金庸知识图谱构建文档

AUTHOR: yuzhang.liu<lyz0409@163.com>

环境准备

软件

- python 3.6+
- mongodb
- neo4j

MongoDB 安装配置

1. 安装 mongodb server

安装地址：https://www.mongodb.com/download-center/community

2. 启动 mongodb 进程服务

打开 `CMD` 终端，运行

```
mongod --dbpath "F:\MongoDB\data"
```

4. 在终端进入 mongo 主程序

```
mongo
```

更多使用教程： http://www.runoob.com/mongodb/mongodb-tutorial.html

Neo4j 安装

1. 安装 java，配置 java 的环境变量

2. 安装neo4j

安装地址：https://neo4j.com/download-thanks/?edition=community&release=3.5.3&flavour=winzip&_ga=2.249237184.658138385.1551513830-703156177.1551513830#

安装教程：https://blog.csdn.net/lihuaqinqwe/article/details/80314895

3. 启动 neo4j

- 打开 cmd
- 执行

```
neo4j.bat console
```

python 依赖库

```
jieba
requests
scrapy
pymongo
py2neo
```

可通过 pip 安装所需的依赖库，考虑到大多数库放在国外服务器，可以走清华源下载

```
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple jieba requests
scrapy pymongo py2neo
```

scrapy

scrapy 简单文档

1.新建项目： `scrapy startproject myspider`

- scrapy.cfg 项目的配置信息，主要为Scrapy命令行工具提供一个基础的配置信息
- items.py 设置数据存储模板，用于结构化数据，如：Django 的 Model
- pipelines 数据处理行为，如：一般结构化的数据持久化
- settings.py 配置文件，如：递归的层数、并发数，延迟下载等
- spiders 爬虫目录，如：创建文件，编写爬虫规则

2.创建爬虫文件：cd myspider->scrapy genspider baidu baidu.com

3.明确目标：编写items.py

4.制作爬虫爬取网页：spiders/xx_spider.py

5.设计管道存储爬取内容：pipelines.py

爬虫爬取数据

scrapy 爬取金庸全套小说

1. 新建 scrapy 项目

```
scrapy startproject xiaoshuo_spider
```

2. 修改 items.py

```
import scrapy
```

```
class XiaoshuoSpiderItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    name = scrapy.Field()    #小说名字
    chapter_name = scrapy.Field()    #小说章节名字
    chapter_content = scrapy.Field()     #小说章节内容
```

3. 创建爬虫文件 xiaoshuo_spider.py

在 `xiaoshuo_spider/xiaoshuo_spider/spiders` 目录下新建 xiaoshuo_spider.py

```
# -*- coding: utf-8 -*-

import re
import scrapy
from urllib.request import urlopen
from pymongo import MongoClient

mongo=MongoClient()
db=mongo["yinyong"]["xiaoshuo"]

re_paragraph = re.compile('(?<=<p>).*?(?=</p>)')

class XiaoshuoSpider(scrapy.Spider):
    name = 'xiaoshuo_spider'
    allowed_domains = ['jinyongwang.com']
    start_urls = [
        'http://www.jinyongwang.com/fei/',
        'http://www.jinyongwang.com/xue/',
        'http://www.jinyongwang.com/lian/',
        'http://www.jinyongwang.com/tian/',
        'http://www.jinyongwang.com/she/',
        'http://www.jinyongwang.com/bai/',
        'http://www.jinyongwang.com/lu/',
        'http://www.jinyongwang.com/xiao/',
        'http://www.jinyongwang.com/shu/',
        'http://www.jinyongwang.com/shen/',
        'http://www.jinyongwang.com/xia/',
        'http://www.jinyongwang.com/yi/',
        'http://www.jinyongwang.com/bi/',
        'http://www.jinyongwang.com/yuan/',
        'http://www.jinyongwang.com/yue/',

    ] #金庸王全套小说
```

```
    #获取小说章节的URL
    def parse(self, response):

        cnt_url = "/".join(response._url.strip("/").split("/")[:-1])
        name =
response.xpath('//div[@class="pu_breadcrumb"]//h3[@class="set"]/font/text()'
).extract_first()
        chapter_names =
response.xpath('//ul[@class="mlist"]//li/a/text()').extract()
        chapter_urls =
response.xpath('//ul[@class="mlist"]//li/a/@href').extract()

        chapters = []
        for chapter_name, chapter_url in zip(chapter_names, chapter_urls):
            chapter_name = chapter_name.replace("\u3000", " ")
            response = urlopen(cnt_url + chapter_url)
            html = response.read().decode("utf-8")
            texts = re_paragraph.findall(str(html))
            chapters.append({"name": chapter_name, "content":
"\n".join(texts)})

        db.save({"book_name": name, "chapters": chapters})
```

4. 开始执行爬虫脚本

```
scrapy crawl xiaoshuo_spider
```

5. 从 mongodb 中导出小说文本，新建 convert.py

```
# -*- coding: utf-8 -*-

import os
from pymongo import MongoClient

mongo = MongoClient()
db = mongo['yinyong']['xiaoshuo']
dirname = "F:/jinyong/data/books"

for book_obj in db.find():
    print("start to process {}...".format(book_obj["book_name"]))

    book_dir = os.path.join(dirname, book_obj["book_name"]+".txt")
    os.makedirs(book_dir, exist_ok=True)
```

```
    for chapter in book_obj["chapters"]:
        chapter_fname = os.path.join(book_dir, chapter['name'])
        with open(chapter_fname, "w", encoding="utf-8") as wf:
            wf.writelines(chapter['content'] + "\n")
```

6. jieba 词性标注获取人名

```
# -*- coding: utf-8 -*-

import jieba
import jieba.posseg as pseg

from pymongo import MongoClient

# 定义 mongodb 连接对象
mongo = MongoClient()
# 使用的 mongodb 指定数据库数据表
db = mongo['jinyong']['xiaoshuo']
save_path = "F:/jinyong/data/persons.txt"

print("start processing...")
persons = []
for book_obj in db.find():
    for chapter in book_obj["chapters"]:
        for word, tag in pseg.cut(chapter['content']):
            if tag == "nr":
              persons.append(word)
print("save to {}...".format(save_path))
# 去重
persons = list(set(persons))
with open(save_path, "w") as wf:
    for word in persons:
        wf.writelines("{}\n".format(word))
```

小说人物的爬取

要对百科中 infobox 内容进行爬取

由于得到人物列表的程序执行比较慢，可以直接下载整理好的人物列表，人物列表地址：https://github.com/liuyuzhangolvz/novel-kg/blob/master/crawl-baike/persons.txt ，下载后将其放在 `F:/jinyong/data/persons.txt` 文件下

1. 新建一个爬虫项目

```
scrapy startproject person_spider
```

2. 在 spiders 目录下新建爬虫程序

在 `person_spider/person_spider/spiders` 目录下新建 `person_spider.py`

！**注意：**PERSONS_FILE 这个常量设置上述解压人物列表（persons.txt）的存放地址

```
# -*- coding: utf-8 -*-

""" 爬取人物百度百科 infobox
"""

import re
import scrapy
from pymongo import MongoClient

# 常量
PERSONS_FILE = 'F:/jinyong/data/persons.txt' # 人物列表存放地址，这里最好用绝对地址哦
DB_NAME = 'jinyong'    # mongodb 库名
TABLE_NAME = 'persons'  # mongodb 表名
DROP_KEYS = ['中文名', '饰演', '配音']  # 要删除的 info 键名
KEYWORDS = ['金庸', '飞狐外传', '雪山飞狐', '连城诀', '天龙八部', '射雕英雄传',
        '白马啸西风', '鹿鼎记', '笑傲江湖', '书剑恩仇录', '神雕侠侣',
```

```
        '侠客行', '倚天屠龙记', '碧血剑', '鸳鸯刀', '越女剑']

# 变量
mongo = MongoClient()  # mongodb 的操作对象，使用默认参数即可
mongo[DB_NAME].drop_collection(TABLE_NAME)  # 删除 persons 数据库，好重写
db = mongo[DB_NAME][TABLE_NAME]  # yz 是 mongo 的数据库， persons 是库 yz 下的一张表
re_split = re.compile(r'[,、，；]')  # 字符串切割正则
re_match = re.compile(r'({})'.format('|'.join(KEYWORDS)))  # 生成关键词正则，表示网页中必须最少含有这些关键词的一个

def strQ2B(content):
    """全角转半角"""
    content = str(content)
    rstring = ""
    for uchar in content:
        inside_code=ord(uchar)
        if inside_code == 12288:                              #全角空格直接转换
            inside_code = 32
        elif (inside_code >= 65281 and inside_code <= 65374): #全角字符（除空格）根据关系转化
            inside_code -= 65248

        rstring += chr(inside_code)
    return rstring

def clean_content(content):
    """
    文本清洗:
    - 去除 html 标签、实体
    - 去除 url
    - 去除 \t \n 等符号
    """

    content = strQ2B(content)
    re_del = re.compile(r'<[^>]+>|\s+|\&\w+\;|http://[a-zA-Z0-9.?/&=:]*',re.S)
    content = re_del.sub("", content)
    content = content.replace(" ","")
    return content


def get_urls():
    """从人物列表中获取url"""
    url_format = 'https://baike.baidu.com/item/{}'
```

```
    urls = []
    with open(PERSONS_FILE, 'r', encoding='utf-8') as rf:
        for line in rf:
            urls.append(url_format.format(line.strip()))
    return urls

class PersonSpider(scrapy.Spider):
    """主爬虫类，需要继承 scrapy 的 Spider 类"""
    name = 'person_spider' # 爬虫名称，和文件名一致
    allowed_domins = ['baike.baidu.com'] # 允许爬取的域名
    start_urls = get_urls() # 要爬去的 url
    fail_handle = open('./fail.txt', 'w', encoding='utf-8') # 失败的文件资源，对于插入失败的人物可能是百科中出现了歧义，所以需要人工检查插入

    def parse(self, response):
        """ 解析页面，这里只存储 basic-info 的内容，且要判断 '金庸' 二字是否出现在页面中
        """
        # 通过 xpath 语法获取当前人物名
        person = response.xpath("//dd[@class='lemmaWgt-lemmaTitle-title']//h1/text()").extract_first()
        if re_match.findall(clean_content(response.xpath("//div[@class='content-wrapper']").extract_first())):
            # names 代表 basic-info 里的键名
            names = response.xpath("//div[contains(concat(' ', normalize-space(@class), ' '), 'basic-info')]//dt[contains(concat(' ', normalize-space(@class), ' '), 'name')]").extract()
            # vals 代表 baisic-info 里的值
            vals = response.xpath("//div[contains(concat(' ', normalize-space(@class), ' '), 'basic-info')]//dd[contains(concat(' ', normalize-space(@class), ' '), 'value')]").extract()
            # 这里使用断言判断 names 的个数和 vals 的能对齐
            assert len(names) == len(vals)
            # 对 names 和 vals 进行规范化
            names = [clean_content(x) for x in names]
            values = []
            for val in vals:
                val = clean_content(val)
                arr = [str(x) + ')' for x in val.split(")")] if val.count(')') > 1 else re_split.split(val)
                if len(arr) > 1:
                    values.append(arr)
                else:
                    values.append(arr[0])
```

```
            if not values:
                # 失败则插入文件
                self.fail_handle.writelines("{} {}\n".format(str(person),
response._url))
            else:
                info = dict(zip(names, values))
                for drop_key in DROP_KEYS:
                    if drop_key in info:
                        del info[drop_key]
                # 插入 mongodb
                db.save({"person": person, "info": info})
        else:
            if person is not None:
                # 失败则插入文件
                self.fail_handle.writelines("{} {}\n".format(str(person),
response._url))
```

3. 修改 settings.py

把 ROBOTSTXT_OBEY 设置为 False，即

```
ROBOTSTXT_OBEY = False
```

4. 执行爬虫程序

```
scrapy crawl person_spider
```

将 mongodb 的数据导入 neo4j

转换脚本

1. 新建 mongo2neo.py

```
# -*-  coding: utf-8 -*-

from pymongo import MongoClient
from py2neo import Node, Relationship, Graph, NodeMatcher

mongo = MongoClient()
db = mongo['jinyong']['persons']

graph = Graph('127.0.0.1:7474', user='neo4j',password='123456')
matcher = NodeMatcher(graph)
graph.delete_all()
```

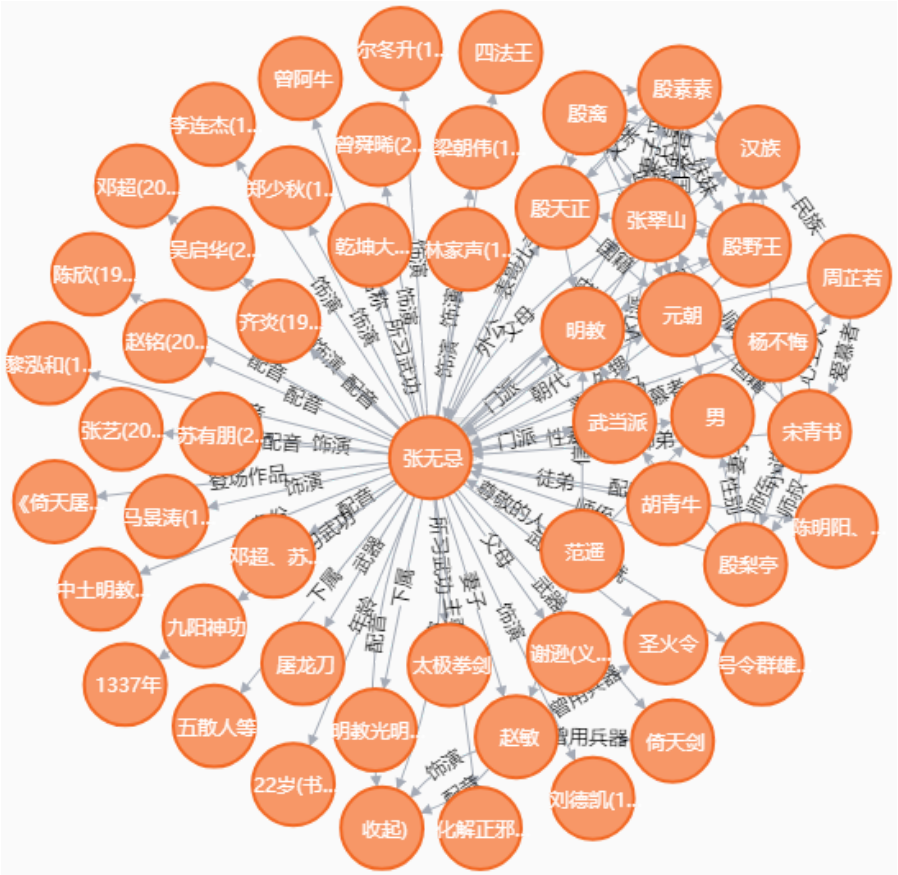
```
def look_and_create(name):
    end = matcher.match("Jinyong", name=name).first()
    if end is None:
        end = Node('Jinyong', name=name)
    return end

def insert_one_data(arr):
    start = look_and_create(arr[0])
    items = [arr[2]] if isinstance(arr[2], str) else arr[2]
    for name in items:
        end = look_and_create(name)
        r = Relationship(start, arr[1], end, name=arr[1])
        graph.create(r)

def insert_datas():
    print('transfering...')
    for data in db.find():
        for key, val in data['info'].items():
            if not key.strip() or not val:
                continue
            insert_one_data([data['person'], key, val])

if __name__ == "__main__":
    insert_datas()
```

2. 执行转换脚本

```
python mongo2neo.py
```

neo4j 可视化

数据导入成功后可输入 http://localhost:7474/browser/ 设置初始用户和密码后即可进行图谱可视化，下图是 `张无忌` 对应的可视化例子



基于 d3.js 问答图谱可视化

原理

- 对查询进行词法分析，得到查询中的“人名”nr， 然后用人名去检索 mongodb 得到相应人物的数据
- 将数据转换成 d3.js 需要的格式传给前端显示， 其中答案（与查询直接相关的内容）会“绿色线”显示，主要实现原理是判断键值对的键是否在查询中，以下以一个例子说明：

查询项：张无忌是什么门派的?
词性标注：张无忌/nr 是/v 什么/r 门派/n 的/uj ？ /x
查询：

```
db.find({"person": "张无忌"})

"""结果
{ "_id" : ObjectId("5cd1942d421aa96e58c928be"), "person" : "张无忌", "info" :
{ "其他名称" : "曾阿牛", "登场作品" : "《倚天屠龙记》二至四卷(第11回到第40回登场)及其衍生
作品[1]", "生日" : "1337年", "年龄" : "22岁(书末)", "性别" : "男", "朝代" : "元朝",
"民族" : "汉族", "门派" : [ "武当派", "明教" ], "身份" : "明教第三十四代教主", "武功"
: [ "九阳神功", "乾坤大挪移", "圣火令神功", "梯云纵", "七伤拳", "降龙十八掌" ], "武器"
: [ "倚天剑", "屠龙刀", "圣火令" ], "主要成就" : [ "化解正邪两道积怨", "号令群雄抗元",
```

```
"挑战六大门派" ] } }
"""
```

将 data["info"] 转成 d3.js 格式:

```
{'edges': [{'label': '其他名称', 'source': 0, 'target': 1, 'type': 'info'},
           {'label': '登场作品', 'source': 0, 'target': 2, 'type': 'info'},
           {'label': '生日', 'source': 0, 'target': 3, 'type': 'info'},
           {'label': '年龄', 'source': 0, 'target': 4, 'type': 'info'},
           {'label': '性别', 'source': 0, 'target': 5, 'type': 'info'},
           {'label': '朝代', 'source': 0, 'target': 6, 'type': 'info'},
           {'label': '民族', 'source': 0, 'target': 7, 'type': 'info'},
           {'label': '门派', 'source': 0, 'target': 8, 'type': 'answer'},
           {'label': '身份', 'source': 0, 'target': 9, 'type': 'info'},
           {'label': '武功', 'source': 0, 'target': 10, 'type': 'info'},
           {'label': '武器', 'source': 0, 'target': 11, 'type': 'info'},
           {'label': '主要成 就', 'source': 0, 'target': 12, 'type': 'info'}],
 'nodes': [{'name': '张无忌'},
           {'name': '曾阿牛'},
           {'name': '《倚天屠龙记》二至四卷(第11回到 第40回登场)及其衍生作品[1]'},
           {'name': '1337年'},
           {'name': '22岁(书末)'},
           {'name': '男'},
           {'name': '元朝'},
           {'name': '汉族'},
           {'name': ['武当派', '明教']},
           {'name': '明教第三十四代教主'},
           {'name': ['九阳神功', '乾坤大挪移', '圣火令神功', '梯云纵', '七伤拳', '降
龙十八掌']},
           {'name': ['倚天剑', '屠龙刀', '圣火令']},
           {'name': ['化解正邪两道积怨', '号令群雄抗元', '挑战六大门派']}]}
```

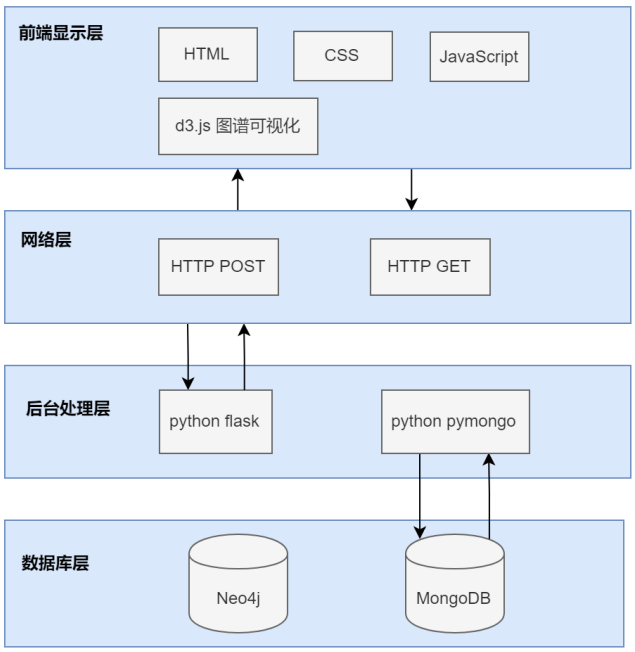
获取答案：因为键名里“门派”在查询中，所以“门派”所在关系是一个答案

安装

源码地址：https://github.com/liuyuzhangolvz/novel-kg/tree/master/kgqa

下载安装完毕后执行 `python app.py` 启动 flask 程序，然后在浏览器中访问 http://127.0.0.1:8000 即可访问

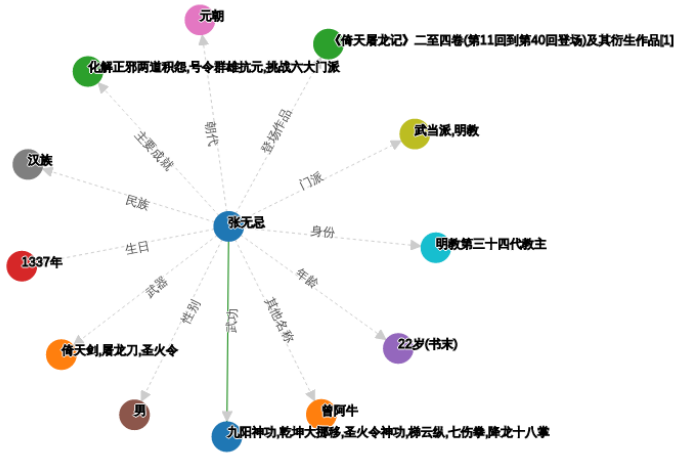
系统架构


前端显示层
HTML
CSS
JavaScript
d3.js 图谱可视化
网络层
HTTP POST
HTTP GET
后台处理层
python flask
python pymongo
数据库层
Neo4j
MongoDB

查询示例

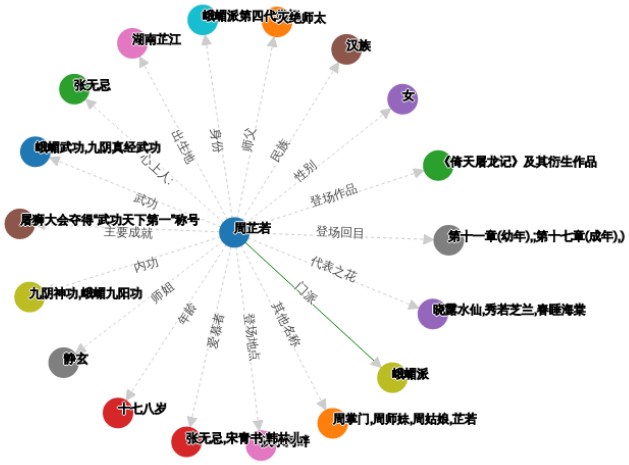
张无忌有哪些武功？

武功: 九阳神功,乾坤大挪移,圣火令神功,梯云纵,七伤拳,降龙十八掌



周芷若是哪个门派的

门派: 峨嵋派



GitHub: https://github.com/liuyuzhangolvz/novel-kg