需求：要实现一个基于深度学习的推荐系统，目标是通过输入一份答卷，输出需要改进提升的题型或生成一份新的答卷，并且包含题目的详细信息、答题情况、知识点和图谱

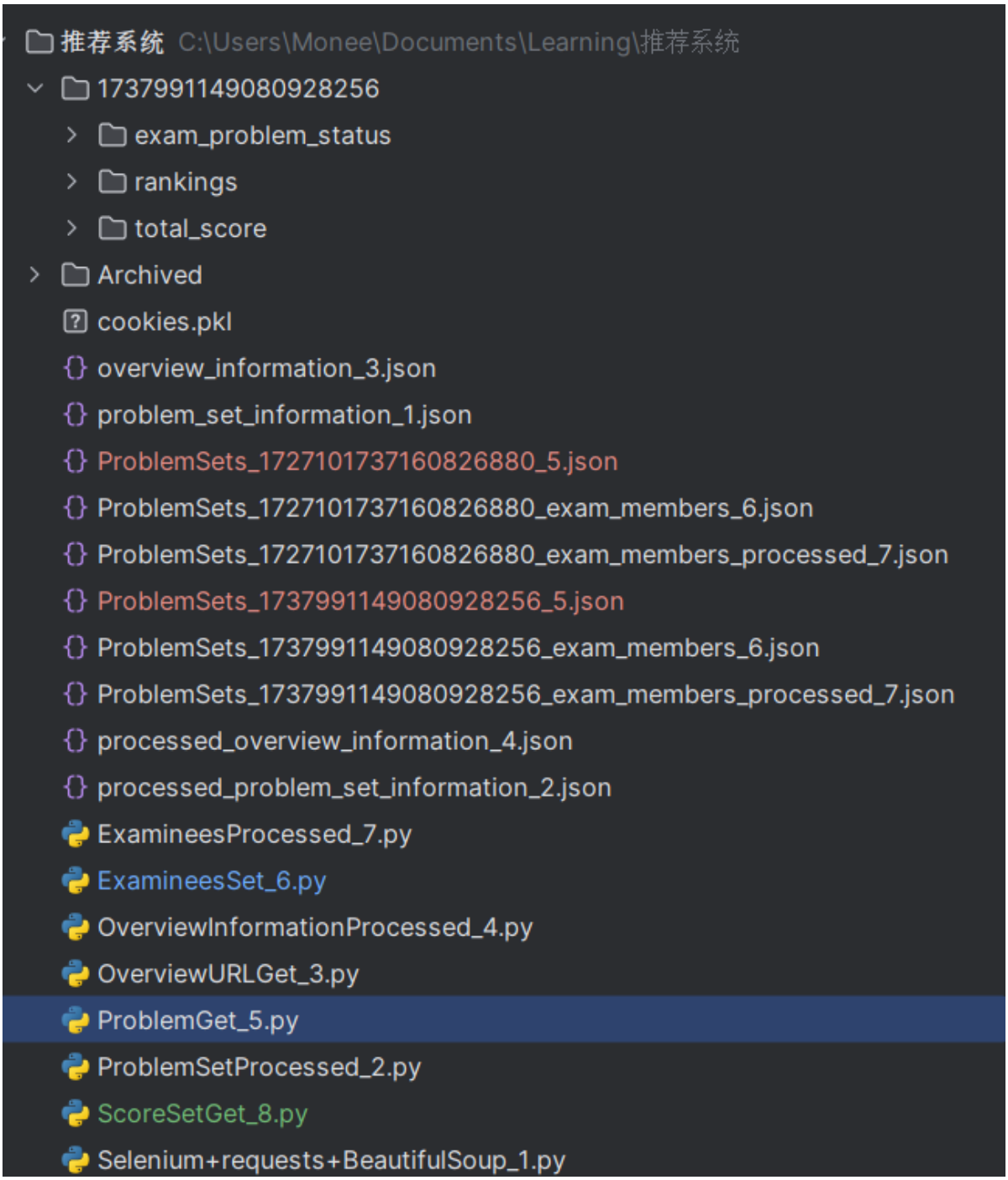
进展：已经完成数据的收集、部分数据的整理

寻找并确定使用DNN（深度神经网络）、LSTM、Transformer 通过处理和分析大量行为数据和内容信息，生成更精准和个性化的推荐。DNN需要向量形式的数据、生成推荐列表。LSTM长短时记忆也很合适，可以追踪一位同学的学习过程。Transformer也很适合全局捕捉依赖。

这也要求数据的整理需要画一些功夫，多整理几类，分别测试各个模型的效果

整理数据反而占用了整个项目中最多的时间。使用模型进行训练只是进行调包调库，进展会比较快。下一个难点大概在于网络的输出和实际需求的映射

规划：预计在暑假搭建出完整的框架、实现最初的需求目标。在这之后进行一些前端后端的设计，方便使用



1. 数据收集与预处理

登录网站：使用`Selenium`模拟浏览器登录过程。

爬取数据：使用`requests`和`BeautifulSoup`爬取所需的数据。

加载并整合数据：将题目集、题目详情、学生答题情况和得分等数据加载到DataFrame中。

关联数据：通过共同的标识符（如题目集ID、题目ID和学生ID）将数据关联起来。需要将题目集与相应的题目和分数对应起来，以便进行全面的数据分析和处理。为了实现这一点，您需要确保每个题目和学生的答题情况都能够关联到正确的题目集和分数。

数据存储：将爬取到的数据存储为JSON或CSV文件，方便后续处理。数据存储在三个JSON文件中：

- `questions.json`：包含C语言课程的题目信息。
- `student_answers.json`：包含学生的答题情况、提交情况和得分情况。
- `students.json`：包含学生的信息和排名情况。

👆是初步的想法

过程中遇到和解决的问题：

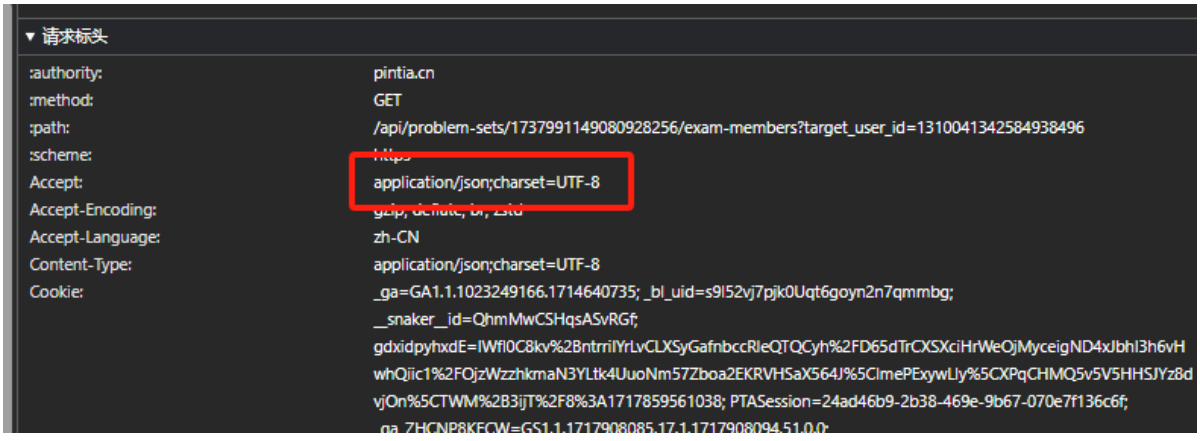
一、cookies和频繁爬取问题：

要爬取网站上的数据使用requests和BeautifulSoup，以及Selenium。因为是需要登录后才能访问数据，使用Selenium模拟浏览器可以规避反复提取cookies的问题。

在爬取过程中遇到些问题，比如爬取次数过多导致登录需要进行滑块拼图验证，使用了检测元素选择器判断是否有无滑块验证，有的话就暂停手动滑动。经过查阅信息，想要实现自动拼图验证是一个比较复杂的问题，涉及到机器视觉和OCR，但这不是本项目的重点，便使用了手动验证的方式。以节约时间。

二、爬取数据类型问题

在爬取题目集的时候，默认发送request得到的是protobuf数据，经过检查HTML的表头发现也确实是application/json;charset=UTF-8。后面是通过主动修改为发送application/json;charset=UTF-8请求获取到的json数据



三、爬取到的文件分类：

为了后续因缺少数据内容而延误进程，这里就能爬就爬。

1、从Overview开始递归爬取

首先通过元素选择，爬取所有题目集的base_url得到所有题目集的url

经过数据处理，提取去重要的problem_set_id和problem_set_name，方便后续批量递归爬取

接下来也是通过元素选择，获取试卷、考生、提交列表等等的baseurl


```
{
    "2023年C语言实验10": {
        "problems": "https://pintia.cn/problem-sets/1737991149080928256/problems",
        "examinees": "https://pintia.cn/problem-sets/1737991149080928256/examinees"
        "submissions": "https://pintia.cn/problem-sets/1737991149080928256/submissi
        "rankings": "https://pintia.cn/problem-sets/1737991149080928256/rankings",
        "subjective_problems": "https://pintia.cn/problem-sets/1737991149080928256/
        "detections": "https://pintia.cn/problem-sets/1737991149080928256/detection
        "exports": "https://pintia.cn/problem-sets/1737991149080928256/manage/expor
    },
    "浙大版《C语言程序设计（第4版）》第5章": {
        "problems": "https://pintia.cn/problem-sets/1727101737160826880/problems",
        "examinees": "https://pintia.cn/problem-sets/1727101737160826880/examinees"
        "submissions": "https://pintia.cn/problem-sets/1727101737160826880/submissi
        "rankings": "https://pintia.cn/problem-sets/1727101737160826880/rankings",
        "subjective_problems": "https://pintia.cn/problem-sets/1727101737160826880/
        "detections": "https://pintia.cn/problem-sets/1727101737160826880/detection
        "exports": "https://pintia.cn/problem-sets/1727101737160826880/manage/expor
    }
}
```

进行数据处理，分类打上tag


```
[
    {
        "problem_set_name": "2023年C语言实验10",
        "link_type": "problems",
        "link": "https://pintia.cn/problem-sets/1737991149080928256/problems",
        "problem_set_id": "1737991149080928256",
        "craw_link_problem": "https://pintia.cn/api/problem-sets/1737991149080928256/problems?limit=500
        "craw_link_examinees": "https://pintia.cn/api/problem-sets/1737991149080928256/exam-members"
    },
    {
        "problem_set_name": "2023年C语言实验10",
        "link_type": "examinees",
        "link": "https://pintia.cn/problem-sets/1737991149080928256/examinees",
        "problem_set_id": "1737991149080928256",
        "craw_link_problem": "https://pintia.cn/problem-sets/1737991149080928256/examinees",
        "craw_link_examinees": "https://pintia.cn/api/problem-sets/1737991149080928256/exam-members"
    },
    {
        "problem_set_name": "2023年C语言实验10",
        "link_type": "submissions",
        "link": "https://pintia.cn/problem-sets/1737991149080928256/submissions",
        "problem_set_id": "1737991149080928256",
        "craw_link_problem": "https://pintia.cn/problem-sets/1737991149080928256/submissions",
        "craw_link_examinees": "https://pintia.cn/api/problem-sets/1737991149080928256/exam-members"
    },
    {
        "problem_set_name": "2023年C语言实验10",
        "link_type": "rankings",
        "link": "https://pintia.cn/problem-sets/1737991149080928256/rankings",
        "problem_set_id": "1737991149080928256"
```


› problem_set_id

接下来依次爬取题目集、考生信息等等

```
C:\Users\Monee\.conda\envs\Common\python.exe C:\Users\Monee\Documents\Learning\推荐系统\ProblemGet_5.py
Already logged in!
Crawling data for problem set: 2023年C语言实验10, ID: 1737991149080928256
Crawling link: https://pintia.cn/api/problem-sets/1737991149080928256/problems?limit=500
Response status code: 200
Response headers: {'Date': 'Sun, 09 Jun 2024 05:35:37 GMT', 'Content-Type': 'application/json', 'Transfer-Encoding': 'chunked', 'Connection'
 'keep-alive', 'Vary': 'origin,access-control-request-method,access-control-request-headers,accept-encoding', 'Set-Cookie':
```

Current File

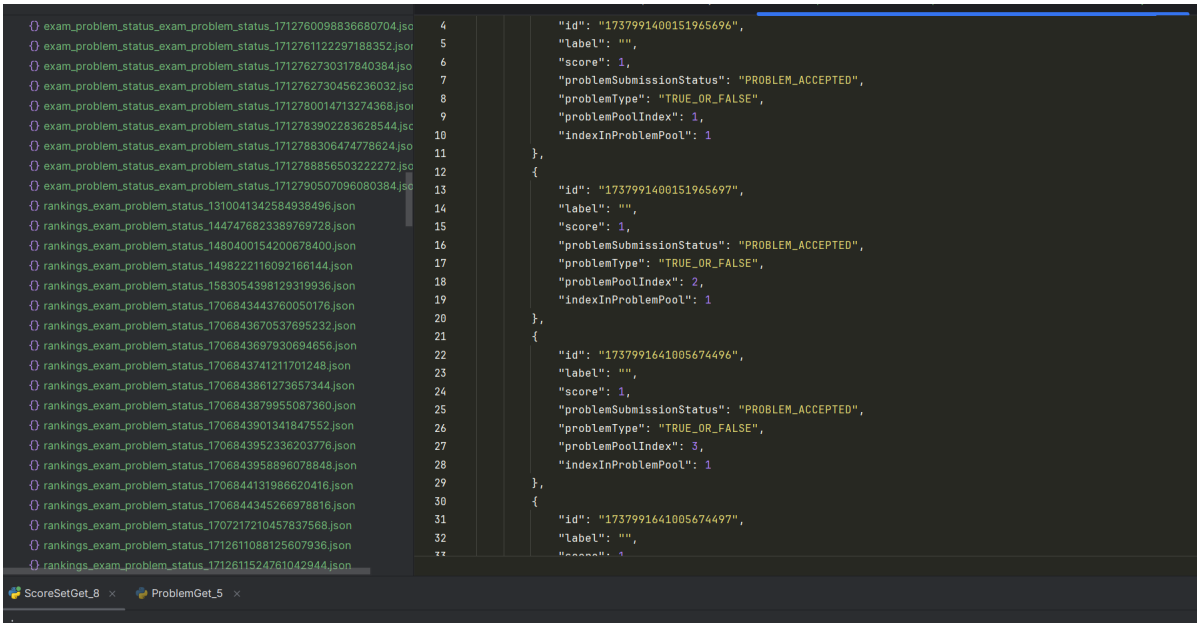
ProblemSets_1737991149080928256_5.json | ProblemSets_1737991149080928256_exam_members_6.json | ProblemSets_17271(


```
            "label": "",
            "score": 1,
            "problemPoolIndex": 1,
            "problemPoolCompositionCount": 1,
            "problemConfig": {
                "solutionVisible": false,
                "answerVisible": false
            },
            "deadline": "1970-01-01T00:00:00Z",
            "problemId": "1397877580286623744",
            "title": "链表是动态存储分配的数据结构。根据需要动态地开辟内存空间，可以比较自由方便地插入新元素（结点），支持随机访问，故使用
            "content": "链表是动态存储分配的数据结构。根据需要动态地开辟内存空间，可以比较自由方便地插入新元素（结点），支持随机访问，故使
            "type": "TRUE_OR_FALSE",
            "author": "张引",
            "difficulty": 1,
            "knowledgePointPaths": [
                {
                    "knowledgePoints": [
                        {
                            "id": "63",
                            "name": "C程序设计",
                            "isLeaf": false,
                            "enName": "C Programming"
                        },
                        {
                            "id": "82",
                            "name": "指针与结构",
                            "isLeaf": false,
                            "enName": ""
```


problemSetProblems › 26


```
[
    {
        "user": {
            "id": "1310041342584938496",
            "nickname": "王子鸣"
        },
        "studentUser": {
            "studentNumber": "202000800166",
            "name": "王子鸣",
            "id": "1307671266799845376"
        },
        "userGroupId": "1704067266309910528",
        "examId": "0"
    },
    {
        "user": {
            "id": "1446661705874468864",
            "nickname": "苟俊豪"
        },
        "studentUser": {
            "studentNumber": "202100800542",
            "name": "苟俊豪",
            "id": "1444906007248146443"
        },
        "userGroupId": "1704067266309910528",
        "examId": "0"
    },
    {
        "user": {
```

这里还没有对题目集进行数据处理，只对考生信息进行了整理，现在在看DNN和Transformer的文献，后续再确定整理处哪些内容



总路线图

1. **数据收集与预处理**
2. **特征工程**
3. **模型选择与训练**
4. **系统实现**
5. **评估与优化**

技术栈

1. **编程语言**：Python
2. **数据处理**：Pandas, NumPy
3. **深度学习框架**：TensorFlow, Keras 、 PyTorch
4. **数据库**：MySQL 或 PostgreSQL，用于存储和管理题目和学生的答题数据
5. **前端框架**：React 或 Vue.js，用于实现推荐系统的前端界面
6. **后端框架**：Flask 或 Django，用于实现系统的API
7. **可视化工具**：Matplotlib, Seaborn, D3.js（前端可视化）

详细步骤

1. 数据收集与预处理

- **数据收集**：从C语言课程的题目集、学生的答题情况、提交情况、得分情况、排名情况等收集数据。
- **数据清洗**：处理缺失值、异常值，规范化数据格式。
- **数据存储**：将清洗后的数据存储在关系数据库中，方便后续查询和分析。

2. 特征工程

- **特征提取**：从题目和答卷中提取特征，如题目类型、难度、知识点、答题时间、得分等。
- **特征选择**：选择对模型训练有用的特征，去除冗余特征。
- **特征编码**：对类别型特征进行编码，如One-Hot Encoding。

3. 模型选择与训练

- **模型选择**：选择合适的深度学习模型，如LSTM、Transformer或深度神经网络（DNN）。
- **模型训练**：使用训练数据训练模型，调整超参数，进行交叉验证。
- **模型评估**：使用测试数据评估模型性能，如准确率、召回率、F1-score等。

4. 系统实现

- **后端API**：使用Flask或Django实现后端API，处理数据请求，调用深度学习模型进行预测。
- **前端界面**：使用React或Vue.js实现前端界面，展示推荐结果和相关信息。
- **推荐算法**：实现基于模型的推荐算法，生成个性化的题目推荐或新的答卷。

5. 评估与优化

- **系统测试**：对系统进行全面测试，确保各模块功能正常。
- **模型优化**：根据模型评估结果，不断优化模型，提升推荐效果。
- **用户反馈**：收集用户反馈，迭代改进系统。

数据库设计

表结构

1. **题目表**（questions）
    - question_id
    - content
    - type
    - difficulty
    - knowledge_point
2. **学生答题表**（student_answers）
    - answer_id
    - student_id
    - question_id
    - answer
    - score
    - submit_time
3. **学生信息表**（students）
    - student_id
    - name

- class
- rank

前端设计

使用React或Vue.js创建用户界面，展示题目推荐结果、题目信息、学生的答题情况等。

后端API设计

使用Flask或Django创建后端API，处理前端请求，调用深度学习模型进行预测。

通过以上步骤和技术栈，逐步实现一个基于深度学习的推荐系统，满足输入答卷并输出改进建议或新答卷的需求。