魔兽世界人物信息知识图谱

《魔兽世界》（英语：World of Warcraft，缩写作 WoW），简称魔兽，是由暴雪娱乐制作的一款大型多人在线角色扮演游戏。魔兽世界的剧情开端点是在《魔兽争霸3：冰封王座》结束之后的故事。我们基于魔兽世界wiki，对于其中人物信息数据进行了爬取、清洗、关系抽取，并建立知识图谱。我们将人物基础信息与关系存在json格式文件中，将人物、属地、组织、种族作为结点生成rdf格式文件，最后将数据导入neo4j数据库进行数据分析，并在此基础上实现了简单的问答功能。

1. 数据收集与整理

1.1 数据来源

我们以魔兽世界中文维基，自由编辑的魔兽资料库 - 灰机wiki (huijiwiki.com)作为本次知识图谱搭建的数据来源网站，该网站是目前据我们所知内容最为完善的中文版魔兽世界在线资料库，其中包括对于《魔兽世界》各版本人物、地图、装备等游戏内容的介绍。其中，与游戏背景故事剧情的推动最为相关的就是其中人物相关的页面，因此我们围绕魔兽世界中文维基中收录的524个官方角色资料进行了数据收集与整理。

1.2 数据爬取

首先我们对魔兽中文维基的多个人物页面进行了浏览，发现每个人物页面都会包含一个人物基本信息的总览表格，其中记录了人物的名称、头衔、等级等基本信息以及阵营、势力等势力信息，另外还有亲属、同伴等人物关系相关的信息。知识图谱这种储存数据的形式非常适合用于记录处理这种形式的数据，于是我们选择对每个人物的基本信息表格进行爬取、解析、整理。

一个典型的人物介绍页面，右侧表格介绍了角色基尔加丹的基本信息、势力信息和人物关系

在数据爬取方面，我们使用了Selenium自动化浏览器工具的API作为框架基础，用python语言搭建了一个爬虫程序。爬虫程序会模仿用户查询维基网站的行为，首先获取所有的人物页面相关链接，然后逐个访问，通过解析网页元素定位到人物信息简介表，将其中的数据以半结构化的形式爬取成简单的键-值对（如 `{"状态": "死亡（剧情）\n可击败（游戏中）"}` ），交由程序进行进一步的细节解析处理。

1.3 数据解析与清洗

首先，我们统计了人物表格中所可能包含的属性类别，并将单个人物的各项数据分为三种不同的大类进行处理：

1. 单项属性：即人物一般只会有对应的单个值的属性，如性别、年龄等等。对于这些属性，我们会按照属性的数据类型（如字符串、枚举类型、整型类型）对数据的文字进行解析。
2. 多项数据：除了单项属性外，人物可能还会包含头衔、所在地点等含有多个值的列表形式的数据。数据源网页中对于这些数据的格式并没有做出要求，格式较为混乱，这里我们结合使用了多个正则表达式，再通过人工处理个别异常的方式，整理出了人物的多项数据。
3. 关系数据：在多项数据的基础上，人物还往往包含人际关系、势力等与其他实体结点相关的数据，这里我们通过正则表达式和人工调整的方法，将相同实体的不同名称对应起来，清洗到具有较高一致性的程度。

最后我们形成了 `.json` 格式的初步整理数据，以供进一步构建知识图谱使用；以下是其中一个人物的数据示例：

```
{       "source":"https://warcraft.huijiwiki.com/wiki/%E9%98%BF%E5%85%8B%E8%92%99%E5%BE%B
        "中文名":"阿克蒙德",
        "前势力":[
            "唤醒者"
        ],
        "势力":[
            "燃烧军团"
        ],
        "头衔":[
            "污染者（The Defiler）",
            "军团的艾瑞达霸王（Eredar Overlord of the Legion Forces）",
            "伟者（Great One）"
        ],
        "导师":[
            "萨格拉斯",
            "萨奇尔"
        ],
        "性别":"男性",
        "状态":[
            "死亡（剧情中）",
            "可击败（游戏中）"
        ],
        "种族":[
            "曼阿瑞（恶魔）"
        ],
        "职业":[
            "术士"
        ],
        "英文名":"Archimonde",
        "身份":[
            "燃烧军团之王（Lord of the Burning Legion）"
        ],
        "门徒":[
            "古尔丹"
        ],
        "阵营":[
            "中立"
        ]
    }
```


2.json转rdf格式

由于后续工作需要将文件导入Neo4j数据库，我们首先需要将其转为rdf格式。

首先确定选取哪些属性成为节点，对此我们对于所有key出现频数进行了一次排序

```
  "中文名": 524,
  "种族": 523,
  "英文名": 523,
  "性别": 480,
  "状态": 466,
  "阵营": 429,
  "所在地": 424,
  "势力": 417,
  "头衔": 308,
  "身份": 278,
  "对联盟玩家态度": 249,
  "对部落玩家态度": 249,
  "职业": 220,
  "等级": 151,
  "前势力": 133,
  "生命值": 122,
  "法力值": 54,
  "门徒": 45,
  "导师": 44,
  ...
```

其中二值（状态、性别、态度等）与单值（中文名、职业、等级、生命值、法力值等）的属性不适合作为结点，仅仅作为每个点的属性即可；频数过低的属性不适合作为结点。

最终我们选择 **人物**、**种族**、**势力**、**所在地** 作为结点。
在rdf中定义如下：

```
group:影踪突袭营
        relation:name "影踪突袭营" ;
        a relation:Group .
```

值得一提的是，其中group的类名为了方便起见，我们直接使用了name相同的内容，其实是不合理的，这导致我们后续处理时需要较强的正则判定再次处理每个节点的名字，相关代码如下

```
def preprocess_item(item):
    item = re.sub(u"\\(.*?\\)", "", item)
    item = re.sub(r'[\'+\-*/’“”‘\?\[\],&†\(\)\"·\n]', '', item)
    item = re.sub(r'\s+', "", item)
    return item
```

最后我们用于转rdf的代码记录在json2rdf.py文件中。

3. 知识图谱导入Neo4j图数据库

为了更好的对知识图谱进行可视化和用于后续应用场景，我们将图谱导入Neo4j图数据库。

Neo4j是一个高性能的NoSQL图形数据库，被广泛的使用于各类知识图谱的存储和应用场景。由于Neo4j数据库本身不提供rdf格式的图谱导入，我们使用了开源插件neosemantics来实现一键式导入。

Neo4j: https://neo4j.com/

neosemantics: https://github.com/neo4j-labs/neosemantics

在新建空的本地数据库后，只需要运行命令就可以导入图谱：

```
call n10s.rdf.import.fetch( "The path of .rdf file","Turtle")
```

通过Cypher语言，我们可以实现简单的图谱查询功能。

例如，通过match关键字查询图谱中的节点信息。

```
match(n) return count(n)         // 查询节点个数
match(n) where not () --> (n) return count(distinct n), labels(n)       // 查询根节点信息
```

```
neo4j$ match (n) where not ()⟶(n) return count(distinct n), labels(n)
```

Table | Text | Warn | Code

	count(distinct n)	labels(n)
1	1	["_GraphConfig"]
2	1	["_NsPrefDef"]
3	379	["Resource", "ns0__Character"]

在本图中共有节点1897个，其中根节点379个，都属于Character类别。

因此，本知识图谱实际上是从多个维度描述了“英雄”之间的关系。以“势力”节点为例，下图为查询各个“英雄”与“势力”关系得到的子图（部分）。

更进一步的，通过函数的组合，可以定向检索图中某些节点的统计量特性，例如下图为对节点度数排序的结果。

image-20211111203928860

利用Cypher查询语言，我们可以进一步实现很多更复杂的图算法来对知识图谱进行学习和推理。Neo4j维护了一个常用的图算法库，Neo4j Graph Data Science Library。

GDS: https://github.com/neo4j/graph-data-science

此处我们给出两个简单的使用GDS中图算法的示例。

在使用之前，首先需要从数据库中生成Graph，此处我们使用所有的节点和关系。

image-20211111213438524

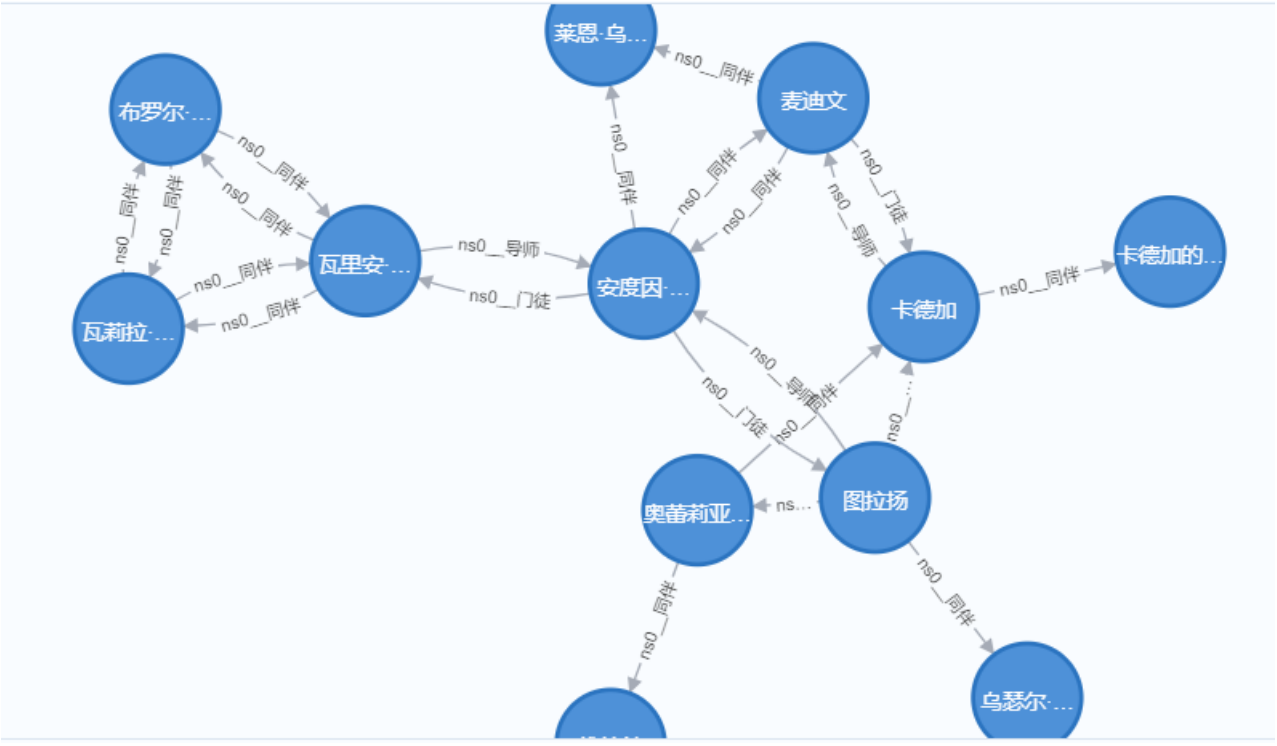
之后，调用Degree-centrality算法，来通过度数评估图中节点的重要性，节点的出度和入度都在该算法计算范围之内。可以看到，在图谱中，“人类”是最受欢迎的节点，图谱中的很多实体都与人类相关。

image-20211111214038659

此外，通过算法我们也可以来对图谱中的隐藏信息进行挖掘。比如，使用邻居节点预测算法，我们可以通过节点的共同朋友节点来预测两个节点是否可能有所联系。

image-20211111220142677

在图谱可视化的结果中我们可以看到，虽然两个节点没有直接相连，但通过其他节点成功预测到了它们之间可能有所联系。



4.知识图谱应用——基于图数据库的知识问答

基于上述工作，我们在知识图谱上尝试了基于图数据库的知识问答这一应用。
在实际执行过程中，用户输入一个问题或只输入关键词，问答算法提取其中的关键字后执行图数据库查找，最终返回对应的结果。
我们目前支持的问句形式有：

(1)查询英雄的属性，包括种族/性别/阵营/所在地/势力/头衔/职业/等级。

(2)查询英雄关系对象，如安东尼达斯的门徒是吉安娜。关系包括亲属/门徒/导师/同伴。

(3)查询所在地/种族/势力的从属关系，包括英雄属于哪些势力，某位置有哪些英雄等。

具体效果如下：

```
问题:阿尔萨斯·米奈希尔属于哪个势力?
回答: 阿尔萨斯·米奈希尔属于的势力有：天灾军团

问题:德米提恩的等级是多少?
回答: 德米提恩的等级是62

问题:哪些英雄属于天灾军团?
回答: 属于天灾军团势力的英雄有：女公爵布劳缪克丝 巫妖王 辛达苟萨 伯瓦尔·弗塔根 达尔坎·德拉希尔
德拉诺什·萨鲁法尔 耐奥祖 阿努巴拉克 鲜血女王兰娜瑟尔 瑞文戴尔男爵
克尔苏加德 玛维恩 阿尔萨斯·米奈希尔

问题:阿格拉玛的头衔是什么?
回答: 阿格拉玛的头衔是(前)伟大的萨格拉斯的副官

问题:哪些英雄属于蜥蜴人?
回答: 种族是蜥蜴人的英雄有：阿卡里达尔

问题:哪些英雄位于暗月岛?
回答: 位于暗月岛的英雄有：月牙

问题:安东尼达斯的门徒有哪些?
回答: 安东尼达斯的门徒是吉安娜·普罗德摩尔 多位达拉然法师

问题:戴勒琳·夏月队长的种族是什么?
回答: 戴勒琳·夏月队长的种族是暗夜精灵

问题:范妮·盘角的所在地是哪里?
回答: 范妮·盘角的所在地是卡桑琅丛林雄狮港
```

在处理时，获得疑问文本后，算法首先对其进行分词，
从中识别出如下实体：属性词、英雄名称、势力名称、种族名称、所在地名称、从属词和关系词。
随后利用问句中的这些实体作为关键词，我们使用py2neo连接neo4j数据库并在数据库中使用cypher语句进行查询，
再对返回结果进行组装， 就得到了回答。