









GPT 辅助综述编写 Python 代码使用说明

数据皮皮侠

- 1、将自己在官网上申请的 token 输入 **sk_token.txt** 文件中

| 名称 | 类型 | 压缩大小 | 密码保护 |
|---|-------------------|------|------|
|  ChatGPT_outline.py | JetBrains PyCharm | 2 KB | 否 |
|  ChatGPT_review.py | JetBrains PyCharm | 2 KB | 否 |
|  crawl_code.py | JetBrains PyCharm | 5 KB | 否 |
|  cURL.txt | 文本文档 | 4 KB | 否 |
|  get_crawl.py | JetBrains PyCharm | 1 KB | 否 |
|  get_valuable_article.py | JetBrains PyCharm | 1 KB | 否 |
|  sk_token.txt | 文本文档 | 0 KB | 否 |
|  使用说明.txt | 文本文档 | 1 KB | 否 |

- 2、将目标网页的 cURL 复制到 **cURL.txt** 中

| 名称 | 类型 | 压缩大小 | 密码保护 |
|---|-------------------|------|------|
|  ChatGPT_outline.py | JetBrains PyCharm | 2 KB | 否 |
|  ChatGPT_review.py | JetBrains PyCharm | 2 KB | 否 |
|  crawl_code.py | JetBrains PyCharm | 5 KB | 否 |
|  cURL.txt | 文本文档 | 4 KB | 否 |
|  get_crawl.py | JetBrains PyCharm | 1 KB | 否 |
|  get_valuable_article.py | JetBrains PyCharm | 1 KB | 否 |
|  sk_token.txt | 文本文档 | 0 KB | 否 |
|  使用说明.txt | 文本文档 | 1 KB | 否 |

- 3、运行 get_crawl.py 获得 crawl.py 文件

- 4、运行 crawl.py 文件即可获得爬取的数据。对结果进行爬取后即可获得文献信息

- 5、如果不想自己处理数据，可以使用我们预先准备好的 crawl_code.py 代码获取文献摘要数据。这个代码会默认在当前目录下生成两个主题命名的文件夹，并且将结果保存为 result.csv

| 名称 | 类型 | 压缩大小 | 密码保护 |
|---|-------------------|------|------|
|  ChatGPT_outline.py | JetBrains PyCharm | 2 KB | 否 |
|  ChatGPT_review.py | JetBrains PyCharm | 2 KB | 否 |
|  crawl_code.py | JetBrains PyCharm | 5 KB | 否 |
|  cURL.txt | 文本文档 | 4 KB | 否 |
|  get_crawl.py | JetBrains PyCharm | 1 KB | 否 |
|  get_valuable_article.py | JetBrains PyCharm | 1 KB | 否 |
|  sk_token.txt | 文本文档 | 0 KB | 否 |
|  使用说明.txt | 文本文档 | 1 KB | 否 |

```
356 print(f"正在爬取第{page}页...\n")
357 res = self.get_other_page(page)
358 self.get_msg(res, page)
359 print(f"\n第{page}页获取成功!\n")
360
361 def save_as_csv(self, df_dic, save_name='result.csv'):
362     field_names = df_dic.keys()
363     rows = zip(*df_dic.values())
364     file_path = f'./{self.topic_1}+{self.topic_2}'
365     if not os.path.exists(file_path):
366         os.mkdir(file_path)
367     save_path = f"{file_path}/{save_name}"
368     with open(f'{save_path}', 'w', newline='', encoding='utf-8') as csvfile:
369         writer = csv.writer(csvfile)
370         writer.writerow(field_names)
371         writer.writerows(rows)
372
373 if __name__ == '__main__':
374     topic1 = '数字金融'
375     topic2 = '企业创新'
376     Cnki = CnkiSpider(topic1, topic2)
377
378     df_dic = Cnki.crawl_paper_data()
379     print("开始导出文件.....")
380     Cnki.save_as_csv(df_dic)
381
```

修改路径的地方

修改主题的地方

6、运行 get_valuable_article.py 文件可以筛选有价值的文献。如果想修改筛选的规则可以自行修改。这个文件会自动读取当前目录下
(可以使用数据皮皮侠官网的 GPT 辅助修改 <http://www.ppmandata.cn/chatgpt>)

```
1 > import ...
4
5 # 1. 读取CSV文件
6 topic1 = "数字金融"
7 topic2 = "企业创新"
8 folder_path = f'./{topic1}+{topic2}/'
9 csv_file = os.path.join(folder_path, 'result.csv')
10 df = pd.read_csv(csv_file)
11
12 # 2. 格式化日期列
13 df['发表时间'] = pd.to_datetime(df['发表时间'])
14
15 # 3. 添加新列 '年份', 并按照发表年份分组
16 df['年份'] = df['发表时间'].dt.year
17 grouped = df.groupby('年份')
18
```

修改主题和路径的地方

```
18
19 # 4. 处理每个年份分组的文章
20 result = pd.DataFrame()
21 for year, group in grouped:
22     # (1) 计算综合指标
23     group['综合指标'] = 0.7 * group['下载频次'] + 0.3 * group['被引频次']
24
25     # (2) 最近三个月内的文章加分
26     three_months_ago = datetime.now() - timedelta(days=90)
27     group.loc[group['发表时间'] >= three_months_ago, '附加分'] = 100
28     group['附加分'].fillna(0, inplace=True)
29
30     # (3) 计算评分并排序
31     group['评分'] = group['综合指标'] + group['附加分']
32     group.sort_values(by='评分', ascending=False, inplace=True)
33
34     # (4) 选择排名前5%的文章
35     num_articles = max(int(len(group) * 0.1), 1)
36     selected_articles = group.head(num_articles)
37     result = pd.concat([result, selected_articles])
38
39     # 打印每个年份选择的的文章数量
40     print(f'年份: {year}, 选择的的文章数量: {len(selected_articles)}')
41
42 # 打印总的文章数量
43 print(f'总的文章数量: {len(result)}')
44
45 # 5. 将结果保存到新的CSV文件中
46 output_file = os.path.join(folder_path, '筛选结果.csv')
47 result.to_csv(output_file, index=False)
```

修改筛选规则的地方

7、运行 ChatGPT_outline.py 获得综述大纲

```
16 temperature=0, # 此参数为控制答案的随机程度, 为0一般总是会得到相同的答案
17 )
18 return response.choices[0].message["content"]
19
20
21 topic_1 = '数字金融'
22 topic_2 = '企业创新'
23 df = pd.read_csv(f"./{topic_1}+{topic_2}/筛选结果.csv")
24 result = ''
25 for i, j, z, x, y in zip(df["作者"], df["所属期数"], df["文献来源"], df["篇名"], df["摘要"]):
26     result += f'"{i}", "{j}", "{z}", "{x}", "{y}"\n'
27
28 prompt = f"""
29 作为一位["{topic_1}"和"{topic_2}"]领域的专家, 您了解该领域所有的前沿知识、专业名词、模型方法、\
30 算法技术等必备的知识, 同时具有较好的文献阅读能力和信息总结能力。
31
32 我即将以csv文件的格式给您发送论文信息, 第一列是作者, 第二列是年(期), 第三列是刊名, 第四列是文献题名\
33 第五列是文章的摘要。
34
35 现在, 我希望根据这些文献的信息, 写一篇专业性较强的综述文章。请你从专业的角度出发, 为我的综述拟一份大纲。 \
36 要求大纲的格式标准, 涵盖一篇标准综述应该拥有的内容。并且注明各个部分需要参考的所有文献, 标注的方式为: (参考文献: 作者, 篇名)
37
38 最后, 为这篇综述文章取一个合适的标题。
39
40 论文信息:
41 ...
42 {result}
43 ...
44 """
```

修改主题的地方

8、运行 ChatGPT_review.py 获得综述

```
crawl_code.py ChatGPT_outline.py ChatGPT_review.py x
16     temperature=0, # 此参数为控制答案的随机程度，为0一般总是会得到相同的答案
17 )
18     return response.choices[0].message["content"]
19
20
21     topic_1 = '数字金融'
22     topic_2 = '企业创新'
23     with open(f'./{topic_1}+{topic_2}/大纲.md', 'r', encoding="utf-8") as file:
24         outline = file.readlines()
25
26     df = pd.read_csv(f'./{topic_1}+{topic_2}/筛选结果.csv')
27     result = ''
28     for i, j, z, x, y in zip(df["作者"], df["所属期数"], df["文献来源"], df["篇名"], df["摘要"]):
29         result += f'"{i}"', '{j}', '{z}', '{x}', '{y}'\n"
30
31
32     prompt = f"""
33     作为一位["{topic_1}"和"{topic_2}"]领域的专家，您了解该领域所有的前沿知识、专业名词、模型方法、\
34     算法技术等必备的知识，同时具有较好的文献阅读能力和信息总结能力。
35
36     我即将以csv文件的格式给您发送论文信息。第一列是作者，第二列是年（期），第三列是刊名，第四列是文献题名\
37     第五列是文章的摘要。
38
39     同时，我会给你发送一份综述的大纲，请你据此写一篇专业性强的综述。大纲中提到的参考文献都可以在论文信息中找到，\
40     请你根据大纲找到对应的参考文献，从中提取出综述需要的内容进行编写。要求在文中引用参考文章的观点或者结论时，需要在这个句\
41     子后面用（作者，年份）标注出来。同时，正文部分不能仅仅罗列其他文章的观点或结论，还需要你自己思考这个结论的意义，并做出\
42     评价，用经济学的原理和方法去解释这一结论或者观点。正文部分不少于2000字。
43
44     综述以markdown的格式展示。
```