

# 2021-2022 学年

# 知识工程专题实践报告

任课教师：吴天星

院 系 人工智能学院

专 业 人工智能

组 别 第三组

题目 面向中文的足球领域知识图谱构建

## 小组成员名单

学号	姓名
58119213	徐丹颖
58119123	黄振鹏
58119311	张静涵
58119318	杨喆
58119321	李雨辰
58119329	王兆阳

# 目录

I Data Preparation.....	6
一、实验目的.....	6
二、实验环境 .....	6
三、实验步骤 .....	6
II Fact Extraction .....	13
一、实验目的.....	13
二、实验环境 .....	13
三、实验步骤 .....	13
四、实验结果与分析 .....	15
III Type Inference.....	18
一、实验目的.....	18
二、实验环境.....	18
三、实验步骤 .....	18
IV Knowledge Fusion.....	19
一、实验目的.....	19
二、实验思路 .....	19
三、代码实现 .....	19
V Ontology Building.....	22

一、实验目的.....	22
二、实验步骤 .....	23
三、实验结果与分析 .....	27
VI Applications.....	30
一、实验目的.....	31
二、实验步骤 .....	31
三、代码实现 .....	35
四、实验结果与分析 .....	38

# 实验分工

- 数据准备、事实抽取：杨喆，徐丹颖，王兆阳，黄振鹏
- 类型推断：李雨辰，张静涵
- 知识融合：杨喆
- 本体构建：李雨辰，张静涵
- 应用-知识查询：李雨辰，杨喆，徐丹颖，王兆阳，黄振鹏
- 应用-可视化：张静涵

# I Data Preparation

## 一、实验目的

为了构建领域知识库，我们首先需要准备相应的数据来为提供数据支撑。为此本次实验中，我们的数据主要分三类进行爬取。为了获取西甲数据，我们爬取了结构化数据，半结构化数据和非结构化数据。

## 二、实验环境

编程语言 python 3.x

依赖库：request, xpath, beautifulsoup4

## 三、实验步骤

### 1. 半结构化数据

首先对于半结构化数据我们主要是从维基百科和百度百科爬取了数据。

维基百科部分首先从西甲维基结果页面的表格作为球队的信息的入口，如下：

参赛球队 [编辑]			
2021–22年西班牙足球甲级联赛参赛队伍共有 20 支。			
中文名称	英文名称	所在城市	上季成绩
体育马德里	Atlético de Madrid	马德里	第 1 位
皇家马德里	Real Madrid C.F.	马德里	第 2 位
巴塞罗那	FC Barcelona	巴塞罗那	第 3 位
塞维利亚	Sevilla FC	塞维利亚	第 4 位
皇家社会	Real Sociedad	圣塞瓦斯蒂安	第 5 位
贝蒂斯	Real Betis	塞维利亚	第 6 位
比利亚雷亚尔	Villarreal CF	比利亚雷亚尔	第 7 位
塞尔塔	Celta de Vigo	维戈	第 8 位
格拉纳达	Granada CF	格拉纳达	第 9 位
毕尔巴鄂	Athletic Bilbao	毕尔巴鄂	第 10 位
奥萨苏纳	CA Osasuna	潘普洛纳	第 11 位
加的斯	Cádiz CF	加的斯	第 12 位
瓦伦西亚	Valencia CF	巴伦西亚	第 13 位
莱万特	Levante UD	巴伦西亚	第 14 位
赫塔菲	Getafe CF	赫塔菲	第 15 位
阿拉维斯	Deportivo Alavés	维多利亚	第 16 位
埃尔切	Elche CF	埃尔切	第 17 位
西班牙人	RCD Espanyol	巴塞罗那	乙组，第 1 位
马略卡	RCD Mallorca	帕尔马	乙组，第 2 位
巴列卡诺	Rayo Vallecano	马德里	乙组，附加赛胜方

在该表格获取每个球队的名称和球队链接：

```

html = etree.HTML(r.text)
table = html.xpath('//*[@id="mw-content-text"]/div[1]/table[2]/tbody//tr')
print(len(table))
teams = []
for i in range(20):
    tr = table[1+i]
    tname = tr.xpath('string(.)').split()[0]
    td = tr.xpath('.//td[1]//a[@href')[0][5:]
    teams.append((tname, td))
print(teams)

```

接下来爬取每个球队的 infobox，每个球队的 infobox 如下：



通过 xpath 解析元素，将每行的表头作为字典的键，表的值作为字典的值，保存到队伍的信息当中。

```

teamsBreatInfo = {}
for i in range(len(teams)):
    # for i in range(1):
        print(teams[i][0])
        teamurl = url + teams[i][1]
        print(teamurl)
        res = requests.get(teamurl)
        html = etree.HTML(res.text)
        breafinfo = {}
        trnum = html.xpath('//*[@class="infobox vcard"]/tbody//tr').__len__()
        for j in range(2, trnum-4):
            tr = html.xpath('//*[@class="infobox vcard"]/tbody//tr['+str(j)+'])[0]
            th = tr.xpath('.//th[1]//text()')[0]
            td = tr.xpath('.//td[1]')[0].xpath('string(.)')
            breafinfo[th] = td
        # print(breafinfo)
        teamsBreatInfo[teams[i][0]] = breafinfo
print(teamsBreatInfo)

```

数据保存如下：

'体育马德里': {'全名': 'Club Atlético de Madrid', '绰号': 'Los Colchoneros (床单军团) Los Rojiblancos (红白军团)', '成立': '1903 年 4 月 26 日, \u2000b118 年前'}

\u200b (1903-04-26) [1]', '城市': '西班牙马德里', '主场': '万达大都会球场', '容纳人数': '68,456[2]', '拥有者': '米盖尔·安赫尔·希尔·马林(英语:Miguel Ángel Gil Marín)(51%)伊丹·奥佛(英语:Idan Ofer)(30%)恩里克·塞雷佐(英语:Enrique Cerezo)(19%)[3][4][5]', '主席': '恩里克·塞雷佐(英语:Enrique Cerezo)', '主教练': '迭戈·西蒙尼', '联赛': '西班牙足球甲级联赛', '2020-21': '西甲, 第1位'}

对于百度百科的半结构化数据，处理方法和维基百科一致，首先在西甲联赛页面获取所有球队的链接：



接下来在每个球队的页面获取到该球队的 infobox 如下：

中文名	巴塞罗那足球俱乐部	主要荣誉	欧冠冠军 (6次)
外文名	FC Barcelona		西甲冠军 (26次) [4]
成立时间	1899年11月29日		国王杯冠军 (31次) [4]
所属地区	西班牙巴塞罗那	容纳人数	98260人
运动项目	足球	主席	豪安·拉波尔塔
角逐赛事	西班牙足球甲级联赛	现任队长	塞尔吉奥·布斯克茨 [4]
主场馆	诺坎普球场	队歌	Cant del Barça
现任主教练	哈维·埃尔南德斯	昵称	巴萨 (Barça) [114]
知名人物	莱奥·梅西、哈维·埃尔南德斯、罗纳尔迪尼奥、约翰·克鲁伊夫、里瓦尔多、佩普·瓜迪奥拉		

处理方式和维基百科的方式一致：

```

baseurl = 'https://baike.baidu.com'
team_dict = {}
for i in range(len(teamList)):
    team_url = baseurl + teamList[i][1]
    print(teamList[i], team_url)
    team_r = requests.get(team_url, headers=headers)
    team_r.raise_for_status()
    team_r.encoding = 'utf-8'
    team_html = etree.HTML(team_r.text)
    dt = team_html.xpath('/html/body/div[1]/div[2]/div/div[1]/div[2]/dt')
    dd = team_html.xpath('/html/body/div[1]/div[2]/div/div[1]/div[2]/dd')
    data_dict = {}
    print(i)
    if len(dt) == 0:
        print(f'this team has no information: {teamList[i][0]}')
    else:
        for j in range(len(dt)):
            data_dict[''.join(dt[j].xpath("string(.)").split())] = ''.join(dd[j].xpath("string(.)").split())
    team_dict[teamList[i][0]] = data_dict
print(team_dict)

```

最后数据保存形式如下：

'毕尔巴鄂竞技足球俱乐部': {'中文名': '毕尔巴鄂竞技俱乐部', '外文名': 'Athletic Bilbao', '成立时间': '1898年', '所属地区': '西班牙毕尔巴鄂', '运动项目': '足球', '角逐赛事': '西班牙足球甲级联赛', '主场馆': '新圣马梅斯球场[2]', '现任主教练': '爱德华多·贝里索', '知名人物': '略伦特、哈维·马丁内斯、伊劳拉、安德尔、埃切贝里亚、乌尔塞斯', '容纳人数': '53,289人', '主席': 'Josu Urrutia', '绰号': '红白箭条、巴斯克雄狮'}

## 2. 非结构化数据

对于非结构化数据，主要是获取每个球队的简介，在后期对简介进行三元组抽取。在这部分，对于每个球队的链接获取方式和上面 infobox 的球队的链接方式一致；而球队的简介如下：

马德里竞技足球俱乐部

10

撤销 | 编辑 | 讨论 | 上传视频

马德里竞技俱乐部 (Club Atlético de Madrid) 是一家位于西班牙马德里的足球俱乐部，成立于1903年4月26日。球队主场为万达大都会球场。 [8] 其一线队参加西班牙足球甲级联赛。 [1]

1903年4月26日，马德里竞技作为毕尔巴鄂竞技的青年队分支由居住在马德里的三名巴斯克学生创立。1921年，马德里竞技正式从毕尔巴鄂竞技独立。1940年，马德里竞技首次夺得西甲联赛冠军。1962年，马德里竞技首次夺得欧洲赛事冠军。自2002年起，马德里竞技一直征战于西班牙顶级联赛。 [9]

马德里竞技曾夺得11次西甲联赛冠军、3次欧冠亚军、3次欧罗巴联赛冠军、3次欧洲超级杯冠军、1次洲际杯冠军。 [1]

通过 BeautifulSoup 解析页面获取到该简介的位置和内容如下：

```
def getIntro(url, obj=''):
...
    获取页面infobox信息
    参数url-页面链接，obj-页面标题
    返回值：属性键值对字典
    ...
    html_text = url
    #print(html_text)
    soup = BeautifulSoup(html_text, 'html.parser')

    title_node = soup.find('dd', class_="lemmaTitle-title J-lemma-title")
    title = title_node.get_text()
    print(title_node.get_text())

    # 找到intro
    if soup.find(attrs={"name": "description"}):
        intro = soup.find(attrs={"name": "description"})['content']
        print("找到intro了:\n", intro)
        #intro=intro.text
        return title, intro
    else:
        print('该页面无intro—', obj)
        return
```

最后保存结果如下：

```
巴塞罗那足球俱乐部
找到intro了：
巴塞罗那足球俱乐部（FC Barcelona），昵称“巴萨（Barça）”，是一家位于西班牙巴塞罗那的足球俱乐部，于1899年11月29日由瑞士人胡安·甘伯创立。球队主场诺坎普球场可容纳接近十万名观众，是全欧洲最大及世界第二大的足球场。巴塞罗那是西班牙足球甲级联赛传统豪门之一，在西班牙国内共赢得26次西甲联赛冠军、31次国王杯（在国王杯历史上高居榜首）、13座西班牙超级杯、3座伊娃杯和2座西班牙联赛杯；在国际上，共赢得了5座欧冠奖杯、4座欧洲优胜者杯、3座国际城市博览会杯、5座欧洲超级杯和3座世俱杯。在IFFHS国际俱乐部排行榜中，巴萨在1997年、2009年、2011年、2012和2015年均排名第一位。
```

对于维基的数据也是一样的处理方式，不再赘述。

除了球队的信息以外，我们还获取了球员的信息。首先是从球队的页面获取到球员的列表如下：

2021-22赛季一线队球员列表				
号码	球员姓名	主要位置	出生日期	FIFA国籍
1	尼古拉·伊万斯	门将	1991-04-26	法国
2	何塞·马里亚·阿门蒂斯	后卫	1995-01-20	乌拉圭
4	杰弗里·孔多比亚	后卫	1993-02-15	中非
5	罗德里戈·博尔热斯	后卫	1994-05-24	阿根廷
6	托尼	中场	1992-01-08	西班牙
7	若昂·普利克斯	前锋	1999-11-10	葡萄牙
8	安托万·格列兹曼	前锋	1991-03-21	法国
9	路易斯·苏亚雷斯	前锋	1987-01-24	乌拉圭
10	安赫尔·科雷亚	前锋	1995-03-09	阿根廷
11	托马斯·勒马尔	中场	1995-11-12	法国
12	雷南·纳沙	后卫	1998-04-08	巴西
13	扬·莫拉塔	门将	1993-01-07	斯洛文尼亚
14	马科斯·略伦特	中场	1995-01-30	西班牙
15	斯特凡·伊维塞维奇	后卫	1991-01-08	黑山
16	埃克托·埃雷拉	中场	1990-04-19	墨西哥
17	丹尼尔·瓦斯	后卫	1989-05-31	丹麦
18	盖列罗	后卫	1989-05-16	巴西
19	马特乌斯·库尼亞	前锋	1999-05-27	巴西
21	西梅奥内·卡拉斯科	前锋	1993-09-04	比利时
22	马里奥·埃尔莫索	后卫	1995-06-18	西班牙
23	霍纳尔多·蒙达瓦	后卫	1994-01-21	莫桑比克
24	西梅·费尔南利科	后卫	1992-01-10	克罗地亚

通过 BeautifulSoup 解析网站获取到该表格如下：

```
#号码
star["number"] = all_tds[0].text
#个人百度百科链接
#star["link"] = 'https://baike.baidu.
#姓名
star["name"] = all_tds[1].text
#国籍
star["country"] = all_tds[2].text
#生日
star["birthday"] = all_tds[3].text
#场上位置
star["location"] = all_tds[4].text
```

最终将数据保存下来。

### 3. 结构化数据

对于结构化数据，我们使用了领域知识库 FBref (<http://fbref.com/en/>) 作为数据补充。在领域知识库 FBref 中，我们得到了西甲联赛各足球俱乐部的球队信息，例如巴塞罗那足球俱乐部：



## 2021-2022 Barcelona Stats (La Liga)

[« Previous Season](#)

**Record:** 19-9-6, 66 points (1.94 per game), 2nd in La Liga (1st Tier)  
**Home Record:** 11-2-4, 35 points **Away Record:** 8-7-2, 31 points  
**Goals:** 63 (1.85 per game), **Goals Against:** 34 (1.00 per game), **Diff:** 29  
**xG:** 59.0, **xGA:** 33.2, **Diff:** 25.8  
**Last Match:** Win 2-1 vs Mallorca  
**Next Match:** Saturday, May 7 at Real Betis  
**Manager:** Xavi  
**Governing Country:** Spain   
**Gender:** Male  
**Champions League:** 3rd place in Group stage (Advanced to Europa League Knockout round play-offs)  
**Europa League:** Lost Quarter-finals to Eintracht Frankfurt  
**Copa del Rey:** Lost Round of 16 to Athletic Club  
**Supercopa de España:** Lost Semi-finals to Real Madrid

Barcelona Stats & History	Stats by Competition ▾	Player Career Details	Match Logs ▾	Affiliated Squads ▾
On this page:				
<a href="#">Standard Stats</a>	<a href="#">Scores &amp; Fixtures</a>	<a href="#">Goalkeeping</a>	<a href="#">Advanced Goalkeeping</a>	<a href="#">Shooting</a>
<a href="#">Goal and Shot Creation</a>	<a href="#">Defensive Actions</a>	<a href="#">Possession</a>	<a href="#">Playing Time</a>	<a href="#">Miscellaneous Stats</a>
			<a href="#">Passing</a>	<a href="#">Regular Season, La Liga</a>
				<a href="#">Pass Types</a>
				<a href="#">Full Site Menu</a>

同时，我们也得到了大量的球员信息：

Player	Nation	Pos	Age	Playing Time			Performance						Per 90 Minutes			Expected			Per 90 Minutes										
				MP	Starts	Min	90s	Gls	Ast	G-PK	PK	PKatt	CrdY	CrdR	Gls	Ast	G+A	G-PK	G+A-PK	xG	npxG	xA	npxG+xA	xG	xA	xG+xA	npxG	npxG+xA	Mat
Sergio Busquets	ESP	MF	33-294	33	33	2,928	32.5	2	0	2	0	0	9	0	0.06	0.00	0.06	0.06	0.6	0.6	1.9	2.5	0.02	0.06	0.08	0.02	0.08	Mats	
Marc-André ter Stegen	GER	GK	30-006	32	32	2,880	32.0	0	0	0	0	0	3	0	0.00	0.00	0.00	0.00	0.0	0.0	0.1	0.0	0.00	0.00	0.00	0.00	0.00	Mats	
Jordi Alba	ESP	DF	33-046	27	27	2,374	26.4	1	9	1	0	0	9	0	0.04	0.34	0.38	0.04	0.38	1.2	1.2	5.4	6.6	0.04	0.21	0.25	0.04	0.25	Mats
Frenkie de Jong	NED	MF	24-359	29	27	2,206	24.5	3	3	3	0	0	6	1	0.12	0.12	0.24	0.12	0.24	4.5	4.5	3.0	7.5	0.18	0.12	0.31	0.18	0.31	Mats
Gerard Piqué	ESP	DF	35-093	27	25	2,092	23.2	1	0	1	0	0	10	1	0.04	0.00	0.04	0.04	0.04	2.1	2.1	0.1	2.1	0.09	0.00	0.09	0.09	0.09	Mats
Gavi Pezz	ESP	MF/PW	17-274	30	24	1,988	22.1	2	5	2	0	0	10	1	0.08	0.23	0.32	0.09	0.32	3.9	3.9	3.1	7.0	0.18	0.14	0.32	0.18	0.32	Mats
Ronald Araújo	URU	DF	23-060	27	22	2,025	22.5	4	0	4	0	0	6	0	0.18	0.00	0.18	0.18	0.18	1.4	1.4	0.4	1.8	0.06	0.02	0.08	0.06	0.08	Mats
Eric García	ESP	DF	21-117	24	21	1,859	20.7	0	0	0	0	0	4	1	0.00	0.00	0.00	0.00	0.02	0.2	0.2	0.2	0.3	0.01	0.01	0.02	0.02	0.02	Mats
Memphis Depay	NED	FV	28-082	24	17	1,633	18.1	11	2	7	4	5	3	0	0.61	0.11	0.72	0.39	0.50	10.5	6.7	3.8	10.5	0.58	0.21	0.79	0.37	0.58	Mats
Sergiño Dest	USA	DF/FW	21-184	21	17	1,512	16.8	0	3	0	0	0	2	0	0.00	0.18	0.18	0.00	0.18	1.5	1.5	2.8	4.3	0.09	0.16	0.26	0.09	0.26	Mats
Ousmane Dembélé	FRA	M/PDF	24-356	18	13	2,121	13.5	1	1	1	0	0	3	0	0.07	0.82	0.89	0.07	0.89	2.8	2.8	7.0	9.8	0.20	0.52	0.73	0.20	0.73	Mats
Ferrán Torres	ESP	FV	22-066	14	13	1,133	12.6	4	4	3	1	1	1	0	0.32	0.32	0.64	0.24	0.56	6.5	5.8	2.8	8.6	0.52	0.23	0.74	0.46	0.68	Mats
Nicolás González	ESP	MF	20-123	27	12	1,114	12.4	2	1	2	0	0	6	0	0.16	0.08	0.24	0.16	0.24	1.4	1.4	1.2	2.6	0.11	0.10	0.21	0.11	0.21	Mats
Pedri	ESP	MF	19-162	12	10	885	9.8	3	1	3	0	0	0	0	0.31	0.10	0.41	0.31	0.41	1.1	1.1	1.1	2.2	0.11	0.11	0.22	0.11	0.22	Mats
Pierre-Emerick Aubameyang	GAB	FV	32-322	13	10	858	9.5	9	1	9	0	0	0	0	0.94	0.10	1.05	0.94	0.15	6.1	6.1	1.0	7.1	0.64	0.10	0.75	0.64	0.75	Mats
Dani Alves	BRA	DF	39-000	10	9	750	8.3	1	2	1	0	0	1	1	0.12	0.24	0.36	0.12	0.36	0.1	0.1	1.5	1.6	0.02	0.18	0.19	0.02	0.19	Mats
Oscar Mingueza	ESP	DF	22-358	16	8	792	8.0	0	1	0	0	0	2	0	0.00	0.11	0.11	0.00	0.11	0.2	0.2	0.9	1.1	0.02	0.10	0.12	0.02	0.12	Mats
Luuk de Jong	NED	FV	31-252	19	6	627	7.0	6	0	6	0	0	1	0	0.88	0.08	0.86	0.86	0.86	3.6	3.6	2.0	3.6	0.52	0.03	0.54	0.52	0.54	Mats
Abdeslam Ezzaïzouli	MAR	FV/PD	20-140	10	6	580	6.4	1	0	1	0	0	4	0	0.16	0.00	0.16	0.16	0.16	1.3	1.3	0.6	1.9	0.20	0.09	0.29	0.20	0.29	Mats
Clément Lenglet	FRA	DF	26-323	18	5	623	6.9	0	1	0	0	0	3	0	0.00	0.14	0.14	0.00	0.14	0.0	0.0	0.6	0.01	0.08	0.09	0.01	0.09	Mats	
Philippe Coutinho	BRA	M/PFW	29-328	12	5	503	5.6	2	0	1	1	1	1	0	0.38	0.00	0.36	0.18	0.18	2.8	2.8	0.3	2.4	0.58	0.06	0.56	0.36	0.42	Mats
Sergi Roberto	ESP	M/PDF	30-088	9	4	400	4.4	2	1	2	0	0	1	0	0.45	0.23	0.68	0.45	0.68	1.6	1.6	0.3	2.0	0.38	0.08	0.44	0.36	0.44	Mats
Jutglà	ESP	FV	23-094	6	4	301	3.3	1	0	1	0	0	1	0	0.30	0.00	0.30	0.30	0.30	1.6	1.6	0.4	2.0	0.47	0.13	0.60	0.47	0.60	Mats
Adama Traoré	ESP	FV/PD	26-101	9	3	309	3.4	0	2	0	0	0	0	0	0.08	0.58	0.58	0.00	0.58	0.6	0.6	0.6	1.2	0.17	0.19	0.36	0.17	0.36	Mats
Antoine Griezmann	FRA	FV	31-046	3	3	263	2.9	0	0	0	0	0	0	0	0.00	0.00	0.00	0.00	0.00	0.4	0.4	0.0	0.4	0.12	0.00	0.12	0.12	0.12	Mats
Martin Braithwaite	DEN	FV	30-335	4	3	235	2.6	2	1	2	0	0	0	0	0.77	0.38	1.15	0.77	1.15	1.0	1.0	1.3	2.3	0.40	0.48	0.88	0.40	0.88	Mats
Ansu Fati	ESP	FV	19-187	6	3	229	2.5	3	0	3	0	0	0	0	1.18	0.00	1.18	1.18	1.18	1.1	1.1	0.7	1.6	0.43	0.27	0.70	0.43	0.70	Mats
Yusuf Demir	AUT	FV	18-338	6	2	192	2.1	0	0	0	0	0	0	0	0.00	0.00	0.00	0.00	0.00	0.1	0.1	0.0	0.1	0.03	0.01	0.04	0.03	0.04	Mats
Neto	BRA	GK	32-291	2	2	180	2.0	0	0	0	0	0	0	0	0.00	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.00	0.00	0.00	0.00	0.00	Mats
Sergio Agüero	ARG	FV	33-338	4	2	151	1.7	1	0	1	0	0	0	0	0.68	0.00	0.60	0.60	0.60	1.1	1.1	0.3	1.3	0.64	0.16	0.80	0.64	0.80	Mats

Sergio Busquets Overview	Stats by Competition ▾	Match Logs ▾	Scouting Report ▾	Player Comparison	Barcelona ▾
On this page:					
<a href="#">Scouting Report</a>					
<a href="#">Similar Players</a>					
<a href="#">Standard Stats</a>					
<a href="#">Player News</a>					
<a href="#">Shooting</a>					
<a href="#">Passing</a>					
<a href="#">Pass Types</a>					
<a href="#">Goal and Shot Creation</a>					
<a href="#">Defensive Actions</a>					
<a href="#">Possession</a>					
<a href="#">Additional Resources</a>					
<a href="#">Full Site Menu</a>					

FBREF Sergio Busquets Scouting Report					

在网页中我们对这些信息直接进行下载，并且调整其格式，信息，得到了如图所示的所有俱乐部球员和球队信息：

	球衣号码	姓名	国籍	生日	职位	所属球队	英文名
0	1	费尔南多	西班牙	1992. 5. 18	门将	阿拉维斯	Pere Pons
1	2	塔奇	西班牙	1997. 9. 10	后卫	球员	Toni Moya
2	3	鲁文·杜	西班牙	1995. 10. 1	后卫	阿拉维斯	Rubén Duarte
3	4	马特·米	美国	1995. 7. 19	后卫		Luis Rioja
4	5	比克托·	西班牙	1989. 11. 5	后卫		Víctor Laguardia
5	6	马马杜·	塞内加尔	1996. 12. 3	中场		Mamadou lum
6	7	马马杜·	塞内加尔	1994. 3. 20	前锋		Mamadou Sila
7	8	托马斯·	西班牙	1987. 10. 1	中场		Tachi
8	9	何塞卢	西班牙	1990. 3. 27	前锋		Ximo Navarro
9	10	约翰·圭	瑞典	1992. 4. 15	前锋		Martin Agirregabiria
10	11	路易斯·	西班牙	1993. 10. 1	中场		Facundo Pellistri
11	12	萨乌尔·	西班牙	1994. 11. 9	后卫		Manu García
12	13	安东尼奥	西班牙	1996. 8. 11	门将		Fernando Pacheco
13	14	马努·加	西班牙	1998. 1. 2	中场		Manu Garcia
14	15	托尼·莫	西班牙	1998. 3. 20	中场		Mamadou Sylla
15	17	埃德加·	西班牙	1990. 1. 2	中场		Gonzalo Escalante
16	18	法昆多·	乌拉圭	2001. 12. 2	中场		Fakundo peristri
17	19	伊万·马	西班牙	1999. 2. 14	中场		John Guidetti
18	20	佩雷·庞	西班牙	1993. 2. 20	中场		Perry Ponce
19	21	马丁·阿	西班牙	1996. 5. 10	后卫		Martin aguirregaviria
20	22	弗洛里安	法国	1991. 5. 20	后卫		Antonio Sivera
21	23	西莫·纳	西班牙	1990. 1. 23	后卫		Loum Ndiaye
22	24	米格尔·	西班牙	1999. 9. 3	前锋		Tomás Pina Isla
23	26	阿卜杜勒	摩洛哥	1999. 3. 10	后卫		Abdul abukar
24	27	哈维·洛	西班牙	2002. 3. 25	后卫		Matt Miazga
25	31	赫苏斯·	赤道几内	2001. 3. 1	门将		Miguel
26	35	阿兰·戈	西班牙	2003. 5. 4	前锋		Alan Godoy

## II Fact Extraction

### 一、实验目的

对维基百科的非结构化文本，以及半结构化数据和领域知识库 FBref 的表格数据进行信息提取，以补充 infobox 中缺失的信息

### 二、实验环境

编程语言：python3

依赖库：Python 标准库 re，百度翻译 API

百科在描述类似事物时常采用比较一致的表达方式，故采用正则表达式是比较高效的方式  
我们利用 python 的 re 模块编写正则表达式对 wiki 百科的非结构化文本进行抽取

### 三、实验步骤

#### 1. 非结构化文本

## (一) 正则表达式编写

对不同的属性和关系编写了多个正则表达式模板进行抽取。

### 1. 成立日期

‘成立于([0-9]{4}年)[,.]\*’,  
‘([0-9]{4}年)[成创]立[,.]\*’

### 2. 位置

‘位于(西班牙\S{3,25})的\S\*俱乐部[,.]’

### 3. 简称

‘简称(\S{1,10})[,.]’

### 4. 球队主场容纳量

‘容纳(\d\*,?\d\*)(人|观众)’

### 5. 球队主场名称

‘主场场馆为(\S\*)球场’,  
‘球[会队]主场([a-zA-Z\s]\*),’,  
‘,以([\S\s]\*)作?为主场’

### 6. 主要竞争对手

‘r’ 对手为\S\*的(\S\*),’

### 7. 球队主场

‘r’ 球[会队]的\*主场(\S{1,5}\s\*\S{1,5})球场’,  
‘r’ 球[会队]主场(\S{1,5}\s\*\S{1,5})体育场’,  
‘r’ 主场场馆为(\S\*)球场’,  
‘r’ 球[会队]主场([a-zA-Z\s]\*),’,  
‘r’,以([\S\s]\*)作?为主场’

## (二) 利用 re 模块的 search 函数得到目标匹配

基于前一部分爬虫所获取的文本与编写的正则表达式配合 re. search 函数进行模板匹配，抽取得到的结果见实验结果部分

## 2. 结构化数据与半结构化数据

对于这两种类型的数据，我们主要对领域知识库 FBref 的数据进行处理。

	球衣号码	姓名	国籍	生日	职位	所属球队	英文名
0	1	费尔南多	西班牙	1992.5.18	门将	阿拉维斯	Pere Pons
1	2	塔奇	西班牙	1997.9.10	后卫	球员	Toni Moya
2	3	鲁文·杜	西班牙	1995.10.1	后卫	阿拉维斯	Rubén Duarte
3	4	马特·米	美国	1995.7.19	后卫		Luis Rioja
4	5	比克托·	西班牙	1989.11.5	后卫		Víctor Laguardia
5	6	马马杜·	塞内加尔	1996.12.3	中场		Mamadou Lum
6	7	马马杜·	塞内加尔	1994.3.20	前锋		Mamadou Sila
7	8	托马斯·	西班牙	1987.10.1	中场		Tachi
8	9	何塞卢	西班牙	1990.3.27	前锋		Ximo Navarro
9	10	约翰·圭	瑞典	1992.4.15	前锋		Martin Agirregabiria
10	11	路易斯·	西班牙	1993.10.1	中场		Facundo Pellistri
11	12	萨乌尔·	西班牙	1994.11.9	后卫		Manu García
12	13	安东尼奥	西班牙	1996.8.11	门将		Fernando Pacheco
13	14	马努·加	西班牙	1998.1.2	中场		Manu Garcia
14	15	托尼·莫	西班牙	1998.3.20	中场		Mamadou Sylla
15	17	埃德加·	西班牙	1990.1.2	中场		Gonzalo Escalante
16	18	法昆多·	乌拉圭	2001.12.2	中场		Fakundo peristri
17	19	伊万·马	西班牙	1999.2.14	中场		John Guidetti
18	20	佩雷·庞	西班牙	1993.2.20	中场		Perry Ponce
19	21	马丁·阿	西班牙	1996.5.10	后卫		Martin aguirregaviria
20	22	弗洛里安	法国	1991.5.20	后卫		Antonio Sivera
21	23	西莫·纳	西班牙	1990.1.23	后卫		Loum Ndiaye
22	24	米格尔·	西班牙	1999.9.3	前锋		Tomás Pina Isla
23	26	阿卜杜勒	摩洛哥	1999.3.10	后卫		Abdul abukar
24	27	哈维·洛	西班牙	2002.3.25	后卫		Matt Miazga
25	31	赫苏斯·	赤道几内	2001.3.1	门将		Miguel
26	35	阿兰·戈	西班牙	2003.5.4	前锋		Alan Godoy

在得到了球员数据后，我们将 FBef 领域库中常用的属性和关系进行分类，筛选出球衣号码，国籍，生日，职位，队长，主教练，得分情况等各种词条作为事实信息。

## 四、实验结果和分析

最终，基于前一部分爬虫所获取的文本与编写的正则表达式配合 re.search 函数进行模板匹配，抽取得到的结果如图：

成立时间：  
体育马德里,成立时间,1903年  
加的斯,成立时间,1910年  
missing at 埃尔切.txt  
塞维利亚,成立时间,1905年  
奥萨苏纳,成立时间,1920年  
巴列卡诺,成立时间,1924年  
巴塞罗那,成立时间,1899年  
格拉纳达,成立时间,1931年  
比利亚雷亚尔,成立时间,1923年  
毕尔巴鄂,成立时间,1898年  
瓦伦西亚,成立时间,1919年  
皇家社会,成立时间,1909年  
皇家马德里,成立时间,1902年  
莱万特,成立时间,1909年  
西班牙人,成立时间,1900年  
贝蒂斯,成立时间,1907年  
赫塔菲,成立时间,1983年  
阿拉维斯,成立时间,1921年  
马略卡,成立时间,1916年

位置：  
体育马德里,位置,西班牙马德里  
加的斯,位置,西班牙安达鲁西亚自治区加的斯  
missing at 埃尔切.txt  
塞维利亚,位置,西班牙安达鲁西亚首府塞维利亚  
奥萨苏纳,位置,西班牙纳瓦拉自治区首府潘普洛纳

图一：对应关系的抽取结果（节选）

通过对领域知识库的球员的数据抽取，我们得到了西甲联赛所有俱乐部球员的信息，例如皇家贝蒂斯俱乐部：

	球衣号码	姓名	国籍	生日	职位	所属球队	英文名
0	1	霍埃尔·罗夫莱斯	西班牙	1990.6.17	门将	皇家贝蒂斯足球俱乐部球员	Rodrigo
1	2	马丁·蒙托亚	西班牙	1991.4.14	后卫	球员 Martin Montoya	
2	3	埃德加·冈萨雷斯	西班牙	1997.4.1	后卫	皇家贝蒂斯足球俱乐部 Edgar Gonzalez	
3	4	保罗·阿库奥库	科特迪瓦	1997.12.20	中场	Nan Nabil Fekir	
4	5	马克·巴特拉	西班牙	1991.1.15	后卫	Nan Mark batra	
5	6	比克托·鲁伊斯	西班牙	1989.1.25	后卫	Nan Bictor Ruiz	
6	7	胡安米	西班牙	1993.5.20	前锋	Nan Willian José	
7	8	纳比勒·费基尔	法国	1993.7.18	中场	Nan Aitor Ruibal	
8	9	博尔哈·伊格莱西亚斯	西班牙	1993.1.17	前锋	Nan Andrés Guardado	
9	10	塞尔希奥·卡纳莱斯	西班牙	1991.2.16	中场	Nan Youssouf Sabaly	
10	11	克里斯蒂安·特略	西班牙	1991.8.11	前锋	Nan Kike Hermoso	
11	12	威廉·若泽	巴西	1991.11.23	前锋	Nan Juanmi	
12	13	鲁伊·席尔瓦	葡萄牙	1994.2.7	门将	Nan Marc Bartra	
13	14	威廉·卡瓦略	葡萄牙	1992.4.7	中场	Nan William Carvalho	
14	15	亚历克斯·莫雷诺	西班牙	1993.6.8	后卫	Nan Alex Moreno	
15	16	赫尔曼·佩泽拉	阿根廷	1991.6.27	后卫	Nan Herman pezela	
16	17	华金	西班牙	1981.7.21	中场	Nan Hua Jin	
17	18	安德烈斯·瓜尔达多	墨西哥	1986.9.28	中场	Nan Cristian Tello	
18	19	埃克托·贝列林	西班牙	1995.3.19	后卫	Nan Ektor berelin	
19	20	迭戈·莱内斯	墨西哥	2000.6.9	前锋	Nan Diego Raines	
20	21	吉多·罗德里格斯	阿根廷	1994.4.12	中场	Nan Guido Rodriguez	
21	22	比克托·卡马拉萨	西班牙	1994.5.28	中场	Nan Víctor Camarasa	
22	23	优素福·萨巴利	塞内加尔	1993.3.5	后卫	Nan Yusuf sabali	
23	24	艾托尔·鲁伊瓦尔	西班牙	1996.3.22	前锋	Nan Borja Iglesias	
24	25	克劳迪奥·布拉沃	智利	1983.4.13	门将	Nan Claudio Bravo	
25	27	罗韦尔·冈萨雷斯	西班牙	2001.1.6	前锋	Nan Lowell Gonzalez	
26	28	罗德里·桑切斯	西班牙	2000.2.16	中场	Nan Joaquín	
27	33	胡安·米兰达	西班牙	2000.1.19	后卫	Nan Juan Miranda	

其中各项关系信息为第一行，以及之后再本体构建部分添加的其他关系。

同时，从百度百科和维基百科抽取出的球员信息如图所示：

number		name	country	birthday	location
0	1	马克-安德烈·特尔施特根	德国	1992-04-30	门将
1	2	塞尔吉尼奥·德斯特	美国	2000-11-03	后卫
2	3	杰拉德·皮克	西班牙	1987-02-02	后卫
3	4	罗纳德·阿劳霍	乌拉圭	1999-03-07	后卫
4	5	塞尔吉奥·布斯克茨	西班牙	1988-07-16	中场
5	6	里基·普奇	西班牙	1999-08-13	中场
6	7	奥斯曼·登贝莱	法国	1997-05-15	前锋
7	8	达尼·阿尔维斯	巴西	1983-05-06	后卫
8	9	孟菲斯·德佩	荷兰	1994-02-13	前锋
9	10	安苏·法蒂	西班牙	2002-10-31	前锋
10	11	阿达马·特拉奥雷	西班牙	1996-01-25	前锋
11	12	马丁·布雷思韦特	丹麦	1991-06-05	前锋
12	13	内托	巴西	1989-07-19	门将
13	14	尼科·冈萨雷斯	西班牙	2002-01-03	中场
14	15	克莱芒·朗格莱	法国	1995-06-17	后卫
15	16	佩德里	西班牙	2002-11-25	中场
16	17	卢克·德容	荷兰	1990-08-27	前锋
17	18	霍尔迪·阿尔瓦	西班牙	1989-03-21	后卫
18	19	费兰·托雷斯	西班牙	2000-02-29	前锋

在通过 3 种方式得到数据以后，我们在 Knowledge Fusion 部分对其进行知识对齐。

## III Type Inference

### 一、实验目的

具有相同特点或属性的实体集合的抽象，如足球球员、足球联赛、足球教练。领域知识图谱多采用自顶向下的方法来构建本体。一方面，相对于开放域知识图谱，领域知识图谱涉及的概念和范围都是固定或者可控的；另一方面，对于领域知识图谱，要求其满足较高的精度。自顶向下是先为知识图谱定义好本体与数据模式，再将实体加入到知识库。该构建方式需要利用一些现有的结构化知识库作为其基础知识库。因此，我们在本体构建之前，首先要对其类型作定义和推断

### 二、实验环境

Protégé 软件 5.5.0

### 三、实验步骤

在类型推断的部分中，在得到了 infobox 和 category 以后，我们对其进行提取类别，结合之后的本体构建工作，我们对其进行分类，提取出了足球俱乐部，足球联赛，球员，主教练，赞助商，球场等作为其类别，进行之后的知识图谱本体构建。

Atletico de Madrid			
			全名 Club Atlético de Madrid
绰号 Los Colchoneros (红黑军团) Los Rojiblancos (红白军团)			成立 1903年4月26日, 119年前[1]
城市 西班牙马德里			主场 万达大都会球场
容纳人数 68,456[2]			赞助商 米盖尔·安赫尔·希门内斯(51%) 伊丹·莫南(30%) 恩里克·萨博尼(19%)[3][4][5]
主席 埃米利奥·塞雷佐			主教练 迭戈·西蒙尼
主客场 西班牙足球甲级联赛 西甲, 第 1 位			网站 官方网站
 主场球衣 客场球衣 第三球衣			
中文名 巴塞罗那足球俱乐部	主要荣誉	欧冠冠军 (5次) 西甲冠军 (26次) [4] 国王杯冠军 (31次) [5]	
外文名 FC Barcelona	容纳人数	98260人	
成立时间 1898年11月29日	主席	豪安·拉波尔塔	
所属地区 西班牙巴塞罗那	现任队长	塞尔希奥·布斯克茨 [6]	
运动项目 足球	队歌	Cant del Barça	
角逐赛事 西班牙足球甲级联赛	昵称	巴萨 (Barça) [1][6]	
主场馆 诺坎普球场			
现任主教练 哈维·埃尔南德斯			
知名人物 莱奥·梅西、哈维·埃尔南德斯、罗纳尔迪尼奥、约翰·克鲁伊夫、里瓦尔多、兹普·瓜迪奥拉			

# IV Knowledge Fusion

## 一、 实验目的

为了获取更为全面的俱乐部球员信息，考虑融合百度百科、维基百科与领域知识库 FBref 的球员信息，其中考虑三者关键属性名称语种差异，为了保证融合准确性首先需对球员名称进行对其融合；俱乐部的关键属性俱乐部名称同样存在语种差异，我们一并进行了融合处理。

## 二、 实验思路

- 1) 将百度百科、维基百科和领域知识库 FBref 的球员信息爬取下来，保存为 dataframe 数据结构形式
- 2) 通过球员球衣号码关键词为匹配键进行不同知识源球员信息对照，考虑到球员信息库差异，并结合百度翻译 api 进行语种自动识别翻译对照，划定相似度衡量阈值进行相似度匹配，进行补充印证
- 3) 将对照后的球员信息合并为一张总表，保存为 csv 文件
- 4) 俱乐部名称同样借助百度翻译 api 处理
- 5) 将融合后信息加入到知识图谱构建中

## 三、 代码实现

百度翻译 api 语种转换：通过百度翻译 api，将所需信息进行分割处理，以指定格式加密发送处理，再将处理后信息捕获解析

```
! # coding=utf-8
def translate(q):
    appid = '20220406001160200' # 填写你的appid
    secretKey = 'hYUtVR6JR9kulHwTH3ap' # 填写你的密钥

    httpClient = None
    myurl = '/api/trans/vip/translate'

    fromLang = 'auto' #原文语种
    toLang = 'zh' #译文语种
    salt = random.randint(32768, 65536)
    # q= 'apple'
    sign = appid + q + str(salt) + secretKey
    sign = hashlib.md5(sign.encode()).hexdigest()
    myurl = myurl + '?appid=' + appid + '&q=' + urllib.parse.quote(q) + '&from=' + fromLang + '&to=' + toLang + '&salt=' + str(salt) + '&sign=' + sign

    try:
        httpClient = http.client.HTTPConnection('api.fanyi.baidu.com')
        httpClient.request('GET', myurl)

        # response是HTTPResponse对象
        response = httpClient.getresponse()
        result_all = response.read().decode("utf-8")
        result = json.loads(result_all)
```

图 1：百度翻译 api（节选）

相似度划分：借助 difflib 库进行字符串信息相似度匹配

```
#相似度度量对比, 交换
def similar_diff_ratio(str1, str2):
    return difflib.SequenceMatcher(None, str1, str2).ratio()
def max_index(list):
    temp = list[0]
    index = 0
    for i in range(len(list)):
        if(list[i]>temp):
            temp = list[i]
            index = i
    return index
#list1是百度中文名, list2是领域知识库翻译后中文名
def simi_name(list1,list2):
    simi = []
    temp = []
    maxindex = [] #正确的下标
    # final_name = [] #更新后对应顺序名
    for i in range(len(list1)):
        for j in range(len(list2)):
            temp.append(similar_diff_ratio(list1[i],list2[j]))
        simi.append(temp)
        maxindex.append(max_index(temp))
        temp = []
    
```

图 2：相似度匹配（节选）

名称信息融合：将相似度匹配后结果作为参考，将名称信息进行融合

```

def FBref_name_append(file1,zh_FBREFname,file2,index):
    df1 = pd.read_csv(file1)
    en_df = np.array(df1['Unnamed: 0'][1:])
    for i in range(len(en_df)):
        en_df[i] = en_df[i].split("\\\\")[0]
    en_FBREFname = list(en_df)
    df2 = pd.read_csv(file2,encoding='utf-8')
    df3 = df2[['球衣号码','姓名','国籍','生日','职位','所属球队']]
    df3['英文名'] = [' '] * df3.shape[0]
        # df3.reset_index(drop=True, inplace=True)
    zh_name = list(np.array(df2['姓名']))
    # print(zh_name)
    # print(zh_FBREFname[i])
    maxindex = simi_name(zh_name,zh_FBREFname[index])
    for i in range(len(maxindex)):
        for j in range(len(maxindex)):
            if(maxindex[j]==i):
                df3['英文名'][i] = en_FBREFname[j]
    for i in range(len(df3)):
        if(df3['英文名'][i]==' '):
            time.sleep(1)
            df3['英文名'][i] = translate_en(df3['姓名'][i])
    return df3

```

图 3: 名称信息融合（节选）

俱乐部名称融合：借助百度翻译 api，以及 Levenshtein 距离、jaccard 相似度叠加作为度量标准进行字符串相似度衡量，进行俱乐部名称信息融合

```

# calculate Levenshtein similarity
leven_mole1 = Levenshtein.distance(w1,b1)
leven_demo1 = max(len(w1),len(b1))
leven_mole2 = Levenshtein.distance(w2,b2)
leven_demo2 = max(len(w2),len(b2))
similarity1 = 1-leven_mole1/leven_demo1
similarity2 = 1-leven_mole2/leven_demo2

# calculate jaccard similarity
w1a = list(w1)
b1a = list(b1)
w1b = set(w1a)
b1b = set(b1a)
d1 = float(len(w1b.intersection(b1b))) / len(w1b.union(b1b))

w1a = list(w2)
b1a = list(b2)
w1b = set(w1a)
b1b = set(b1a)
d2 = float(len(w1b.intersection(b1b))) / len(w1b.union(b1b))

# 判断相似度是否大于阈值
if(similarity1*0.3+similarity2*0.2+d1*0.3+d2*0.2 > 0.55):
    temp.append(wiki_final[i][0])
    temp.append(baidu_final[j][0])

```

图 4: 俱乐部名称融合 (节选)

## V Ontology Building

### 一、实验目的

1980 年, 本体论 (Ontology) 哲学概念 “本体” 被引入到人工智能领域用来刻画知识。本体是共享概念模型的明确的形式化规范说明, 该定义体现了本体的四层含义: 概念模型、明确、形式化、共享。本体是实体存在形式的描述, 往往表示为一组概念定义和概念之间的层级关系, 本体框架形式树状结构, 通常被用来为知识图谱定义 schema。因此, 我们希望构建一个全面的, 具有多层次复杂关系的本体关系。

## 二、实验步骤

### 1. 运行环境

在本体构建的过程中，我们使用了 Protégé 软件 5.5.0 版本。Protégé 软件是斯坦福大学医学院生物信息研究中心基于 Java 语言开发的本体编辑和知识获取软件。它是一种本体开发工具，是基于知识的编辑器，属于开放源代码软件。Protégé 软件主要用于语义网中本体的构建，是语义网中本体构建的核心开发工具，同时提供了本体概念类，关系，属性和实例的构建。

### 2. 导入数据

由于在数据提取阶段，我们已经获取了西甲所有球队及球员的三元组信息，并保存为 csv 表格形式，如图所示：

奥萨苏纳足球俱乐部							
	A	B	C	D	E	F	G
1	球衣号码	name	国籍	生日	职位	所属球队	英文名
2	1	塞尔希奥·埃雷拉	西班牙	1993.6.5	门将	奥萨苏纳足球俱乐部球员	Sergio Herrera
3	2	纳乔·比达尔	西班牙	1995.1.24	后卫	球员	Roberto Torres
4	3	胡安·克鲁斯	西班牙	1992.7.28	后卫	奥萨苏纳足球俱乐部	Darko Brašanac
5	4	乌奈·加西亚	西班牙	1992.9.3	后卫		Kike Barja
6	5	戴维·加西亚	西班牙	1994.2.14	后卫		Unai Dufur
7	6	奥耶尔·圣胡尔霍	西班牙	1986.5.25	中场		José Ángel
8	7	霍恩·蒙卡约拉	西班牙	1998.5.13	中场		Horn moncajola
9	8	达尔科·布拉沙纳茨	塞尔维亚	1992.2.12	中场		Javier Ontiveros
10	9	奇米·阿维拉	阿根廷	1994.2.6	前锋		Chimi Avila
11	10	罗伯托·托雷斯	西班牙	1989.3.7	中场		Roberto Torres
12	11	基克·巴尔哈	西班牙	1997.4.4	前锋		Juan Cruz Armada
13	12	豪梅·格劳	西班牙	1997.5.5	中场		Nacho Vidal
14	13	胡安·佩雷斯	西班牙	1996.7.15	门将		Aridane Hernández
15	14	鲁文·加西亚	西班牙	1993.7.14	前锋		Reuven Garcia
16	15	若纳斯·拉马略	安哥拉	1993.6.10	后卫		Unai Garcia
17	16	科特	西班牙	1989.9.5	后卫		Juan Pérez
...	...	...	...	...	...	...	...

因此我们使用 Protégé 软件对本体先创建规则。以奥萨苏纳俱乐部为例，我们用规则：

Class: @F2

SubclassOf:@F3

定义“奥萨苏纳俱乐部的球员”属于“球员”这一类；

用规则：

Individual:@B\* Types:@F2

Facts:@A1 @A\* (xsd:integer)

Facts:@C1 @C\*

Facts:@D1 @D\*

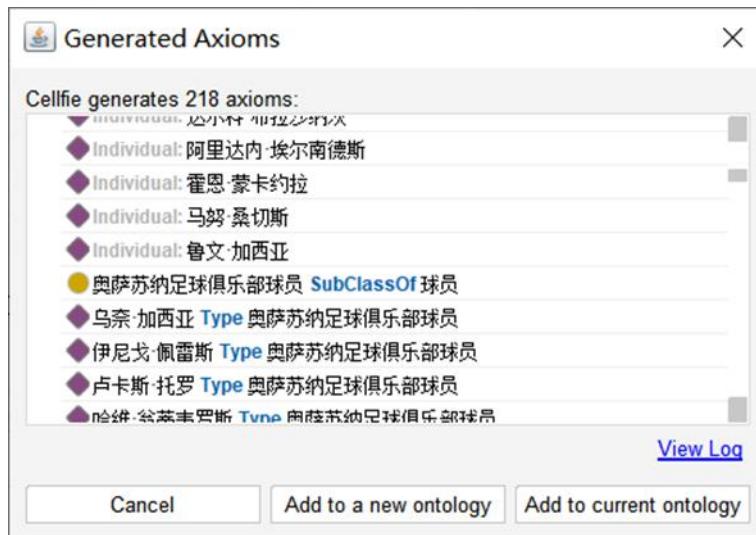
Facts:@E1 @E\*

Facts:@F1 (ObjectProperty) @F4

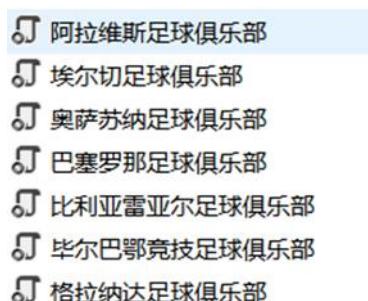
定义 B 这一列的所有数据为“奥萨苏纳俱乐部的球员”类型，并且他们的具体属性数据包括：A 列为球衣号码，类型为整型；CDE 列分别为国籍，生日，以及在球队中的职位。同时，这些球员都与“奥萨苏纳足球俱乐部”这一节点建立关系，关系名称为“所属球队”。具体实现如图所示：

	Sheet Name	Start Column	End Column	Start Row	End Row	Rule
✓	奥萨苏纳足球俱乐部	A	G	1	+	Class:@F2 SubClassOf:@F3
✓	奥萨苏纳足球俱乐部	A	G	2	+	Individual:@B* Types:@F2 Facts:@A1 @A* (xsd:integer) Facts:@C1 @C* Facts:@D1 @D* Facts:@E1 @E* Facts:@G1 @G* Facts:@F1(ObjectProperty) @F4
✓						

经过规则推理，我们得到了如图所示的本体与实例关系：



接下来，我们将西甲所有足球俱乐部的规则信息全部保存为 json 文件，例如：

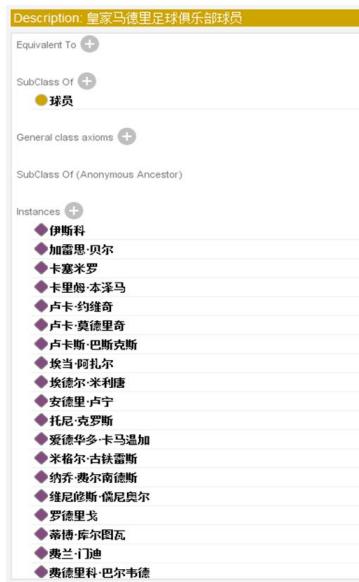


### 3. 本体分类

完成了规则和数据的导入后，我们在 Protégé 软件中对所有的本体进行更进一步的分类和创建，并且人工添加了例如主席，主教练，俱乐部拥有者（赞助商）等本体，与西甲俱乐部，球员，球场共同构成整个西甲联赛的本体信息。同时，我们还新建了球员的子类，分别是各个俱乐部的球员，例如“巴塞罗那足球俱乐部球员”为“球员”的子类。



在每个子类中，我们加入了该类别的所有实例，以“皇家马德里足球俱乐部球员”为例，我们将其现役的所有球员作为实例添加其中：

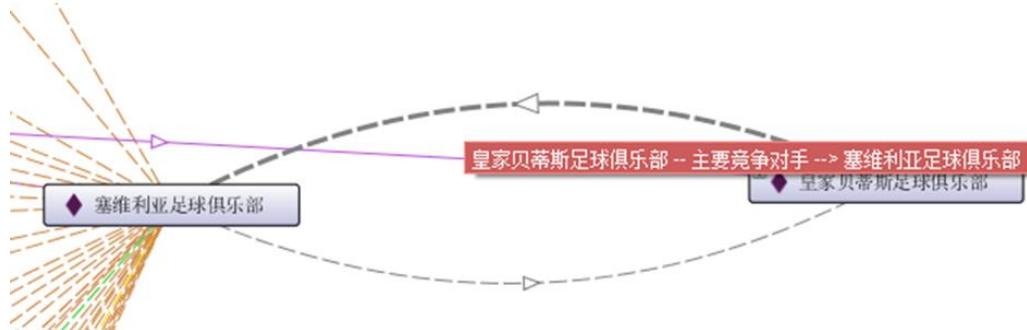


#### 4. 关系创建

我们又对不同实例之间的关系做出了扩充，例如主要竞争对手，就任主席，所属球队，执教，球队主场，赞助商是，队长是等关系：

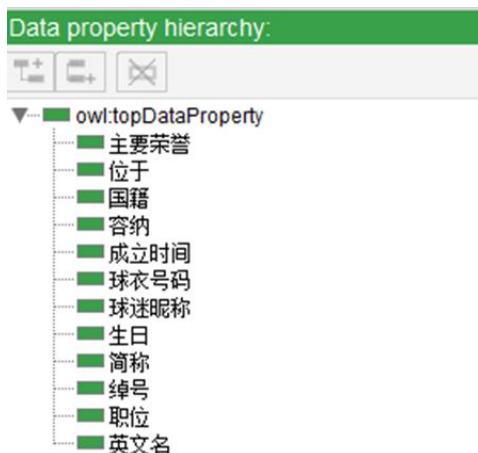


以下图为例：塞维利亚足球俱乐部与皇家贝蒂斯足球俱乐部互相之间是“主要竞争对手”的关系：



## 5. 数据属性

在完成了本体创建与分类，关系创建之后，我们又对部分实例的数据属性做出了扩充。例如，主要荣誉，位于（地点），国籍等具体信息：



以下图为例，巴塞罗那足球俱乐部的队长是塞尔吉奥，球队主场是诺坎普球场，主要荣誉有 5 次欧冠冠军，26 次西甲冠军，成立时间为 1899 年等等：

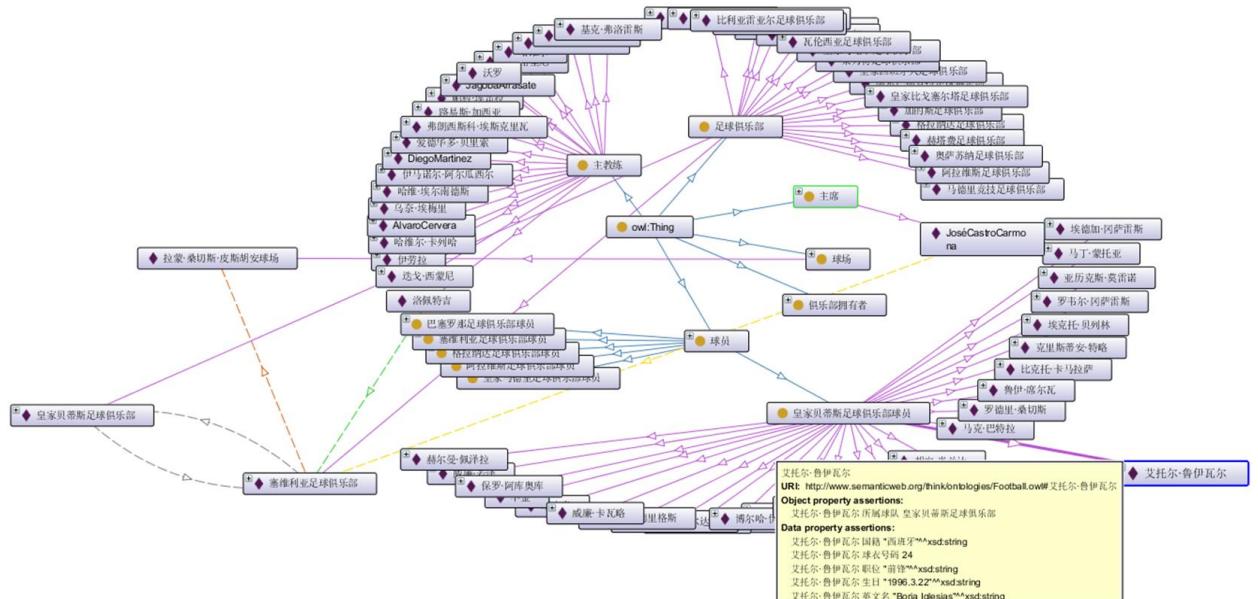


以它的队长塞尔吉奥为例，他的英文名和国籍，生日，职位，球衣号码等信息也都被加入：



### 三、实验结果和分析

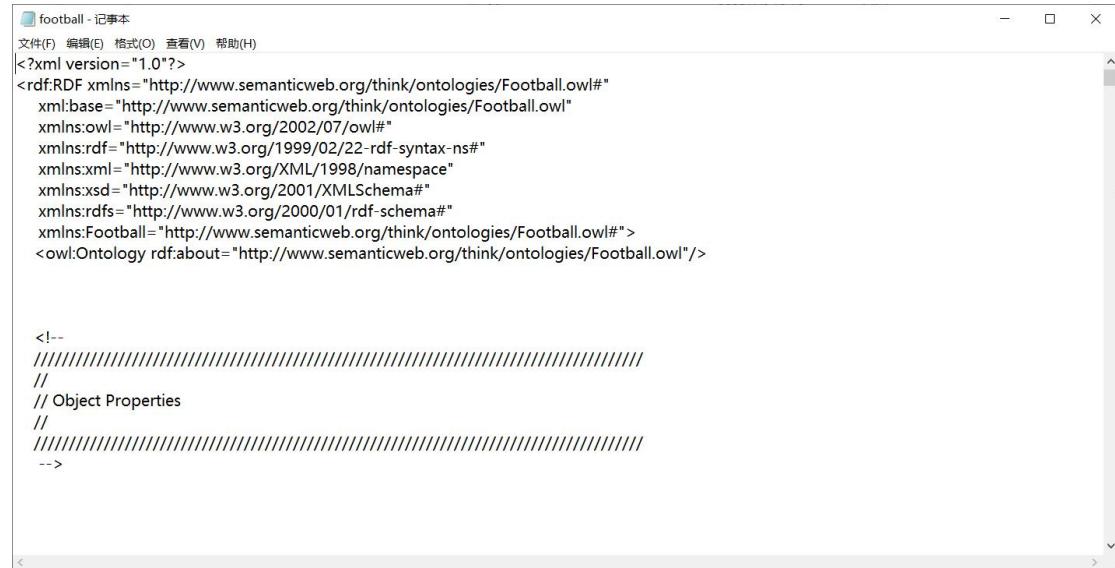
在完成了西甲联赛所有数据的知识图谱本体构建后，我们在 Protégé 软件中得到了如下的可视化图谱：



我们得到了西甲联赛的所有俱乐部，俱乐部球员，球场信息，主教练信息，以及各个实例之

间的关系。此外，当我们关注某一个特定实例时，还可以知道他的具体属性信息。

生成的 football.owl 文件如图所示：



```
football - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.semanticweb.org/think/ontologies/Football.owl#"
    xmlns:owl="http://www.semanticweb.org/think/ontologies/Football.owl#"
    xmlns:rdf="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xml="http://www.w3.org/XML/1998/namespace"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:Football="http://www.semanticweb.org/think/ontologies/Football.owl#">
<owl:Ontology rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#"/>

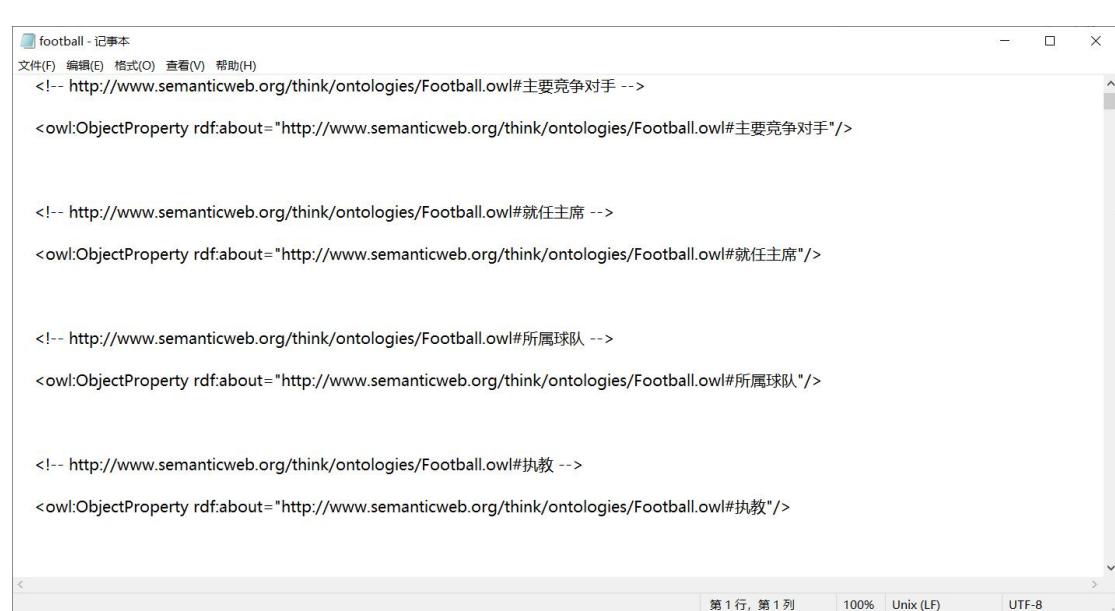
<!--

// Object Properties
//
--&gt;

&lt;!-- http://www.semanticweb.org/think/ontologies/Football.owl#主要竞争对手 --&gt;
&lt;owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#主要竞争对手"/&gt;

&lt;!-- http://www.semanticweb.org/think/ontologies/Football.owl#就任主席 --&gt;
&lt;owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#就任主席"/&gt;

&lt;!-- http://www.semanticweb.org/think/ontologies/Football.owl#所属球队 --&gt;
&lt;owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#所属球队"/&gt;

&lt;!-- http://www.semanticweb.org/think/ontologies/Football.owl#执教 --&gt;
&lt;owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#执教"/&gt;</pre>
```

```

football - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

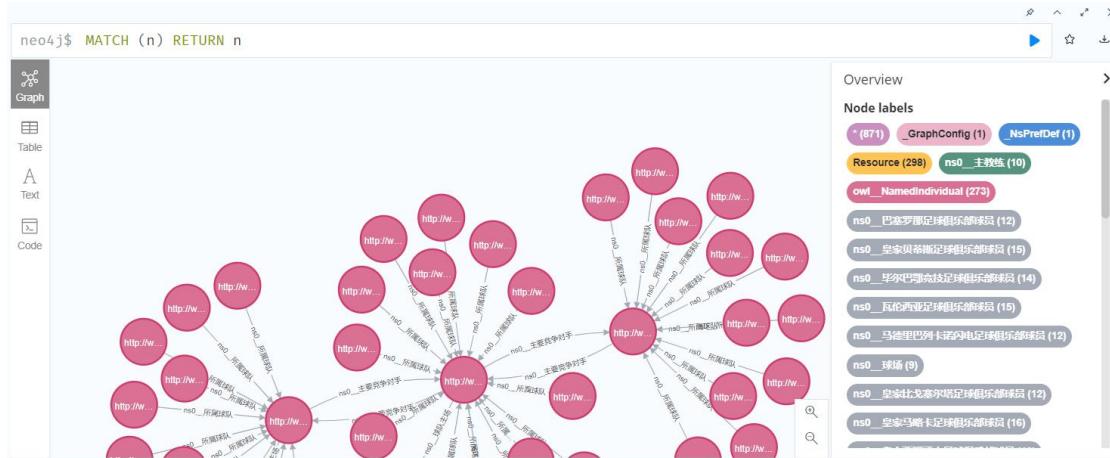
<!-- http://www.semanticweb.org/think/ontologies/Football.owl#DiegoMartinez -->
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#DiegoMartinez">
  <rdf:type rdf:resource="http://www.semanticweb.org/think/ontologies/Football.owl#主教练"/>
  <执教 rdf:resource="http://www.semanticweb.org/think/ontologies/Football.owl#格拉纳达足球俱乐部"/>
</owl:NamedIndividual>

<!-- http://www.semanticweb.org/think/ontologies/Football.owl#JagobaArrasate -->
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#JagobaArrasate">
  <rdf:type rdf:resource="http://www.semanticweb.org/think/ontologies/Football.owl#主教练"/>
  <执教 rdf:resource="http://www.semanticweb.org/think/ontologies/Football.owl#奥萨苏纳足球俱乐部"/>
</owl:NamedIndividual>

<!-- http://www.semanticweb.org/think/ontologies/Football.owl#JosuUrrutia -->
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#JosuUrrutia">
  <rdf:type rdf:resource="http://www.semanticweb.org/think/ontologies/Football.owl#主席"/>

```

最终，在后续 neo4j 图数据库的验证中，我们得知，这张知识图谱一共包含了 871 个节点以及 169 个关系。





根据抽样测试，我们随机选取了 50 个节点，人工校验其节点信息是否与真实信息匹配，最后得出结论，47 个节点的所有信息都正确，有 3 个节点的属性数据，“英文名”这一项存在不匹配，可能是因为匹配时的翻译错误。总体来说，本次知识图谱搭建的准确率很高，三元组数量有一定工作量。

## VI Applications-Knowledge Querying and Visualization

## 一、实验目的

在知识库与人类的交互中，我们会自然地想到问答系统。知识库问答是目前的研究热点。知识库问答（knowledge base question answering, KB-QA）即给定自然语言问题，通过对问题进行语义理解和解析，进而利用知识库进行查询、推理得出答案。我们希望将用户输入问答系统的自然语言转化成知识库的查询语句。为了使操作简化，我们希望在 python 终端直接输入自然语句，然后得到的输出是返回的知识库节点信息和数据。此外，我们还希望对我们的知识图谱数据进行可视化。

## 二、实验步骤

### 1. 运行环境

Python3.8, neo4j-community-4.4.1。

Neo4j 是一个高性能的图形数据库，它可以将结构化数据存储在网络上而不是表中。同时，它还是一个嵌入式的、基于磁盘的、具备完全的事务特性的 Java 持久化引擎，具有成熟数据库的所有特性。

### 2. 图谱数据文件转换

由于在 neo4j 图数据库中，支持的导入文件格式为 rdf 格式，而在本体构建过程中，我们利用 Protégé 软件生成的图谱文件为 owl 文件，经过实验测试，我们发现两种文件格式并不兼容，因此我们调用了 java 包 rdf2rdf-1.0.1-2.3.1，作为第三方工具将 owl 文件转换为 rdf 格式。rdf2rdf-1.0.1-2.3.1 支持的转换类型如下图所示：

Supported extensions	
rdf, rdfs, owl, xml	RDF/XML
nt	N-Triples
ttl	Turtle
n3	N3
trig, xml	TriX
trig	TriG

在完成了 rdf2rdf-1.0.1-2.3.1 的安装后，我们将其移动到 neo4j/plugins 目录下，并且修改其配置文件，在 neo4j/neo4j.conf 文件中添加以下内容：

```
dbms.unmanaged_extension_classes=semantics.extension=/rdf
```

然后，我们在命令行终端输入：

```
java -jar rdf2rdf-1.0.1-2.3.1.jar football.owl football.turtle
```

将我们得到的 owl 图谱文件转换为 rdf 格式。转换结果见实验结果部分。

### 3. 将知识图谱数据导入 neo4j 图数据库

接下来，我们重新启动 neo4j 客户端，并在其中输入语句：

```
call n10s.rdf.import.fetch("file:///C:/Users/think/Desktop/1/football.turtle",
"RDF/XML")
call n10s.rdf.import.fetch("file:///D:/football.turtle", "RDF/XML", {shortenUrls:
true})
```

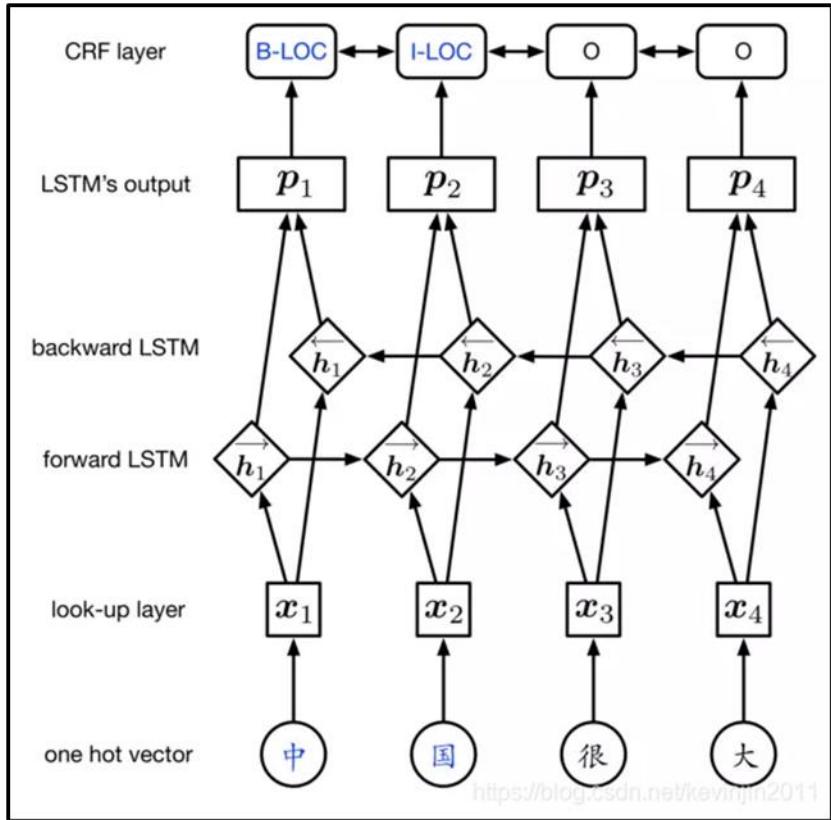
至此，我们完成了 owl 文件与 rdf 文件的转换，并且支持其在 neo4j 终端进行导入、查询等操作。Neo4j 中的图谱数据详见实验结果部分。

### 3. 自然语言查询

我们希望将命名实体识别（NER）和 Neo4j 查询语句（Cypher）结合，从而实现自然语言转换到知识图谱的查询功能。

#### 1) 命名实体识别（NER）

在命名实体识别（NER）中，我们使用 BiLSTM+CRF 模型。BiLSTM-CRF 输入是词向量，输出每个单词预测的序列标注，整个模型分为三步骤，第一步：单词输入，单词进入 look-up layer 层，使用 CBOW、Skip-gram 或者 glove 模型映射为词向量；第二步：词向量进入 BiLSTM 层，通过学习上下文的信息，输出每个单词对应于每个标签的得分概率；第三步：所有的 BiLSTM 的输出将作为 CRF 层的输入，通过学习标签之间的顺序依赖信息，得到最终的预测结果，如下图所示：



## 2) Synonyms 同义词对齐

Synonyms 用于自然语言理解的很多任务：文本对齐，推荐算法，相似度计算，语义偏移，关键字提取，概念提取，自动摘要，搜索引擎等。同时，它还对进行了命名实体识别后的结果与问题模板的关键词进行匹配，如图所示：

词语	2016词林改进版	知网	Synonyms	人工标准
"轿车", "汽车"	0.82	1.0	0.73	0.98
"宝石", "宝物"	0.83	0.17	0.71	0.96
"旅游", "游历"	1.0	1.0	0.59	0.96
"男孩子", "小伙子"	0.81	1.0	0.88	0.94
"海岸", "海滨"	0.94	1.0	0.68	0.93
"庇护所", "精神病院"	0.96	0.58	0.64	0.90
"魔术师", "巫师"	0.85	0.58	0.66	0.88
"中午", "正午"	1.0	1.0	0.81	0.86
"火炉", "炉灶"	0.98	0.58	0.85	0.78

在我们西甲联赛的知识图谱中，我们也将同义词近义词作了对齐，方便自然语言查询。例如“简称”和“绰号”是一组同义词。

## 3) 自然语言分词

为了符合 cypher 语句的查询方式，我们将输入的自然语言进行分词处理，提取出我们希望查询的关键词，具体代码见实验代码部分。

#### 4. 用 cypher 语句查询模板转换自然语言

我们使用查询模板来具象化 4 种特定类型的问题。

1) 查询一个 entity 的所有信息：（例：甲的出生年月？）

```
MATCH (a:entity {name: '甲'})
```

```
RETURN a.property
```

2) 与 entity 有关的某一类关系值：（甲球队的所有球员？）

```
MATCH (a:entity {name: '甲'}) - [r:relation]-(b)
```

```
RETURN r
```

3) 查询与 a 有对应关系的 b：（例：甲隶属于哪个球队？）

```
MATCH (a:entity {name: '甲'}) - [:relation]-(b)
```

```
RETURN b
```

4) 查询 a 与 b 的关系：（甲和乙是什么关系？）

```
MATCH (a:entity1 {name: '甲'}) - [r:relation]-(b:entity2 {name: '乙'})
```

```
RETURN r
```

#### 5. 连接 python 终端和 neo4j

neo4j 目前是图数据库的主流，neo4j 的 Cypher 语法简单直观，但是不便于流程化。如果习惯在 python 环境下处理数据，那么还是要用到 python 的 neo4j 库，即 py2neo。因此，考虑到我们知识库问答系统的输入和输出，我们使用 py2neo 包直接输出在 neo4j 中查询的结果。具体代码和测试结果在后文中详细阐述。

#### 6. 图谱数据可视化

除去 neo4j 图数据库中本身的可视化效果以外，我们还针对节点的具体信息生成动态网页。

例如，以西甲联赛所有球队的所在位置为例，我们生成了

<http://localhost:8888/view/render.html> 这样一个动态网页来可视化每个球队的所在位置。

### 三、代码实现

其他词探索.ipynb：实现了知识库问答查询的所有功能。

```
Type Markdown and LaTeX: α²

In [4]: 1 from LAC import LAC
          C:\Anaconda3\lib\site-packages\socks.py:58: DeprecationWarning: Using or importing the ABCs from 'collections' instead of from 'collections.
          abc' is deprecated, and in 3.8 it will stop working
          from collections import Callable

In [5]: 1 lac = LAC(mode = 'rank')

In [5]: 1 text = u'西甲的第一名是谁'
          2 lac_result = lac.run(text)
          3 lac_result

Out[5]: [[u'西甲', '的', '第一名', '是', '谁'], ['nz', 'u', 'n', 'v', 'r'], [3, 0, 2, 0, 1]]

In [6]: 1 texts = [u'马德里竞技俱乐部在西甲里面排名多少', '皇家贝蒂斯足球俱乐部的排名和皇家马德里俱乐部的排名那个比较好']
          2 lac_results = lac.run(texts)
          3 lac_results

Out[6]: [[[u'马德里竞技俱乐部', '在', '西甲', '里面', '排名', '多少'],
          ['ORG', 'p', 'nz', 'f', 'v', 'r'],
          [3, 0, 3, 1, 2, 1]],
[[u'皇家贝蒂斯足球俱乐部', '的', '排名', '和', '皇家马德里俱乐部', '的', '排名', '那个', '比较好'],
          ['ORG', 'u', 'vn', 'c', 'ORG', 'u', 'vn', 'r', 'a'],
          [2, 0, 2, 0, 3, 0, 2, 1, 2]]]

In [7]: 1 text = '塞维利亚俱乐部的出生年月是什么时候'
          2 lac_result = lac.run(text)
          3 lac_result

Out[7]: [[u'塞维利亚俱乐部', '的', '出生', '年月', '是', '什么时候', '?'],
          ['ORG', 'u', 'm', 'n', 'v', 'n', 'v'],
          [3, 0, 2, 2, 0, 2, 0]]
```

#### 分词（节选）

```
13     '职位'
14 ]
15 n_sim = []
16 for i in n_list:
17     n_sim.append(synonyms.compare(n, i))
18 # print(n_sim)
19 max_value = max(n_sim)
20 idx = n_sim.index(max_value)
21 return n_list[idx], max_value

In [8]: 1 get_closest_prop('名字')
Out[8]: ('绰号', 0.893)

In [45]: 1 def get_entity_info(sentence):
          2     lac_res = lac.run(sentence)
          3     # print(lac_res)
          4     max_sim = -1
          5     max_n = ''
          6     # print(list(enumerate(lac_res[2])))
          7     for index, i in enumerate(lac_res[2]):
          8         if i > 0:
          9             # print(index)
          10            tmp_n, tmp_sim = get_closest_prop(lac_res[0][index])
          11            if tmp_sim > max_sim:
          12                max_sim = tmp_sim
          13                max_n = tmp_n
          14        entity = ''
          15        for index, i in enumerate(lac_res[1]):
          16            if i == 'PER' or i == 'LOC' or i == 'ORG':
          17                entity = lac_res[0][index]
          18        # print(entity, max_n)
          19        SPARQL_SEN = 'MATCH (a:ns0_足球俱乐部) WHERE ?uri = "http://www.semanticweb.org/think/ontologies/Football.owl#' + entity + '"\n        RETURN a.ns0_'
          20        print(SPARQL_SEN)
          21    return SPARQL_SEN

In [46]: 1 q1 = get_entity_info('皇家马略卡足球俱乐部的昵称是什么')
          2
          3 print(q1)
          MATCH (a:ns0_足球俱乐部) WHERE ?uri = "http://www.semanticweb.org/think/ontologies/Football.owl#皇家马略卡足球俱乐部"
          RETURN a.ns0_绰号
```

#### 问题转换（节选）

```

MATCH (a:entity{name:"皇家马德里俱乐部"}) - [r:主要竞争对手]->(b)
RETURN r

Out[39]: 'MATCH (a:entity{name:"皇家马德里俱乐部"}) - [r:主要竞争对手]->(b) \nRETURN r'

In [28]: 1 def get_entitys_better(sentence):
2     lac_res = lac.run(sentence)
3     # print(lac_res)
4     max_sim = -1
5     max_r = ''
6     # print(list(enumerate(lac_res[2])))
7     for index, i in enumerate(lac_res[2]):
8         if i > 0:
9             # print(index)
10            tmp_n, tmp_sim = get_closest_relation(lac_res[0][index])
11            if tmp_sim > max_sim:
12                max_sim = tmp_sim
13                max_r = tmp_n
14    entity_list = []
15    for index, i in enumerate(lac_res[1]):
16        if i == 'PER' or i == 'LOC' or i == 'ORG':
17            entity_list.append(lac_res[0][index])
18    # print(entity_list, max_r)
19    SPARQL_SEN = 'MATCH (a:entity {name:' + entity_list[0] + '}), (b:entity2 {name:' + entity_list[1] + '}) \nWHERE a.property>b.property'
20    # print(SPARQL_SEN)
21    return SPARQL_SEN

```

```

In [27]: 1 get_entitys_better('皇家马德里俱乐部赛季排名是否比塞维亚俱乐部好?')
Out[27]: 'MATCH (a:entity1 {name:"皇家马德里俱乐部"}), (b:entity2 {name:"塞维亚俱乐部"}) \nWHERE a.property>b.property\nRETURN a,b'

```

```

In [40]: 1 def get_relation_between(sentence):
2     lac_res = lac.run(sentence)
3     # print(lac_res)
4     max_sim = -1
5     max_r = ''
6     # print(list(enumerate(lac_res[2])))
7     for index, i in enumerate(lac_res[2]):
8         if i > 0:
9             # print(index)
10            tmp_n, tmp_sim = get_closest_relation(lac_res[0][index])

```

## 问题转换（节选）

```

29     return aim_list[1][max_index]
30
31 def which_template(q_part, sentence):
32     temp = 0
33     lac_res = lac.run(sentence)
34     # print(lac_res)
35     entity_list = []
36     for index, i in enumerate(lac_res[1]):
37         if i == 'PER' or i == 'LOC' or i == 'ORG':
38             entity_list.append(lac_res[0][index])
39     if(len(entity_list)>1):
40         get_relation_between(sentence) #q3
41     else:
42         for i in range(len(l1_list)):
43             if(q_part == l1_list[i]):
44                 get_entity_info(sentence) #q1
45                 temp = 1
46             if(temp == 1):
47                 get_entity_relation(sentence) #q2

```

## 测试

```

In [47]: 1 #aim_list已固定，只需修改输入的sentence
2 sentence1 = '皇家马略卡足球俱乐部的昵称是什么'
3 sentence2 = '卡尔洛·安切洛蒂在哪个球队执教'
4 sentence3 = '皇家贝蒂斯足球俱乐部和塞维利亚足球俱乐部是什么关系？'
5 q_part = question_part(sentence1, aim_list)
6 which_template(q_part, sentence1)
7 q_part = question_part(sentence2, aim_list)
8 which_template(q_part, sentence2)
9 q_part = question_part(sentence3, aim_list)
10 which_template(q_part, sentence3)

MATCH (a: ns0_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家马略卡足球俱乐部"}) 
RETURN a.ns0_绰号
MATCH (a: ns0_主教练`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#卡尔洛·安切洛蒂"}) - [:`ns0_执教`]->(b)
RETURN b
MATCH (a: ns0_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家贝蒂斯足球俱乐部"})-[r] ->(b: ns0_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#塞维利亚足球俱乐部"})
RETURN r

```

## 测试结果（节选）

```

In [48]: 1 import json
2 from py2neo import Graph, Node, Relationship, NodeMatcher, Subgraph
3
4 graph = Graph('http://localhost:7474', auth=("neo4j", "Lyc111lyc"))
5

In [ ]: 1 matcher = NodeMatcher(graph)

In [54]: 1 graph.run(q1)
        "LosBermellones"

In [57]: 1 graph.run(q2)
        [{"identity": 1083, "labels": ["Resource", "ns0_足球俱乐部", "owl_NamedIndividual"], "properties": {"ns0_简称": "皇马", "ns0_主要荣誉": "西班牙足球甲级联赛冠军(34次)", "ns0_位于": "西班牙首都马德里", "ns0_成立时间": "1902年", "ns0_球迷昵称": "美凌格", "uri": "http://www.semanticweb.org/think/ontologies/Football.owl#皇家马德里足球俱乐部"}}

In [62]: 1 graph.run(q3)
        [{"identity": 1039, "start": 1278, "end": 1164, "type": "ns0_主要竞争对手", "properties": {}}
        {"identity": 912, "start": 1164, "end": 1278, "type": "ns0_主要竞争对手", "properties": {}}

In [3]: 1 #matcher = NodeMatcher(graph)
2 #matcher.match().first()

In [2]: 1 #result=graph.run("MATCH (n) RETURN n").data() #读取neo4j数据库中的所有数据
2 #record=result[0] #读取表中的第一个数据
3 #record.get('n').get('uri') #假设数据库中有一个属性(或者说是列名字)是name
4

```

Py2neo 接口 (节选)

数据可视化.ipynb: 实现了西甲联赛足球俱乐部所在位置的动态网页

```

In [ ]: 1 import pandas as pd
2 from pyecharts import options as opts
3 from pyecharts.charts import Geo
4 from pyecharts.faker import Faker
5 from pyecharts.globals import ChartType
6 import os

In [67]: 1 df=pd.read_csv('百度球队三元组.csv')
2 df=df.iloc[20:40]
3
4 df['object'][38]='西班牙首都马德里1'
5 df['object'][39]='西班牙首都马德里2'
6 df

Out[67]:


|    | subject      | predicate | object                  |
|----|--------------|-----------|-------------------------|
| 20 | 加的斯足球俱乐部     | 位置        | 西班牙安达鲁西亚自治区加的斯          |
| 21 | 埃尔切足球俱乐部     | 位置        | 西班牙巴伦西亚自治区埃尔切           |
| 22 | 塞维利亚足球俱乐部    | 位置        | 西班牙安达卢西亚自治区首府塞维利亚       |
| 23 | 奥萨苏纳足球俱乐部    | 位置        | 西班牙纳瓦拉自治区首府潘普洛纳         |
| 24 | 巴塞罗那足球俱乐部    | 位置        | 西班牙巴塞罗那市                |
| 25 | 格拉纳达足球俱乐部    | 位置        | 西班牙安达鲁西亚自治区格拉纳达省首府格拉纳达  |
| 26 | 比利亚雷亚尔足球俱乐部  | 位置        | 西班牙巴伦西亚自治区卡斯特利翁省比利亚雷亚尔市 |
| 27 | 毕尔巴鄂竞技足球俱乐部  | 位置        | 西班牙北部巴斯克自治区比斯开省毕尔巴鄂市    |
| 28 | 瓦伦西亚足球俱乐部    | 位置        | 西班牙瓦伦西亚                 |
| 29 | 皇家比戈塞尔塔足球俱乐部 | 位置        | 西班牙加利西亚自治区比戈市           |
| 30 | 皇家社会足球俱乐部    | 位置        | 西班牙巴斯克地区的圣塞巴斯提安         |
| 31 | 皇家西班牙人足球俱乐部  | 位置        | 西班牙加泰罗尼亚自治区巴塞罗那         |



In [73]: 1 location_dic={
2     '西班牙安达鲁西亚自治区加的斯':[-6.18, 36.32],
3     '西班牙巴伦西亚自治区埃尔切':[-0.42, 38.15],
4     '西班牙安达卢西亚自治区首府塞维利亚':[-5.59, 37.23],
5     '西班牙纳瓦拉自治区首府潘普洛纳':[-1.38, 42.49],
6     '西班牙巴塞罗那市':[2.05, 41.23],

```

```

In [73]: 1 location_dic=[  
2     "西班牙安达鲁西亚自治区加的斯":[-6.18,36.32],  
3     "西班牙巴伦西亚自治区埃尔切":[-0.42,38.15],  
4     "西班牙安达卢西亚自治区首府塞维利亚":[-5.59,37.23],  
5     "西班牙纳瓦拉自治区首府潘普洛纳":[-1.38,42.49],  
6     "西班牙巴塞罗那市":[2.05,41.23],  
7     "西班牙安达鲁西亚自治区格拉纳达省会格拉纳达":[-3.41,37.13],  
8     "西班牙巴伦西亚自治区卡斯特利翁省比利亚雷亚尔市":[-0.06,39.56],  
9     "西班牙北部巴斯克自治区比斯开省毕尔巴鄂市":[-2.58,43.15],  
10    "西班牙瓦伦西亚":[-0.22,39.28],  
11    "西班牙加利西亚自治区比戈市":[-8.43,42.14],  
12    "西班牙巴斯克地区的圣塞巴斯提安":[-3.21,41.88],  
13    "西班牙加泰罗尼亚自治区巴塞罗那市":[2.11,41.23],  
14    "西班牙安达鲁西亚首府塞维利亚":[-5.59,37.23],  
15    "西班牙首都马德里":[-3.39,40.24],  
16    "西班牙首都马德里":[-3.40,40.20],  
17    "西班牙首都马德里":[-3.42,40.30],  
18    "西班牙马略卡帕尔马":[2.39,39.34],  
19    "西班牙东部巴伦西亚市":[-0.22,39.28],  
20    "西班牙马德里自治区赫塔费":[-3.42,40.18],  
21    "西班牙北部巴斯克自治区维多利亚":[-2.40,42.51]  
22 }  
  
In [74]: 1 # 基础数据  
2 # 基础数据  
3  
4 keys=df['object']  
5 values=df['subject']  
6  
7 c=GeoO  
8  
9 for (i,j) in location_dic.items():  
10    c.add_coordinate(i,j[0],j[1])  
11  
12 c.add_schema(maptypes="西班牙")  
13 c.add(  
14     "球队所在地",  
15     [list(z) for z in zip(keys, values)],  
16     type_=ChartType.EFFECT_SCATTER,  
17   )  
18 c.set_series_opts(label_opts=opts.LabelOpts(is_show=False))  
19 c.set_global_opts(title_opts=opts.TitleOpts(title="西甲全部球队所在地"))  
20  
21  
22 # 打开html  
23 os.system("render.html")  
24

```

Out[74]: 0

```

In [28]: 1 # 没用的 英文名  
2  
3 eng=["Cádiz", "Elche", "Sevilla", "Pamplona", "ciudad de barcelona", "Granada", "Villarreal", "Ciudad de Bilbao", "Valencia", "Vigo", "San Sebastián",  
4 print(eng)  
5

```

['Cádiz', 'Elche', 'Sevilla', 'Pamplona', 'ciudad de barcelona', 'Granada', 'Villarreal', 'Ciudad de Bilbao', 'Valencia', 'Vigo', 'San Sebastián', 'Barcelona', 'Sevilla', 'Madrid', 'Palma de Mallorca', 'Valencia', 'Getafe', 'Victoria']

## 四、实验结果和分析

原先得到的 owl 文件部分截图如下：

```

<!-- http://www.semanticweb.org/think/ontologies/Football.owl#主要竞争对手 -->
<owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#主要竞争对手"/>

<!-- http://www.semanticweb.org/think/ontologies/Football.owl#就任主席 -->
<owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#就任主席"/>

<!-- http://www.semanticweb.org/think/ontologies/Football.owl#所属球队 -->
<owl:ObjectProperty rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#所属球队"/>

<!-- http://www.semanticweb.org/think/ontologies/Football.owl#执教 -->

```

经过转换后的 rdf 文件部分截图如下：

```

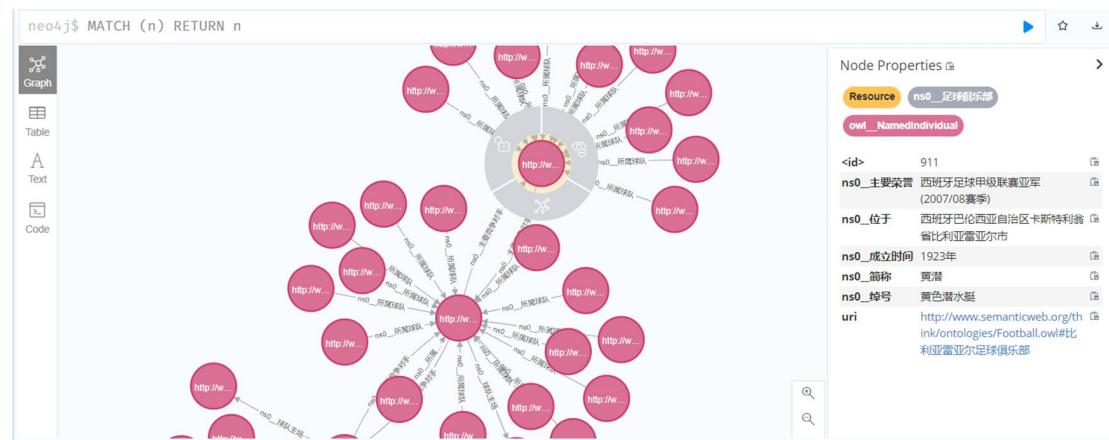
<rdf:Description rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#主要竞争对手">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#就任主席">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.semanticweb.org/think/ontologies/Football.owl#所属球队">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>

```

经过导入后，我们得到的 neo4j 图数据库，我们得知，这张知识图谱一共包含了 871 个节点以及 169 个关系：





同义词对齐结果实例：

```

1 import synonyms
2 # print('国籍：', synonyms.nearby('国籍'))
3 print(synonyms.compare('简称', '绰号'))
```

0.197



```

1 get_entity_info('马略卡的昵称是什么')
MATCH (a:entity{name:"马略卡"})
RETURN a.绰号
'MATCH (a:entity{name:"马略卡"})\nRETURN a.绰号'
```

自然语言分词结果实例：

```

In [5]: 1 text = u'西甲的第一名是谁'
2 lac_result = lac.run(text)
3 lac_result

Out[5]: [[['西甲', '的', '第一名', '是', '谁'], ['nz', 'u', 'm', 'v', 'r'], [3, 0, 2, 0, 1]]]

In [6]: 1 texts = [u'马德里竞技俱乐部在西甲里面排名多少', '皇家贝蒂斯足球俱乐部的排名和皇家马德里俱乐部的排名那个比较好']
2 lac_results = lac.run(texts)
3 lac_results

Out[6]: [[[马德里竞技俱乐部', '在', '西甲', '里面', '排名', '多少'],
['ORG', 'p', 'nz', 'f', 'v', 'r'],
[3, 0, 3, 1, 2, 1]],
[['皇家贝蒂斯足球俱乐部', '的', '排名', '和', '皇家马德里俱乐部', '的', '排名', '那个', '比较好'],
['ORG', 'u', 'vn', 'c', 'ORG', 'u', 'vn', 'r', 'a'],
[2, 0, 2, 0, 3, 0, 2, 1]]]

In [7]: 1 text = '塞维利亚俱乐部的出生年月是什么时候？'
2 lac_result = lac.run(text)
3 lac_result

Out[7]: [[['塞维利亚俱乐部', '的', '出生', '年月', '是', '什么时候', '?'],
['ORG', 'u', 'vn', 'n', 'v', 'n', 'w'],
[3, 0, 2, 2, 0, 2, 0]]]

```

为了验证我们的知识库问答系统的准确性，我们使用了三个有代表性的自然语言问句，分别代表了 1) 查询一个 entity 的所有信息：皇家马略卡足球俱乐部的昵称是什么？ 2) 查询与 a 有对应关系的 b：卡尔洛·安切洛蒂在哪个球队执教？ 3) 查询 a 与 b 的关系：皇家贝蒂斯足球俱乐部和塞维利亚足球俱乐部是什么关系？

我们对三个问题进行测试，得到的 cypher 查询语句如下：

## 测试

```

1 #aim_list已固定，只需修改输入的sentence
2 sentence1 = '皇家马略卡足球俱乐部的昵称是什么'
3 sentence2 = '卡尔洛·安切洛蒂在哪个球队执教'
4 sentence3 = '皇家贝蒂斯足球俱乐部和塞维利亚足球俱乐部是什么关系？'
5 q_part = question_part(sentence1, aim_list)
6 which_template(q_part, sentence1)
7 q_part = question_part(sentence2, aim_list)
8 which_template(q_part, sentence2)
9 q_part = question_part(sentence3, aim_list)
10 which_template(q_part, sentence3)

MATCH (a:`ns0_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家马略卡足球俱乐部"})
RETURN a.ns0_绰号
MATCH (a:`ns0_主教练`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#卡尔洛·安切洛蒂"}) - [:ns0_执教] -> (b)
RETURN b
MATCH (a:`ns0_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家贝蒂斯足球俱乐部"}) - [r] -> (b:`ns0_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#塞维利亚足球俱乐部"})
RETURN r

```

然后我们将这些查询语句在 neo4j 数据库中进行查询验证，分别为：

1) MATCH (a:`ns0\_足球俱乐部`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家马略卡足球俱乐部"})  
`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家马略卡足球俱乐部"})  
RETURN a.ns0\_绰号



a.ns0_绰号
"LosBermellones"

Started streaming 1 records after 6 ms and completed after 6 ms.

2) MATCH (a:`ns0\_\_主教练`  
`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#卡尔洛·安切洛蒂"}) - [:`ns0\_\_执教`] -> (b)  
RETURN b



3) MATCH (a:`ns0\_\_足球俱乐部`  
`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#皇家贝蒂斯足球俱乐部"})-[r] -(b:`ns0\_\_足球俱乐部`  
`{uri:"http://www.semanticweb.org/think/ontologies/Football.owl#塞维利亚足球俱乐部"})  
RETURN r



我们将这些查询的数据与 python 终端输入自然语言后直接得到的数据输出进行对比：

```

1 import json
2 from py2neo import Graph, Node, Relationship, NodeMatcher, Subgraph
3
4 graph = Graph('http://localhost:7474', auth=("neo4j", "Lycilillyc"))
5
6
7 matcher = NodeMatcher(graph)
8
9 graph.run(q1)
10 "LosBermellones"
11
12 graph.run(q2)
13
14 {"identity": 1083, "labels": ["Resource", "ns0__足球俱乐部", "owl_NamedIndividual"], "properties": {"ns0_简称": "皇马", "ns0_主要荣誉": "西班牙足球甲级联赛冠军(34次)", "ns0_位于": "西班牙首都马德里", "ns0_成立时间": "1902年", "ns0_球迷昵称": "美凌格", "uri": "http://www.semanticweb.org/think/ontologies/Football.owl#皇家马德里足球俱乐部"}}
15
16 graph.run(q3)
17
18 {"identity": 1039, "start": 1278, "end": 1164, "type": "ns0__主要竞争对手", "properties": {}}
19 {"identity": 912, "start": 1164, "end": 1278, "type": "ns0__主要竞争对手", "properties": {}}

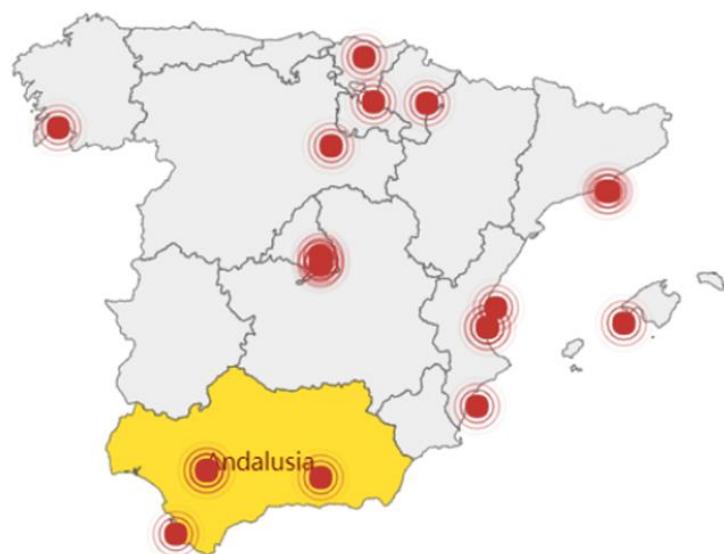
```

可以看到，两者完全一样，我们的知识库问答系统的查询功能健壮且完善。

网页动态可视化中，我们对各个球队的位置信息进行可视化，并且显示其具体信息：

## 西甲全部球队所在地

● 球队所在地



## 西甲全部球队所在地

● 球队所在地

