HW7实验报告

1811516 余樱童

代码大致流程

1. 数据获取

本次数据获取采用了两种方式

（1）使用爬虫在网页上爬取文本信息，获得非结构化数据。再进行三元组抽取。

（2）半人工标注，从网页上获取表格型的数据（结构化数据），再用程序处理成三元组

2. 使用构建好的数据创建知识图谱
3. 用户输入问题，使用`sparql`查询语言从知识图谱中查找答案

本次代码使用的是`harvesttext`这个库，该库支持分词分句、内容清晰、实体链接等内容，GitHub网址如下https://github.com/blmoistawinde/HarvestText。

数据获取 & 可视化

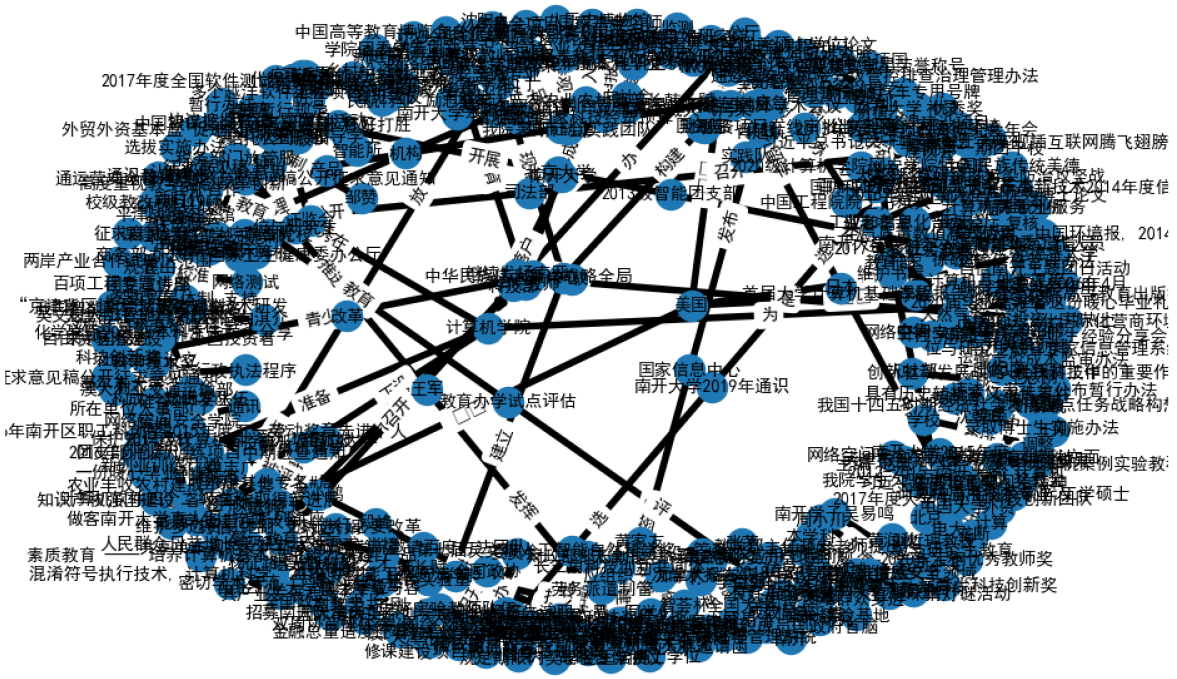
非结构化数据

爬虫部分还是和HW6差不多，主要爬取的是计算机学院相关的网页。重点是三元组抽取。

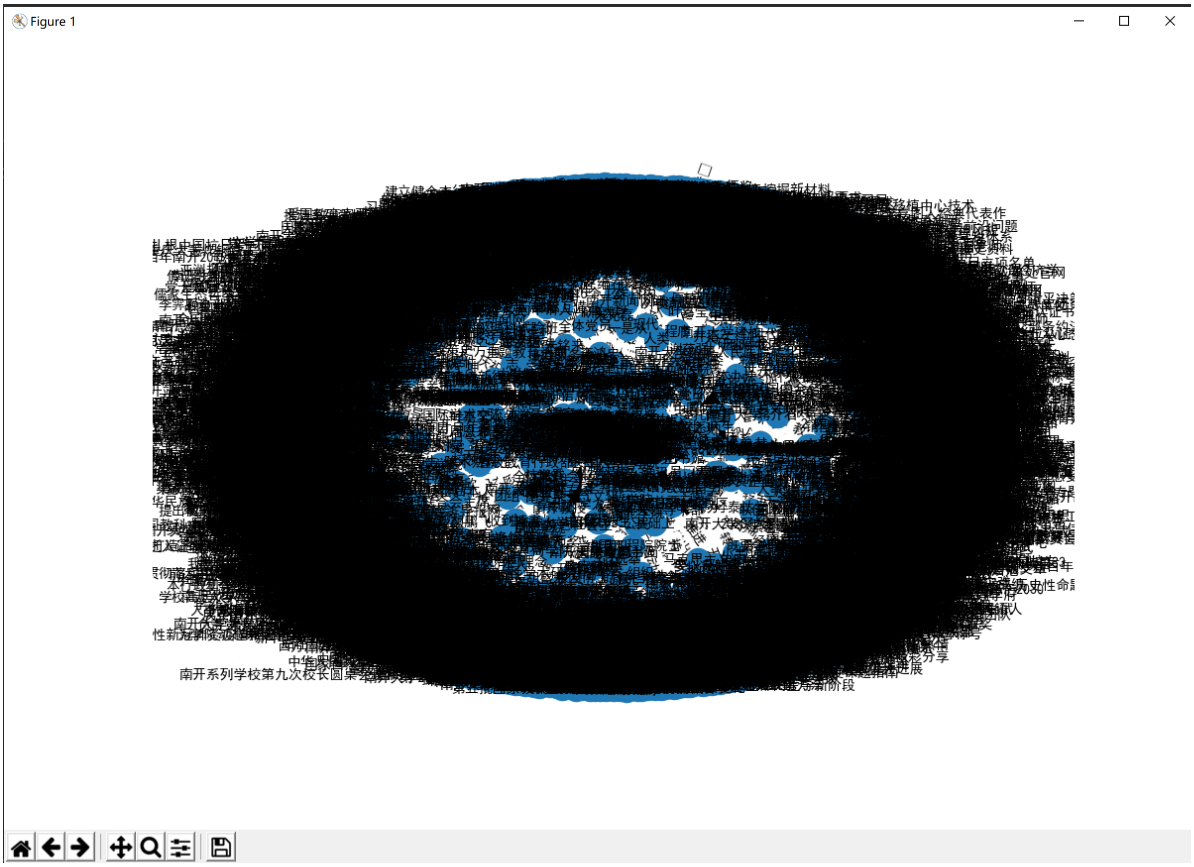
大致流程如下：

1. 从`harvesttext`这个库中，获得内置的清华领域词典，将词典中的内容作为“其他专名”加入到命名实体类别的词典中。
2. 对爬虫获得的数据，以一个句子为单位进行处理。对每个句子进行命名实体和类别的识别，将命名实体和类别加入到命名实体类别的词典中。
3. 将命名实体词典加入到`harvesttext`实例化出来的模型`ht`中。
4. 再次遍历爬虫的数据，对每一句进行处理，若句子符合一定条件，则进行三元组抽取，遍历抽取到的三元组，若该三元组的内容还算正常（代码中是判断抽取出来的sub和obj或者其一部分是不是实体），就加入三元组的列表中。

获得的知识图谱如下。



一开始是对南开大学相关网页进行的处理，最后获得的知识图谱如下。



因为数据太杂乱了，后面就改成了仅对计算机学院的内容进行处理。（但是因为数据量比较小，效果也不是很好。）

结构化数据

本部分，人工从网页上扒下来了计算机学院师资队伍的数据，因为网页上的数据是下面这样的。

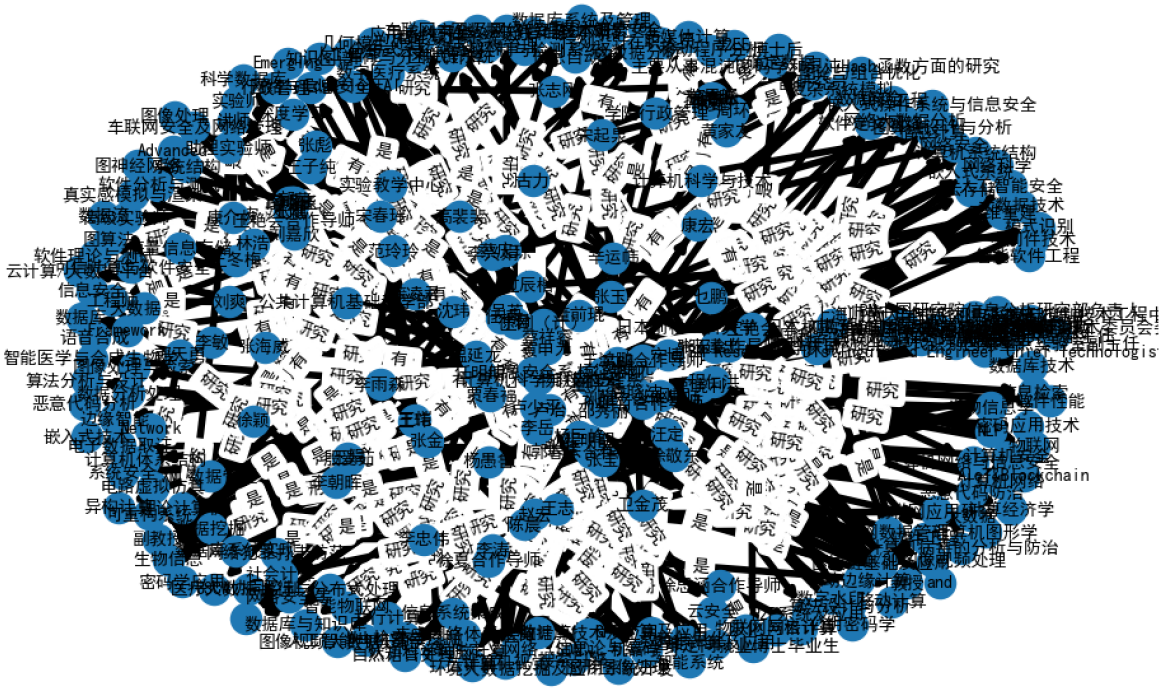
姓名	职称	所属部门	研究方向
白刚	教授	物联网工程系	人工智能，机器学习，模式识别，计算机视觉
程仁洪	教授	计算机科学与技术系	数据库技术及应用,软件性能,智能控制与检测系统
程明明	教授	计算机科学与技术系	人工智能、计算机视觉、图像视频大数据分析、计算机图形学
贾春福	教授	信息安全系	网络与信息安全；可信计算与软件安全；恶意代码分析; 密码应用技术
刘健	教授	计算机科学与技术系	大数据、人工智能、生物信息学、类脑计算、智能医学与合成生物学
刘哲理	教授	信息安全系	数据安全；人工智能安全
刘晓光	教授	计算机科学与技术系	云存储 搜索引擎 区块链系统 云计算/大数据平台 计算经济学
李庆诚	教授	计算机科学与技术系	嵌入式操作系统与信息安全
李涛	教授	物联网工程系	异构计算, 机器学习, 智能物联网
邵秀丽	教授	计算机科学与技术系	人工智能、数据分析、智能系统、CSCW协同控制
卫金茂	教授	计算机科学与技术系	机器学习，数据挖掘，生物信息学
汪定	教授	信息安全系	公钥密码学，系统安全，人工智能安全

非常整齐，处理起来也比较方便，所以就直接复制粘贴，后面放到代码中处理了。

处理完后的数据大概是下面的样子。

```
['蔡祥睿', '属于', '信息安全系']
['信息安全系', '有', '蔡祥睿']
['蔡祥睿', '研究', 'NLP']
['蔡祥睿', '研究', '机器学习']
['蔡祥睿', '研究', '深度学习']
['董晓东', '是', '讲师']
['董晓东', '属于', '计算机科学与技术系']
['计算机科学与技术系', '有', '董晓东']
['董晓东', '研究', '计算机网络']
['董晓东', '研究', '未来网络体系结构']
['董晓东', '研究', '软件定义网络']
['董晓东', '研究', '云计算']
['古力', '是', '讲师']
['古力', '属于', '信息安全系']
['信息安全系', '有', '古力']
['古力', '研究', '主要从事混沌密码学和混沌Hash函数方面的研究']
['过辰楷', '是', '讲师']
['过辰楷', '属于', '公共计算机基础教学部']
['公共计算机基础教学部', '有', '过辰楷']
['过辰楷', '研究', '智能软件工程']
['过辰楷', '研究', '移动程序分析']
['过辰楷', '研究', '医疗大数据']
['高裴裴', '是', '讲师']
['高裴裴', '属于', '公共计算机基础教学部']
['公共计算机基础教学部', '有', '高裴裴']
['高裴裴', '研究', '情感计算']
['高裴裴', '研究', '语音合成']
['康宏', '是', '讲师']
['康宏', '属于', '物联网工程系']
['物联网工程系', '有', '康宏']
['康宏', '研究', '数据库技术']
['康宏', '研究', '中间件技术']
['康宏', '研究', 'J2EE']
['李兴娟', '是', '讲师']
['李兴娟', '属于', '计算机科学与技术系']
['计算机科学与技术系', '有', '李兴娟']
['李兴娟', '研究', '网络安全']
['李兴娟', '研究', '']
['李兴娟', '研究', '无线和移动网络']
['李兴娟', '研究', '应用系统开发']
['李敏', '是', '讲师']
```

知识图谱是下面的样子（相当规整）。



知识图谱构建

知识图谱的图使用的是`rdflib`库。

遍历所有三元组，直接`add`就加到图里了。

代码如下：

```
# 创建知识图谱
def build_KG(self, SVOs, ht_SVO):
    namespace0 = Namespace(self.default_namespace)  # 创建命名空间
    g = Graph()  # 建图
    type_word_dict = {"实体":set(),"谓词":set()}
    # 遍历所有的三元组，扩充词典并且将其加入图中
    for (s,v,o) in SVOs:
        type_word_dict["实体"].add(s)
        type_word_dict["实体"].add(o)
        type_word_dict["谓词"].add(v)
        g.add((namespace0[s], namespace0[v], namespace0[o]))
    ht_SVO.add_typed_words(type_word_dict)
    return g
```

问答系统

问答系统主要由以下几个部分组成。

```
类成员：
    ht_SVO              harvesttext的实例化模型
    ht_e_type           harvesttext的实例化模型
    q_type2templates    词典，根据提供的实体类型确定回答的模板
```

```
    q_type2search         词典，根据提供的实体类别，改变参数，调用获得sparql查询语句的函数
    q_template2answer     词典，根据回答的模板，获得生成回答的函数
类函数：
    get_sparql(self, x=None, y=None, z=None, limit=None)
        将参数填入sparql查询语句中空缺的位置，获得sparql查询语句
    get_default_answer(self, x = "", y = "", z = "")
        生成回答的函数的一种（问句中什么都没抽取到时，使用这个函数）
    get_default_answers(self, entities, answers)
        生成回答的函数的一种（answers是一个列表，若非空，这个函数将其转换为由"、"连接的句子）
    parse_question_SVO(self, question, pinyin_recheck=False, char_recheck=False)
        对问句进行解析，返回问句关键实体及其类型
    extract_question_e_types(self, question, pinyin_recheck=False,
char_recheck=False)
        将问句中的实体用该实体的类别替换
    match_template(self, question, templates)
        使用最短编辑距离查找最匹配的模板
    search_answers(self, search0)
        使用库函数，执行sparql语句
    add_template(self, q_type, q_template, answer_function)
        增加问题的模板
    answer(self, question, pinyin_recheck=False, char_recheck=False)
        核心函数。调用上面各种函数来获得当前问题的答案。
```

获得一个问题的答案的流程大致如下：

1. 对问句进行实体链接，获得问句的关键实体及其类别。
2. 根据抽取到的问句的关键实体的类别，调用`get_sparql()`函数，并更改相应的参数。
3. 执行sparql查询语句获得结果。
4. 根据问题获得所有对应的回答模板。
5. 使用最短编辑距离获得最匹配的模板。
6. 根据模板输出回答。

根据三元组的内容，程序员也可以在代码中加入问题的模板

例如：

```
answer_func = lambda entities, answers: "他" + "、".join("".join(x) for x in
answers)
QA.add_template(("#实体#",), "#人名#干了哪些事？", answer_func)
```

当输入类似“张三干了哪些事？”样式的问题时，回答的模板将会类似于“他做完了信息检索第七次作业”。

结果展示

非结构化数据

```
问：刘哲理教授荣获什么？
答：2020年宝钢优秀教师奖
问：学院第一期团干校和开学典礼是什么关系？
答：举行
问：龚克干了哪些事？
答：他会见微软亚洲研究院副院长。
问：exit
退出成功
```

结构化数据

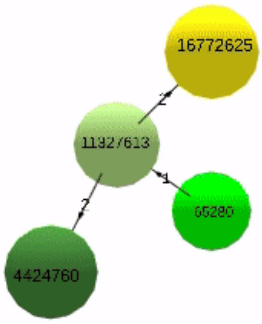
```
Yingtong Yu@DESKTOP-TFSO748 MINGW64 /d/课程/大三上/信息检索/hw7
$ D:/DBIS/ToUsePyTorch/Anaconda3/envs/ei_env/python.exe d:/课程/大三上/信息检索/hw7/KG.py
Building prefix dict from the default dictionary ...
Loading model from cache C:\Users\YINGTO~1\AppData\Local\Temp\jieba.cache
Loading model cost 0.871 seconds.
Prefix dict has been built successfully.
问：杨巨峰老师研究什么？
答：计算机视觉、多媒体计算、机器学习
问：杨巨峰老师属于什么？
答：计算机科学与技术系
问：程明明老师的职称是什么？
答：教授
问：程明明老师研究什么？
答：计算机图形学、人工智能、计算机视觉、图像视频大数据分析
问：谁研究计算机视觉？
答：杨巨峰、任博、白刚、王恺、徐君、李岳、程明明
问：exit
退出成功
```

两种方式对比之后，发现其实对非结构化数据进行三元组抽取效果不太好~~（奇差无比）~~，因为`harvesttext`使用的是无监督的抽取，而且虽然缩小了网站的范围，爬取的内容还是开放领域的，数据也比较少，所以抽取效果不是很好，三元组内容有点奇怪。

相比之下，结构化数据的问答效果就很好，基本上满足一定的格式，就是有问必答。

跑题

本来是想用`zincbase`这个库来做3D的可视化的，效果可能类似下面这样。



但是调用库中的函数时，`publish redis`一直失败，最后就不了了之了。:-X