数据采集与预处理

一、 数据采集

这段代码实现了从新浪新闻网站爬取不同栏目的新闻链接，然后并行地进行请求和解析，最后将解析得到的新闻标题、URL 和正文内容保存到 Excel 文件中

1.首先导入需要的模块

```
import time
import sys
import os
import requests
import pandas as pd
from bs4 import BeautifulSoup
from multiprocessing import Pool
```

2．定义不同新闻栏目及其对应的链接

```
news_dict = {
    '国内':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2510
&k=&num=50&page={}',
    '国际':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2511
&k=&num=50&page={}',
   '娱乐':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153
&lid=2513&k=&num=50&page={}',
    '军事':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2514
&k=&num=50&page={}',
    '科技':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2515
&k=&num=50&page={}',
```

```
    '财经':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2516
&k=&num=50&page={}',
    '股票':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2517
&k=&num=50&page={}',
    '金融':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2509
&k=&num=50&page={}',
   '体育':
'https://feed.mix.sina.com.cn/api/roll/get?pageid=153&lid=2517
&k=&num=50&page={}'
}

}
```

3. 获取新闻链接列表的函数

```
def get_URL(url):
   #发送请求获取页面内容，并将返回的 json 数据库解析为字典
    page = requests.get(url, headers=headers).json()
    links = []
    #遍历解析得到数据，获取新闻的链接并添加到列表中
    for j in range(50):
        urls = page['result']['data'][j]['url']
        links.append(urls)
    return links
```

4. 解析新闻内容的函数

```
def parse(url):
    #发送请求获取新闻页面内容
    res = requests.get(url)
    res.encoding = 'utf-8'
    soup = BeautifulSoup(res.text, 'html.parser')
    #提取新闻标题
    title = soup.select(".main-title")[0].text
    article_content = ""
    #提取新闻正文内容
    article = soup.select('.article p')[:-1] # 末端的消息来源不
需要
    for p in article:
        article_content += p.text.strip()

    return {
        'url': url,
```

```
        'title': title,
        'content': article_content
    }
```

5. 将数据写入 excel 文件的函数

```
def write_to_excel(data):
    #将数据转换为 DataFrame 对象
    df = pd.DataFrame(data)
    root = ".//newsCollection//"
    if not os.path.exists(root):
        os.mkdir(root) #创建目录
        print('目录创建成功')
    path = os.path.join(root, "news_collection.xlsx")
    try:
        df.to_excel(path, index=False)#将数据写入 excel 文件
        print("---数据已成功写入 news_collection.xlsx 文件---")
    except IOError:
        print('抱歉，写入失败')
```

6.主函数进行对 headers 和对每个 url 的处理

```
def main():
    global headers  #定义全局变量
    #这里对 headers 进行伪装
    headers = {'User-Agent':
               'Mozilla/5.0 (Windows NT 10.0; WOW64) '
               'AppleWebKit/537.36 (KHTML, like Gecko) '
               'Chrome/55.0.2883.87 Safari/537.36'}
    pool = Pool(8) # 创建一个进程池
    all_records = []
   #遍历不同新闻栏目及其对应的链接
    for name, url in news_dict.items():
        print(f"开始处理 {name} 栏目...")
        link_list = []
        try:
            for i in range(1, 51):  #遍历不同页面，获取每个页面的
新闻链接，并添加到链接列表中
                links = get_URL(url.format(i))
                link_list.extend(links)
        except:
            print(f"{name} 栏目链接获取失败")
            continue
        else:
            print(f"{name} 栏目链接已经全部获取")
```

```
        res_list = []
        for url in link_list:
            res = pool.apply_async(func=parse, args=(url,))
            res_list.append(res)

        record_list = []
        #使用进程池并发送请求解析新闻内容
        count = 0
        for res in res_list:
            count += 1
            try:
                result = res.get(timeout=10)
                print(f'{name} 栏目第 {count} 个链接获取成功')
            except Exception as e:
                print(f'{name} 栏目第 {count} 个链接获取失败:',
str(e))
                continue
            record_list.append(result)

        all_records.extend(record_list)
        print(f'{name} 栏目处理完成\n')
        write_to_excel(all_records)
    pool.close()
    pool.join()
```

7. 执行程序

```
if __name__ == "__main__":
    sys.setrecursionlimit(100000) # 设置默认递归深度
    main()
```

二、 数据清洗

1. 导入需要的模块

```
import pandas as pd
import re
```

2. 进行数据清洗

```
def clean_data(text):
    # 数据清洗：去除特殊字符和多余空格
    cleaned_text = re.sub(r"[^\w\s]", "", str(text))
    cleaned_text = re.sub(r"\s+", " ", cleaned_text)
    return cleaned_text
```

3. 进行数据脱敏

```
def anonymize_data(text):
    # 数据脱敏：将关键信息替换为占位符
    anonymized_text = re.sub(r"\d{4}-\d{2}-\d{2}", "XXXX-XX-XX", text)
    anonymized_text = re.sub(r"\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-
Z|a-z]{2,}\b", "XXX@XXX.com", anonymized_text)
    return anonymized_text
```

4. 进行对文件的读取

```
# 读取 Excel 文件
df = pd.read_excel("news_collection.xlsx")
```

5. 进行数据预处理

```
# 数据预处理
df['title'] = df['title'].apply(clean_data)
df['content'] = df['content'].apply(clean_data)
```

6. 打印处理后的数据

```
# 打印处理后的数据
print(df.head())
```

7. 保存 news_collection_processed.xlsx

```
# 将处理后的数据写入新的 Excel 文件
df.to_excel("news_collection_processed.xlsx", index=False)
```

三、 总结

通过使用 Pandas 和正则表达 式对新闻内容进行处理去除特殊字符和多余空格将关键信息替换为占位符最后，处理后的数据被保存为一个 Excel 文件， 以便进行后续的分析和可视化操作，。整个过程旨在清理和规范，使其更适合进一步的数据分析工作。