智能问答系统

abel - 2022年7月4日



整体技术栈选择

整体的技术架构：

后端flask/flask-cors 实现API化；

前端bootstrap/vue/jquery/jinjia模版

知识图谱可视化部分：d3/echart/vue实现

对话系统模块：python3/numpy/sklearn/pandas/ChatterBot引擎

正常流程应该如下：

用户通过浏览器访问我们的wsgi host的网站前端网页，然后可以搜索关键词/输入初始问题，请求访问WSGI的API和中间逻辑层Application部分，我们的对话系统应该模块化，把训练好的模型load在WSGI的flask API层，我们的知识图谱可视化应该在Flask api层完成数据的动态加载，给网站的前端d3/echart的图谱可视化提供动态数据和关系维护：

图像

对话系统模块方案1:模版方法区分出时间/地点

初始问题进入，先可以使用简单的模版方法，进去区分：用户是询问时间、地点还是询问实体、实体属性或者其他，并提取问题中的关键词。其中询问实体部分就需要提前构建好知识图谱，具体是将独立的知识转化为三元组形式 。这一步模版方法逻辑先：

1. 区分中英文，从而api端进入不同的对话模型

（知道啦中英文后，我们后续对话部分，就可以知道是进入模型是corpus.english 还是chatterbot.chinese 哪一种开源数据集的模型啦）训练ChatterBot内嵌英文语料库

1. 根据模版方法区分是地点查询、时间查询，还是实体查询，还是其他，从而进入地点/时间的api 函数，或者进入下一轮对话模型

基于模板的问答方法属于比较传统的方法，设计者需要提前设定模板，依据问题相关部分，选取 不同模板，生成答案。这种方法优点是:可以获得比较 准确的答案，回答响应速度快。缺点:需要耗费大量人力进行模板校对，以及模板库维护。但是，针对问答领 域多跳复杂问题，最新的模板方法也能提供解决思路， 当前该方法研究重点更侧重于模板自动生成，克服耗时耗力难题。 这里我们只有用他来区分时间，地点和实体或者其他。模版可以使用json/python pickle存储，根据不同的量级。

对话系统模块方案2:知识图谱的知识抽取和查询

1.对于知识图谱需要的知识抽取，我们会从 **CausalityEventExtraction ComplexEventExtraction SequentialEventExtration** 3中选择1个，作为抽取算法

2. 最好设计成从已有的知网pdf中能自动ocr内容变成txt后，作为素材，传入效果最好的抽取算法中，然后抽取出3元组之后，代码支持导出为json/或者xml

3. 上面说过，实现了基于flask+echart+vue+d3实现知识图谱的可视化，也会设计用户系统（注册 登录，首页，上传json三元组的功能）

4. 数据输入方式：这里应该不是只知识图谱的上传，我觉得应该是需要提取的素材的上传，然后运行抽取算法，那么这里就先改成支持上传txt文件吧

（这里用户可以手动标注**3**元组，因为虽然可以用机器学习准确率找到最好的知识抽取算法，但是有时候还是需要手工标注一些**3**元组，去掉明显的错误）

5. 这部分可以帮助用户搜索或者问问题的时候，提供相关的知识图谱可视化，同时知识图谱的知识抽取的3元组可以帮助回答基于实体的问题，例如：用户的问题中出现啦“IOT是做什么的？”，如果在知识图谱的查询“IOT”中可以得到结果，我们的对话系统就不应该遵循上以基于已知对话的集合生成响应，而是进行知识图谱语义查询的优先级超过ChatterBot的自动生成的对话。

找知识库中的实体以及与实体相关的知识，在结构化形式的知识图谱上 进行查询，最高效的方法是利用结构化查询语句，类似 SQL、SPARQL 语句等，然而对普通用户来说，设计规范 的查询语句存在困难。基于知识图谱的语义解析问答 系统实现，需要两个关键步骤:(1)使用语义解析器将问 题转换成机器能够理解和运行的语义表示;(2)使用该语义产生结构化查询语言，对知识图谱进行查询，并从 返回的实体集合中寻找答案 .

比如语义分析：作者A和作者B共同编写的知网论文有哪些？

Select ?x where {

?x( 作者） （作者a），

?x(作者） （作者b),

}

在三元组中根据类Sql查找

MATCH p=(知识抽取)-[r:最早研究|综述研究|最新研究| 网络文献]->()  
RETURN p LIMIT 5  
 对检索结果限制在 5 条以内，检索结果如图 8(a)，从知网文献与网络文献的角度对知识抽取领域的研究成果进行 脉络梳理，帮助研究者快速了解领域概貌。除此之外，研究 人员也可根据几个相同关键词的标引，快速匹配出相似度最 高的文献，并以图的形式展示如图 8(b)，以免漏掉重要文献， 提高检索效率。

如果有精力，我们的所有提取的3元组的原始材料应该放入**Elasticsearch**中，我们可以查找的时候，提供出Elasticsearch的链接，溯源到原始的文档。

  

说明

对话系统模块方案3:对话上下文训练，基于机器学习的上下文对话，对付非时间/地点/实体的对话

我们总体方案已经选择啦ChatterBot后，框架支持创建了四个训练类，包括ListTrainer、ChatterBotCorpusTrainer、TwitterTrainer、UbuntuCorpusTrainer。

我们针对知网的知识对话，我们只需要使用头2种：

1、ListTrainer：训练样本由用户输入的string类型的语句列表构成，按照语句的先后顺序放在statement（专门存放对话语句的列表）中。

2、ChatterBotCorpusTrainer：训练样本由自身提供的语料库构成，按照语句先后顺序放在statement中。

3. 我们平时web在线对话系统保存下用户和机器人的对话，先根据中英文，分成2类，分别作为中文/英文语料，添加到**ListTrainer**的**list**或者**ChatterBotCorpusTrainer**的文件中，进行模型训练。

4. 训练好的模型作为模块，变成函数化可以调用，继承进入flask api端