Python大作业实验报告

组号： 19

组员1 姓名与学号：朱彦安20373384

组员2 姓名与学号：黄雨石20376156

组员3姓名与学号：吴中源20373354

一、实验任务：

任务介绍：

在线问题库平台，设计一个问题平台，帮助学生完成问题的自我评估。

任务要求：

- 您需要使用像 `TKinter` 和 `PyQt5` 这样的 `GUI` 库来设计 `GUI`，或者使用 `Python` 支持的前端和后端框架来进行 开发和设计。
- 该平台包括：自动识别 `PDF` 文件或图片的文本提取系统、文本结果编辑系统、自动或手动问题记录系统、问题生成系统、问题回答系统和结果分析系统。
- `GUI` 必须是可演示性的，并且可以根据您的功能添加其他功能，但它们必须易于使用

二、已完成任务

已完成基本实验要求：

- 能够通过上传 `PDF` 文件或问题的图片来识别相关的内容。
- 它有一个简单而漂亮的 `UI` 来编辑已识别的结果，以调整不准确的内容。
- 根据确定的内容手动或自动生成问题列表，然后将它们放入问题库。
- 问题生成系统，支持随机问题和手动问题选择。
- 该问答系统，方便用户自行测试。
- 结果反馈功能。

已完成可选要求

- 使用登录和注册功能来区分用户。
- 错题模式。记录用户的错误问题，方便用户查看，也可以根据错题进行设置。
- 喜爱（题目）功能。记录用户最喜欢的问题，问题可以根据最喜欢的问题进行设置。
- 背诵（题目）功能。直接提供答案，供用户背诵。
- 历史答案分析功能，可以分析用户之前的结果。

三、总体设计方案

总体概述及使用方法

上交的文件夹中内容如下：

```
19+朱彦安+2
|___code
|    |___ ...    ----------------> 代码与部分测试数据
|___MRC
|    |___ ...    ----------------> MRC训练代码
|_____大作业报告.pdf    ----------> 实验报告
|_____小航搜题.mp4    ------------> 讲解视频
```

`code` 部分代码主要分为**图形界面、数据库、识别三个模块**处理。由前端界面提供接口，识别模块进行处理后返回识别到的文本，数据库对数据进行增删查改等基本操作，大体结构可有如下目录树呈现，而以上三个模块分别位于 `ui`、`db`、`OCR_and_PDF` 文件夹中

```
code
|___db
|    |___ ...    ---------------------------> 数据库代码
|___db_data
|    |___ question.db    -------------------> 航概原始题库
|___img
|    |___ ...    ---------------------------> 图形界面所需或生成图片
|___MRC
|    |___model
```

```
|    |   |___ ...   ----------------------> 训练好的MRC模型
|    |___ MRC_request.py    -------------> 调用模型代码
|___OCR_and_PDF
|    |___ fileProcess.py    -------------> 识别部分代码
|____test_cases
|    |___contribute
|    |   |___ ...   ----------------------> 贡献题目测试数据
|    |___MRC
|    |   |___ ...   ----------------------> MRC测试数据
|    |___search
|        |___ ...   ----------------------> 搜索题目测试数据
|___ ui
|    |___ ...   -------------------------> 图形界面代码
|___ requirements.txt   -----------------> 环境要求
|___ run.py    --------------------------> 程序运行入口
|___ uninstall.py   ----------------------> 卸载本地数据
```

除模块分界明了，该工程运行方法也十分简单，在通过`requirements.txt`配置好环境后（`Python3.8`）：

- 运行`run.py`即可运行整个程序，**且第一次初始化由于需要加载原始题库会比较久一点**
- 运行`unstall.py`则会删除本地保存的所有用户数据

tips：

- 还需注意识别图片、`pdf`、`txt`文件的路径不得包含中文，且项目路径最好不要包含中文
- 无返回按钮界面，直接叉掉即可返回

具体设计介绍

识别处理

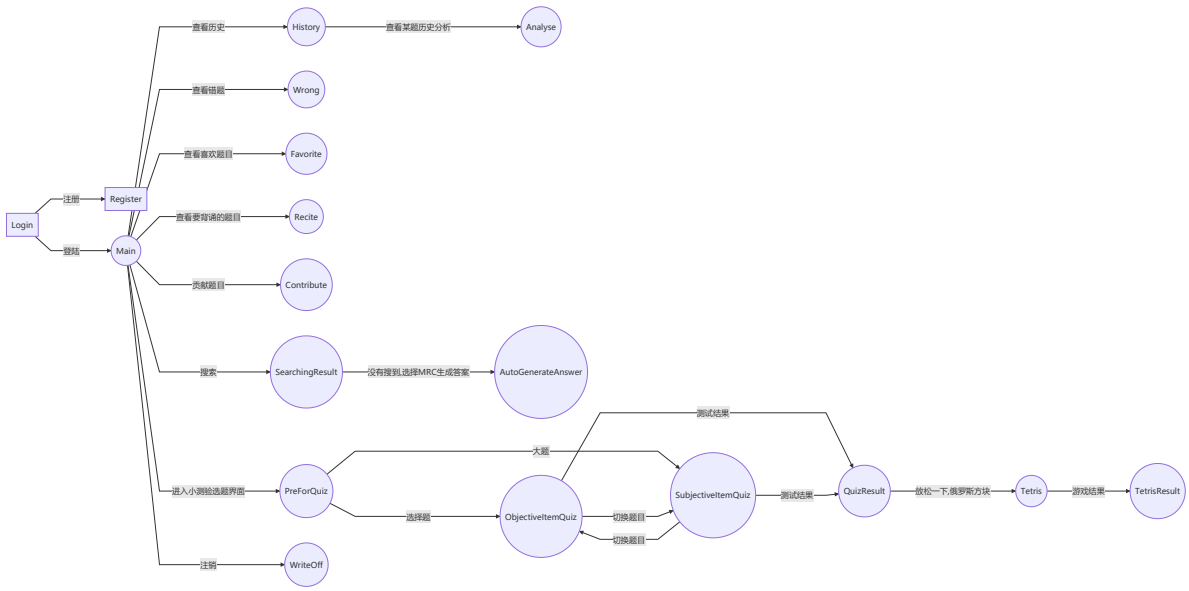
文件读入模块主要包括`PDF`文件读入、图片文件读入以及文本文件读入。从前端获取到文件路径后，该模块会通过后缀名来判断文件类型，并依据文件类型来选择具体的读入函数。

`PDF`文件读入主要依赖于`pdfminer`库。在读入`PDF`文件后，首先会加载一个`PDF`文档分析器，从文档中获取数据。同时需要加载`PDF`文档对象用来保存获取的数据，它和`PDFParser`是相互关联的。最后还需要加载一个`PDF`资源管理器，用来储存共享资源，比如字体或者图像。完成上述步骤后即可调用`pdfminer`库的内置函数进行文本读取

`OCR`识别模块的实现主要依赖于`openCV`以及`easyOCR`两个库。为了解决图片模糊、颜色对比度小等影响光学字符识别的问题，本模块首先将读入的图片从`RGB`图转换为灰度图，以便去除无关颜色的影响，提高识别准确率。随后，本模块借助`OpenCV`实现了对图像的形态学处理。由于拍摄清晰度低的问题可能导致字体边缘不连续，因此本模块使用了腐蚀操作，将图像的边界点消除，让图像沿着边界点向内收缩，使单个文字的边框相连接，由此减少无关笔迹对于题目主体的干扰。随后再进行膨胀操作，使字体的线条变粗。由于在腐蚀操作时无关笔迹等噪声已经从字体中被去除，因此此时进行膨胀可以有效还原字体本身的形态。完成上述操作后的图片将被进行全局阈值二值化处理。全局阈值二值化首先设定阈值`t`，根据阈值`t`将灰度图像分为两部分，灰度值大于`t`的像素点将被设为白色，小于阈值`t`的像素点将被置为黑色。考虑到题目字体一般为黑色，二值化可以有效提高题目文字与背景颜色的区分度，大幅提高`OCR`识别准确率。本模块使用了开源的`easyOCR`进行文本提取。`easyOCR`支持对中英文进行识别，同时支持`GPU`加速。预处理完毕后的图像即可交由`easyOCR`进行识别，并将识别出来的文本返回给用户界面。

UI

主要使用`PyQt5`库来完成，其中界面设计通过`Qt Designer`绘制然后由`pyuic`库导出代码`*.py`，即将不同的界面分为`*.py`和`*Controller.py`组成，其中`*.py`中为确定界面控件、布局等功能的类，`*Controller.py`中为操控界面控件、切换界面等逻辑以及提供与后端的数据库交互的接口，具体界面间关系由下图所示：



最后为了美观使用了一些图片与`qt_materia`库进行渲染，由于需要分析历史数据，于是还使用了`numpy`、`opencv-python`、`matplotlib`两个库进行数据处理与可视化

数据库

采用`sqlite`作为嵌入式数据库，方便移植在本地环境进行使用。经分析，本项目对数据库有如下需求，存储题库实现搜题和练习，存储用户信息实现用户系统，存储用户相关信息。因此设计了如下表单。

- **表`question_bank`：题库表**

字段	类型	备注
`question`	`varchar(1024)`	总问题
`answer`	`varchar(512)`	答案
`type`	`varchar(20)`	题目类型
`select_question`	`varchar(512)`	选择题题面
`A`	`varchar(256)`	选项A
`B`	`varchar(256)`	选项B
`C`	`varchar(256)`	选项C
`D`	`varchar(256)`	选项D
`provider`	`varchar(25)`	供题者
`question_id`	`integer`	题目id

- **表`user`：注册用户表**

字段	类型	备注
`name`	`varchar(25)`	用户名
`password`	`varchar(25)`	用户密码
`highest_score`	`int`	用户最高分

用户表的下属表：以用户名加对应字段组成表名

- **表 `_like`：用户喜欢的题**

字段	类型	备注
`question_id`	`int`	题目id
`Table_id`	`integer`	表id

- **表 `_recite`：用户要背诵的题**

字段	类型	备注
`question_id`	`int`	题目id
`Table_id`	`integer`	表id

- **表 `_history`：用户的历史记录**

字段	类型	备注
`question_id`	`int`	题目id
`table_id`	`integer`	表id
`answer`	`varchar(512)`	用户答案
`score`	`float`	得分
`practice_time`	`int`	练习次数
`history_type`	`varchar(20)`	历史类型

- **表 `_wrong`：用户的错题本**

字段	类型	备注
`question_id`	`int`	题目id
`table_id`	`integer`	表id
`last_answer`	`varchar(512)`	最后一次错误答案

设计了 `DB` 类对数据库操作进行封装管理，`user` 类对用户行为进行管理，`questionBank` 类对题库行为进行管理。

- **题库系统**

  通过单例模式创建的 `questionBank` 类来管理题库系统，提供了如下接口：

方法	备注
`add_question(self, question_and_answer, provider)`	添加主观题
`add_select_question(self, question_and_answer, select_question, provider)`	添加选择题
`get_question(self, question_id)`	通过题号获取问题
`search_question(self, s)`	搜索问题
`get_questions(self, num)`	根据题目数返回题单
`check_repeat(self, question)`	查找该题是否在题库中

- **用户系统**

  通过单例模式创建的`user`类来管理用户系统，提供了如下接口：

方法	备注
`register(self, name, password)`	注册用户
`login(self, name, password)`	用户登录
`exit(self)`	退出登录
`add_data(self, table, question_answer)`	向指定用户下属表单添加数据
`get_*_table(self)`	获取对应的用户下属表单
`get_question_infor(self, question_id, table_id)`	获取用户指定问题的相关信息
`get_history_score(self, question_id)`	获取用户对该题的练习历史
`get_question_number(self)`	获取用户各题单的题目数
`get_tetris_score(self)`	获取用户俄罗斯方块历史记录
`add_wrong_with_answer(self, question_id, my_answer)`	添加错题与错误答案
`add_test_history(self, question_id, my_answer, score)`	添加练习历史与练习情况
`delete_history(self, table_id)`	删除历史
`delete_data(self, table, question_id)`	删除表单数据
`destroy(self)`	注销当前账户

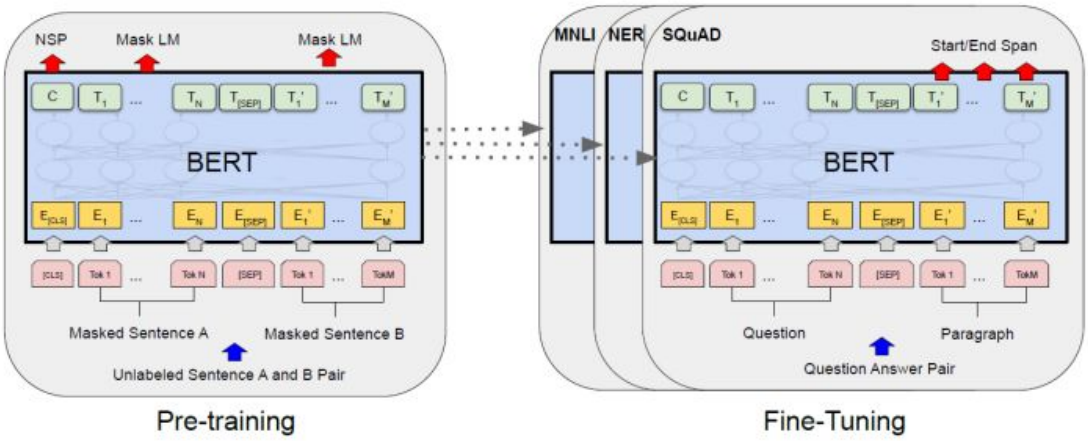
四、创新之处

选用原始航概题库

总所周知每个北航人都必须修航概，而我们的原始题库包含1507道航概习题，均由小组成员花大功夫收集以及手打而成，可以供大一修该课学生进行刷题、背题、备考等等，十分方便，具有很高的实用价值

新增机器阅读理解功能

- **灵感来源：**考虑到题库中不一定包含所有的题目，且有部分题目的答案可以直接从文本中得到，且我们刚好在平时的实验室打杂日常中恰好处理过这个问题（该部分代码由于未在本地跑过，而是使用配置好的服务器因此未列出环境要求），于是本项目增加了机器阅读理解功能，可以直接从已有文本中依据问题抽取答案。
- **整体概述：**为实现更加通用高效的机器阅读理解模型，我们收集和爬取了大量网络上开源的片段抽取式问答数据集和实体识别数据集，通过相似性匹配、掩码替换、模板生成、滑动窗口划分等方法获取120万条问答数据（包括文娱、社交、疫情、军事、医学、电商、新闻等领域），构建大型中文阅读预训练数据集。基于$RoBERTa$中文语言模型训练机器阅读理解问答模型，利用语言模型强大语义建模能力，实现信息精准提取和答案精准定位。完成2轮预训练后，在通用领域和垂直领域，长文本和实体文本级上的效果远好于现有网络开源的（`Huggingface`上）最好的阅读理解预训练模型。代码基于`Hugginface`上开源的chinese_pretrain_mrc_roberta_wwm_ext_large改进而成



- **训练数据：**
  - **数据来源：**

数据集名称	领域	类型
$DuReader$	百科	自由问答
$CMRC$ 2018	百科	片段抽取
$ChineseSquad$片段抽取	百科	片段抽取
中医数据	中医	片段抽取
$WebQA$	百科	片段抽取
疫情数据	疫情	片段抽取
军事数据	军事	片段抽取
中文法律阅读理解数据集$CJRC$	法律	片段抽取
繁体$DRCD$	百科	片段抽取
$CCKS$2020 金融事件抽取	金融	时间抽取
医疗科普知识阅读_$CCKS$ 2021：面向中文医疗科普知识的内容理解	医疗	自由问答+片段抽取
中文电子病历的医疗实体及事件抽取2021	医疗	待解析
地址结构化解析数据集—天池	地址	实体识别待解析
中文医疗信息处理挑战榜$CBLUE$	医疗	待解析
电商NER数据	电商	实体识别待解析
文娱NER数据	文娱	待解析
搜狐新闻数据$(SogouCS)-NER$任务数据	新闻	实体识别待解析
$CLUENER$2020 中文细粒度命名实体识别	百科	实体识别

数据集名称	领域	类型
$BosonNLP$命名实体	百科	实体识别
人民网$2004NER$模型	新闻	实体识别
影视—音乐—书籍实体标注数据	文娱	实体识别
微博命名实体识别数据集$WeiboNER$	社交	实体识别
简历实体数据集$ResumeNER$	社交	实体识别

    - **数据构造方法：**
        - **数据标注**（$\frac{\text{抽取式问答}}{\text{实体}}$）：模糊匹配寻找最相似的文本片段作为答案
        - **格式整理：**处理数据格式，包含领域、文本、问答对
        - **负例生成：**
            - 随机从数据中获取`context`，保留`title`（50%）
            - 将正样本中作为答案的句子删除（20%）
            - 使用$BM25$算法采样高分文档（30%）

    我们在各领域广泛选择数据，保证了训练数据具有代表性；通过模糊匹配寻找最相似文本片段为答案的方法，我们分别针对抽取式问答和实体类问答获取数据，保证了模型在长短文本表现更均衡；通过细致的格式整理，我们保证了来自不同领域的不同类型数据可以被统一训练；通过负例生成与训练，我们解决了“能不能回答”的问题，进一步提升了模型的理解能力、泛用性与鲁棒性。

- **训练结果：**

    单个答案完整分数：

$$\text{精确率} = \frac{\text{预测答案与正确答案公共长度}}{\text{预测段落长度}}$$

$$\text{召回率} = \frac{\text{预测答案与正确答案公共长度}}{\text{标注段落长度}}$$

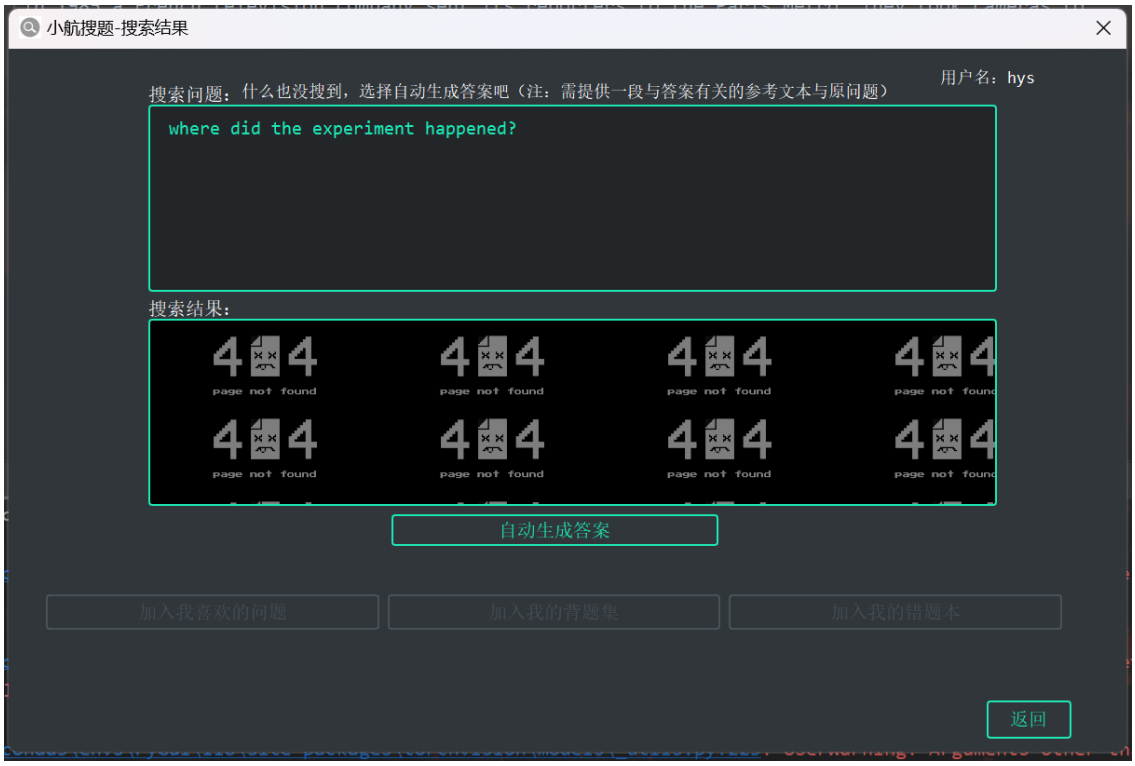
$$F_1 = \frac{2 \times \text{精确率} \times \text{召回率}}{\text{精确率} + \text{召回率}}$$

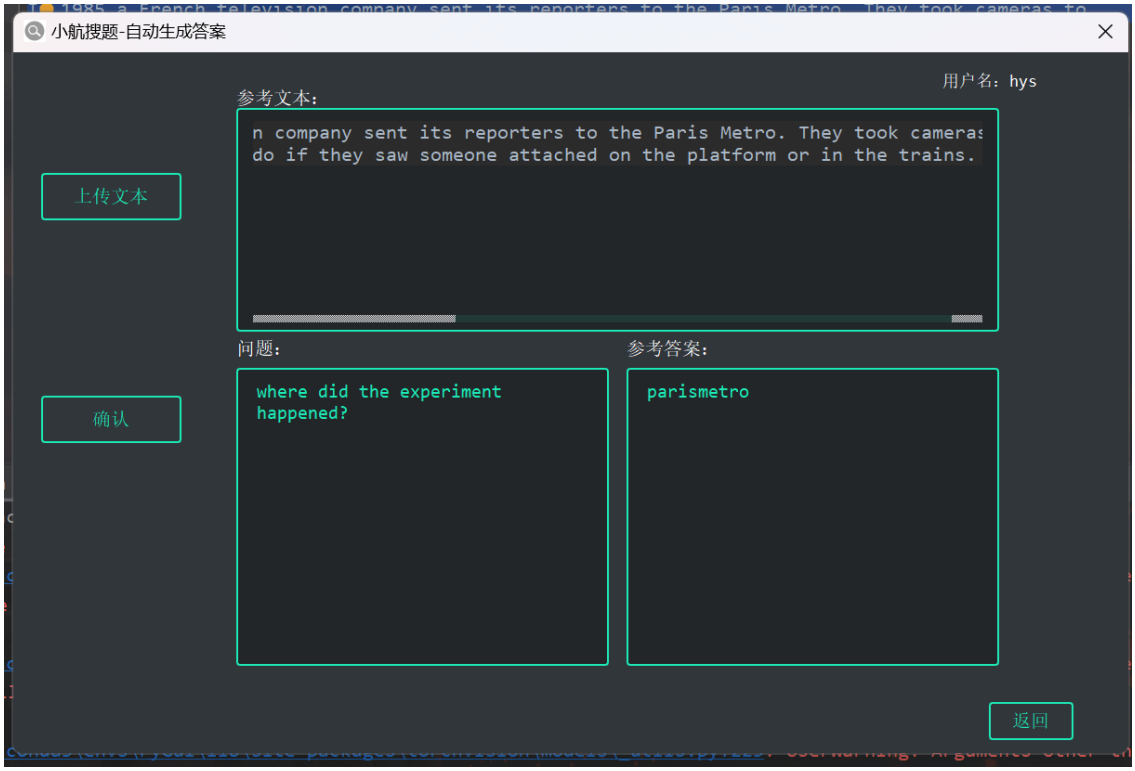
    下表中数据均为F_1的计算结果

模型	验证集（350）	地点集（100）	地点集（2886）
HF线下版本	32.08	17.91	20.87
HF在线版本	36.18	22.12	−−
$our\ model\ v1.0$	45.95	34.42	28.57
$our\ model\ v2.0$	54.87	−−	47.23

- **功能使用：**

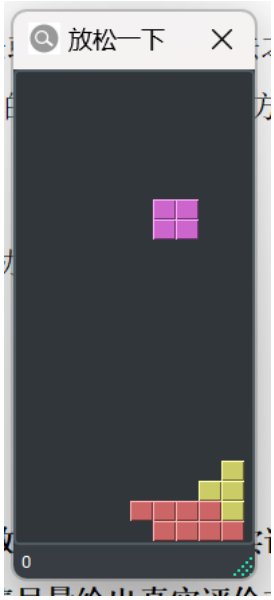
在小航搜题中，若某道题在题库中无法被搜寻到，即会弹出“自动生成答案”按钮，只需在“参考文本”中输入答案可能的来源文本，再在“问题”中输入问题，即可自动从文本中抽取答案。





新增俄罗斯方块游戏功能

紧张的学习怎么能没有放松！为了使小航搜题的功能更加人性化，同时为了更深入地学习`UI`界面，我们增加了俄罗斯方块游戏，同时可以查看本地用户游戏纪录与自己的游戏纪录。益智轻松的俄罗斯方块既能提升用户体验，又能促使我们更好地掌握课程所学的前端设计内容。



按住方向键可以左移、右移、顺时针旋转、逆时针旋转，按`p`暂停

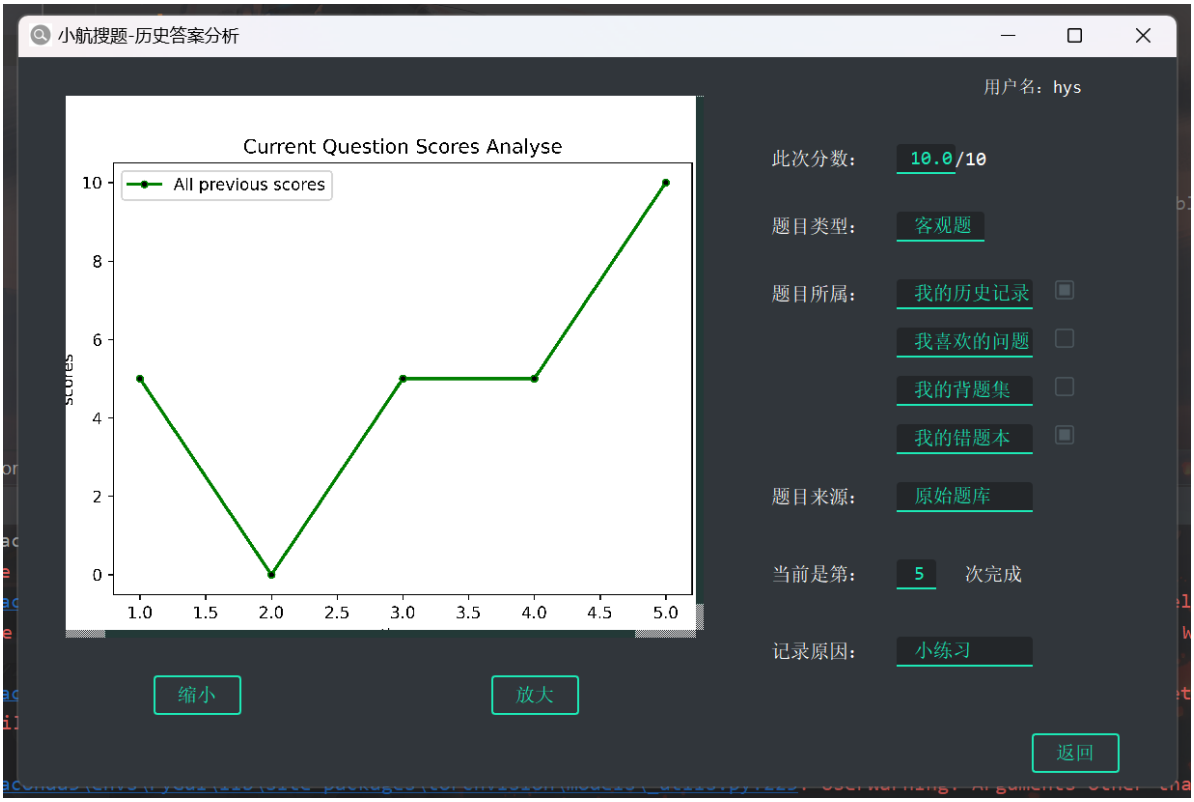
结果可以显示如上，觉得不满意可以试试再来一次，不过由于考虑到学生应以学习为重，我们是不会让您再玩的，而是会出现友情提示

使用模糊匹配增加搜题输入容错率

使用了`fuzzywuzzy`库中的`fuzz.partial_ratio(S1,S2)`部分匹配函数来进行模糊匹配计算相似度，能有效的减少因输入文本中的错误或缺失导致的搜题失败或者返回结果错误的情况，增强了搜题效率与准确度。

历史数据可视化呈现

使用`matplotlib`以及`opencv-python`中的函数将历史某题中的全部得分用折线图在界面重现，同时支持拖动与放大缩小



五、实验难点总结

图片预处理

由于`OCR`识别的准确度受图片质量影响很大，因此在图片预处理的过程中我们遇到了很多的问题，包括究竟是选择全局阈值二值化还是局部阈值二值化，二值化的阈值究竟设定为多少比较合适，如何使文字变得更加清晰等等。我们进行了大量的实验，请教了在计算机视觉方面有所研究的学长，才最终选择了一个合适的阈值，同时选择通过边界腐蚀和边界膨胀来提升字体的清晰程度。

机器阅读理解模型训练

由于我们选用的机器阅读理解模型属于大型模型，加之我们收集的数据量达百万条之多，笔记本电脑的算力根本无法满足，因此如何训练模型困扰了我们许久。最终我们选择了租用服务器（还有一部分训练是由学校实验室设备支持完成的），利用服务器的`GPU`资源快速进行模型训练，

`UI`设计

页面间切换从属设计以及部分控件的使用，由于`PyQt5`这个库每个控件下的函数非常多，非常乱，好些函数网上搜出来斌没有使用的示例，因此要自己尝试，而页面间关系随着代码量的上升维护起来逐渐困难，且过一段时间自己会忘了这部分代码是干什么用到，这个问题也没有太好的解决办法，只能重新读一遍代码

后端数据存储

因为目前我们还没有学习数据库相关知识，因此要想实现题库和用户系统的设计，必须从零开始学习数据库。一开始我们采用了更推荐初学者学习的`mysql`作为数据库，一边学习一边构建我们的数据库系统。后面在数据库工作接近尾声的时候我们觉得`mysql`作为需要输入登录密码的数据库，并不是非常适合用于这种需要方便移植，本地使用的嵌入式使用场景。经过进一步的学习，我们选用了更适合作为嵌入式数据库的轻量级数据库`sqlite`来替代原有的数据库系统，解决了移植需要`mysql`环境和输入密码的繁琐步骤。但是`sqlite`并不是完全兼容`mysql`语句的，在移植过程中也出现了不少需要更改的地方，例如在`mysql`语句中，自增关键字为`auto_increment`，而在`sqlite`中自增关键字为`autoincrement`，并且只能用于`integer`类型的主键上，又例如`mysql`支持右外连接，而`sqlite`仅

在2022.7更新的最新版本中才支持，移植过程出现了不同有人无法正常使用，有人又可以的情况，排查这些不兼容的问题花了不少功夫。

搜题实现

根据文本搜索题库内容也是困扰了我们很久的问题，因为本身数据库仅支持较准确的匹配，并不能很好的满足我们需要对各种输入不准确以及不在题库中的题目匹配的情况。一开始我们使用了将输入字符串切片用有限长的有效字符串去使用数据库的匹配功能进行匹配，希望能达到抗干扰的效果。但后面发现这么做的效果并不是很好，还是会出现切片中混有干扰字符导致无法成功匹配的情况。于是后面我们换用了`Levenshtein`计算编辑距离来查找最接近的字符串，但这又出现了阈值难以确定的问题，因为有时输入的并不是完整题目，所以即便是最接近的串可能编辑距离也很大，无法很好的确定判断题目是否在题库内的阈值选取。最后我们选择了`fuzzywuzzy`库中`fuzz.partial_ratio(S1,S2)`函数进行部分匹配返回百分比相似度来判断，经过数据筛选确定出较为合适的阈值，在经过反复测试确定这个方案最符合我们的需求。

六、课程学习总结

课程收获

- **夯实了`Python`基础知识**：得益于老师细致入微的讲解，通过本次暑期`Python`课程我们对于`Python`的诸多基础知识有了更加深入的认识。这其中就包括了动态语言与静态语言的区别、`Python`程序的结构、`Python`中的数据类型等等。尤其是老师对列表、字典、元组等复合数据类型的讲解降低了我们在使用他们的过程中出错的可能性。
- **巩固了面向对象思想**：在大二下学期，我们的专业课程中包含了“面向对象的设计与构造”（以下简称`OO`），其中就详细讲解了面向对象的内容，不过其是以`Java`语言为载体。在本次课程中，我们以`Python`语言为载体，学习了在`Python`中面向对象的有关知识。这不仅是对面向对象思想的巩固，更是对`OO`课程所学知识的迁移，使我们不再仅仅只会使用`Java`，而是将面向对象的思想扩展到其他语言。
- **提升了团队协作能力**：本次暑期课的大作业具有很大的挑战性，需要一个小组的同学来共同攻坚克难。在完成大作业的过程当中，我们不断地沟通想法、不断地修正方案、不断地解决`BUG`，秉持着“高内聚、低耦合”的思想，分工明确，以模块化的形式进行设计，逐步实现了一个又一个功能，如同拼积木一般将整个项目一块块地搭建完成。整个过程对于我们的团队协作能力、工程能力、资料搜寻能力都有极大的提升。

难点分析：

- 英文授课，在知识获取时需要进行二次加工，增加了听课时的脑力消耗
- 大作业比较难，需要好多技术栈支持，而这些内容相当一部分是大三才开始接触的，比如数据库

评价与建议：

老师、助教、课程都很友好，特别是助教答疑非常耐心，建议就是继续保持

七、参考资料

- https://iowiki.com/pyqt/pyqt_quick_guide.html
- https://blog.csdn.net/weixin_34268610/article/details/93036528
- https://blog.csdn.net/u010368839/article/details/78963843
- https://pymysql.readthedocs.io/en/latest/
- https://www.yiibai.com/sqlite/outer-join.html
- https://github.com/muziing/PyQt_practice