

鬼灭之刃知识图谱项目报告

本项目依据成员的兴趣，构建了动漫《鬼灭之刃》的知识图谱。本图谱的数据来源为[Fandom鬼灭之刃wiki](#)和[萌娘百科鬼灭之刃](#)的词条。项目构建了以人物、组织、技能为主要实体的知识图谱，实现步骤主要有以下几点：

- 通过爬虫爬取网页wiki上的半结构化数据和文字数据
- 对爬取的数据分别进行数据处理，半结构化数据直接转化为三元组，文字数据进行知识抽取
- 对上述得到的三元组进行合并，以及人工检查清洗并转化为rdf/xml格式的文件
- 对得到的知识图谱导入neo4j数据库进行可视化，以及构建Q&A应用

OpenKG网址：[鬼灭之刃知识图谱 - 图谱 - 开放知识图谱 \(openkg.cn\)](#)

项目组成员

- 周添 （学号：12221046）
- 傅雨婷 （学号：3170100647）
- 张童晨 （学号：22221258）

目录

项目组成员
知识图谱的数据预处理
数据来源
实体对象爬取
半结构化实体关系抽取和转化
中文繁体简体转换
数据清洗
知识图谱的文本知识抽取
实体识别
纯文本知识抽取
三元组转换为RDF
知识图谱的应用（可视化和知识问答）
将知识图谱导入neo4j数据库
知识图谱可视化与查询
知识问答

知识图谱的数据预处理

数据来源

本项目所用的半结构化数据均来源于 Fandom 的鬼灭之刃中文维基百科网站，该网站的网址为 <https://kimetsu-no-yaiba.fandom.com/zh/wiki/>。该网站对于《鬼灭之刃》这部作品的世界观和故事情节有着详细的介绍，并提供了丰富的人物特征与关系信息。而深度知识抽取使用的文字数据，是将萌娘百科鬼灭之刃词条，网址为 <https://zh.moegirl.org.cn/鬼灭之刃/>，与 Fandom 百科词条信息合并获取的。

实体对象爬取

关于《鬼灭之刃》知识图谱构建所需的实体数据，鬼灭之刃的 Fandom 百科网站上，以半结构化数据形式进行展现。下面这张图片，显示的是作为主角团三人组中的"灶门炭治郎"、"嘴平伊之助"及"我妻善逸"等实体在目录网址上展示的形式。根据分析发现不同实体的网址是有相同的前缀，再加上以各个人物实体名称结合而成。根据这个规律我们可以爬取该网站的人物实体信息。



我们通过BeautifulSoup和requests_html库对网站上所有的实体数据进行爬取，具体指令如下：

```
1 from requests_html import HTMLSession
2
3 def Extraction_entity(keywords,web):
4     all_entities=[]
5     current_keywords=keywords.copy()
6     init_keywords=keywords.copy()
7     while len(current_keywords) >= 1:
8         seed = current_keywords.pop(0)
9         print('visiting', seed)
10        url = web + seed
11        session = HTMLSession()
12        response = session.get(url)
13        a_list = response.html.find('a')
14        current_ents = []
15        for a in a_list:
16            if a.attrs.get('class', '') == ('category-page__member-link', ):
17                current_ents.append(a.attrs['title'])
18        for t in current_ents:
19            if 'Template' in t: continue
20            if 'Category' in t:
21                if t not in init_keywords:
22                    current_keywords.append(t)
23                    init_keywords.append(t)
```

```

24         else:
25             all_entities.append(t)
26         return(all_entities)
27
28 #wikipedia of DemonSlayer
29 web ='https://kimetsu-no-yaiba.fandom.com/zh/wiki/'
30
31 #some keywords are listed here
32 keywords = ['Category:鬼杀队',
33             'Category:十二鬼月',
34             'Category:血鬼术',
35             'Category:呼吸法',
36             'Category:柱',
37             'Category:无限城',
38             'Category:人类',
39             'Category:鬼',
40             'Category:主角团'
41            ]
42 all_entities=Extraction_entity(keywords,web)

```

爬取得到的实体对象，如下图所示：

['不死川富綱', '不死川玄彌', '二十一歲三人組〈惡友組〉', '伊黑小芭內', '嘴平伊之助', '培育者', '宇髄天元', '富岡義勇', '尾崎', '後藤', '悲鳴嶼行冥', '我妻善逸', '斑紋', '時透無一郎', '柱', '栗花落香奈乎', '水柱', '炎柱', '煉獄杏壽郎', '煉獄禰壽郎', '甘露寺蜜璃', '產屋敷天音', '產屋敷彼方', '產屋敷玖伊那', '產屋敷耀哉', '產屋敷輝利哉', '神崎葵', '雷門炭治郎', '雷門禰豆子', '桑野匡近', '胡蝶三姐妹', '胡蝶忍', '胡蝶香奈惠', '蝶屋敷', '鎧鴉', '隱', '骰子牛前輩', '鱗瀧左近次', '上弦鬼', '下弦鬼', '十二鬼月', '半天狗', '可樂', '哀絕', '恨之鬼', '憎珀天', '猗窩座', '病葉', '積怒', '空喜', '零余子', '鳴女', '月之呼吸', '無慘的血鬼術招式列表', '珠世的血鬼術列表', '血鬼術', '呼吸', '呼吸法', '岩之呼吸', '戀之呼吸', '日之呼吸', '月之呼吸', '水之呼吸', '混合呼吸', '火之呼吸', '火神神樂', '炎之呼吸', '花之呼吸', '蛇之呼吸', '蟲之呼吸', '雷之呼吸', '霞之呼吸', '音之呼吸', '風之呼吸', '不死川富綱', '二十一歲三人組〈惡友組〉', '伊黑小芭內', '宇髄天元', '富岡義勇', '悲鳴嶼行冥', '時透無一郎', '桑島慈悟郎', '水柱', '炎柱', '煉獄杏壽郎', '煉獄禰壽郎', '甘露寺蜜璃', '胡蝶忍', '胡蝶香奈惠', '鱗瀧左近次', '不死川富綱', '下野弘', '不死川弘', '富岡義勇', '悲鳴嶼行冥', '時透無一郎', '柱', '栗花落香奈乎', '煉獄杏壽郎', '甘露寺蜜璃', '雷門禰豆子', '雷門炭十郎', '雷門炭吉', '繼國緣壹', '胡蝶忍', '下弦之陸', '佛堂鬼', '佩狼', '十二鬼月', '半天狗', '妓夫太郎', '姑獲鳥', '愈史郎', '手鬼', '朱紗丸', '沼鬼', '猗窩座', '玉霞', '珠世', '病葉', '矢琶羽', '稻玉擔岳', '雷門炭治郎', '雷門禰豆子', '重磨', '累', '茶茶丸', '蜘蛛鬼「哥哥」', '蜘蛛鬼「姊姊」', '蜘蛛鬼「母親」', '蜘蛛鬼「父親」', '蜘蛛鬼家族', '血鬼術', '赤影彤', '轆轤', '釜鴿', '零余子', '響凱', '鬼王', '鬼舞辻無慘', '魔夢', '鳴女', '黑死牟', '不死川富綱', '伊黑小芭內', '富岡義勇', '悲鳴嶼行冥', '時透無一郎', '甘露寺蜜璃', '雷門炭治郎', '繼國緣壹', '黑死牟', '不死川玄彌', '嘴平伊之助', '尾崎', '我妻善逸', '村田', '栗花落香奈乎', '雷門炭治郎', '桑野匡近', '骰子牛前輩', '產屋敷天音', '產屋敷家', '產屋敷彼方', '產屋敷日夏', '產屋敷玖伊那', '產屋敷耀哉', '產屋敷輝利哉', '鬼舞辻無慘', '嘴平伊之助', '我妻善逸', '雷門炭治郎', '雷門禰豆子', '中原澄', '寺內清', '栗花落香奈乎', '神崎葵', '胡蝶三姐妹', '胡蝶忍', '胡蝶香奈惠', '蝶屋敷', '高田菜穗', '蝶屋敷', '嘴平伊之助', '我妻善逸', '栗花落香奈乎', '雷門炭十郎', '雷門炭治郎', '鎧免', '不死川玄彌', '愈史郎', '雷門禰豆子', '悲鳴嶼行冥', '繼國緣壹', '半天狗', '半天狗分身列表', '可樂', '哀絕', '恨之鬼', '憎珀天', '積怒', '空喜', '鳴女', '上弦鬼', '半天狗', '墮姬', '妓夫太郎', '猗窩座', '玉霞', '稻玉擔岳', '重磨', '鳴女', '黑死牟', '下弦之陸', '下弦鬼', '佩狼', '姑獲鳥', '累', '轆轤', '釜鴿', '零余子', '響凱', '魔夢', '佩狼', '半天狗', '半天狗分身列表', '可樂', '哀絕', '恨之鬼', '憎珀天', '積怒', '空喜', '月之呼吸', '雷之呼吸', '岩之呼吸', '水之呼吸', '炎之呼吸', '霞之呼吸', '風之呼吸', '悲鳴嶼行冥', '甘露寺蜜璃', '雷門炭十郎', '雷門炭吉', '雷門炭治郎', '繼國緣壹', '黑死牟', '富岡義勇', '村田', '真狹', '雷門炭治郎', '鎧免', '鱗瀧左近次', '混合呼吸', '花之呼吸', '蛇之呼吸', '蟲之呼吸', '煉獄杏壽郎', '煉獄禰壽郎', '甘露寺蜜璃', '戀之呼吸', '嘴平伊之助', '岩之呼吸', '戀之呼吸', '月之呼吸', '蛇之呼吸', '蟲之呼吸', '音之呼吸', '栗花落香奈乎', '胡蝶香奈惠', '伊黑小芭內', '胡蝶忍', '我妻善逸', '桑島慈悟郎', '稻玉擔岳', '混合呼吸', '音之呼吸', '時透無一郎', '宇髄天元', '不死川富綱', '桑野匡近', '伊黑小芭內', '悲鳴嶼行冥', '時透無一郎', '桑島慈悟郎', '煉獄杏壽郎', '甘露寺蜜璃', '胡蝶忍', '胡蝶香奈惠', '不死川富綱', '下野弘', '不死川弘', '富岡義勇', '悲鳴嶼行冥', '時透無一郎', '柱', '栗花落香奈乎', '煉獄杏壽郎', '甘露寺蜜璃', '雷門禰豆子', '雷門炭十郎', '雷門炭吉', '繼國緣壹', '胡蝶忍', '雷門炭治郎', '雷門禰豆子', '半天狗', '沼鬼', '珠世', '重磨', '鬼舞辻無慘', '黑死牟', '稻玉擔岳', '黑死牟', '月之呼吸', '無慘的血鬼術招式列表', '珠世的血鬼術列表', '血鬼術', '佛堂鬼', '蜘蛛鬼大姐', '鬼舞辻無慘', '茶茶丸', '歷任鬼王列表', '雷門炭治郎', '鬼舞辻無慘', '蟲之呼吸']

半结构化实体关系抽取和转化

以"我妻善逸"人物实体为例，其数据页面如下图所示。可以看到"我妻善逸"的主要信息在右侧以知识卡片的形式进行呈现。

我妻善逸

“如果只会蠢之型的我是废物,那不会蠢之型的你就是垃圾。”

—— 我妻善逸

あがつまぜんいつ
我妻善逸 (我妻善逸 Agatsuma Zenitsu?), 是漫画作品《鬼灭之刃》及其衍生作品中的主要配角。
是故事开始时鬼杀队的一员,也是与炭治郎同期通过最终试炼的队士。

目录

1. 外貌

2. 个性

3. 经历

4. 关系

5. 能力

5.1. 雷之呼吸

6. 你知道吗

7. 参考资料

8. 外部链接

9. 导览

外貌

造型为制服上披着三角形图案的黄色羽织,留着金色中短发的圆眉少年。

我妻善逸

动画 漫画 舞台

日文名あがつまぜんいつ我妻善逸

罗马字Agatsuma Zen'itsu

首次登场漫画第6话、动画第4集

资讯

物种人类

性别男

年龄16岁

身高164.5cm

体重58kg

出生日期9月3日

出生地东京府牛込区

我们通过字典的键值来对右侧的知识卡片部分进行半结构化实体关系抽取和转化。最终得到爬取到的三元组关系,以"我妻善逸"人物实体为例,如下图所示:

```
In [101]: relations
Out[101]: [('我妻善逸', '罗马字', 'Agatsuma Zen'itsu'),
('我妻善逸', '首次登场', '漫画第6话、动画第4集'),
('我妻善逸', '物种', '人类'),
('我妻善逸', '性别', '男'),
('我妻善逸', '年龄', '16岁'),
('我妻善逸', '体重', '58kg'),
('我妻善逸', '髮色', '黑髮→金髮'),
('我妻善逸', '瞳色', '金瞳'),
('我妻善逸', '所属', '鬼杀队'),
('我妻善逸', '職業', '鬼杀队士'),
('我妻善逸', '親屬', '竈門禰豆子 (妻子) 竈門炭治郎 (大舅) 栗花落香奈乎 (内嫂)'),
('我妻善逸', '狀態', '生存'),
('我妻善逸', '能力', '雷之呼吸'),
('我妻善逸', '武器', '黃色日輪刀'),
('我妻善逸', '興趣', '花札、雙六'),
('我妻善逸', '喜歡的食物', '甜食、高級料理 (例如鰻魚)'),
('我妻善逸', '日本聲優', '下野紘'),
('我妻善逸', '臺灣聲優', '江志倫'),
('我妻善逸', '香港配音員', '杜景煜'),
('我妻善逸', '中國配音員', '李蘭陵'),
('我妻善逸', '美國配音員', 'Aleks Le'),
('我妻善逸', '舞台演員', '植田主輔')]
```

中文繁体简体转换

由于直接爬取得到的实体与关系信息均为繁体,我们利用zhconv库进行繁简体转换,具体代码如下:

```
1 def trans_to_zhans_entites(use_entities):
2     trans_entities=[]
3     for i in use_entities:
4         trans=zhconv.convert(i, 'zh-hans')
5         trans_entities.append(trans)
6     return(trans_entities)
```

数据清洗

对于爬取得到的实体对象信息与实体关系信息，我们按照以下步骤进行数据清洗，具体过程如下。

1. 去除无效信息

在所爬取到的三元组关系中，我们仅保留职业、亲属、能力等知识图谱构建所需的信息，因此对于其他键值都进行跳过不提取。具体代码如下

```
1 #remove unwanted info
2 wanted_relations = []
3 for d in relations:
4     if d[1] in ['日文名', '状态', '罗马字', '体重', '出生日期', '年龄', '身高', '死因', '物种', '动画', '漫画', '首次登场', '日本声优', '舞台演员', '香港配音员', '中国配音员', '美国配音员', '台湾声优', '喜欢的食物', '兴趣', '使用者', '出生地', '伙伴', '日文']: continue
5     wanted_relations.append(d)
```

2. 校正错误格式

从网站上直接爬取的关系数据，往往存在着一些格式错误。比如在亲属这个属性中，实体“不死川实弥”的“父亲”、“母亲”、“弟弟”等关系均在括号内连于实体名称后，如下图所示。

```
['不死川实弥',
 '亲属',
 '不死川泰悟（父亲）不死川志津（母亲）不死川玄弥（大弟）不死川琴（弟弟）不死川弘（弟弟）不死川就也（弟弟）不死川寿美（妹妹）不死川贞子（妹妹）'],
```

对此我们通过以下代码来将每个关系分别转化为三元组。

```
1 import re
2 clean_relations=[]
3 for d in wanted_relations:
4     if d[1] not in ['亲属']:
5         clean_relations.append([d[0],d[2],d[1]])
6         continue
7     else:
8         if d[2][-1] == ')':
9             relation = d[2][:-1].strip().split(' ')
10            # print(relation)
11            for relate in relation:
12                if ' (' in relate:
13                    r = relate.strip().split(' (')
14                    # print(r)
15                    r[0]=clean_r(r[0])
16                    clean_relations.append([d[0], r[0], r[1]])
17            elif ',' in relate:
18                r = relate.strip().split(' (')
19                e = r[0].split(',')
20                for i in e:
21                    clean_relations.append([d[0], i, r[1]])
```

最终将该亲属关系整理为如下格式的三元组。

['不死川实弥', '不死川恭悟', '父亲'],
['不死川实弥', '不死川志津', '母亲'],
['不死川实弥', '不死川玄弥', '大弟'],
['不死川实弥', '不死川琴', '弟弟'],
['不死川实弥', '不死川弘', '弟弟'],
['不死川实弥', '不死川就也', '弟弟'],
['不死川实弥', '不死川寿美', '妹妹'],
['不死川实弥', '不死川贞子', '妹妹'],

知识图谱的文本知识抽取

实体识别

- 模型：使用的是双向长短期记忆网络（BiLSTM）
- 标注集：采用BMOES，由于最后面向的实体主要为人名，因此主要标注了人名，标注格式如下

- | | | |
|----|---|--------|
| 1 | 炭 | B-NAME |
| 2 | 治 | M-NAME |
| 3 | 郎 | E-NAME |
| 4 | 前 | O |
| 5 | 往 | O |
| 6 | 花 | O |
| 7 | 街 | O |
| 8 | 执 | O |
| 9 | 行 | O |
| 10 | 任 | O |
| 11 | 务 | O |

- 数据集：起初是指采用了BosonNLP_NER_6C实体语料，如下图所示，对每个句子中的人名等实体进行了标注（人名 `person_name`），只需要将标注转化为上方的BMOES即可。

- 

- 但是最终训练出来的结果不太理想，如下所示，“炭治郎”人名根本无法识别，而“花街”可以识别出来，究其原因实体识别的对象是日本动漫鬼灭之刃中的人名，因此是日文名的中文翻译，所以训练的人名全为中国人名，是无法识别出来的。

- 1 ['炭', '治', '郎', '前', '往', '花', '街', '执', '行', '任', '务', '。']
- 2 ['O', 'O', 'O', 'O', 'O', 'B-ORG', 'E-ORG', 'O', 'E-ORG', 'O', 'O']

- 因此，为了能够更好地识别日文名，我们先从gitee上找到一个[日文名中文语料库](#)，该语料库中的日文名是繁体字的，所以先转换为简体，然后替换BosonNLP_NER_6C实体语料库中的人名，形成我们制作的实体语料数据，然后进行训练。

- | | |
|----|-------|
| 1 | 阿保藩 |
| 2 | 阿保剛 |
| 3 | 阿保今雄 |
| 4 | 阿保順子 |
| 5 | 阿保雅行 |
| 6 | 阿保郁夫 |
| 7 | 阿倍比羅夫 |
| 8 | 阿倍古美奈 |
| 9 | 阿倍吉俊 |
| 10 | 阿倍毛人 |
| 11 | 阿倍首名 |
| 12 | 阿倍為任 |
| 13 | 阿倍野橋 |
| 14 | 阿倍正之 |
| 15 | 阿比留瑠比 |

-

- ```
Epoch 30, step/total_step: 300/366 81.97% Loss:0.0027
Epoch 30, step/total_step: 305/366 83.33% Loss:0.0036
Epoch 30, step/total_step: 310/366 84.70% Loss:0.0011
Epoch 30, step/total_step: 315/366 86.07% Loss:0.0069
Epoch 30, step/total_step: 320/366 87.43% Loss:0.0037
Epoch 30, step/total_step: 325/366 88.80% Loss:0.0088
Epoch 30, step/total_step: 330/366 90.16% Loss:0.0010
Epoch 30, step/total_step: 335/366 91.53% Loss:0.0011
Epoch 30, step/total_step: 340/366 92.90% Loss:0.0007
Epoch 30, step/total_step: 345/366 94.26% Loss:0.0008
Epoch 30, step/total_step: 350/366 95.63% Loss:0.0071
Epoch 30, step/total_step: 355/366 96.99% Loss:0.0004
Epoch 30, step/total_step: 360/366 98.36% Loss:0.0001
Epoch 30, step/total_step: 365/366 99.73% Loss:0.0003
Epoch 30, Val Loss:0.2211
```

- 最后经过训练进行测试得到如下结果

- ```

1 % 原本的语料库训练出来的结果
2 ["灶", "门", "炭", "治", "郎", "前", "往", "花", "街", "执", "行", "任",
   "务", "。"]
3 ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'B-ORG', 'E-ORG', 'O', 'E-ORG', 'O',
   'O']
4
5 % 替换了日文名的语料库训练出来的结果
6 ["灶", "门", "炭", "治", "郎", "前", "往", "花", "街", "执", "行", "任",
   "务", "。"]
7 ['O', 'O', 'M-NAME', 'M-NAME', 'E-NAME', 'O', 'O', 'B-ORG', 'E-ORG',
   'O', 'O', 'O', 'O']
8
9 % 将训练日文名加入测试样本的结果
10 ["阿", "保", "刚", "前", "往", "花", "街", "执", "行", "任", "务", "。"]
11 ['B-NAME', 'M-NAME', 'E-NAME', 'O', 'O', 'B-ORG', 'E-ORG', 'O', 'O',
    'O', 'O']

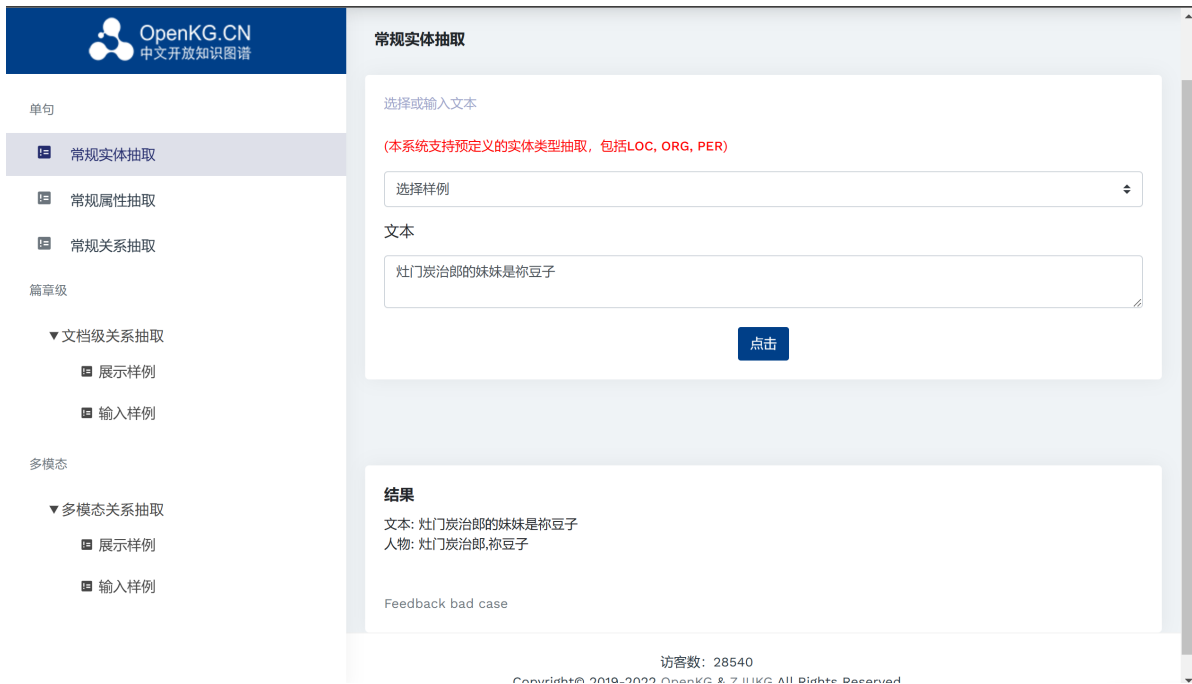
```

```
[deepke] zt@zwj-PowerEdge-T640: /projects/entity_recognition$ python test_model_old.py
['灾', '治', '师', '前', '往', '花', '街', '执', '行', '任', '务', '.']
/home/zt/.conda/envs/deepke/lib/python3.8/site-packages/torch-1.10.0-py3.8-linux-x86_64.egg/torch.nn/modules/rnn.py:694: UserWarning: RNN module weights are
not part of single contiguous chunk of memory. This means they need to be compacted at every call, possibly greatly increasing memory usage. To compact w
eights again call flatten_parameters(). (Triggered internally at ../aten/src/ATen/native/cudnn/RNN.cpp:925.)
    result = _VF.lstm(input, batch_sizes, hx, self.flat_weights, self.bias,
[["'0'", "'0'", "'0'", "'0'", "B-ORG", "E-ORG", "'0'", "E-ORG", "'0'", "'0'"]
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$ vim test_model.py
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$ python test_model.py
['灶', '门', '灾', '治', '前', '往', '调', '查', '无', '限', '列', '车', '事', '件', '遇', '上', '炎', '柱', '炼', '狱', '杏', '寿', '郎', '.', '.']
/home/zt/.conda/envs/deepke/lib/python3.8/site-packages/torch-1.10.0-py3.8-linux-x86_64.egg/torch.nn/modules/rnn.py:694: UserWarning: RNN module weights are
not part of single contiguous chunk of memory. This means they need to be compacted at every call, possibly greatly increasing memory usage. To compact w
eights again call flatten_parameters(). (Triggered internally at ../aten/src/ATen/native/cudnn/RNN.cpp:925.)
    result = _VF.lstm(input, batch_sizes, hx, self.flat_weights, self.bias,
[["'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'M-NAME'", "'E-NAME'"]]
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$ vim test_model.py
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$ python test_model.py
['灶', '门', '灾', '治', '前', '往', '富', '网', '义', '勇', '的', '介', '绍', '下', '拜', '麟', '流', '为', '师', '.']/home/zt/.conda/envs/deepke/lib/pyt
hon3.8/site-packages/torch-1.10.0-py3.8-linux-x86_64.egg/torch.nn/modules/rnn.py:694: UserWarning: RNN module weights are not part of single contiguous ch
unk of memory. This means they need to be compacted at every call, possibly greatly increasing memory usage. To compact weights again call flatten_parameter
s(). (Triggered internally at ../aten/src/ATen/native/cudnn/RNN.cpp:925.)
    result = _VF.lstm(input, batch_sizes, hx, self.flat_weights, self.bias,
[["'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "M-NAME", "E-NAME", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'"]
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$ vim test_model.py
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$ python test_model.py
['灾', '治', '富', '网', '义', '勇', '的', '介', '绍', '下', '拜', '麟', '流', '为', '师', '.']
/home/zt/.conda/envs/deepke/lib/python3.8/site-packages/torch-1.10.0-py3.8-linux-x86_64.egg/torch.nn/modules/rnn.py:694: UserWarning: RNN module weights are
not part of single contiguous chunk of memory. This means they need to be compacted at every call, possibly greatly increasing memory usage. To compact w
eights again call flatten_parameters(). (Triggered internally at ../aten/src/ATen/native/cudnn/RNN.cpp:925.)
    result = _VF.lstm(input, batch_sizes, hx, self.flat_weights, self.bias,
[["'B-NAME'", "'M-NAME'", "'E-NAME'", "'0'", "'0'", "'M-NAME'", "'M-NAME'", "'E-NAME'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'", "'0'"]
(deepke) zt@zwj-PowerEdge-T640: /projects/entity_recognition$
```

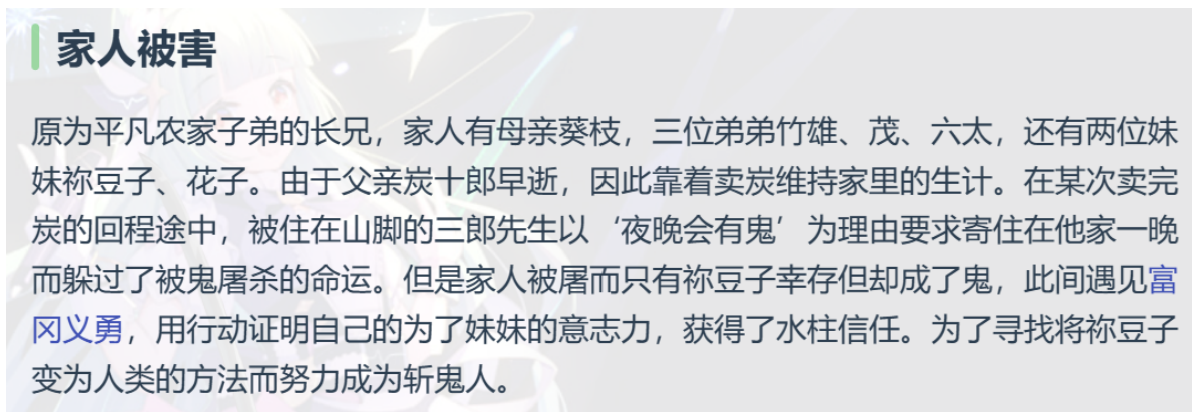
- 可以看出比起原有的语料库，进行日文名替换的实体语料库训练出来的结果能够比较好地识别出人名
的结尾符号，但是开始符号却比较难识别出来。用训练的日文名替换的结果进行对比，可以看出我们训练的模型泛化效果还是不太理想。

纯文本知识抽取

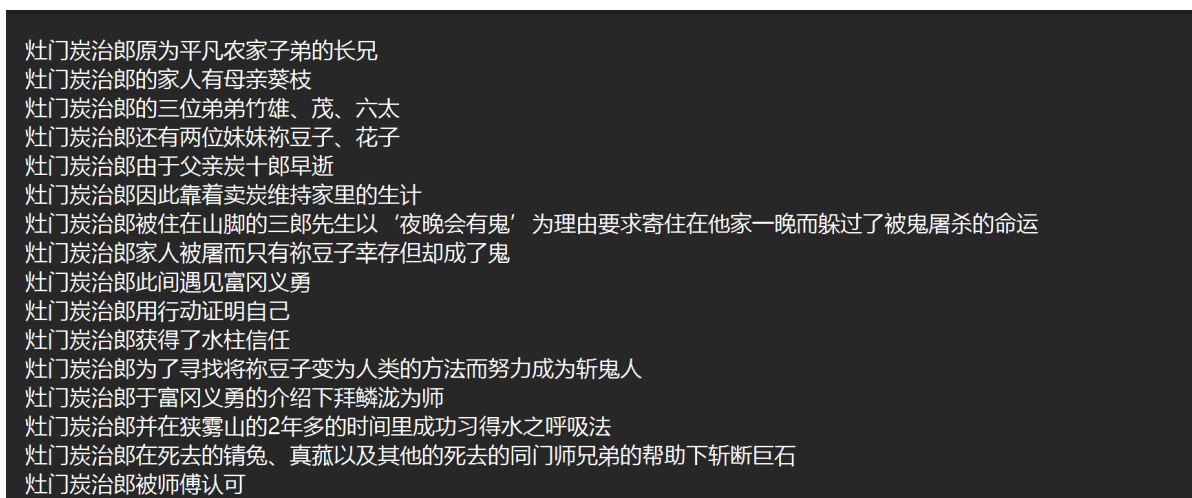
相比而言，DeepKE进行实体识别和纯文本知识抽取的效果比较好，因此我们最后放弃了使用自己训练的模型，取而代之的是采用DeepKE的模型进行纯文本的知识抽取。



对于抽取的样本，我们先做了如下的预处理。



- 如上图所示，数据源是百科词条，存在：没有主语、同一实体别名较多等问题。因此我们首先对爬取到的文本按照，、；、。、！、？进行分句。
- 然后给每句话的句首按照人物实体添加完整的主语。
- 最后我们人工删去一些又明显问题的短句，形成了抽取的样本。如下图所示。

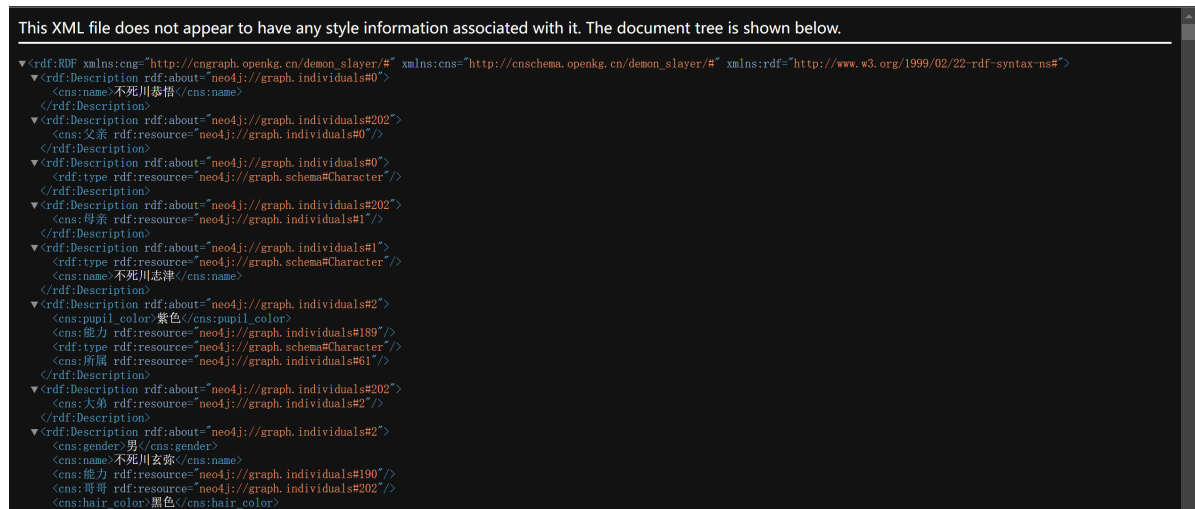


- 最后通过DeepKE进行知识抽取，并将抽取结果中同一人物实体的别名进行规范合并，最后与先前的半结构化三元组合并成为最终的三元组。

三元组转换为RDF

我们采取rdf/xml格式转换了知识图谱的三元组，并使用

`http://cnschema.openkg.cn/demon_slayer/#` 作为URI默认的命名空间，形成了如下所示的rdf/xml格式的文件。



知识图谱的应用（可视化和知识问答）

将知识图谱导入neo4j数据库

使用docker安装neo4j

```
1 | docker pull neo4j:4.4.12
```

运行容器并建立映射，设置用户名和密码

```
1 | sudo docker run -d --name neo4j-4.4.12-container \
2 | -p 7474:7474 -p 7687:7687 \
3 | -v /home/touch/neo4j4.4.12/data:/data \
4 | -v /home/touch/neo4j4.4.12/logs:/logs \
5 | -v /home/touch/neo4j4.4.12/conf:/var/lib/neo4j/conf \
6 | -v /home/touch/neo4j4.4.12/import:/var/lib/neo4j/import \
7 | -v /home/touch/neo4j4.4.12/plugins:/var/lib/neo4j/plugins \
8 | --env NEO4J_AUTH=neo4j/123456 neo4j:4.4.12
```

使用neo4j库，配合neosemantics和APOC插件，编写python脚本将获得的数据导入到neo4j数据库中

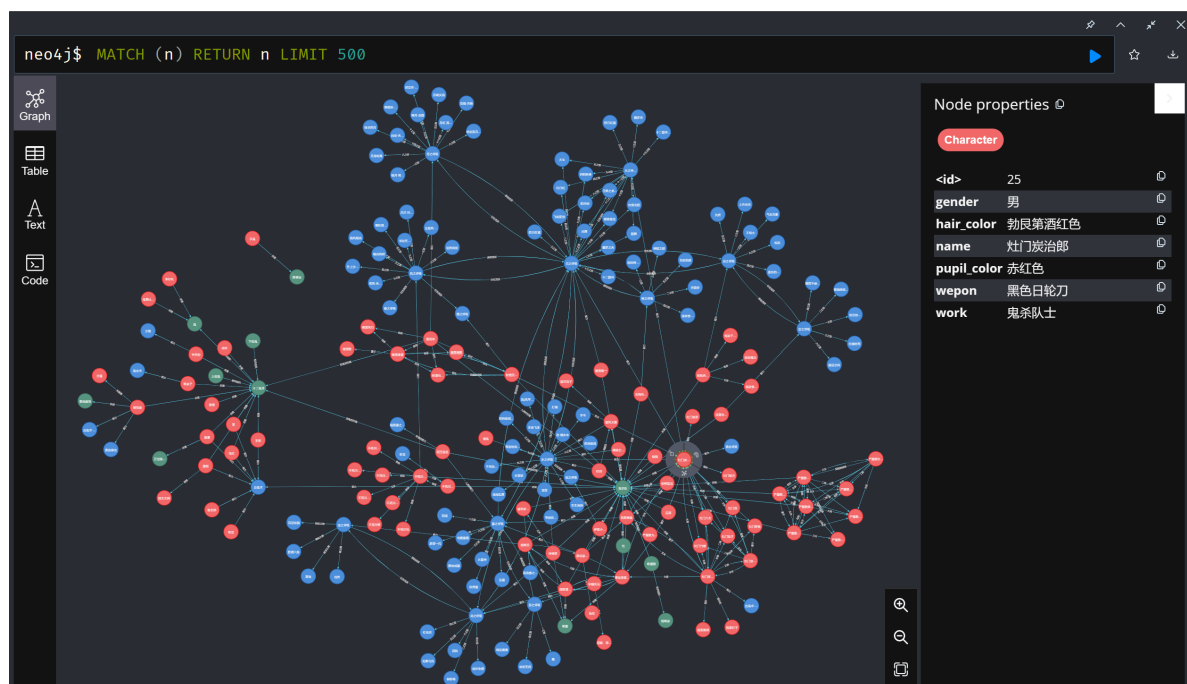
```
1 | python ./src/create_graph.py
```

知识图谱可视化与查询

导入了neo4j数据库之后，可以使用网页连接设置的端口，输入cypher语句进行数据库操作，如显示所有节点：

```
1 | MATCH (n) RETURN (n)
```

结果如下，可以看出每个节点实体按照实体类型进行标签分类，红色为人物实体（如灶门炭治郎、富冈义勇），蓝色为技能实体（如日之呼吸、水之呼吸），绿色为组织实体（如鬼杀队、十二鬼月），通过有向线段进行实体关系的连接，并且每个实体都可以点击查看其属性（性别、发色、瞳色、武器、职业）。



知识问答

使用上面建立的知识图谱，利用cypher语句的查询，可以进行知识问答，如询问："性别为男性的有谁？"提取出关键词"性别"和"男性"，作为输入即可获得答案，如下：

```
1 result = session.execute_read(relation_name_y_query_name_x, relation="性别",  
  name_y="男")  
2 for x in result:  
3     print(x)
```

部分结果如下

```
不死川实弥  
不死川玄弥  
产屋敷耀哉  
产屋敷辉利哉  
伊黑小芭内  
佛堂鬼  
半天狗  
后藤  
嘴平伊之助  
宇髓天元  
富冈义勇  
悲鸣屿行冥  
愈史郎  
我妻善逸  
手鬼  
时透无一郎  
村田  
桑岛慈悟郎  
灶门炭十郎  
灶门炭治郎
```

