阳纸业并给予公司“买入”评级',
 '点评：',
```

'市场担忧减持对公司股价造成较大波动，预计该担忧部分影响了公司的股价表现',
'参考美国陆军直升机的编制情况，我们预测未来我国陆军直升机需求达到1500架,市场空间约为1800亿元；根据《通用航空“十三五”规划》，我们预测国内民用直升机到2020年约有600架的缺口，公司民机业务有望提升',
'盈利预测及估值',
'其中泽星投资持有公司24.78%的股权，买方则由春华资本和方源资本各持股50%',
'公司业绩稳健成长，经营逐步改善，本次股权转让引入知名PE，有望为公司经营带来积极变化，维持“买入”评级',
'本次公司公告了哈飞集团、中直有限的简化版财务报表，按照同一控制下企业合并的规则，2016-2018年股东中航科工的营收将分别增厚184.82/149.44/191.30亿元，净利润将分别增厚1.76/6.14/2.08亿元，归母净利润将分别增厚2.07/3.90/1.79亿元',
'在民机科研方面，公司深入开展“一机首飞”、“一机论证”、“两机取证”',
'据东方网10月1日消息，在我国国庆70周年阅兵表演中直20首次公开亮相',
'公司有望凭借老挝和广西项目的区位优势和林浆纸一体化带来的成本优势，扩大在南方市场文化纸和包装纸的份额',
'EQT是公司上市前就参与投资的战略PE股东，投资期间利用其独特的“实业加速模型”帮公司持续扩张并改善治理结构，也为公司能成功做大上市做出的重要贡献',
'事件：',
'根据公司关于日常关联交易的公告，公司2019年度预计上限同比增长26%左右，我们预计公司2019-2021年实现营收162.48/206.71/264.68亿元，同比增长24.35%、27.23%、28.05%，实现归母净利润分别为6.51/8.61/11.46亿元，同比增长27.50%、32.31%、33.12%，对应EPS分别为1.10/1.46/1.94元/股，目前股价对应2019-2021年PE分别为38.81/29.33/22.04倍，给予“增持”评级',
'自2017年政府加强外废进口管制以来，我国外废进口量大幅下降',
'风险提示',
'本次集团相关直升机资产收购完成后，上市公司营收与净利润将大幅增厚（1）中直有限（中航直升机）：主要从事直升机研制、营销、服务、运营，主营业务包括直升机制造业和通航运营业，可研制和批量生产多种型号直升机和转包生产多种航空零部件',
'我们预计，公司特殊用户、民用直升机两方向将持续不断推进',
'这一短期难以弥补的原材料缺口有望催化国废价格上涨，带动纸价上行，加速领先和落后企业分化，利好太阳纸业等原料端自给率高的龙头企业',
'航空产品交付节奏慢于预期；资产整合进度低于预期',
'公司发布公告，EQT（LeaderHoldingLimited，简称“卖方”）将转让其持有的泽星投资100%股权给SonataCompanyLimited（简称“买方”），总对价为5.57亿美元',
'目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间',
'买方为知名的投资机构春华资本和方源资本，其合伙人团队多为高盛或淡马锡等知名机构就职背景的华人知名投资人，也有如像阿里爸爸、咕咚、快手、蚂蚁金服等知名投资案例，兼具国际化视野和本土经验，预计有望为公司带来更多外部合作机会或医药新零售模式升级的机会',
'预计到2020-2021年，废纸名义缺口量每年可能超过1000万吨',
'自建速生林、采购外废制浆造纸提升原材料渠道自主可控程度，降低原材料价格波动对成本的冲击',
'我们认为，此举彰显了中航工业集团对军民用直升机业务长期发展的信心，从中直股份业绩可以看出直升机业务正处于高景气周期，随着集团资产整合的加速推进，直升机研制及部分总装资产未来有望注入上市公司体内，增厚业绩同时完善直升机资产完整性，打造我国直升机龙头上市公司',
'股东科工拟发股收购中直有限100%、哈飞10.21%、昌飞47.96%股权11月28日公司公告，股东科工拟发行15亿股内资股（向集团发行12.5亿股，向天保投资发行2.5亿股），每股价格4.19元港币（折合3.79元人民币）收购以下3项资产，收购对价为56.88亿元',
'行业整合出清，龙头逆势扩张',
'林浆纸一体化构建核心优势，盈利能力稳中有进',
'提升盈利中枢和稳定性，形成并扩大市场竞争中的产品质量和价格优势，更好地分享供给侧改革带来的机遇',
'废纸稀缺性渐显，或成业绩增长催化剂',
'此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持',
'风险提示：',
'在民机销售方面，各型直升机销售稳中有进，各主打产品逐步站稳市场，努力推进“一带一路市场的开拓”',
'盈利预测与评级：维持营收增速预测，19-21年公司营收为164.6/204.1/249.0亿元，毛利率为13.95%/14.12%/14.3%，净利润为6.19/7.65/9.30亿元，EPS为1.05/1.3/1.58元，按照10月24日收盘价44.07元，对应PE为41.96/33.98/27.92x',
'老挝和广西项目是太阳打开南方市场的战略重镇',
'投资要点',

```
 '11月28日公司公告，公司股东中航科工拟发行内资股收购中直有限100%股权、哈飞集团10.21%股权、昌飞集团47.96%股权',
 '并购整合不顺商誉减值风险；新建并购速度不达预期风险',
 '股权转让引入知名PE，有望带来积极变化，维持“买入”评级维持19~21年EPS预测为1.87/2.42/3.05元，现价对应19~21年PE为39/30/24倍',
 '经过多年建设和发展，公司在山东建立起较为完善的原料采购和纸品销售渠道',
 '风险提示：公司重大资产重组进度低于预期，军工订单低于预期',
 '本次股权转让落地有助于解除减持预期对公司股价上涨的压制，形成利好',
 '（3）昌飞集团：我国直升机科研生产基地，具备研制和批产多品种、多系列、多型号直升机和航空零部件生产的能力，主要产品有直8、直10、直11等军用直升机，AC310、AC311、AC313等民用直升机',
 '公司依托自身丰富的生产运营经验，以前瞻性眼光布局海外基地，十年深耕终有所成',
 '（1）航空工业集团、天保投资合计持有的中直有限100%股权（注：中直有限直接&间接持有中直股份16.03%股份）；',
 '在环保和市场力量共同作用下，大批“小规模、高污染、高能耗”纸企被淘汰出局',
 '直升机资产加速整合，公司中长期受益',
 '文化纸和包装纸价格出现大幅下滑；老挝和广西项目进展速度不及预期',
 '我国造纸工业自2011年起进入“增速下行、优化结构、提高质量”的新阶段',
 '交易完成后，中航科工合计持有中直股份50.80%股份',
 '本次交易属于上市公司主要股东之间的股权变动，符合国家军民融合发展和航空强国战略，中航工业集团整合旗下直升机业务，整体注入到中航科工港股上市平台',
 '（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权',
 '本次交易完成前，科工持股公司34.77%股份；收购完成后，科工合计持有中直股份50.80%股份，实现了50%以上的绝对控股',
 '等工作，2018全年累计试飞5280架次/8849小时',
 '（2）哈飞集团：主要从事多款直升机和系列飞机的研发、制造与销售，产品体系包括直9、直19军用直升机，AC312、AC352民用直升机和运12E、运12F固定翼飞机',
 '首次覆盖给予“买入”评级，合理估值区间12.60~14.50元预测公司2019-2021FY的营业收入分别为225.3、238.0、273.9亿元，同比增速分别为3.5%、5.6%和15.1%；净利润分别为19.7、22.1和25.3亿元，同比增速分别为-11.8%、12.2%和14.0%；BPS分别为5.54、6.30和7.18元/股；对应PB分别为1.65、1.45和1.27倍']
```

方法一：课本原始爬山算法，子模函数为集合覆盖函数

1. 提取关键词，得到关键词集合W

```
txt['content'] = txt['content'].str.replace(r'[^\u4e00-\u9fa5]','')
txt['content'] = txt['content'].str.replace(' ','')
import jieba
txt['content'] = txt['content'].apply(lambda x:' '.join(jieba.cut(x)))
```

读入停用词

```
file = open("stop.txt",encoding='utf-8')
stopword = file.readlines()
stopwords = [word.strip() for word in stopword]
tfidf1 = TfidfVectorizer(stop_words=stopwords, ngram_range=
(1,1)).fit(txt['content'])
tfidf2 = TfidfVectorizer(stop_words=stopwords, ngram_range=(1,1))
```

```
x_res = tfidf2.fit_transform(txt['content'])
feature = {v: k for k, v in tfidf1.vocabulary_.items()}
W_matrix = np.argsort(-x_res.todense())[:, :150]
df = pd.DataFrame(np.vectorize(feature.get)(W_matrix))
feature_array = np.array(tfidf1.get_feature_names())
tfidf_sort = np.argsort(re.toarray()).flatten()[::-1]
W_ = feature_array[tfidf_sort][:150]
W=[]
for i in W_:
    W.append(i)
print(W)
```

```
['游戏', '上周', '服务器', '车型', '新车', '车展', '奇缘', '冰雪', '万吨', '动画电影',
'票房', '传媒', '上涨', '行业', '合资', '库存', '板块', '自主', '车企', '特斯拉', '多
款', '衍生品', '广州', '期货价格', '提供商', '钢材', '下跌', '智能网', '市场', '环比',
'百度', '三七', '互娱', '新车型', '元吨', '覆盖', '铁矿石', '新能源', '全新', '估值', '本
周', '龙头', '主题', '特材钢', '千吨', '基础架构', '异构', '姊妹', '钢联', '全面铺开',
'长安', '水基', '阳泉', '系列', '文化', '采购', '加速', '卖点', '超市', '内地', '明后',
'久立', '百花齐放', '粉矿', '广汽', '交易量', '社会', '机会', '中信', '成交量', '国产',
'产品', '报告', '研高纳', '冷轧', '近几年', '热轧', '搭载', '上映', '福特', '当代', '重
磅', '呈现', '螺纹钢', '下降', '业绩', '主流', '钢铁', '明年', '线材', '换代', '完美',
'数据中心', '中厚板', '交互', '万平方米', '诸多', '比特', '需求', '助力', '同比', '内容',
'建议', '亮相', '出版', '两条', '产业化', '影视', '一个月', '公司', '高性能', '电影', '螺
纹', '澳洲', '电动车', '明显改善', '特钢', '提振', '美元', '时代', '情绪', '涨跌幅', '厂
商', '集团', '汽车', '钢厂', '涨幅', '数据', '投资', '创造', '东方', '国外', '增长', '配
置', '亿美元', '焦炭', '开工', '股份', '二期', '把握', '利好', '跌幅', '头部', '热点',
'同步', '世界', '销售额', '稳健', '效应', '能力']
```

2. 将文档里的所有包含关键词的句子汇总成集合S

```
S=[]
# for i in range(100):
for k in sent[0]:
    if k!='':
        S.append(k)
print(len(S))
```

```
64
```

```
cover_list=[]
cover_list_number=[]
for se in S:
    word=jieba.cut(se)
    words=[]
    word_have=[]
    for wordt in word:
            words.append(wordt)
    #words现在包含的是这个句子中分的词
    for i in words:
        if i in W:
            word_have.append(i)
    word_have=list(set(word_have))
```

```
    if word_have!=[]:
        for i in range(0,len(word_have)):
            word_have[i]=w.index(word_have[i])

    cover_list.append(word_have)
    cover_list_number.append(len(word_have))
```

```
cover_list[2]
```

```
[8]
```

```
cover_list_number[2]
```

```
1
```

```
true_false_list1=[]
true_false_list2=[]
for i in range(0,len(w)):
    true_false_list1.append("false")
    true_false_list2.append("false")
import copy
cover_list_number1=copy.deepcopy(cover_list_number)
cover_list_number2=copy.deepcopy(cover_list_number)
```

3. 原始爬山算法

```
#原始的爬山算法：
import time
k=10
#计时
start =time.perf_counter()
#选择的句子
list_selected1=[]
origin_cover_list_number1=copy.deepcopy(cover_list_number1)
current_selected=np.argmax(cover_list_number1)
while len(list_selected1) <k:
    for j in range(0, len(sentence)):
        for i in range(0,origin_cover_list_number1[j]):
            #每次遍历查找使得增加的关键词最多的句子
            #如果在true_false_list中已经有了这个关键词，那么这个句子中包含的关键词个数就
要-1，因为是要看新增的关键词个数，为此还要维护一个origin_cover_list_number
            if true_false_list1[cover_list[j][i]]==True:
                cover_list_number1[j]=cover_list_number1[j]-1

    current_selected=np.argmax(cover_list_number1)
```

```
    #选出该句子后，将cover_list_number中的对应位置置为0，这样下一次current_selected就不会选择它了
    cover_list_number1[current_selected]=0
    list_selected1.append(current_selected)
    for i in range(0,origin_cover_list_number1[current_selected]):
        true_false_list1[cover_list[current_selected][i]]=True
end = time.perf_counter()
es1=end-start
```

```
print("运行时间")
es1
```

```
运行时间
```

```
0.004040100000565872
```

```
summary=[]
for i in list_selected1:
    summary.append(sentence[i])
summary="。".join(summary)
summary=summary+"。"
print(summary)
```

```
我们认为，此举彰显了中航工业集团对军民用直升机业务长期发展的信心，从中直股份业绩可以看出直升机业务正处于高景气周期，随着集团资产整合的加速推进，直升机研制及部分总装资产未来有望注入上市公司体内，增厚业绩同时完善直升机资产完整性，打造我国直升机龙头上市公司。首次覆盖给予“买入”评级，合理估值区间12.60~14.50元预测公司2019-2021FY的营业收入分别为225.3、238.0、273.9亿元，同比增速分别为3.5%、5.6%和15.1%；净利润分别为19.7、22.1和25.3亿元，同比增速分别为-11.8%、12.2%和14.0%；BPS分别为5.54、6.30和7.18元/股；对应PB分别为1.65、1.45和1.27倍。（3）昌飞集团：我国直升机科研生产基地，具备研制和批产多品种、多系列、多型号直升机和航空零部件生产的能力，主要产品有直8、直10、直11等军用直升机，AC310、AC311、AC313等民用直升机。目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间。自2017年政府加强外废进口管制以来，我国外废进口量大幅下降。直升机需求不断扩大，军民融合市场空间广阔军改后陆军航空兵扩编和海军陆战队建设，我国军用直升机需求不断扩大。自建速生林、采购外废制浆造纸提升原材料渠道自主可控程度，降低原材料价格波动对成本的冲击。预计到2020-2021年，废纸名义缺口量每年可能超过1000万吨。投资要点。据东方网10月1日消息，在我国国庆70周年阅兵表演中直20首次公开亮相。
```

```
print("覆盖率")
cnt=0
for i in true_false_list1:
    if i==True:
        cnt=cnt+1
print(cnt/len(w))
```

```
覆盖率
0.15333333333333332
```

方法二：通过lazy改进课本原始爬山算法，子模函数为集合覆盖函数

```
import time

k=10
start =time.perf_counter()
list_selected2=[]
origin_cover_list_number2=copy.deepcopy(cover_list_number2)
current_selected=np.argmax(cover_list_number2)
while len(list_selected2) <k:
    while current_selected!=np.argmax(cover_list_number2):
        #此时最大的是句子2，那就先把它的和true_false_list比较，看此时的关键词个数
cover_list_number 有多少
        current_selected=np.argmax(cover_list_number2)
        #第一次做
        if
cover_list_number2[current_selected]==origin_cover_list_number2[current_selected
]:
            for i in range(0,origin_cover_list_number2[current_selected]):
                if
cover_list_number2[current_selected]==origin_cover_list_number2[current_selected
]:
                    if true_false_list2[cover_list[current_selected][i]]==True:

 cover_list_number2[current_selected]=cover_list_number2[current_selected]-1

    #得到句子2实际覆盖了20   cover_list_number [current_selected]=20  3
    cover_list_number2[current_selected]=-1
    list_selected2.append(current_selected)
    for i in range(0,origin_cover_list_number2[current_selected]):
        true_false_list2[cover_list[current_selected][i]]=True
end = time.perf_counter()
es2=end-start
```

```
print("运行时间")
es2
```

```
运行时间
```

```
0.0019093999999313382
```

```
print("覆盖率")
cnt=0
for i in true_false_list2:
    if i==True:
        cnt=cnt+1
print(cnt/len(W))
```

```
覆盖率
0.15333333333333332
```

```
summary2=[]
for i in list_selected2:
    summary2.append(sentence[i])
summary2="。".join(summary2)
summary2=summary2+"。"
print(summary2)
```

```
我们认为，此举彰显了中航工业集团对军民用直升机业务长期发展的信心，从中直股份业绩可以看出直升机业务正处于高景气周期，随着集团资产整合的加速推进，直升机研制及部分总装资产未来有望注入上市公司体内，增厚业绩同时完善直升机资产完整性，打造我国直升机龙头上市公司。首次覆盖给予“买入”评级，合理估值区间12.60~14.50元预测公司2019-2021FY的营业收入分别为225.3、238.0、273.9亿元，同比增速分别为3.5%、5.6%和15.1%；净利润分别为19.7、22.1和25.3亿元，同比增速分别为-11.8%、12.2%和14.0%；BPS分别为5.54、6.30和7.18元/股；对应PB分别为1.65、1.45和1.27倍。（3）昌飞集团：我国直升机科研生产基地，具备研制和批产多品种、多系列、多型号直升机和航空零部件生产的能力，主要产品有直8、直10、直11等军用直升机，AC310、AC311、AC313等民用直升机。（2）哈飞集团：主要从事多款直升机和系列飞机的研发、制造与销售，产品体系包括直9、直19军用直升机，AC312、AC352民用直升机和运12E、运12F固定翼飞机。公司发布公告，EQT（LeaderHoldingLimited，简称“卖方”）将转让其持有的泽星投资100%股权给SonataCompanyLimited（简称“买方”），总对价为5.57亿美元。目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间。本次股权转让落地有助于解除减持预期对公司股价上涨的压制，形成利好。自2017年政府加强外废进口管制以来，我国外废进口量大幅下降。（1）航空工业集团、天保投资合计持有的中直有限100%股权（注：中直有限直接&间接持有中直股份16.03%股份）；。直升机需求不断扩大，军民融合市场空间广阔军改后陆军航空兵扩编和海军陆战队建设，我国军用直升机需求不断扩大。
```

可以看到，通过lazy使得算法的运行时间降低了，仅仅使用集合的覆盖函数作为子模函数去完成文本摘要事实上还没有考虑很多问题， 所以接下来主要是比较一下四种方法的效率。

方法三 modified-greedy-algorithm

1. 定义子模函数

```
# MMR V是原文句子集合
def MMR(V, S, λ=4):
    #如果摘要集合的元素个数为0，则最后的MMR值也为0
    if len(S) == 0:
        return 0
    U = list(range(V.shape[0]))
    S_R = U[:] #S_R是原文句子集合-已选的摘要句子S-R
    for s in S:
        S_R.remove(s)
    #从剩余的句子中去选择
    res = 0
    for i in S_R:
```

```
    #候选句子与当前已有的摘要集合的冗余度
        res +=cosine_similarity(V[i], V[S]).sum()
    if len(S) == 1: return res
    for i in S:
        S_i = S[:]
        S_i.remove(i)
        res -=λ *cosine_similarity(V[i], V[S_i]).sum()/2
    return res
```

2. modified_greedy_algorithm

```
def modified_greedy_algorithm(X, V, f, B, r):
    G = []#对应第一行伪代码 #候选集合G，先初始化为空集
    #原文句子集合V
    U = list(range(V.shape[0]))#对应第二行伪代码 #候选原文句子集合
    while len(U) != 0:#对应第三行伪代码
        L = []
        for u in U:
            L.append((f(V,G + [u]) - f(V,G)) / (X[[u]].sum()) ** r)
        k = U[np.argmax(L)]#对应第四行伪代码
        if X[G + [k]].sum() <= B and f(G + [k])- f(G) >= 0:
            G += [k]#摘要集加入该句子 #对应第五行伪代码
        U.remove(k)#候选集移除被选择的句子 #对应第六行伪代码
    #在跳出最外层循环后，并没有立刻返回结果
    f_v=[]
    for t in range(X.shape[0]):
        if X[[t]].sum() <B:
            f_v.append(f(V,[t]))
    v = np.argmax(f_v)
    if f(V,G) > f(V,[v]):
        return G
    else:
        return v
```

```
vec = CountVectorizer()
X = vec.fit_transform(sent[0])
# print(X.shape)
tfidf = TfidfTransformer(use_idf=True).fit(X)
X_tf = tfidf.transform(X)
# print(tf.shape)
```

```
start = time.time()
modified_greedy_algorithm_S = modified_greedy_algorithm(X,X_tf,MMR,50, 0.3)
modified_greedy_algorithm_t = time.time() - start
modified_greedy_algorithm_t
```

```
170.2492344379425
```

```
modified_greedy_algorithm_S
```

```
[0, 2, 47, 27, 6, 54, 62, 43, 3, 5, 7]
```

```
summary_1=[]
for i in modified_greedy_algorithm_S:
    summary_1.append(sent[0][i])
summary_1="。".join(summary_1)
summary_1=summary_1+"。"
print(summary_1)
```

。风险提示。（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权。首次覆盖太阳纸业并给予公司“买入”评级。交易完成后，中航科工合计持有中直股份50.80%股份。此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持。根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间。我国直升机制造业的主力军，自主研发助推新品交付与民品替代中直主营航空产品直升机，现有核心产品包括直8、直9、直11、AC311、AC312、AC313等型号直升机及零部件，近年来已逐步完成主要产品型号的更新换代。预计到2020-2021年，废纸名义缺口量每年可能超过1000万吨。市场担忧减持对公司股价造成较大波动，预计该担忧部分影响了公司的股价表现。公司发布公告，EQT（LeaderHoldingLimited，简称“卖方”）将转让其持有的泽星投资100%股权给SonataCompanyLimited（简称“买方”），总对价为5.57亿美元。。

方法四 lazy改进后的modified-greedy-algorithm

```
def modified_greedy_algorithm_lazy(X, V, f, B, r):
    G = []#对应第一行伪代码 #候选集合G，先初始化为空集
    #原文句子集合V
    U = list(range(V.shape[0]))#对应第二行伪代码 #候选原文句子集合
    score=[]
    for u in U:
        score.append(f(V,[u])/ X[[u]].sum() ** r)#得到增益分数
    while len(U) != 0:#对应第三行伪代码
        current_selected = np.argmax(score)
        #优先对最大的进行增益分数的更新计算
        score_ = (f(V,G + [U[current_selected]]) - f(V,G))/
X[[U[current_selected]]].sum() ** r
        score[current_selected] = score_
        idx =[]
        #第一次肯定要进入while循环
        while (current_selected not in idx) and (score < np.max(score)):
            idx.append(current_selected)
            current_selected = np.argmax(score)
            score_ = (f(V,G + [U[current_selected]]) - f(V,G))/
X[[U[current_selected]]].sum() ** r
            score[current_selected] = score_
        k = U[current_selected]
        score.remove(current_selected)
        if X[G + [k]].sum() <= B and f(G + [k])- f(G) >= 0:
            G += [k]#摘要集加入该句子 #对应第五行伪代码
        U.remove(k)#候选集移除被选择的句子 #对应第六行伪代码

    #在跳出最外层循环后，并没有立刻返回结果
    f_v=[]
    for t in range(X.shape[0]):
```

```
        if X[[t]].sum() <B:
            f_v.append(f(V,[t]))
    v = np.argmax(f_v)
    if f(V,G) > f(V,[v]):
        return G
    else:
        return v
```

```
start = time.time()
modified_greedy_algorithm_lazy_S = modified_greedy_algorithm_lazy(X,X_tf,MMR,50,
0.3)
modified_greedy_algorithm_lazy_t = time.time() - start
```

```
modified_greedy_algorithm_lazy_S
```

```
[0, 2, 47, 27, 6, 54, 62, 43, 3, 5, 7]
```

```
modified_greedy_algorithm_lazy_t
```

```
22.79802703857422
```

```
summary_2=[]
for i in modified_greedy_algorithm_lazy_S:
    summary_2.append(sent[0][i])
summary_2="。".join(summary_2)
summary_2=summary_2+"。"
print(summary_2)
```

```
。风险提示。（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权。首次覆盖太阳纸业并给予公司“买入”评级。交易完成后，中航科工合计持有中直股份50.80%股份。此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持。根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间。我国直升机制造业的主力军，自主研发助推新品交付与民品替代中直主营航空产品直升机，现有核心产品包括直8、直9、直11、AC311、AC312、AC313等型号直升机及零部件，近年来已逐步完成主要产品型号的更新换代。预计到2020-2021年，废纸名义缺口量每年可能超过1000万吨。市场担忧减持对公司股价造成较大波动，预计该担忧部分影响了公司的股价表现。公司发布公告，EQT（LeaderHoldingLimited，简称“卖方”）将转让其持有的泽星投资100%股权给SonataCompanyLimited（简称“买方”），总对价为5.57亿美元。
```

方法五 基于Monotone Submodular Objectives算法

1. 定义子模函数

```
#使用kmeans将文档中的句子集合分成num个类，为f中的奖励多样性项做准备的
def generate_cluster(X_tf, X, num_):
```

```
    num_=int(0.2 * len(range(X.shape[0])))
    m = X_tf.toarray()
    model = KMeans(num=num_)
    model = kmeans.fit(m)
    labels = kmeans.predict(m)
    cluster = {k: [] for k in range(num_)}
    for i in range(len(labels)):
        cluster[labels[i]].append(i)
    return cluster
# f=L(s)+λ*R(s) 最大化它
def monotone_submodulars(V, S, cluster,λ=4):
    if len(S)==0:
        return 0
    res = 0
    #计算第一部分覆盖率
    for i in V:
        res1 = cosine_similarity(i.reshape(1, -1), V[S]).sum()#原文中待选的句子i与候
选摘要集S的相似度
        res2 =cosine_similarity(i.reshape(1, -1), V).sum()
        res += min(res1, res2)
    res_ = 0
    #计算第二部分冗余性
    for k in range(len(cluster)):
        S_and_Pk = list(set(S) & set(cluster[k]))
        res1 = 0
        for j in S_and_Pk:
            res1 +=cosine_similarity(V[j], V).sum()
        res_ += np.sqrt(res1)
    return res + λ * res_
```

2. monotone_submodular_objectives

```
def monotone_submodular_objectives(X, V, cluster, f, B, r):
    G = []#对应第一行伪代码 #候选集合G，先初始化为空集
    #原文句子集合V
    U = list(range(V.shape[0]))#对应第二行伪代码 #候选原文句子集合
    while len(U) != 0:#对应第三行伪代码
        L = []
        for u in U:
            L.append((f(V,G + [u],cluster) - f(V,G,cluster)) / (X[[u]].sum()) **
r)
        k = U[np.argmax(L)]#对应第四行伪代码
        if X[G + [k]].sum() <= B and f(G + [k],cluster)- f(G,cluster) >= 0:
            G += [k]#摘要集加入该句子 #对应第五行伪代码
        U.remove(k)#候选集移除被选择的句子 #对应第六行伪代码
    #在跳出最外层循环后，并没有立刻返回结果
    f_v=[]
    for t in range(X.shape[0]):
        if X[[t]].sum() <B:
            f_v.append(f(V,[t],cluster))
    v = np.argmax(f_v)
    if f(V,G,cluster) > f(V,[v],cluster):
        return G
    else:
        return v
```

```
cluster=generate_cluster(X_tf, X,3)
start = time.time()
monotone_submodular_objectives_S =
monotone_submodular_objectives(X,X_tf,cluster,monotone_submodulars,50, 0.3)
monotone_submodular_objectives_t = time.time() - start
monotone_submodular_objectives_t
```

```
515.7717483043671
```

```
monotone_submodular_objectives_S
```

```
[0, 2, 25, 15, 18, 27, 47, 6, 54, 12, 23, 51, 64, 62, 38, 19, 55, 21, 22]
```

```
summary_3=[]
for i in monotone_submodular_objectives_S:
    summary_3.append(sent[0][i])
summary_3="。".join(summary_3)
summary_3=summary_3+"。"
print(summary_3)
```

```
。风险提示。风险提示：。点评：。事件：。首次覆盖太阳纸业并给予公司“买入”评级。（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权。交易完成后，中航科工合计持有中直股份50.80%股份。此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持。目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间。直升机资产加速整合，公司中长期受益。EQT股权转让完成，新战略股东有望带来新变化。我们预计，公司特殊用户、民用直升机两方向将持续不断推进。根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间。在民机销售方面，各型直升机销售稳中有进，各主打产品逐步站稳市场，努力推进“一带一路市场的开拓”。投资要点。风险提示：公司重大资产重组进度低于预期，军工订单低于预期。盈利预测及估值。老挝和广西项目是太阳打开南方市场的战略重镇。
```

方法六 使用lazy改进的基于Monotone Submodular Objectives算法

```
def monotone_submodular_objectives_lazy(X, V, cluster, f, B, r):
    G = []#对应第一行伪代码 #候选集合G，先初始化为空集
    #原文句子集合V
    U = list(range(V.shape[0]))#对应第二行伪代码 #候选原文句子集合
     score=[]
    for u in U:
        score.append(f(V,[u],cluster)/ X[[u]].sum() ** r)#得到增益分数
    while len(U) != 0:#对应第三行伪代码
        current_selected = np.argmax(score)
        #优先对最大的进行增益分数的更新计算
        score_ = (f(V,G + [U[current_selected],cluster]) - f(V,G,cluster))/
X[[U[current_selected]]].sum() ** r
```

```
        score[current_selected] = score_
        idx =[]
        #第一次肯定要进入while循环
        while (current_selected not in idx) and (score < np.max(score)):
            idx.append(current_selected)
            current_selected = np.argmax(score)
            score_ = (f(V,G + [U[current_selected]],cluster) - f(V,G,cluster))/
X[[U[current_selected]]].sum() ** r
            score[current_selected] = score_
        k = U[current_selected]
        score.remove(current_selected)
        if X[G + [k]].sum() <= B and f(G + [k])- f(G) >= 0:
            G += [k]#摘要集加入该句子 #对应第五行伪代码
        U.remove(k)#候选集移除被选择的句子 #对应第六行伪代码
    #在跳出最外层循环后，并没有立刻返回结果
    f_v=[]
    for t in range(X.shape[0]):
        if X[[t]].sum() <B:
            f_v.append(f(V,[t],cluster))
    v = np.argmax(f_v)
    if f(V,G,cluster) > f(V,[v],cluster):
        return G
    else:
        return v
```

```
start = time.time()
monotone_submodular_objectives_lazy_S =
monotone_submodular_objectives(X,X_tf,cluster,monotone_submodulars,50, 0.3)
monotone_submodular_objectives_lazy_t = time.time() - start
monotone_submodular_objectives_lazy_t
```

```
68.4156174659729
```

```
monotone_submodular_objectives_lazy_S
```

```
[0, 2, 25, 15, 18, 27, 47, 6, 54, 12, 23, 51, 64, 62, 38, 37, 55, 22, 19]
```

```
summary_4=[]
for i in monotone_submodular_objectives_lazy_S:
    summary_4.append(sent[0][i])
summary_4="。".join(summary_4)
summary_4=summary_4+"。"
print(summary_4)
```

。风险提示。风险提示：。点评：。事件：。首次覆盖太阳纸业并给予公司“买入”评级。（2）航空工业集团持有的哈飞集团10.21%股权；（3）航空工业集团持有的昌飞集团47.96%股权。交易完成后，中航科工合计持有中直股份50.80%股份。此前EQT通过泽星投资持股公司股份数高达24.78%，且公告过将全部减持。目前文化纸行业集中度已然较高，包装纸领域尚有较大整合空间。直升机资产加速整合，公司中长期受益。EQT股权转让完成，新战略股东有望带来新变化。我们预计，公司特殊用户、民用直升机两方向将持续不断推进。根据绝对/相对估值法，给予公司12.60~14.50元的合理估值区间。在民机销售方面，各型直升机销售稳中有进，各主打产品逐步站稳市场，努力推进“一带一路市场的开拓”。股权转让落地解除减持预期对股价的压制。风险提示：公司重大资产重组进度低于预期，军工订单低于预期。老挝和广西项目是太阳打开南方市场的战略重镇。投资要点。

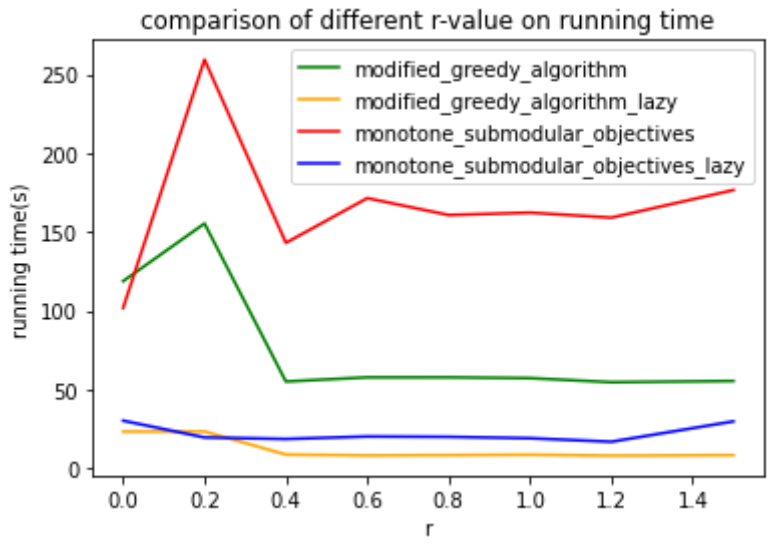
比较后面四种算法在不同r的情况下的运行时间及覆盖率：

```
r_list = [0, 0.2, 0.4, 0.6, 0.8, 1,1.2, 1.5]
run_time_modified=[]
modified_summary=[]

run_time_modified_lazy=[]
modified_lazy_summary=[]

run_time_monotone=[]
monotone_summary=[]

run_time_monotone_lazy=[]
monotone_lazy_summary=[]

for r in r_list:
    print("budget = 40, r ={}".format(r))
    start = time.time()
    modified_greedy_algorithm_S = modified_greedy_algorithm(X,X_tf,MMR,40, r)
    modified_greedy_algorithm_t = time.time() - start
    modified_summary.append(modified_greedy_algorithm_S)
    run_time_modified.append(modified_greedy_algorithm_t)

    start = time.time()
    modified_greedy_algorithm_lazy_S =
modified_greedy_algorithm_lazy(X,X_tf,MMR,40, r)
    modified_greedy_algorithm_lazy_t = time.time() - start
    run_time_modified_lazy.append(modified_greedy_algorithm_lazy_t)
    modified_lazy_summary.append(modified_greedy_algorithm_lazy_S)

    cluster=generate_cluster(X_tf, X,3)
    start = time.time()
    monotone_submodular_objectives_S =
monotone_submodular_objectives(X,X_tf,cluster,monotone_submodulars,40, r)
    monotone_submodular_objectives_t = time.time() - start
    run_time_monotone.append(monotone_submodular_objectives_t)
    monotone_summary.append(monotone_submodular_objectives_S)

    start = time.time()
    monotone_submodular_objectives_lazy_S =
monotone_submodular_objectives(X,X_tf,cluster,monotone_submodulars,40, r)
    monotone_submodular_objectives_lazy_t = time.time() - start
    run_time_monotone_lazy.append(monotone_submodular_objectives_lazy_t)
    monotone_lazy_summary.append(monotone_submodular_objectives_lazy_S)
```

```
budget = 40, r =0
budget = 40, r =0.2
budget = 40, r =0.4
budget = 40, r =0.6
budget = 40, r =0.8
budget = 40, r =1
budget = 40, r =1.2
budget = 40, r =1.5
```

比较算法效率 运行时间

```
import matplotlib.pyplot as plt
x_axix=[]
for i in rs:
    x_axix.append(i)
plt.title('comparison of different r-value on running time')
plt.plot(x_axix, run_time_modified, color='green',
label='modified_greedy_algorithm')
plt.plot(x_axix, run_time_modified_lazy,  color='orange',
label='modified_greedy_algorithm_lazy')
plt.plot(x_axix, run_time_monotone,  color='red',
label='monotone_submodular_objectives')
plt.plot(x_axix, run_time_monotone_lazy,  color='blue',
label='monotone_submodular_objectives_lazy')
plt.legend()
plt.xlabel('r')
plt.ylabel('running time(s)')
plt.show()
```

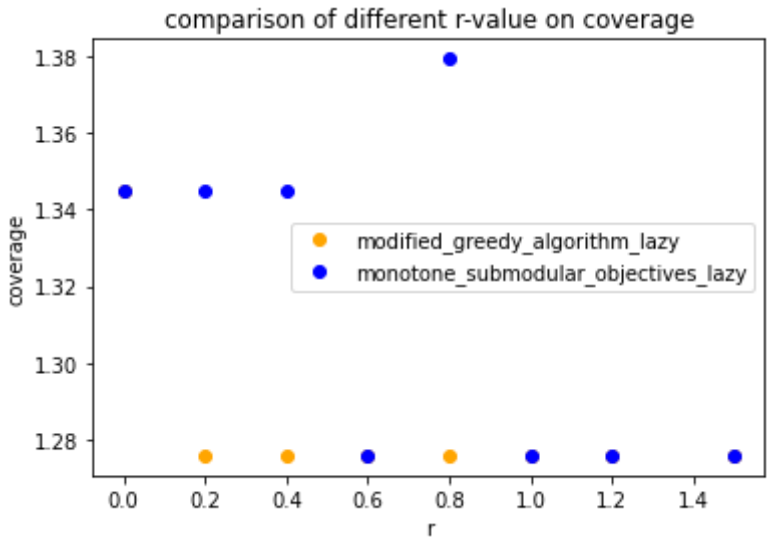


可以清楚看到，modified_greedy_algorithm运行时间是要比基于Monotone Submodular Objectives算法要低的，并且经过lazy改进后的两张算法运行时间都比较低，可以说通过lazy使得算法效率改进了若干倍，同时当r取值为0.4-1.4之间时，四种算法的运行效率基本处于稳定状态

比较覆盖率

```
total=X[X_tf.shape[0]-1].sum()
modified_lazy=[]
monotone_lazy=[]
```

```
for i in S_MMR_lazy_summary:
    x_MMR_lazy.append(X[i].sum()/total)
for i in sub_lazy_summary:
    x_sub_lazy.append(X[i].sum()/total)
plt.title('comparison of different r-value on coverage')
plt.plot(x_axix, modified_lazy,  'ro',color='orange',
 label='modified_greedy_algorithm_lazy')
plt.plot(x_axix, monotone_lazy, 'ro', color='blue',
 label='monotone_submodular_objectives_lazy')
plt.legend()
plt.xlabel('r')
plt.ylabel('coverage')
plt.show()
```



可以看到，基于Monotone Submodular Objectives算法在覆盖率的表现上基本都要比modified-greedy-algorithm好

r=0.8的时候表现整体最优

这仅仅是针对第一页的前4篇文章生成的摘要，现在用后面四种方法对400篇文章生成100篇摘要，进而比较随着文章长度的变化，算法运行时间的变化以及覆盖率的变化

```
def
generate_summary(passages,r,modified_greedy_algorithm,modified_greedy_algorithm_
lazy,monotone_submodular_objectives,monotone_submodular_objectives_lazy,generate
_cluster):
    vec = CountVectorizer()
    X = vec.fit_transform(passages)
    tf_transformer = TfidfTransformer(use_idf=True).fit(X)
    X_tf = tf_transformer.transform(X)

    start = time.time()
    modified_greedy_algorithm_S = modified_greedy_algorithm(X,X_tf,MMR,40, r)
    modified_greedy_algorithm_t = time.time() - start

    start = time.time()
```

```
    modified_greedy_algorithm_lazy_S =
modified_greedy_algorithm_lazy(X,X_tf,MMR,40, r)
    modified_greedy_algorithm_lazy_t = time.time() - start

    cluster=generate_cluster(X_tf, X,3)
    start = time.time()
    monotone_submodular_objectives_S =
monotone_submodular_objectives(X,X_tf,cluster,monotone_submodulars,40, r)
    monotone_submodular_objectives_t = time.time() - start

    start = time.time()
    monotone_submodular_objectives_lazy_S =
monotone_submodular_objectives(X,X_tf,cluster,monotone_submodulars,40, r)
    monotone_submodular_objectives_lazy_t = time.time() - start
    return X_tf.shape[0],modified_greedy_algorithm_t,
modified_greedy_algorithm_lazy_t, monotone_submodular_objectives_t,
monotone_submodular_objectives_lazy_t,len(modified_greedy_algorithm_lazy_S)/
X_tf.shape[0],len(monotone_submodular_objectives_lazy_S)/X_tf.shape[0]
```

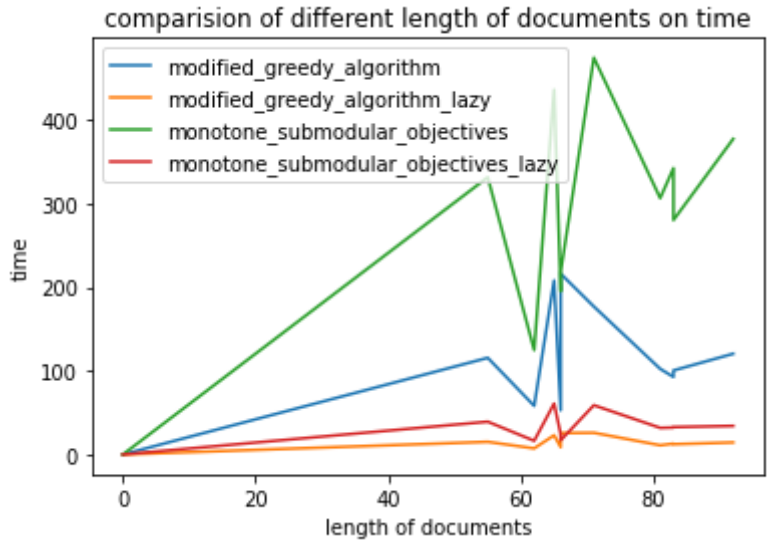
```
import copy
passages=copy.deepcopy(sent)
column = ['length_of_doc','modified_greedy_algorithm_t',
'modified_greedy_algorithm_lazy_t','monotone_submodular_objectives_t',
'monotone_submodular_objectives_lazy_t','modified_lazy_coverage','monotone_lazy_
coverage']
result = pd.DataFrame(np.zeros((len(sent), len(column))),columns=column)
t=0
for i in range(100):
    passages[i]=list(set(passages[i]))

 res=generate_summary(passages[i],0.8,modified_greedy_algorithm,modified_greedy_
algorithm_lazy,monotone_submodular_objectives,monotone_submodular_objectives_laz
y,generate_cluster)
    for j in range(len(column)):
        result[names[j]].iloc[t] = res[j]
    t=t+1
result[1].to_csv('result.csv')
#results.to_csv('result.csv')
```

```
res = pd.read_csv("result.csv", sep = ",")
```

在不同的文章长度下的运行效率：

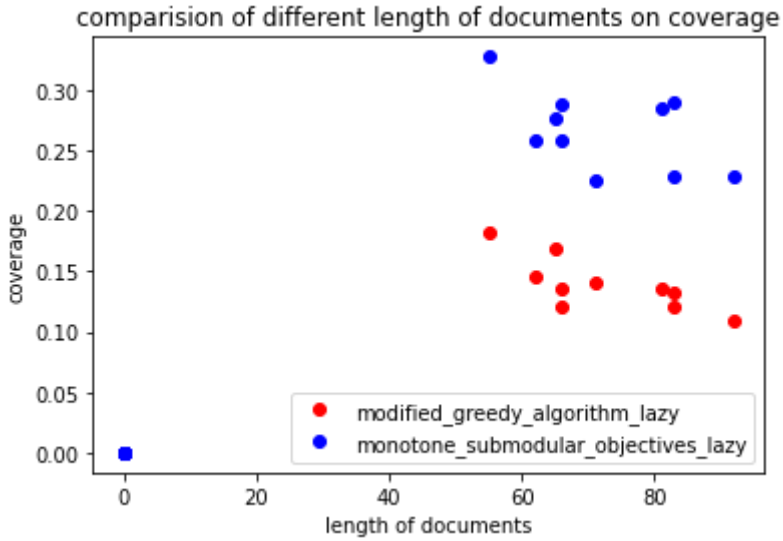
```
length = np.array(res["length_of_doc"])
length_ = res["length_of_doc"][np.argsort(length)[:-1]]
modified = res["modified_greedy_algorithm_t"][np.argsort(length)[:-1]]
modified_lazy = res["modified_greedy_algorithm_t_lazy"][np.argsort(length)[:-1]]
monotone_lazy = res["monotone_submodular_objectives_t_lazy"][np.argsort(length)
[:-1]]
monotone = res["monotone_submodular_objectives_t"][np.argsort(length)[:-1]]

plt.title("comparision of different length of documents on time")
plt.xlabel("length of documents")
plt.ylabel("time")
plt.plot(length_, modified, label = "modified_greedy_algorithm")
```

```
plt.plot(length_, modified_lazy, label = "modified_greedy_algorithm_lazy")
plt.plot(length_, monotone, label = "monotone_submodular_objectives")
plt.plot(length_, monotone_lazy, label = "monotone_submodular_objectives_lazy")
plt.legend()
plt.show()
```



在不同的文章长度下的覆盖率：

```
length = np.array(res["length_of_doc"])
length_ = res["length_of_doc"][np.argsort(length)[:-1]]
modified_c = res["modified_lazy_coverage"][np.argsort(length)[:-1]]
monotone_c = res["monotone_lazy_coverage"][np.argsort(length)[:-1]]

plt.title("comparision of different length of documents on coverage")
plt.xlabel("length of documents")
plt.ylabel("coverage")
plt.plot(length_, modified_c,  'ro',label = "modified_greedy_algorithm_lazy")
plt.plot(length_, monotone_c, 'bo', label =
"monotone_submodular_objectives_lazy")
plt.legend()
plt.show()
```


comparision of different length of documents on coverage
coverage
length of documents
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
20
40
60
80
modified_greedy_algorithm_lazy
monotone_submodular_objectives_lazy