# 基于人类疾病本体词库的智能搜索

## 1 项目 Requirements

操作系统：Ubuntu 16.04

### 1.1 本体词库

选择人类疾病本体词库作为该项目的本体词库（官网：<http://www.disease-ontology.org/>），其所有资源文件均可在Github（https://github.com/DiseaseOntology/HumanDiseaseOntology）仓库中下载，包括本体词库OWL文件HumanDO.owl，及子集测试文件GOLD.owl。

### 1.2 MySQL数据库

本体词库加载到内存中是一件非常耗时的工作（大概需要15分钟左右），如果每一次搜索都要加载一次本体词库，显然是不合理的，所以要将本体词库（HumanDO.owl）持久化，存储到MySQL 数据库中，也便于后期检索。

安装方式（默认最新版本）：

root@ubuntu:~# sudo apt-get install mysql-server

root@ubuntu:~# apt install mysql-client

root@ubuntu:~# apt install libmysqlclient-dev

期间会弹出设置root账户的密码框，输入两次相同密码。

登录mysql:

root@ubuntu:~# mysql –u root –p

创建名为owl的数据库：

mysql> CREATE DATABASE IF NOT EXISTS owl DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;

### 1.3 Python 编程语言

Ubuntu16.04 自带 python 2.7的解释器 因此不需要额外安装。

### 1.4 NLTK第三方库

用于文本处理，包括分词、词性标注、分块等。

安装方式：

root@ubuntu:~# sudo pip install -U nltk

nltk数据包、第三方库的下载：

root@ubuntu:~# python

Python 2.7.12 (default, Nov 19 2016, 06:48:10)

[GCC 5.4.0 20160609] on linux2

Type "help", "copyright", "credits" or "license" for more information.

>>>import nltk

>>> nltk.download()

### 1.5 ontospy 第三方库

用来加载并处理OWL文件，即把OWL/RDF格式文件加载到内存中，解析各个ontology的属性值及其之间的关系。

安装方式：

root@ubuntu:~# sudo pip install ontospy

默认自动安装该库的依赖。

### 1.6 MySQLdb第三方库

操作数据库的接口，主要用于ontonlogy持久化及其查询。

安装方式：

root@ubuntu:~# sudo pip install mysql-python

默认自动安装该库的依赖。

## 2 项目文件结构

├── disease\_ontology.py # 定义DiseaseOntology类及其方法

├── gold.owl # 解析OWL时用到的测试文件

├── HumanDO.owl # 人体疾病本体词库文件

├── keyword\_retrieval.py # 关键词检索模块的程序

├── main.py # 主模块的程序

├── owl\_parser.py # 解析人体疾病本体词库文件并将其持久化

├── perform.py # 两种方法性能比较

├── process\_1.png # 参考的流程图之一

├── process\_2.jpg # 参考的流程图之一

├── project.docx # 项目主要信息介绍

├── test.sql # 部分测试用的SQL语句

└── text\_processor.py # 用户输入文本的处理，包括预处理、语义分割等

## 3 项目模块

主要包含4个模块：数据持久化模块、文本处理模块、关键词检索模块、主程序处理

### 3.1数据持久化模块

源文件：owl\_parser.py

基本流程：

1）加载HumanDO.owl文件；

2）获取节点深度为0的ontology节点(根节点)列表；

3）迭代获取每个节点的基本属性，并写入数据库。基本属性有：

do\_id：本体词的id

do\_uri：本体词的URI

do\_name：本体词的名称

do\_definition：本体词的定义

do\_links：本体词的相关链接（暂时未用，便于以后扩展）

do\_synonyms：本体词的同义词

do\_parents\_uri：本体词父类的URI

do\_ancestors\_uri：本体词祖先的URI

do\_deprecated：本体词的是否过期

4）递归执行2）、3）两步，直到所有节点都写入到数据库中。

前台执行数据持久 化的命令：

python owl\_parser.py

后台执行数据持久化的命令：

python owl\_parser.py > output.txt 2> error.txt &

执行时间较长，建议后台执行。

### 3.2文本处理模块

源文件：text\_processor.py

基本流程：

1）将用户输入的内容进行分词；

2）对单词进行词性标注，便于下一步的语义分割和逻辑推理

3）对已标注的单词，但不包括停止词、动词及其变形、形容词及其变形、数词及其变形，进行词性矫正，包括两类：

(1) 该词出现在布朗语料库中，则用布朗预料库标记的词性；

(2) 该词没有出现在布朗语料库中，则用自定义的标注规则标注词性。主要是考虑到部分词尾在人体疾病本体词库中出现的频率较高，因此有指定词尾的单词可标识相应的词性。常见的词尾有oma \ ome \ tis \ sis \ mia \ nia \ cer等。

4）构建语法树。构造正则语法器分析句法，并生成语法树。

5）获得NP-块。选择NP子树，将其叶节点组合成字符串，从而得到NP-块。

6）去重得到关键词

执行文本处理模块的命令：

python text\_processor.py

### 3.3 关键词检索模块

源文件：keyword\_retrieval.py

基本流程：

给定关键字word，

1）在所有本体词的name中模糊搜索含有关键字的记录；

2）读取记录的所有字段，并生成ontology对象。其中对于parents、ancestors、children进行扩展查找；

3）在所有本体词的synonym中模糊搜索含有关键字的记录；

4）读取记录的所有字段，并生成ontology对象。修改name属性，使其含有指定关键词word，对parents、ancestors、children进行扩展查找；

5）合并2）、4）的结果即为改关键字的检索结果。

执行文本处理模块的命令：

python keyword\_retrieval.py

### 3.4 主程序处理模块

源文件：main.py

基本流程：

1）获取用户输入的文本；

2）创建文本处理器，提取文本的关键词

3）创建关键词检索器，根据关键词检索获取其相关信息。

4）格式化输出相关信息。

执行主程序模块的命令：

python main.py

## 4 性能比较说明

传统搜索方法：基于用户输入的内容的关键字单纯地进行搜索。

执行性能比较程序的命令：

python perform.py

测试用例1：

用户输入："The health authorities now know how to deal with the diarrhea"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 结果集 | | | 时间（秒） |
| 总数 | 有效数 | 查准率 |
| 基于本体的搜搜 | 21 | 18 | 85.71% | 3.986325 |
| 采用传统方法的搜索 | 28 | 18 | 64.29% | 6.569376 |

测试用例2：

用户输入：Incidence rates of pulmonary tuberculosis are high among new immigrants.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 结果集 | | | 时间（秒） |
| 总数 | 有效数 | 查准率 |
| 基于本体的搜搜 | 2 | 2 | 100% | 2.901673 |
| 采用传统方法的搜索 | 209 | 2 | 0.95% | 24.85385 |