# 医疗知识问答机器人——小康

## 一、选题背景

### 1.1 开发背景

#### 1.1.1智能问答机器人概述

智能问答机器人是以聊天界面为基础，可以是文本聊天，也可以是语音聊天，通过聊天解决用户需求的一种服务模式，类似于虚拟助理。智能问答机器人已经在电商、通讯、金融等领域广泛应用，医疗行业同样也有相当大的需求。医疗机构可以借助AI对话平台，将具有对话能力的智能聊天机器人植入到挂号/门诊管理系统中，患者即可通过相关平台的网站、APP、公众号/服务号、小程序等进行对话交互，随时获取服务信息。

#### 1.1.2 智能问答机器人与医疗领域结合

iiMedia Research（艾媒咨询）数据显示，2020年中国移动医疗用户规模达到6.61亿人。在2020年新冠疫情影响下，居民对于互联网医疗平台使用需求进一步提升，推动用户规模快速增长。2018年中国人口老龄化率达到17.2%，近几年这一数据呈现逐年上升的趋势。在人口老龄化背景下，互联网医疗为老年人看诊购药及后续医疗服务等提供极大便利。而年轻居民对于健康的关注度也愈发提升，为互联网医疗营造有利条件，整体发展环境呈现积极态势。

据调查研究，在医疗领域聊天机器人中使用聊天机器人，就可以让每次查询平均节省4分钟以上的时间。因此，很多重复性、基础的咨询工作就可以委派给人工智能去做。

以印度医疗机构Lal PathLabs的聊天机器人为例，它可以为患者提供24/7全天候聊天服务，例如立即查看待处理报告的状态、查找附近的检测中心、显示医疗项目的信息和价格、进行门诊预约等。在启用了聊天机器人后，Lal PathLabs的电话呼叫量明显减少，减低了人工成本，另外患者与医疗机构能够进行及时有效的沟通，效率直线上升。

### 1.2 需求分析

#### 1.2.1 患者端需求

国内市场，就医流程复杂、就医体验差、医患关系紧张是当今医疗系统的几大痛点。据卫生部调查数据显示，中国约有近半居民“有病不就医”。

互联网医疗，是互联网在医疗领域的新应用，弥补了我国稀有且分布不均的医疗资源与日益增长的医疗健康需求之间的巨大缺口，缩短传统诊疗程序，并大大减少相关医疗成本，极大的改变了人们的生活方式。国家对于互联网医疗的态度也日渐明显。2018年4月12日，李克强主持召开国务院常务会议，确定发展“互联网＋医疗健康”措施，缓解看病就医难题、提升人民健康水平。

医疗机构或平台可以利用聊天平台，打造预问诊、医疗智能客服等不同形式的聊天机器人，让患者无需挂号预约，即可了解简单的医学知识，对自己的情况有正确的认知。智能对话体验让整个就医体验更加简单高效，“看病难”的问题也许将不再那么难了。医疗领域的问答机器人能够实实在在为患者带来效率和体验优化。

#### 1.2.2 医疗机构端需求

医疗领域对智能问答机器人的需求点如下：

首先，降低医疗机构经营成本。传统人工客服流动性大、培训成本高、服务水平良莠不齐，无法满足日益提升的客户需求。智能问答机器人能够快速拥有一套完整的知识体系，更快更好地与用户之间建立交互渠道，降低医疗机构人工成本，且机器人工作时间不受限制，可24小时响应用户需求。

其次，提高医护人员工作效率。一些简单、重复问题消磨了人工客服大量的时间、精力，智能问答机器人可以解决一些常识性问题，让医疗人员专攻高精尖医疗难题，促进社会办医的质量提升。

医疗机构要拥抱对话体验，离不开对话平台，将语音识别、机器学习、自然语言理解等AI技术应用到医疗产品中。

### 1.3 产品介绍

华南理工大学2018级汇丰金融科技班同学组队开发的医疗知识问答机器人小康，是基于医疗知识的问答机器人，面向普通民众和患者提供医疗知识问答功能。小康的诞生，让对医疗知识有疑问的普通人随时随地身边都有一个“智能医生”，有什么问题随时可以获得解答。

## 二、选用技术介绍

### 2.1 Vue

本项目使用Vue框架来搭建前端。Vue是一套构建用户界面的渐进式框架。只关注视图层，采用自底向上增量开发的设计,目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

Vue.js有如下优点：

1. 体积小：压缩后只有33k；

2. 更高的运行效率：基于虚拟DOM，一种可以预先通过JavaScript进行各种计算，把最终的DOM操作计算出来并优化的技术，由于这种DOM操作属于预处理操作，并没有真实的操作DOM，所以叫做虚拟DOM；

3. 双向数据绑定：让开发者不用再去操作DOM对象，把更多的精力投入到业务逻辑上；

4. 生态丰富、学习成本低：市场上拥有大量成熟、稳定的基于vue.js的ui框架及组件，拿来即用实现快速开发；对初学者友好、入门容易、学习资料多；

### 2.2 Django

本项目的后台使用Django为框架来实现后台与服务器的操作连接以及逻辑处理。

Django是高水准的Python编程语言驱动的一个开源Web应用框架，由Python写成，采用了MTV的框架模式，即模型M，视图V和模版T。使用这种架构，程序员可以方便、快捷地创建高品质、易维护、数据库驱动的应用程序。这也正是OpenStack的Horizon组件采用这种架构进行设计的主要原因。另外，在Dj ango框架中，还包含许多功能强大的第三方插件，使得Django具有较强的可扩展性。Django 框架的核心组件有：

1. 用于创建模型的对象关系映射；
2. 为最终用户设计较好的管理界面；
3. URL 设计；
4. 设计者友好的模板语言；
5. 缓存系统。

### 2.3 TfidfVectorizer

在进行相似度计算之前，需要将每条问句都转换成向量的形式，即将每条问句都映射到一个向量空间，我们使用的方法是TF-IDF(词频-反向文档频率)对问句文本进行向量化。

TF-IDF = TF \* IDF

TfidfVectorizer是Python的sklearn库中的一个模块，能够帮助我们计算出词语的TF-IDF系数，从而简化我们向量化语句的过程。

### 2.4 余弦相似度

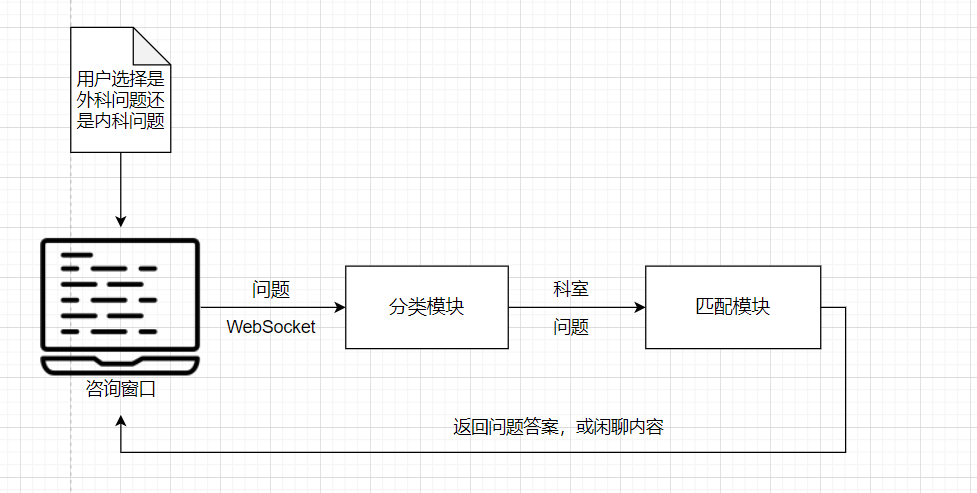
余弦相似度 (Cosine Similarity) 通过计算两个向量的夹角余弦值来评估他们的相似度。将向量根据坐标值，绘制到向量空间中，求得他们的夹角，并得出夹角对应的余弦值，此余弦值就可以用来表征这两个向量的相似性。夹角越小，余弦值越接近于1，它们的方向越吻合，则越相似。可以通过计算两个词向量之间的余弦相似度来判断两个语句之间的相似程度。

### 2.4 Keras

Keras是基于Theano的一个深度学习框架，它的设计参考了Torch，用Python语言编写，是一个高度模块化的神经网络库，支持GPU和CPU。

## 三、产品设计

### 3.1 架构设计



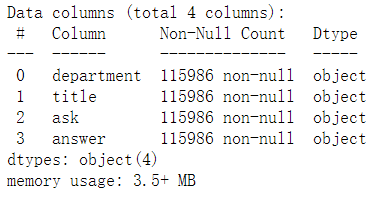
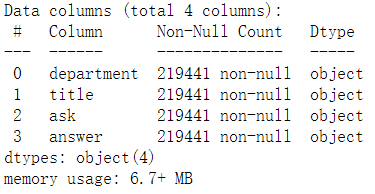
我们的项目整体上分为三个部分，第一部分是前端的咨询窗口，也是一个聊天窗口，我们采用的是现在非常流行的Vue框架进行开发。前端主要负责的是与用户进行交互，接收用户的输入的问题，返回给后端

第二部分是分类模块，分类模块将用户输入的问题具体归类到某一科室。

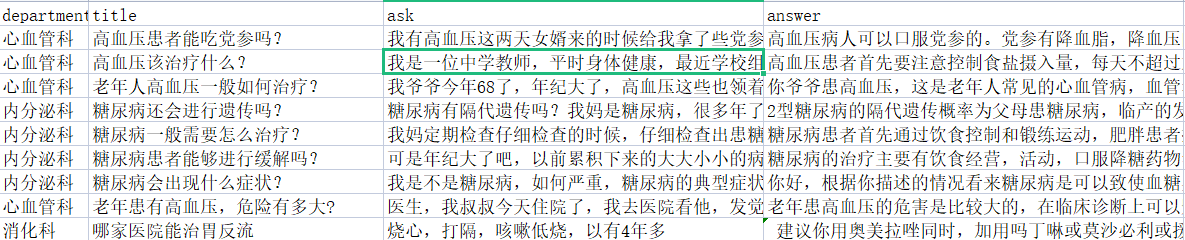
第三部分是匹配模块，接收分类模块传入的科室分类，遍历对应科室的所有问答集中的问题，计算余弦相似度，返回最大值的问题的答案，或者调用闲聊机器人进行闲聊。

### 3.2 数据清洗

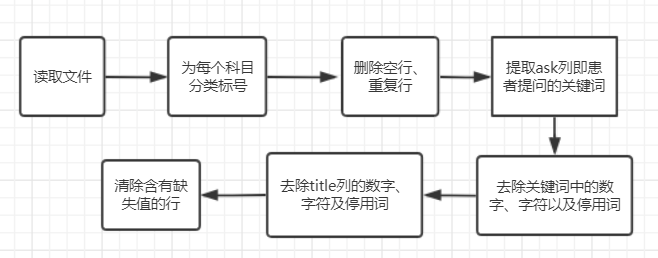
数据集是在CSDN上下载得来，分为内科以及外科两个类别，列名及数据量的大小大致分别如下图所示：



可以看出数据集的结构大概分为四个部分：具体细分的科目类别，精简化后的患者提问即问题的标准问法，患者的真实提问以及医生给出的回答、建议。部分数据集的截图如下所示：

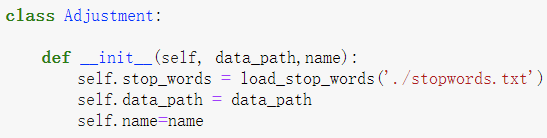


由于两个数据集的数量太过庞大，即使是每个细分的科目也有少则几千条，多则上万条的数据量，所以我们需要对这个数据集做一定的处理，以便后续进行进一步的匹配。数据处理的流程如下图：

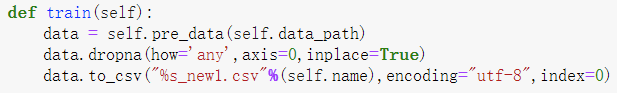


创建一个类Adjustment封装上述过程使用到的方法，方便直接创建类并调用类方法从而实现数据处理的目的，类中含有的方法如下表所示：

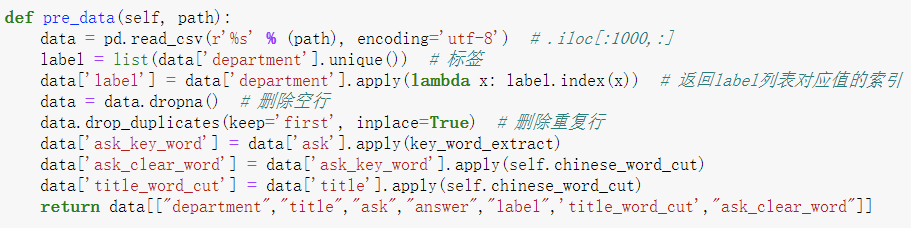
|  |  |
| --- | --- |
| 函数名称 | 作用 |
| train(self) | 控制数据进入数据处理的方法，并删除返回值的缺省行，最后将处理后的数据输出到一个新的文件 |
| pre\_data(self, path) | 加载数据集，并将数据进行分词、添加科目标号等的数据处理的方法 |
| chinese\_word\_cut(self, words) | 去除停用词、数字、字母的数据处理的方法 |



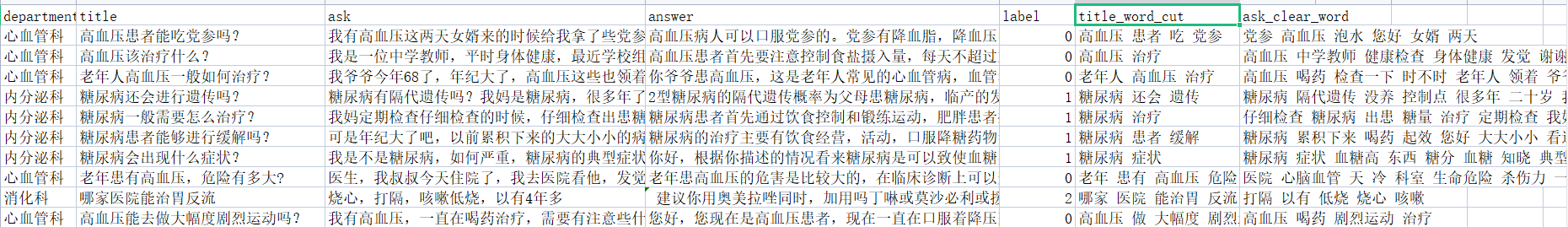
初始化类时加载停用词表，读取文件所在目录以及文件名，为接下来的数据处理做准备。



调用pre\_data方法进行数据处理并将处理后的结果保存在data中，去除含有缺失值的行并输出到新文件中。



在pre\_data方法中读取数据集，用unique方法为每个科目标上标号，标号的数目决定了最后拟合的模型需要输出的层数。之后删除数据集中的空行和重复行，调用key\_word\_extract方法提取患者提问的关键词，这个方法的实质是调用jieba库的analyse.extract\_tags方法提取关键词。接着调用chinese\_word\_cut方法进一步用jieba库对提取的关键词以及title列中的标准问题进行分词，并去除其中的数字、字符以及停用词，从而形成最后的提取词列。最后生成的新数据文件部分截图如下图所示：



可以看出最后经过处理的数据除了原始数据的四列之外，还新增了title列的分词结果列、ask列的分词结果列以及科目的标号列。这样就对后续的匹配算法的实现提供了基础。

但同时，数据处理过程还是有一些不足和有待改进的地方：在最后分词的结果中还是有部分如“您好”，“感觉”之类的看起来对匹配算法并无帮助但是却很难处理的词汇，或许可以找到更好的停用词表从而使分词后的结果更加准确而有意义。

### 3.3 前端

前端采用vue框架及Element ui组件库进行设计。

1. Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。
2. Element ui是一套为开发者、设计师和产品经理准备的基于Vue的组件库，提供了配套设计资源，帮助网站快速成型。由饿了么公司前端团队开源。

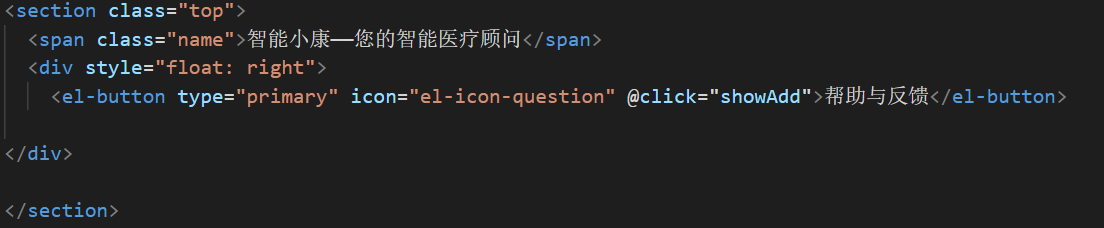
前端结构大致分为两个板块：websocket通信框架以及页面组件。WebSocket是一种通信协议，使得客户端和服务器之间的数据交换变得更加简单，允许服务端主动向客户端推送数据；浏览器和服务器只需要完成一次握手，两者之间就直接可以创建持久性的连接，并进行双向数据传输。页面部分，也就是components文件夹，里面存放前端页面设计的代码，共同组成了前端页面。

聊天页面的设计如下图所示：

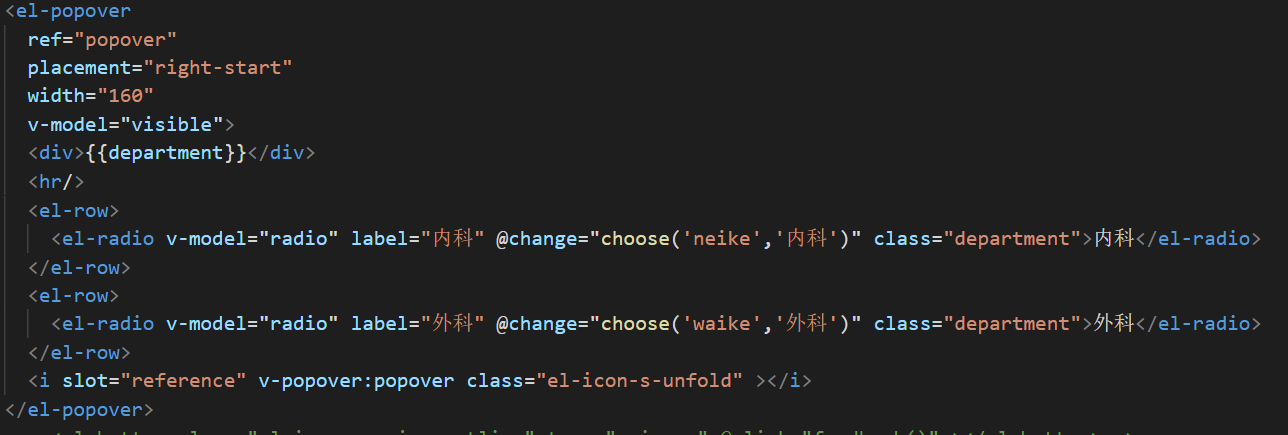


整个对话页面的主色调为蓝白色，背景图有医护人员的卡通形象及花束、云朵等，带有活泼的感觉，使人赏心悦目。对话页面分为三个页面组件组合而成：标题行、侧边栏以及聊天框。每个部件都是通过css样式的设计设置其大小并限制其位置，使其呈现图中的效果。在加载页面时初始化websocket并持续监听，以便随时发送接收信息。

标题行的组件有机器人的名字智能小康和我们的slogan“您的智能医疗顾问”，在标题栏的最右边有一个带着问号icon的帮助与反馈按钮，用户可以通过这个按钮了解到我们的公共邮箱，并通过邮件方式与我们联系。



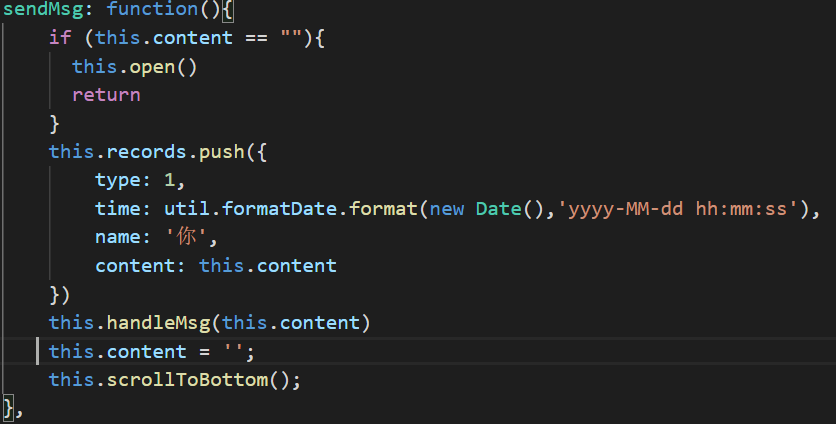
侧边栏的组件有医生的头像以及一个菜单按钮，菜单按钮是一个带嵌套的弹出框。用户可以选择内科或者外科。



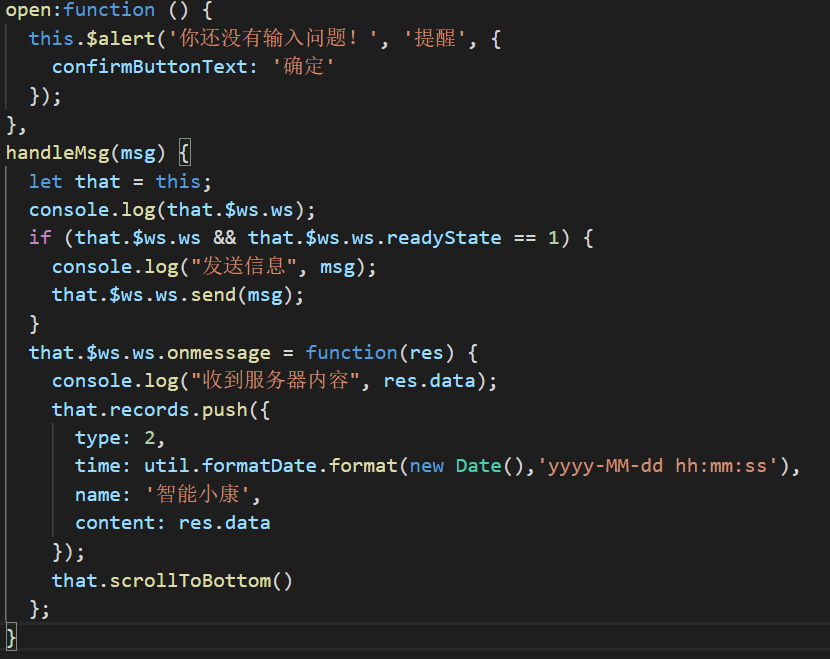
用户选择后会弹出选择成功的消息，用户选择的内容通过websocket发送至djiango交由后端。



聊天框分为聊天消息的展示框以及输入框。展示框中我们采用常规的医生头像在左边，用户头像在右边的布置。输入框的组件有一个文本域用于输入内容以及一个发送按钮，用户点击发送按钮后调用sendMsg方法，在展示框上显示此条发送信息。



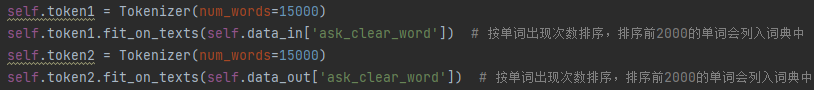
handleMsg(msg)将消息框中的内容通过websocket发送至djiango交由后端处理，若为空则提示用户为输入信息。

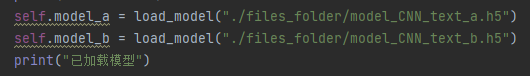


### 3.4 后端

#### 3.4.1分类部分

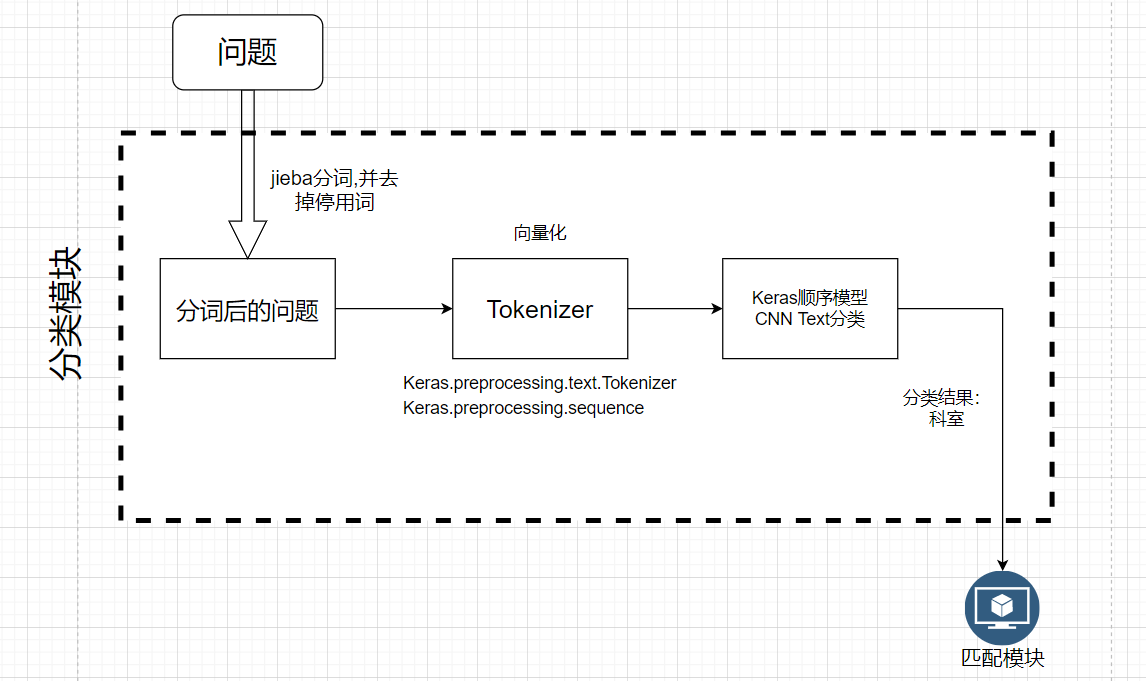
预加载Tokenizer分词器和训练好的两个分类模型以及初始化GetClassification类。







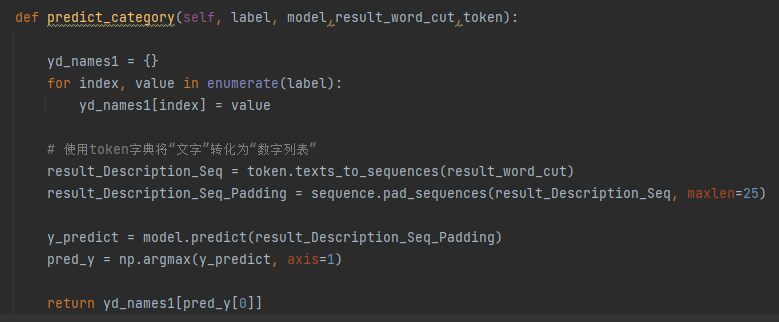
分类模块的运行流程图如下：



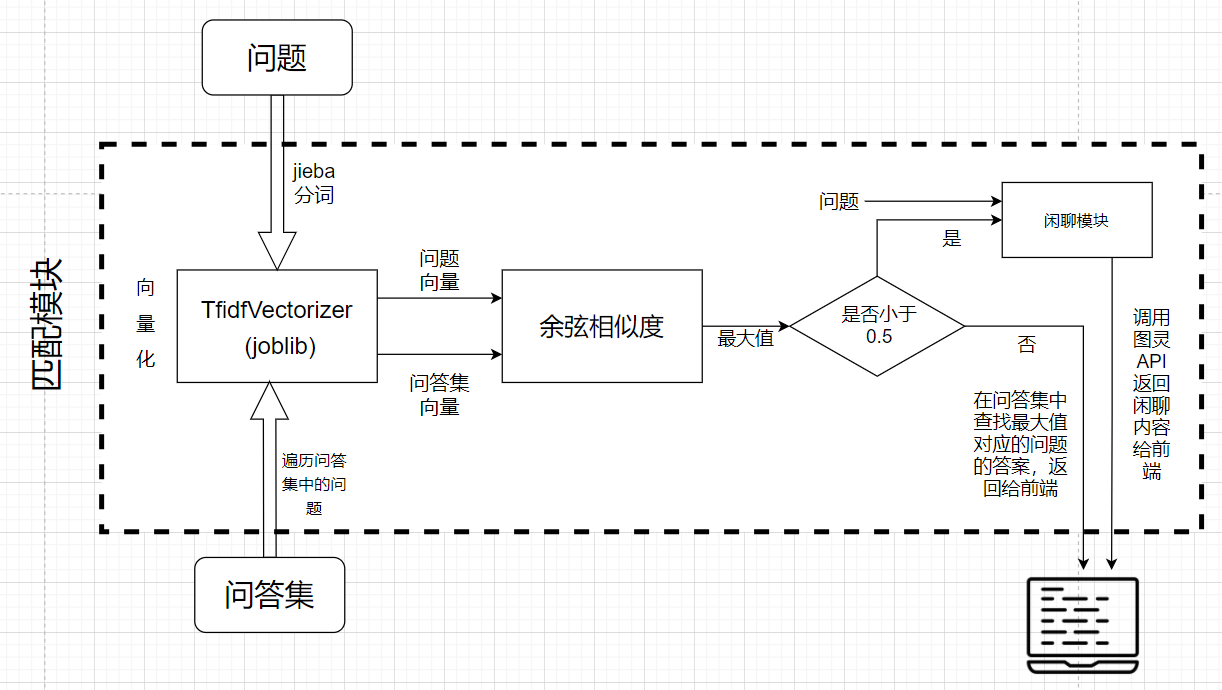
GetClassification类的方法如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| get\_category() | 控制获得具体科室分类的运行流程和代码逻辑 |
| predict\_category() | 分词结果向量化后，传入模型中，得到具体科室的分类结果 |
| query\_word\_cut() | 将用户问题分词，并去除停用词 |

predict\_category()方法的截图如下，运行流程即通过tokenize.texts\_to\_sequences()和sequence.pad\_sequences()方法将分词后的问题向量化并标准化，随后传入对应的分类模型中，根据结果，找到所对应的标签，得到分类后的科室结果。



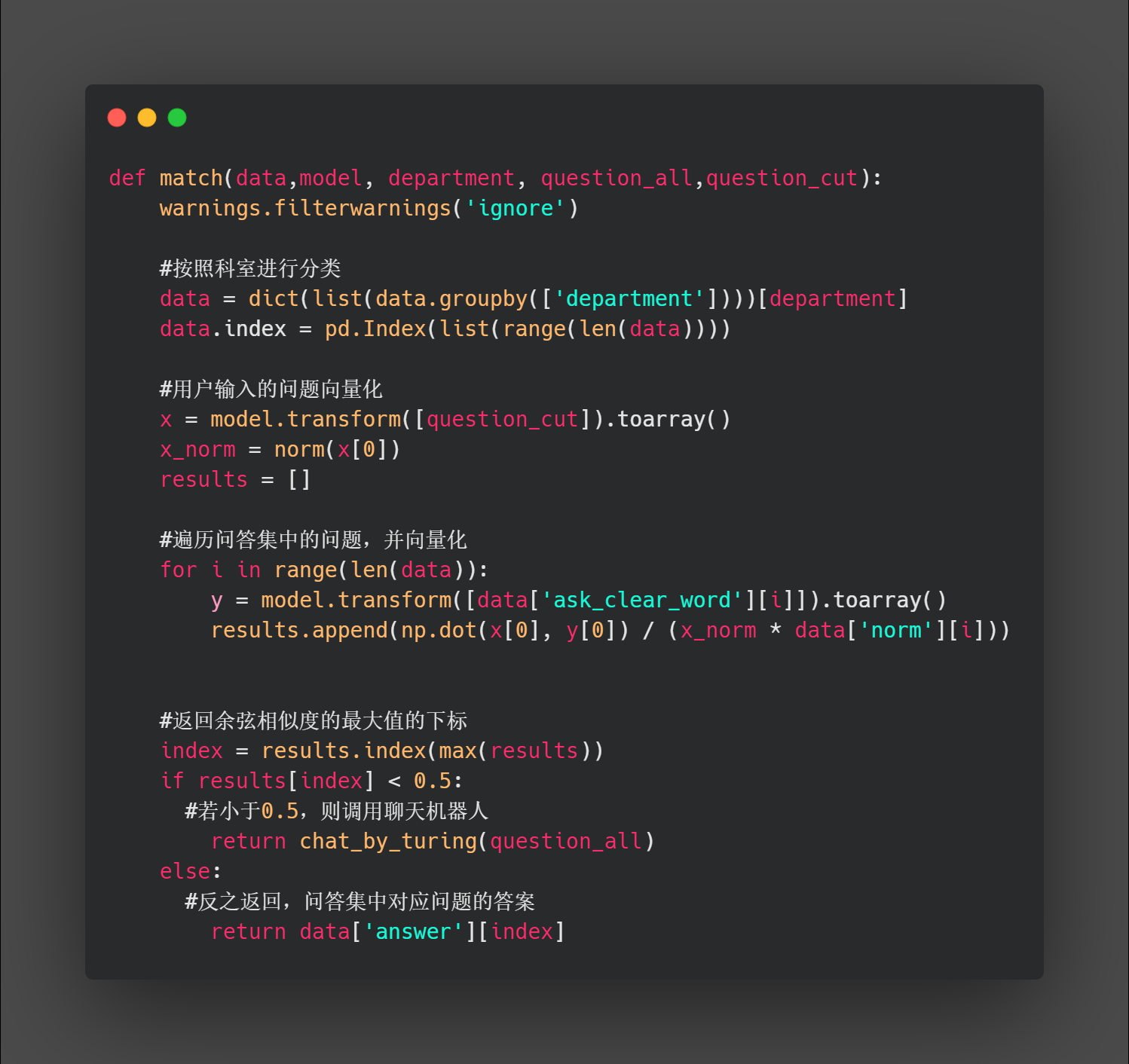
#### 3.4.2匹配部分



匹配模块的流程如上图所示。在数据处理阶段，我们已经针对内科和外科，分别训练出了一个词向量模型TfidfVectorizer，并保存为joblib格式。

当接收到用户提出的问题后，先调用我们自己构建的分词模块，对用户输入的问题进行一个分词，再传入匹配模块中，使用预先保存下来的词向量模型TfidfVectorizer进行问题向量化。随后遍历问答集中的所有问题，同时也调用词向量模型对每个问题进行向量化，然后计算问题向量和问答集向量之间的余弦相似度，返回余弦相似度的最大值。若余弦相似度的最大值小于0.5，则说明用户咨询的问题，在问答集中没有，为了避免回复“问答集中没有此类问题”之类的尴尬话语，以提升用户的体验，我们将问题转发给闲聊模块，闲聊模块是一个基于图灵API的聊天机器人，当余弦相似度最大值小于0.5时，调用闲聊机器人与用户聊天；当余弦相似度最大值大于等于0.5时，则返回问答集中对应问题的答案。

关键代码：



### 3.5 模型训练

#### 3.5.1 分类模型训练

**数据处理**



读入数据后，得到“label”标签总共有多少类，即最后模型需要输出的层数。

使用keras.preprocessing.text.Tokenizer和 keras.preprocessing.sequence这两个Keras框架的包，进行文本向量化的处理。

**Tokenizer分词器：**

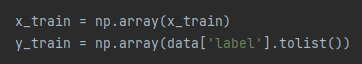
Fit\_on\_texts() 将我们要分词的文本内容输入进去，分词器会将里面所有的文本进行拆分，并且将所有拆分后的不重复词进行排序，形成一个词汇表。每一个词都对应一个独有的索引。

Texts\_to\_sequences()利用生成的词汇表，将data[“ask\_clear\_word”]的句子向量化，得到“数字列表”。

**Sequence：**

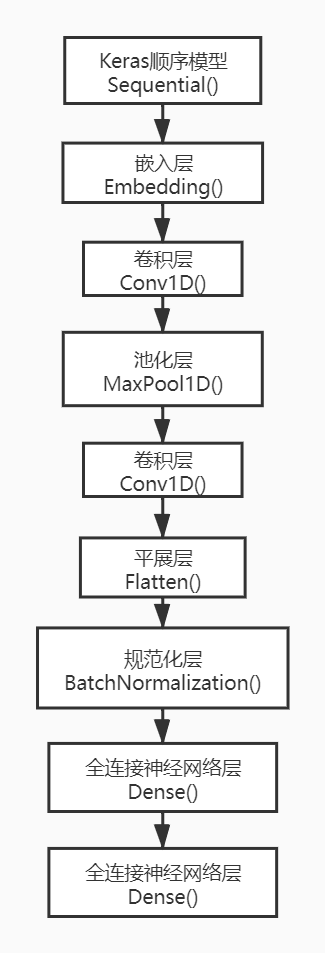
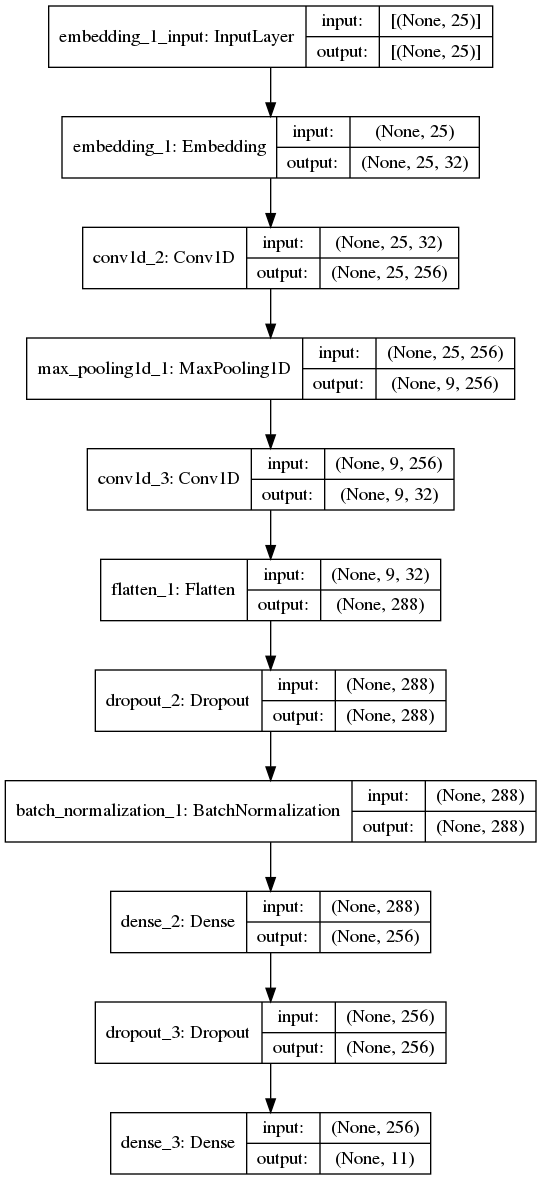
Pad\_sequences()进行序列填充或截断，通过截长补短让所有“数字列表”长度一致，maxlen即为最后得到的序列长度。padding的默认值为“pre”，即当数字列表长度短于maxlen时，在序列前面填充默认值0；当数字列表长度大于maxlen时，截断序列前面的数。

数据处理后即得到最后的训练样本x\_train和y\_train。



**构建模型**

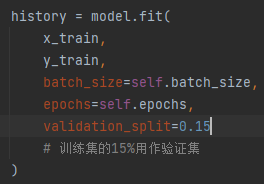
使用keras框架构建顺序模型，其中添加Embedding层、Conv1D层、MaxPool1D层、Flatten层、BatchNormalization层、Dense层，具体构建流程和数据输入输出维度如图所示。



构建keras顺序模型的代码如下图。

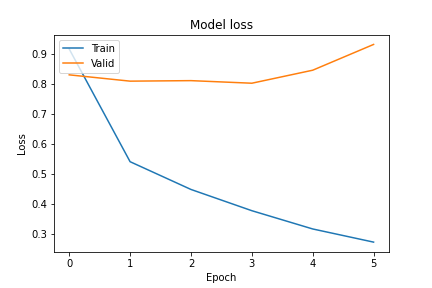
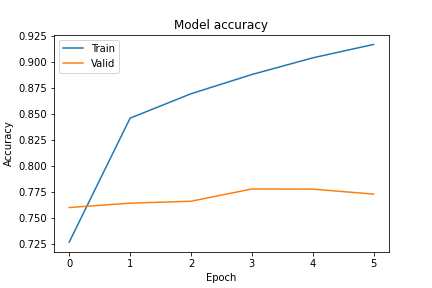


最后使用x\_train和y\_train对模型进行训练，并保存。

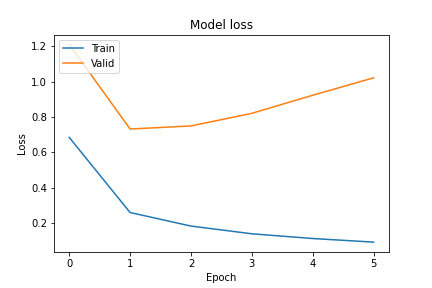
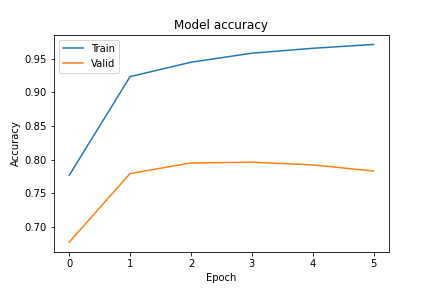


**分析准确率和损失值图**

内科数据模型训练结果的准确率和损失值图如下，分析可知迭代次数Epoch设置为1即可。



外科数据模型训练结果的准确率和损失值图如下，分析可知迭代次数Epoch设置为1或2即可。



#### 3.5.2 词向量模型构建

采用的是python的sklearn库中的TfidfVectorizer模块进行词向量模型的构建，以外科的词向量模型为例，核心代码如下：



## 四、不足与改进

1. 类别较少。我们现在只有2个大类（内科和外科），20个小类，所以用户可以咨询的问题的类别较少，日后可以向问答集中添加更多的类别，增加多样性。
2. 不稳定。由于我们的类别较少，所以当用户输入一个不属于问答集中的类别的医疗问题时，我们要么返回的答案很奇怪，要么只能调用聊天机器人回答。
3. 响应速度。正常情况下，我们的响应速度在2~3秒，当一些小类别的数据量比较大时，我们的响应速度会比较慢。所以日后，一方面，我们会考虑优化我们的算法模型，可以尝试一下余弦相似度之外的匹配算法模型；另一方面，我们发现我们的词向量大部分都是稀疏向量，因此我们未来会考虑压缩稀疏向量，减少向量的维度，从而提升算法的计算速度。

## 五、项目管理

### 5.1 代码管理

本项目代码通过在gitee上搭建的git仓库进行管理。

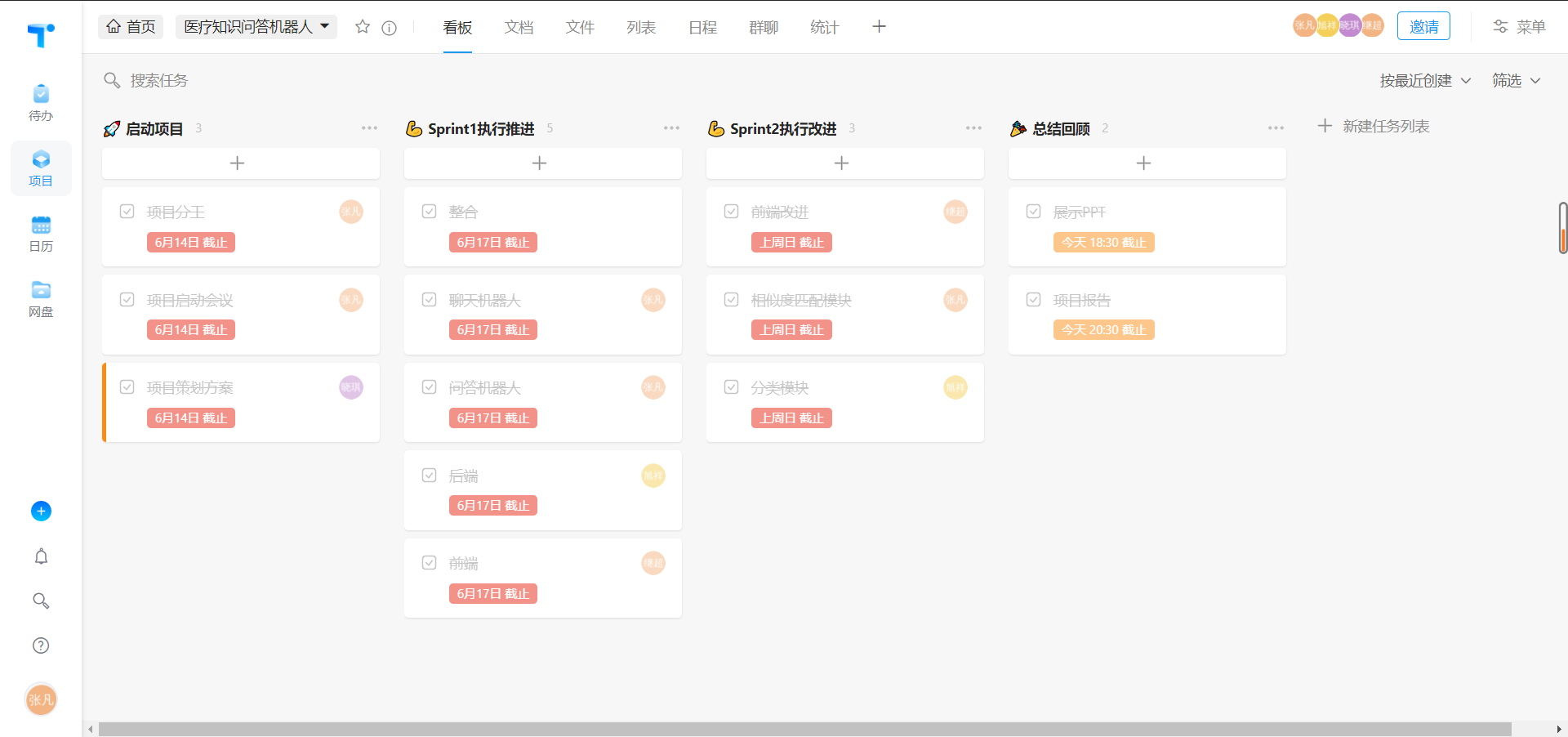




### 5.2 小组管理

由组长张凡提议，在teambition平台上进行可视化的项目进展和协作，实现Agile敏捷开发全流程。

如下图所示，我们的项目进程大概分为四个部分，第一部分是启动项目，包括项目的分工，以及项目的初步规划；第二部分，也是第一轮sprint，我们做出了我们第一版的通用问答机器人；第三部分，也是第二轮sprint，我们在第一版的基础之上，我们进行了优化改进，做出了第二版的通用问答机器人（最终版）；第四部分，进行总结回顾，制作项目报告和PPT，准备答辩。



### 5.3 小组会议

本项目使用腾讯会议进行共享屏幕和多人会议，以方便在线交流和协作。

## 六、分工

|  |  |  |
| --- | --- | --- |
| **角色** | **姓名** | **主要工作** |
| **组长** | 张凡 | 项目整体规划，词向量模型构建，匹配模块，闲聊机器人 |
| **组员** | 曾旭详 | Keras模型构建，分类模块，后端与前端交互 |
| **组员** | 杨继超 | 数据获取与预处理，前端界面设计，后端与前端交互 |
| **组员** | 沈晓琪 | 初期策划案撰写，前端界面设计 |