基于医疗知识图谱的问答系统

**一、应用场景**

当今社会处于互联网飞速发展的时代，同时人们也越来越关心自己以及家人的身体健康，许多人通过互联网获取信息来判断自己的身体状况。然而互联网上存在大量无关信息以及广告等会误导患者，并且耽误患者的治疗时间，使患者不能得到及时的治疗。因此，本文构建的基于医疗知识图谱的问答系统是具有现实意义的。该系统可以根据用户输入的问题，将用户的提问意图以及问句中的疾病实体信息进行提取，再根据得到的信息在构建的知识图谱中进行查询并返回准确地结果。

**二、解决方案**

总的来说，本系统主要分为两个方面，一是问句解析，其中由两个子任务构成：使用BiLSTM-CRF算法对于用户问句中的实体进行识别以及使用BERT-TextCNN模型对于用户的提问意图进行分类；二是建立一个医疗领域的知识库，总共划分为8种实体，11种实体关系，并最终基于Neo4j图数据库整合成一个含有三万五千个实体，二十九万多条实体关系的医疗领域知识图谱，以便对用户的提问进行回答。主要的解决方案将在以下小节展开介绍。

**2.1 基于BiLSTM-CRF算法提取用户问句中的实体信息**

本文的命名实体识别相关模块主要由两个部分构成，分别是BiLSTM层和CRF层，具体的模型结构如图2-1所示：

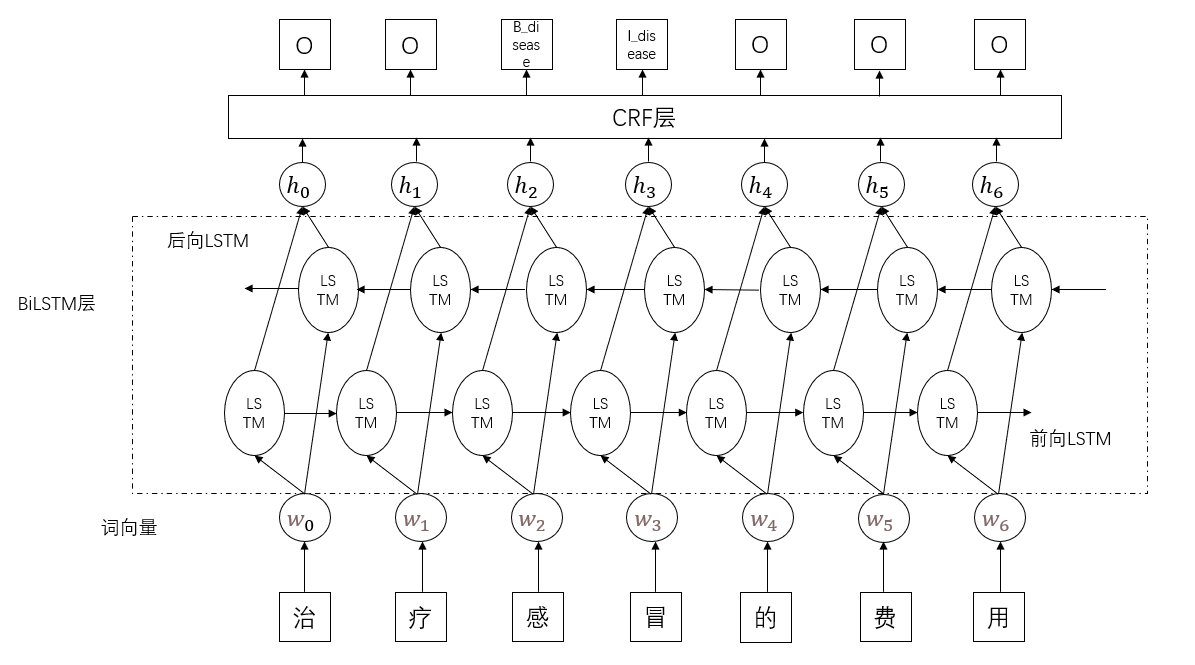


图2-1 BiLSTM-CRF模型结构

（1）BiLSTM层

BiLSTM是LSTM的优化，对于LSTM的基本原理以及缺陷，已经在第二章详细介绍过，这里就不再进行赘述。即便LSTM模型已经解决了文本的远程依赖问题，但是它的一个缺陷是无法编码从后到前的信息，也就是它只能是单向编码，这就使它无法更好的从后文学习语义信息来矫正前文。当我们在需要更为具体的一个分类时，需要注意症状词，情感词以及否定词之间的影响。如“他发烧温度高得不行，得赶快降温”，这句话的“不行”是对温度高的程度的一种修饰，如果只是通过LSTM的单向信息来对语义学习，效果会不是很好，这样并不能很准确地理解上下文信息。因此本文使用BiLSTM模型来提取用户问句中的实体信息，这样可以有效的避免后文信息量大于前文时出现一些问题。BiLSTM将正向文本和逆向文本信息分别采用不同的方式编码，最后将两种编码拼接得到包含上下文特征信息的特征向量，最后再通过全连接层激活函数得到各类别的得分。

（2）CRF层

从上文对BiLSTM层的介绍，可以知道BiLSTM层可以输出每个标签的预测分数来对实体进行预测，那么既然已经可以预测，为什么还需要CRF层呢。原因是如果我们仅使用BiLSTM层来对词语进行分类标签会出现标签序列不合理的情况。譬如使用BIO进行标注时，B代表实体的开头，I代表实体的中间部分，O代表无关词语，假设对于科室的标注符号为“department”，对于身体的标注为“body”，一个正确的预测序列I应该是在B的后面，一个序列的首个词应该为O或B，并且当B的后一个输出I和B不是同一类时也是无效标签。但是由于BiLSTM的输出是选择得分最大的标签作为该序列的输出标签，当出现如图2-2所示的情况时：“I- department”，“B-body”，“O”，“B- department”，“I-body”，这显然是不合理的。BiLSTM无法保证每次预测的标签都是正确的，因此需要在后面加一层CRF来解决预测标签不合理的问题。

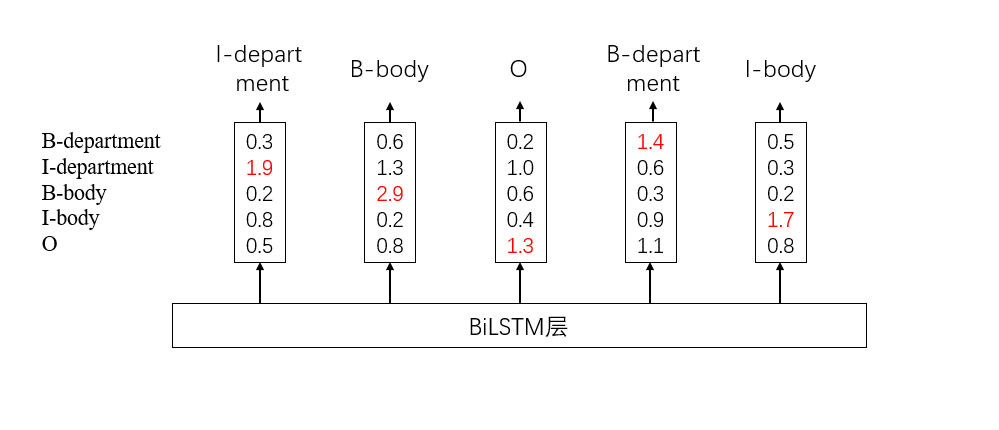


图2-2 BiLSTM标签预测

CRF层的主要作用是向最终的预测标签加一些约束来确保其有效性，它在模型训练时通过转移矩阵根据标签之间的依赖关系来学习到一些依赖规则，过滤掉不合法的输出标签，计算出每个token的最优输出标签。它的约束规则可以是：句子中第一个单词的标签应该以“B-”或“O”开头，而不是“I-”，“B-”后的“I-”必须是同一类等等。这种约束条件是多种多样的，CRF层的作用就是为了避免产生这些错误结果。

（3）数据集与评价指标

在进行NER的实验中，除了模型的构建，数据集的选取也同样是至关重要的，数据集的好坏根本程度上决定了模型实验结果的好坏。这里使用了ChineseBLUE中的cMedQANER数据集，它含有body，department，disease，drug等11个实体，训练集，验证集，测试集加起来共21万个token。本文采用精确率、召回率和F1值来作为评价指标。

**2.2 基于BERT-TextCNN模型对用户意图进行分类**

本文的意图识别相关模块主要是由BERT层和TextCNN两个部分组成的，具体模型结构如图2-3所示：

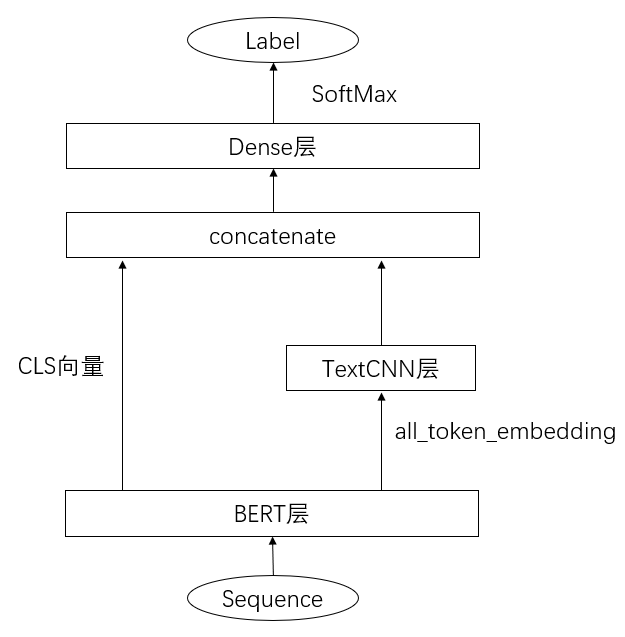


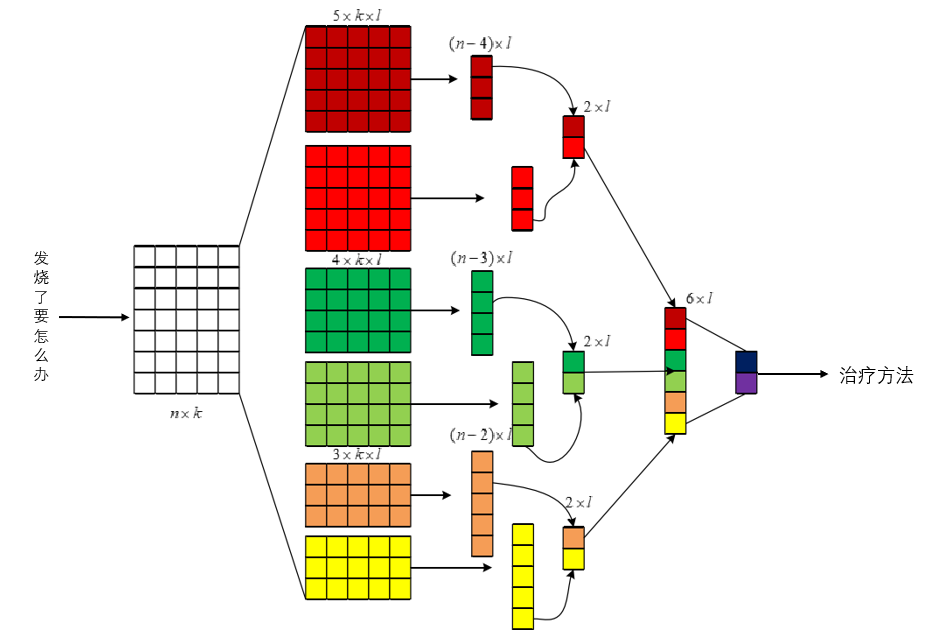
图 2-3 BERT-TextCNN模型结构

（1）BERT层

BERT-TextCNN模型的第一层是嵌入层，该部分的作用是将输入的sequence语句转换为对应的向量表示，也就是常说的编码层。该模块通常的做法是使用词袋模型、Word2Vec或者使用BERT等，本文所使用的方法是BERT。总的来说是将BERT模型输出的embedding层作为下一个模块的输入，CLS层则与下一模块的输出做concatenate来作为整个模型的输出。

（2）TextCNN层

TextCNN模型的构造十分简单，训练速度快，它的结构如图2-4所示：



卷积层，分别用两次3\*3，4\*4，5\*5的卷积核进行卷积

全连接层，将向量进行拼接再降维最后经过SoftMax输出预测标签

池化层，采用maxpooling

TextCNN由三部分构成，分别是卷积层、池化层和全连接层。其卷积层采用一维卷积来对序列特征进行提取，在这里filter设置为2，激活函数设置为Relu，输出设置为256。其下一部分为池化层，池化层的作用是对每个区域进行下采样得到一个特征值作为该区域的特征输出，使得通过卷积层后的向量维度保持一致，并且能够捕捉到句子最突出的特征，然后将特征进行拼接和输出。之后使用最大池化来精简特征值。第三步就是全连接层了，在进行第三步之前，需要将上一个模块输出中的CLS层与TextCNN池化后的向量进行拼接，得到拼接后的向量再进入全连接层，之后经过Relu激活函数输出，再接入SoftMax函数得到各标签概率，最后将得分最高的标签进行输出。

（3）数据集和评价指标

在意图识别部分本文使用了CMID数据集作为训练和测试的数据集，该数据集共有4个大类以及所划分的36个小类。但是由于所构建的知识图谱并不能包括这么多类别，所以针对本文所使用的知识图谱，自定义划分了13个类别，分别是定义、病因、预防、临床表现(病症表现)、相关病症、治疗方法、所属科室、传染性、治愈率、禁忌、化验/体检方案、治疗时间以及其他。从CMID数据集中提取所需要的13个类别作为本次训练的数据集。由于本次的数据集经过筛选后数量太少，训练集只有七千多个样本，测试集只有八百多样本，因此就不再划分验证集。

本文针对意图识别任务采用的评价指标为精确率P，召回率R和F1值，与上一小节相同。

**2.3 基于知识图谱的答案检索**

构建医疗问答系统首先需要建立一个用于提供答案检索的医疗领域的知识库，本文的数据来源于http://www.xywy.com/，这是一个信息半结构化的医疗相关网站，该网站含有常见的疾病及其症状、并发症治疗方式、周期等信息。这些信息非常丰富而且数据质量高，可以用于构建知识图谱。本文使用liuhuanyong爬取到的数据并进行整理，形成适用于该项目的医疗知识图谱。

**2.3.1 构建知识图谱**

爬取到的数据共计8808条，数据格式如图2-5所示：

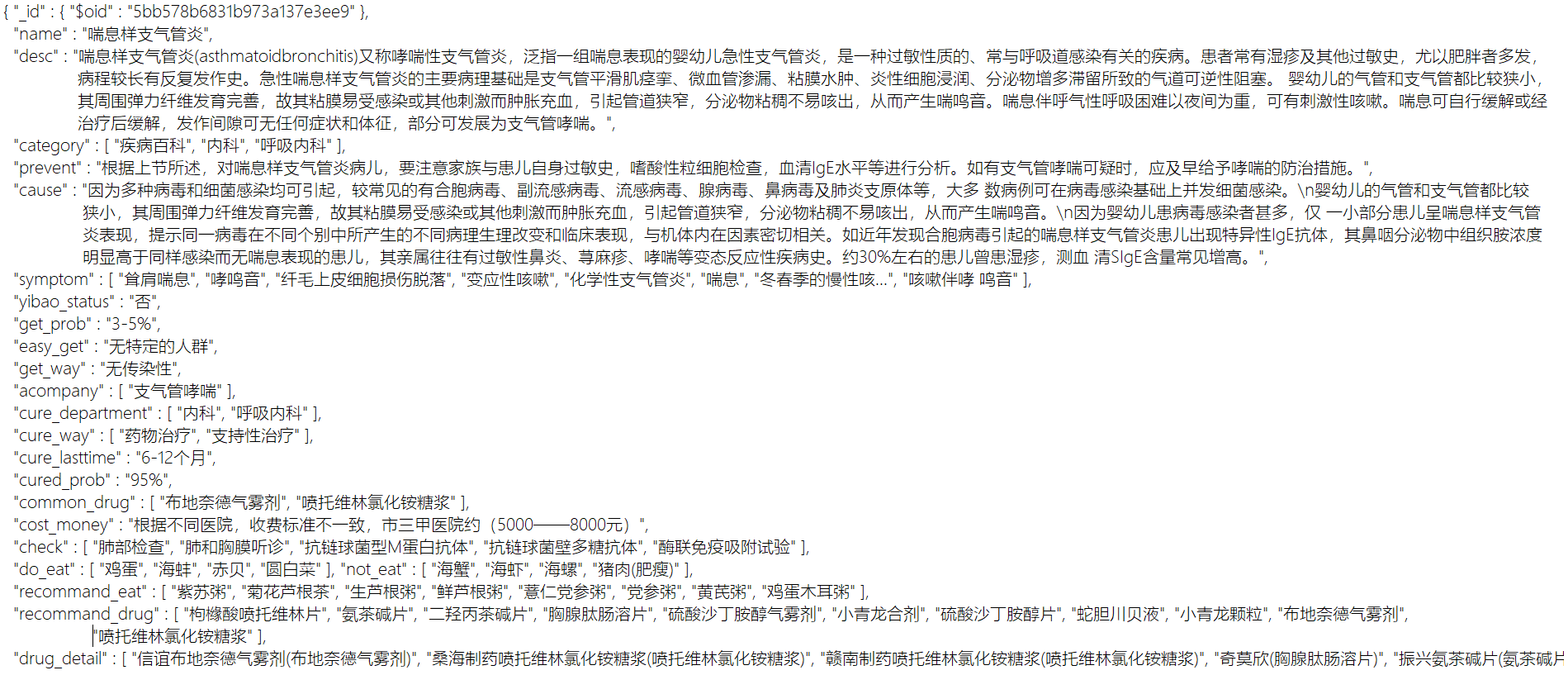


图 2-5 爬虫爬取到的数据

知识图谱的Schema定义了该图谱的领域本体，它包含了这一个领域中的所有概念、属性以及关系，是一种结构化的表达。任何一条数据都要通过Schema格式的定义才能被知识图谱更新。一般情况，加入知识图谱的数据是三元组的形式，即<实体，实体关系，属性>。本文定义的Schema如图2-6：

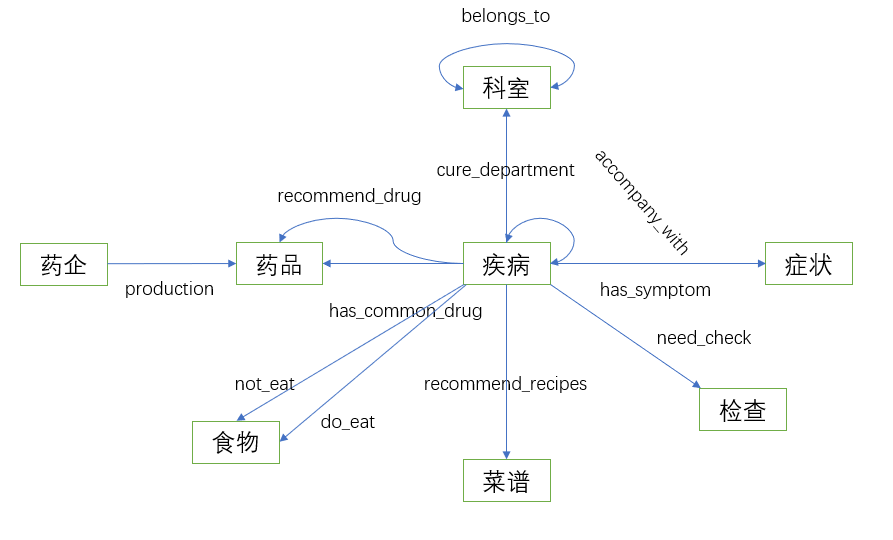


图 2-6 Schema结构图

由常识可知，每个疾病都有其对应的实体关系，如疾病与症状之间是疾病-症状关系(has\_symptom)，疾病和检查之间是疾病-检查(need-check)关系。在本文中，根据Schema结构定义了包括疾病在内的八个实体以及十一种实体关系。并以疾病为中心构建了包括并发症，食物，症状所需检查等八个属性。具体类型如表2-3所示：

表 2-3 实体定义

|  |  |  |
| --- | --- | --- |
| 实体类型 | 实体数量 | 举例 |
| 疾病 | 8807 | 支气管哮喘，感冒 |
| 症状 | 5998 | 胸闷，低热 |
| 科室 | 54 | 内科，儿科 |
| 检查 | 3352 | 血常规，骨髓分析 |
| 药品 | 3975 | 布美他尼片，十一味金色丸 |
| 食物 | 366 | 赤贝，鸡蛋 |
| 药企 | 7919 | 东信药业，海南制药厂 |
| 菜谱 | 4506 | 清蒸鲤鱼，红枣花生汤 |
| 合计 | 34977 |  |

为了在各实体之间建立联系，共构建了十一类实体关系，如疾 病 症 状 (has\_symptom)、 并 发 疾 病(accompany\_with)、常用药品(common\_drug)、宜吃食物(do\_eat)等，具体关系如表2-4所示：

表 2-4 实体关系定义

|  |  |  |
| --- | --- | --- |
| 实体关系 | 实体数量 | 举例 |
| belongs\_to | 37 | <神经内科，属于，内科> |
| recommend\_drug | 59465 | <不孕不育，推荐药品，麒麟丸> |
| accompany\_with | 12024 | <言语障碍，并发症，自闭症> |
| production | 17112 | <太安堂药业，生产药品，麒麟丸> |
| has\_common\_drug | 14647 | <胰腺癌，常用药品，尿嘧啶替加氟片> |
| has\_symptom | 54710 | <面神经瘤，症状，面部肌肉痉挛> |
| not\_eat | 22239 | <痛经，不能吃，鸡肝> |
| do\_eat | 22230 | <低血糖，可以吃，莲子> |
| recommend\_recipes | 40221 | <肠结核，推荐食谱，黄豆粥> |
| need\_check | 39417 | <牙结石，需要检查，牙髓温度试验> |
| cure\_department | 8806 | <毒蛇咬伤，所属科室，急诊科> |
| 合计 | 290908 |  |

由于本文是针对疾病的问答，所以对知识图谱中的属性也是以疾病为中心来定义。本文所定义的疾病相关属性有：

表 2-5 疾病属性定义

|  |  |  |
| --- | --- | --- |
| 属性类型 | 中文含义 | 举例 |
| prevent | **预防措施** | **注意卫生，劳逸结合** |
| get\_prob | **发病几率** | **0.05%** |
| cure\_way | **治疗方式** | **药物治疗，支持性治疗** |
| name | **疾病名称** | **梭形细胞脂肪瘤** |
| cure\_lasttime | **治愈时间** | **2-4周** |
| cured\_prob | **治愈概率** | **65%-80%** |
| cause | **发病原因** | **食管平滑肌痉挛、缺血** |
| cure\_department | **治疗科室** | **内科、消化内科** |
| easy\_get | **易感人群** | **无特殊人群** |
| desc | **疾病描述** | **白塞，好发于男性青壮年…** |

目前知识图谱一般都采用图数据库的方式进行存储，这样可以对各存储结点的属性以及之间的关系有更清晰的认知。当前最为流行的图数据库是Neo4j，因此本文选择使用Neo4j来对已经定义好的结构化的数据进行存储。具体方法是通过Py2neo模块对Neo4j数据库进行连接，并根据定义好的Schema分别执行Cypher语句创建实体结点，实体关系以及疾病的相关属性。图谱的部分展示如图2-7：

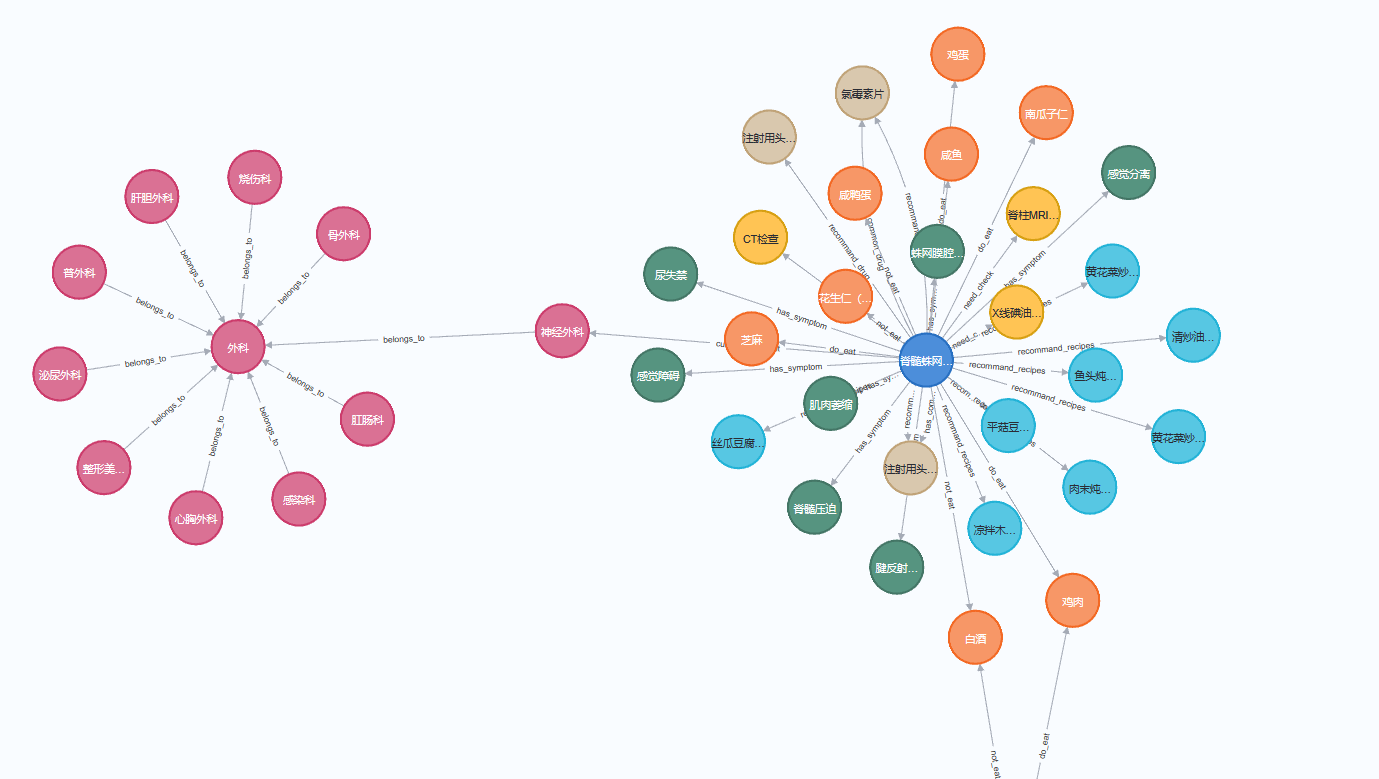


图 2-7 知识图谱展示(部分)

**2.3.2 设计答案检索方法**

在答案检索的过程中，用户输入的问句信息会作为text\_reply函数的输入，这个函数主要由三个部分构成：闲聊机器人(gossip\_robot)、多轮对话回复以及医疗问答机器人(medical\_robot)。用户输入的语句会在该函数开始时被分为7个类别：greet、goodbye、deny、isbot、accept、accept1、diagnose。其中前五类都表示闲聊：打招呼，说再见，否定，询问AI是谁，表示赞同。由于在进行对话中，用户对于有些问题的描述并不是很清晰，所以需要进一步的询问。这时用户会回答表示同意或否定的话语，为了避免将此时表示同意的意图和闲聊中的意图混淆，这里将两种情况下可能的回答分开进行标注。最后一类为有关疾病的类别，这里为了简便，把不是之前类别的意图都归为疾病类别。

如果判断用户输入的信息是闲聊类别，则通过gossip\_robot函数随机从对应类别的语料库抽取一条作为回复。具体不同类别的回复内容如表2-6所示：

表 2-6 闲聊回复内容

|  |  |
| --- | --- |
| 类别 | 回复 |
| greet | 我是智能医疗诊断机器人，有什么可以帮助你吗 |
| goodbye | 再见，很高兴为您服务 |
| deny | 很抱歉没帮到您 |
| isbot | 我是医疗诊断机器人 |
| accept | 没关系，还有其它问题要问么 |

如果判断用户输入的信息是diagnose类别，那么接下来的回答就会从medical\_robot函数中返回。medical\_robot函数主要由三个部分组成：问句解析(semantic\_parser)，知识图谱查询(neo4j\_searcher)以及回复(get\_answer)。

对于问句解析(semantic\_parser)函数，这个函数主要是得到用户输入语句的意图和该意图的置信度，以及实体信息。得到问句意图后，可以得到该意图下的查询以及回复模板。由于共有13种意图类别，这里只展示其中的病因以及临床表现(病症表现)模板，如图2-8、2-9，其余类别就不做过多介绍。

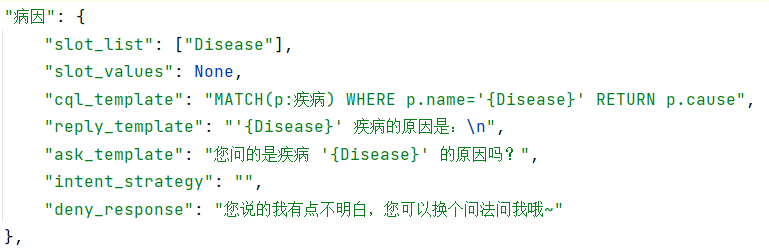


图 2-8 病因回复及查询模板

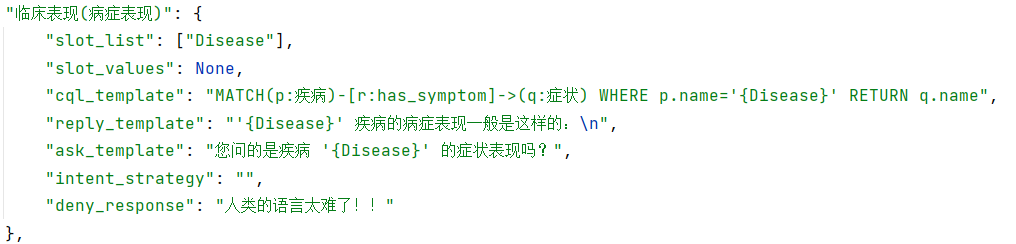


图 2-9 临床表现(病症表现)回复及查询模板

通过上图可知，每个种类的模板一般都有七种信息，分别是槽位列表(slot\_list)：当前槽位的实体类别，如：Disease。槽位信息(slot\_values)：当前槽位的具体实体类型，如过敏性感冒，心脏病等。查询模板(cql\_template)：neo4j数据库查询语句，如当前意图分类为病因，根据slot\_values查询该疾病的病因。回复模板(reply\_template)：这是针对不同的意图类别所设置的不同的回复语句。询问模板(ask\_template)：该语句一般在意图置信度为clarify时使用，当无法判断用户是否为该意图时回复该语句，可以用做多轮对话。意图策略(intent\_strategy)：这里有三个策略根据意图置信度来划分，当意图识别中的confidence分别为大于0.8,0.2-0.8以及小于0.2时，其对应的意图策略true，clarify以及deny，关于划分这三种策略的目的将会在get\_answer函数中详细介绍。否定回答(deny\_response)则是当用户对答案给予否定后的回复。

得到了用户问句意图的回复模板之后，接下来要对模板进行填槽，得到适用于该问句的回复语句。通过对问句进行命名实体识别后的实体信息进行提取，把识别到的疾病实体填充到slot\_values槽位中，得到该问句的slot\_values信息。再根据意图识别获得的置信度信息，可以对intent\_strategy槽位进行填充，得到回复策略。

通过上文可知，在semantic\_parser函数中得到了三种不同的回复策略以及疾病的实体信息，在get\_answer函数中对于不同的回复策略使用不同的查询语句以及回复方案。

当回复策略为accept时，表明对用户的意图判断正确，把疾病实体信息填入对应的cql\_template槽位获得cql查询语句，并在Neo4j图数据库中查询。如果没有在数据库中找到该实体信息，会在该意图下的模板新增reply\_answer键，并赋值为“抱歉我不太知道您说的这种疾病，您可以问我其它的病哦”作为回复语句。如果数据库中存在该疾病实体但是并没有相关的属性，这时reply\_answer会被赋值为这种模板："抱歉我不知道" + slot\_values + "的" + BIM.predict(text).get("intent")。如存在心脏病疾病但是并没有关于心脏病的治愈率回复抱歉我不知道心脏病的治愈率。如果数据库中存在该疾病及相关信息，则会按照填充实体后的回复模板以及数据库中查询得到的信息进行回答。

当回复策略为clarify时，表明对该问句的意图并不是很清楚，因此需要向用户提问是否为当前判断出来的意图，提问语句按照ask\_template进行填槽并存放在reply\_answer中。接下来使用cql语句对判断出的实体以及意图进行查询，如果不存在该实体，那么也就不需要再向用户提问意图是否正确，将reply\_answer赋值为“抱歉我不太知道您说的这种疾病，您可以问我其它的病哦”即可。如果存在该实体，无论是否存在该实体属性都将回答存放在回复模板中新增的choice\_answer里。这样无论用户输入问题中的实体是否存在，都只需使用reply\_answer进行回复。若实体存在且向用户提问澄清，得到肯定意图(accept1)后，便可以使用choice\_answer来进行回复；如果得到用户的否定意图，那么会进入闲聊机器人的deny类别语料库中随机挑选一个语句作为回复。

如果回复策略为deny，表明没有理解用户的意图，此时只需回复模板中的deny\_response语句即可。

通过上述介绍可以发现，本文还使用了多轮对话。具体实现方式是使用了一个json文件来进行记录，记录最近一次的所有槽位模板信息。如果已经提及过某一实体，在下一次提问中若还是该实体的相关信息，提问时可以不用提及该实体。在问句解析函数(semantic\_parser)中，若未识别到问句实体则会将None值填槽，此时会读取json文件中的数据将槽位值为None的信息进行填写，由此可以进行多轮对话。在每次提及新的实体时，也会更新json文件来保存模板信息以此来达到多轮对话的目的。

neo4j\_searcher函数是针对得到的cql查询语句在Neo4j数据库中进行查找，将查找到的答案按照“、”间隔并返回为字符串格式。如果不存在数据则返回为空。

**三、实验结果及功能测试**

**3.1实验结果**

（1）BiLSTM-CRF实体识别实验结果

在基于BiLSTM-CRF的命名实体识别的实验中，使用两层BiLSTM作为CRF层的输入，BiLSTM隐藏单元设置为128，词嵌入的维度设置为200，两层的Dropout分别设置为0.5和0.25，最大输入序列长度为80，epoch设置为80，batch size 为32。测试集的测试结果如表3-1所示：

表3-1 BiLSTM-CRF测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Precision (%) | Recall (%) | F1 (%) | Support |
| body | 66.98 | 65.75 | 66.36 | 219 |
| treatment | 69.31 | 50.00 | 58.09 | 140 |
| test | 58.54 | 52.17 | 55.17 | 46 |
| disease | 74.88 | 80.05 | 77.38 | 391 |
| crowd | 79.01 | 90.14 | 84.21 | 71 |
| time | 57.89 | 36.67 | 44.90 | 30 |
| symptom | 69.49 | 73.87 | 71.62 | 222 |
| drug | 50.00 | 43.33 | 46.43 | 60 |
| feature | 92.59 | 89.29 | 90.91 | 28 |
| physiology | 83.78 | 72.09 | 77.50 | 43 |
| department | 66.67 | 75.00 | 70.59 | 8 |
| average | 71.04 | 69.79 | 70.41 | 1258 |

如表3-1所示共11类实体1258个测试实例，平均F1值为70.41%，仔细观察不同实体的表现可以发现，该模型在test类，time类以及drug类的表现效果较差，导致了整体实验结果的效果不佳。其主要原因可能是因为这些类别所含的一些字段使模型判断错误导致整体识别有误。

下面测试对实际用户的输入信息来进行识别，编写MedicalNerModel类对用户的输入进行实体识别并输出实体以及实体类别。

例如用户输入“感觉很不舒服，很难受，是不是黄斑囊样水肿”，输出：[{'string': '感觉很不舒服，很难受，是不是得了黄斑囊样水肿', 'entities': [{'word': '黄斑囊样水肿', 'type': 'disease'}]}]。从输出格式可以看出输出数据是按照字典格式保存，其中string为用户输入的内容，entities也为字典格式，其中包含了实体word，也包含了实体的类别type。模型准确地识别出了“黄斑囊样水肿”这一实体并准确地判断出其类别为“disease”。

该模型不仅可以识别用户输入的单个实体，还能识别已知类别的多个实体，如用户输入“孩子感觉很不舒服，一直流鼻涕，明天上午要去耳鼻喉科看看是不是得了过敏性鼻炎，顺便看看需不需要买阿莫西林”，这句话包含了5个实体内容分别为crowd：孩子，symptom：流鼻涕，department：耳鼻喉科，disease：过敏性鼻炎，drug：阿莫西林。下面测试一下模型的输出：[{'string': '孩子感觉很不舒服，一直流鼻涕，明天上午要去耳鼻喉科看看是不是得了过敏性鼻炎，顺便看看需不需要买阿莫西林', 'entities': [{'word': '孩子', 'type': 'crowd'}, {'word': '流鼻涕', 'type': 'symptom'}, {'word': '耳鼻喉科', 'type': 'department'}, {'word': '过敏性鼻炎', 'type': 'disease'}, {'word': '阿莫西林', 'type': 'drug'}]}]，可以看出输出与预期相符，实体与类别都正确。

(2) BERT-TextCNN意图识别实验结果

在基于BERT-TextCNN的意图识别实验中，使用12层的BERT模型作为TextCNN的输入，使用Adam优化器，设置学习率为5e-6，TextCNN中卷积核的大小分别为3,4,5，单个卷积核的数量为256，epoch设置为10，batch size为8，测试结果如表3-2所示：

表3-2 BERT-TextCNN测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | Precision (%) | Recall (%) | F1 (%) | Support |
| 定义 | 72 | 56 | 63 | 41 |
| 病因 | 55 | 68 | 61 | 90 |
| 预防 | 76 | 54 | 63 | 24 |
| 临床表现(病症表现) | 60 | 57 | 59 | 141 |
| 相关病症 | 70 | 39 | 50 | 18 |
| 治疗方法 | 71 | 74 | 78 | 169 |
| 所属科室 | 100 | 100 | 100 | 9 |
| 传染性 | 90 | 100 | 95 | 9 |
| 治愈率 | 89 | 75 | 81 | 32 |
| 禁忌 | 58 | 45 | 51 | 31 |
| 化验/体检方案 | 82 | 96 | 89 | 28 |
| 治疗时间 | 81 | 87 | 84 | 15 |
| 其他 | 65 | 64 | 64 | 202 |
| average | 75 | 71 | 72 | 809 |

如表3-2所示共13个意图类别809个测试样例，准确率为75%，其中禁忌类、病因以及临床表现类准确度不高，原因可能是有些提问的方法与其它类别类似导致不能有效的进行分类。

下面编写BertIntentModel类来对用户的提问意图进行预测并输出预测概率以便进行后续的工作。例如用户输入语句“孩子得了过敏性鼻炎，这个病需要怎么治疗”，模型输出{'intent': '治疗方法', 'confidence': 0.9328970909118652}，可以看出用户的意图是疾病的治疗方法，预测正确。再如输入“孩子得了过敏性鼻炎，要去医院挂什么科”，用户的意图为想询问去哪个科室看病，模型返回的结果为{'intent': '所属科室', 'confidence': 0.8030179142951965}，可见该模型可以正确地预测用户的提问意图并可以返回置信率，以便用于后续的工作。

**3.2 系统测试**

（1）Neo4j图数据库测试

对Neo4j数据库查询时使用Cypher语句进行封装，如当查询心脏病的病因时采用如下语句“MATCH (m:`疾病`) where m.name = '心脏病' return m.name as name, m.cause as cause”。所得到的查询结果如图5-2所示。算法层主要是调用命名实体识别以及意图识别算法来对用户的问句进行解析。而表现层则是针对算法层解析出的问句信息依据不同的回复策略进行回复并展示给用户。

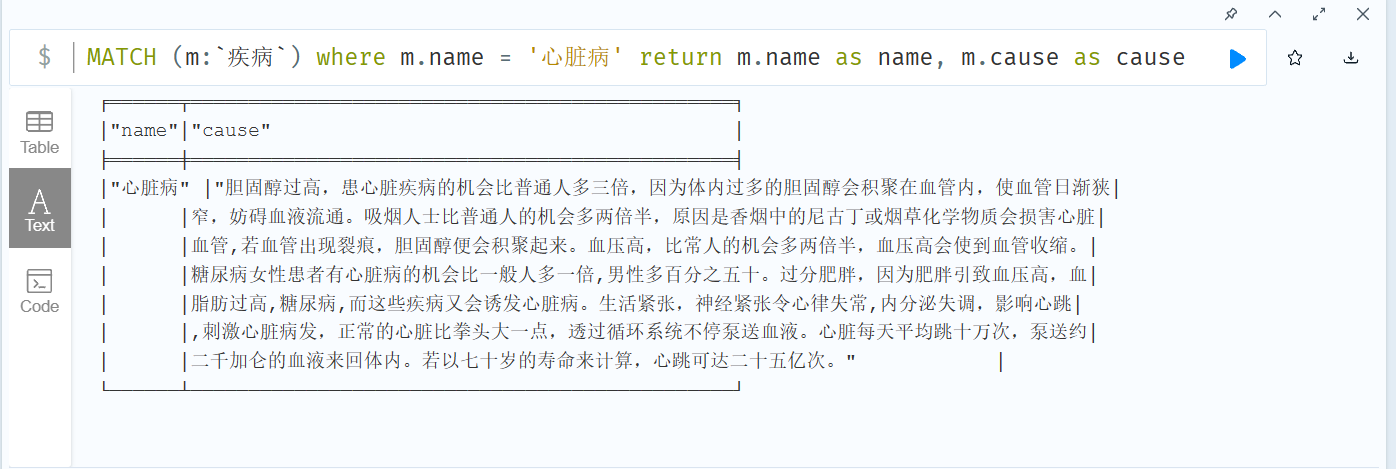


图 3-1 查询结果

（2）医疗问答系统测试

下面对于医疗问答系统整体的功能进行测试，用户输入提问语句，智能问答助手会在对话框中进行输出。

下面针对闲聊对话进行测试，如图3-2所示：

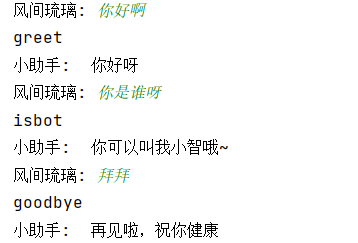


图 3-2 闲聊对话测试

上图是对智能问答机器人的闲聊测试，分别对其进行问好、询问以及再见，它都准确地预测出了闲聊意图并给予回复。

之后进行有关医疗问答的测试，测试时使用甲沟炎作为疾病实体。例如这里提问“甲沟炎是一种什么样的病”，此时智能问答机器人的回复如图3-3测试样例所示，由图可知该问句在第一次意图识别时被分为diagnose类，因此进入medical\_robot来进行第二次意图识别以及实体识别。通过打印出的信息可以看到，甲沟炎这一疾病实体信息被准确地识别，但是由于意图识别的置信度是0.61，此时的意图回复策略为clarify，因此机器人会向用户进行澄清。在得到肯定的回复后(accept1类别)，选择choice\_answer语句进行回复。

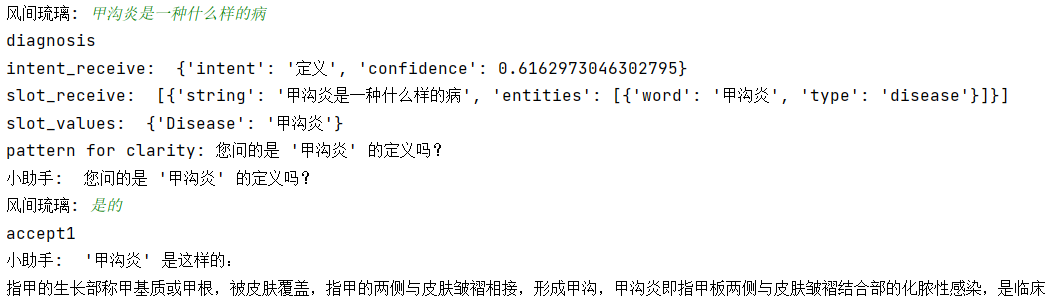


图 3-3 测试样例

由于每次重启问答系统都会将保存最近一次信息的json文件清空，当用户进行询问后，则会将回复的模板信息写入文件中，如图3-4所示：

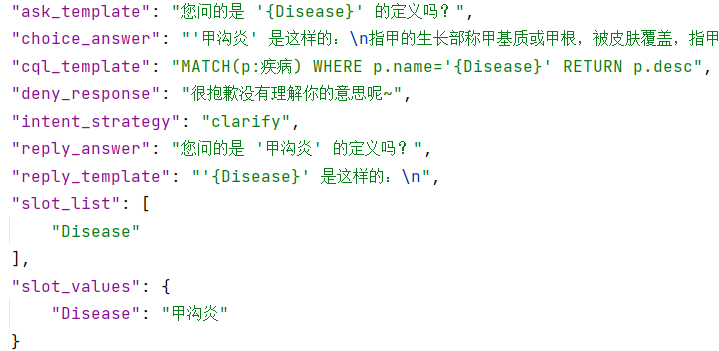


图 3-4 json文件信息

上图所示信息是进行多轮对话的关键，下面将进行多轮对话测试，具体如图3-5 所示：

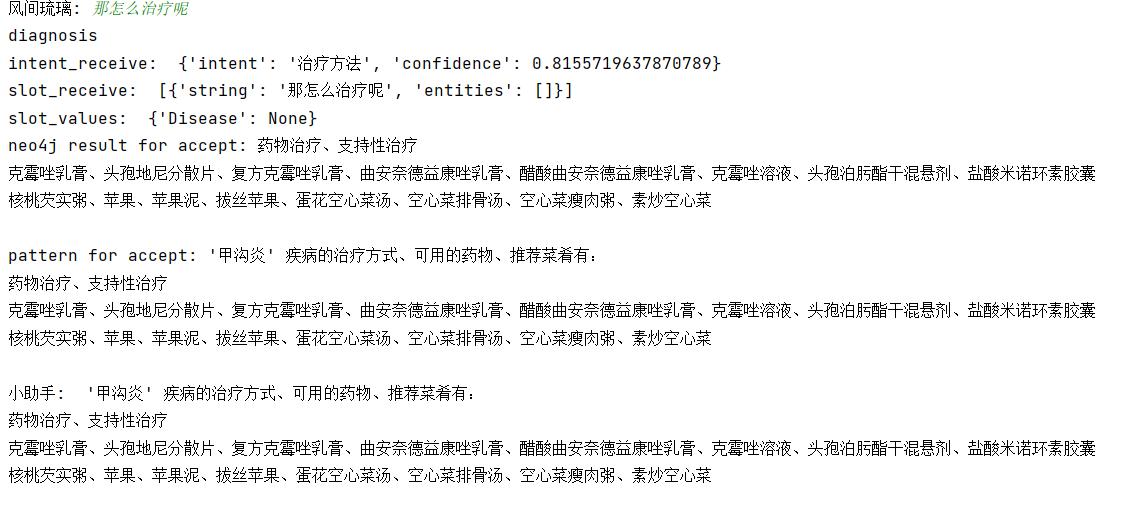


图 3-5 多轮对话测试

由图可知，输入的信息为“那怎么进行治疗呢”，这里并没有提到疾病的实体，因此这里的疾病实体使用最近一次对话中的实体“甲沟炎”。由于此次问句的意图识别置信度大于0.8，回复策略为accept，所以直接对查询语句进行填槽并使用对Neo4j图数据库进行查询。查询结果如图中“neo4j result for accept”所示，最终的输出是由填槽后的“reply\_template”以及“neo4j result for accept”组成，如“pattern for accept”。当进行到下一轮对话时，由于实体以及意图不同，json文件中的信息也进行了更新，更新后的模板使用治疗方法下的回复模板，并将上一轮提问中的实体填入相应的槽位，具体如图3-6所示：

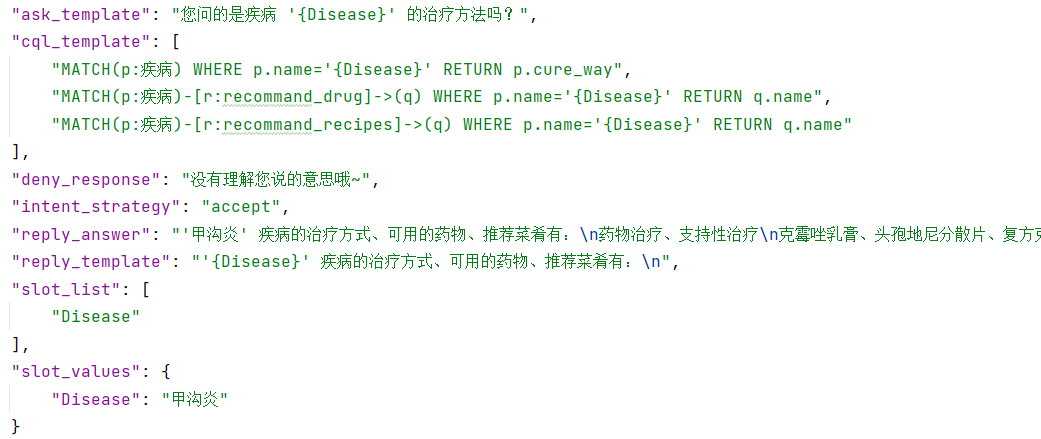


图 3-6 更新后的json文件

针对一些可能出现的状况，本项目也进行了基本的回复设置。例如识别到的疾病实体在数据库中没有查询到，则会回复“抱歉我不太知道您说的这种病，您可以问我其它的病哦”，如提问“内分泌紊乱是怎么回事”，即使已经识别到了疾病实体以及问句意图，但是由于数据库中没有该实体信息，所以会按照回复模板进行回答，如图3-7所示。如果存在该疾病的实体但是数据库中没有该疾病的相关属性，那么会按照“抱歉我不知道+<疾病>的+<问句意图>”，如存在心脏病的疾病实体但是并没有关于心脏病的治愈率，回复抱歉我不知道心脏病的治愈率。

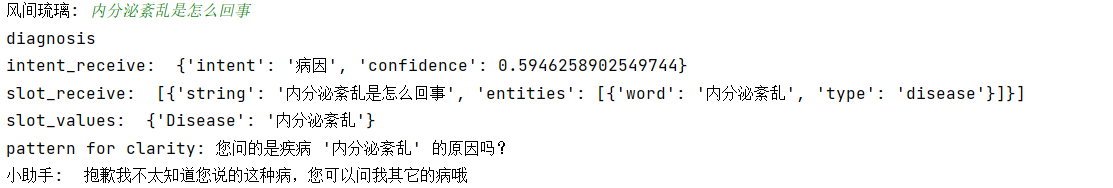


图 3-7 回复设置

**四、系统使用说明**

1、BiLSTM-CRF模型的模型参数存在knowledge\_extraction\bilstm\_crf\checkpoint下的best\_bilstm\_crf\_model.h5文件中。模型训练代码在此目录的train.py文件中，基于该模型的命名实体识别可以在app.py文件中运行，得出如3.1小节的实验结果。

2、BERT-TextCNN模型的模型参数存在nlu\bert\_intent\_recognition\checkpoint下的best\_model.weights文件中。模型训练代码在此目录的train.py文件中，基于该模型的命名实体识别可以在app.py文件中运行，得出如3.1小节的实验结果。

3、使用医疗问答系统首先要构建医疗知识图谱，需要配置neo4j图数据库，并配置相应的端口号、用户名、密码至build\_kg\_utils.py文件的MedicalExtractor函数里，并运行该文件，即可在neo4j图数据库中构建相应的结点以及关系。构建完成后在cmd命令行输入neo4j.bat console，在弹出的窗口中输入用户名和密码即可进入neo4j的可视化界面，输入相关命令来对结点以及相应的关系进行查找。

4、在使用医疗问答系统前需要打开本地的neo4j服务，具体步骤如3所描述。之后只需运行modules.py文件，即可在编辑器的对话框中进行提问。