Assignment 4 - Ontology Learning

Zefan Liang

School of Electrical Engineering and Computer Science

University of Ottawa

10th April, 2019

CSI 5180 Topics in AI - Ontologies and Semantic Web

Caroline Barrière

Table of content

1 Introduction

This report contains the whole content of Ontology Learning‘s requirements.

2 Corpus Building

First step is to find some knowledge good documents for the topic.

My topic is related to a famous German soccer club, Borussia Dortmund. First of all, I chose some latest news of Borussia Dortmund from the British Broadcasting Corporation(BBC) . I chose it because it is the world's oldest national broadcasting organisation and the largest broadcaster in the world. The news in BBC is very authoritative and reliable. Every news in BBC about Borussia Dortmund is quite short, so I chose 10 latest news here. I make them from docu1.txt to docu10.txt.

After that I chose the Wikipedia of Borussia Dortmund. The reason why I chose Wikipedia because it is the largest and most popular general reference work on the World Wide Web. It will provide comprehensive information of Borussia Dortmund. I put the content in Borussia docu11.txt.

Then I need to construct a corpus from those documents. Here I use Python to construct them in a large txt file while every document occupies one line. This file is named result.txt.

The Python code is as follow:

```
######################
import re
import os
```

```
import sys

import codecs

import shutil

def merge_file():

path = "documents\\docu"

resName = "result.txt"

if os.path.exists(resName):

os.remove(resName)

result = codecs.open(resName, 'w', 'utf-8')

num = 1

while num <= 11:

name = num

fileName = path + str(name) + ".txt"

source = open(fileName, 'r')

line = source.readline()

line = line.strip('\n')

line = line.strip('\r')

while line!="":

#line = unicode(line, "utf-8")

line = line.replace('\n',' ')

line = line.replace('\r',' ')

result.write(line+ ' ')

line = source.readline()

else:

print ('End file: ' + str(num))
```

```
result.write('\r\n')

source.close()

num = num + 1

else:

print ('End All')

result.close()

if __name__ == '__main__':

merge_file()

######################
```

3 Term Extraction

a) The second step is to find the terms in the corpus. Test a couple term extractor.

First I chose Termine here. It is an online terminology extractor tool which is quite easy to use.

The TerMine demonstrator intergrates C-Value multiword term extraction and AcroMine acronym recognition. C-value is a domain-independent method for automatic term recognition (ATR) which considers the following four characteristics:

[1] the occurrence frequency of the candidate term

[2] the frequency of the candidate term as part of other longer candidate terms

[3] the number of these longer candidate terms

[4] the length of the candidate term

The source of Termine is http://www.nactem.ac.uk/software/termine/

After upload my corpus on Termine, I got a table there:

Rank	Term	Score			
1	borussia dortmund	74.742424	11	jadon sancho	7
2	bayern munich	24.8125	11	head coach	7
3	champions league	24.444445	11	manchester city	7
4	young player	16	14	marco reus	6
5	borussia dortmund gmbh	14.066542	14	bundesliga title	6
6	uefa champions league	10.777745	14	world cup	6
7	robert lewandowski	8.333333	17	lucien favre	5.965517
8	co kgaa	8	18	michael zorc	5.947369
8	premier league	8	19	uefa cup	5.666667
10	signal iduna park	7.924812	20	real madrid	5
			20	english club	5

Figure 1 - the result of Termine

Next I tried another term exaction tool Translated Labs which is also an online free Term Extractor.

It uses Poisson statistics, the Maximum Likelihood Estimation and Inverse Document Frequency between the frequency of words in a given document and a generic corpus of 100 million words per language. It uses a probabilistic part of speech tagger to take into account the probability that a particular sequence could be a term. It creates n-grams of words by minimizing the relative entropy.

The source of Translated Labs is https://labs.translated.net/terminology-extraction/

And I also got a table as follow:

#	Extracted term	Score
1	bayern munich	65%
2	borussia dortmund	55%
3	ruthless first-half display	53%
4	lewandowski lewandowski	52%
5	achraf hakimi	52%
6	jurgen klopp	52%
7	borussia monchengladbach	52%
8	pin-point header	52%
9	matchday squad	52%
10	first-team manager	51%

Figure 2 - the result of Translated Labs

Compared to Termine which gives 950 exacted terms, Translated Labs just shows 20 exacted terms. So I think it is not suitable to do statistics analysis.

b) Now Comparing the result of the term extractors with my list of instances, classes and properties that I had defined. What is similar, what is missing?

For Termine, I chose the terms whose score are over 2.5. So here is 86 exacted terms to analyse and compare.

By comparison, I made a table as follow:

	Exactly the same	similar	Missing
Classes	8 (+ 4)	12 (+ 9)	3
instances	8 (+ 5)		
properties	2 (+ 1)		5
No relation	29		

Table 1 - the statics of 86 exacted terms

There are 8 terms of classes from my ontology and the exacted terms are exactly the same where the brackets mean it repeated appeared for 4 times. There are 12 classes in my ontology are similar to the exacted terms(9 are repeated), while my ontology lacks 3 classes by compared with the exacted terms.

The instances and properties can be explained in the same way. What's more, there are 29 exacted terms which has no relations with my ontology.

By analyzing, I found some features:

[1]. For the classes, there are many similar classes like "uefa cup" which is closed to my defined class in my ontology "UEFA_Champions_League".

[2]. Some of the classes (exactly the same or similar) repeated for many times like German_Cup.

[3]. The missing classes include some classes like "supervisory board". It is a department of Borussia Dortmund.

[4]. The instances of my ontology do not have some similar one and missing one here.

[5]. Many properties are missing here, including "is club caption", "is top player".

[6]. There are a large amount of exacted terms which do not have some relationships. Some of them are the other soccer clubs and soccer players in Europe while some of them are hard to explain like "German top flight".

So we can make some conclusions:

[1]. The definitions of classes are missing by analyzing the exacted terms, I should add these classes.

[2]. The instances are defined exactly because many of them are the soccer player's names of Borussia Dortmund.

[3]. Many properties are missing, I should add these properties.

[4]. There are many terms which have no relationships with my ontology, so the chosen documents may be not very representative.

4 Taxonomy induction

(a) Pattern-Based Approach:

For Pattern-Based approach, I simply manually pre-program 4 patterns and search for them by Python. I use Python regular expression here.

The Python code is as follow:

```
########
import re
resName = open("result.txt")
lines = resName.read()
p1 = re.compile(r'([a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+) is a ([a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+) .*',re.IGNORECASE)
p2 = re.compile(r'([a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+) including ([a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+ [a-zA-Z0-9]+) .*',re.IGNORECASE)
p3 = re.compile(r'([A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+) and related ([A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+) .*',re.IGNORECASE)
p4 = re.compile(r'([A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+) in ([A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+ [A-Za-z0-9]+) .*',re.IGNORECASE)
m1 = p1.findall(lines)
m2 = p2.findall(lines)
```

```
m3 = p3.findall(lines)


if m1 or m2 or m3 or m4:
   print (m1)
   print (m2)
   print (m3)
########
```

I defined 3 patterns, incuding:

[1]. (A is a B)

This is the result:

```
[('or simply Dortmund', 'German sports club'), ('Holland the pressure', 'bit less and')]
```

It shows Borussia Dortmund is a German sports club. So it can also show that Borussia Dortmund should be a subclass of sports club.

```
[('The founders', 'Franz and Paul Braun'), ('and friends', 'at the game and'), ('The allegations', 'made in documents published')]
```

It shows Borussia Dortmund is founded by Franz and Paul Braun, so they should be the instances of founder.

[2]. (A including B)

```
[('Senior manager', 'Watzke and Zorc')]
```

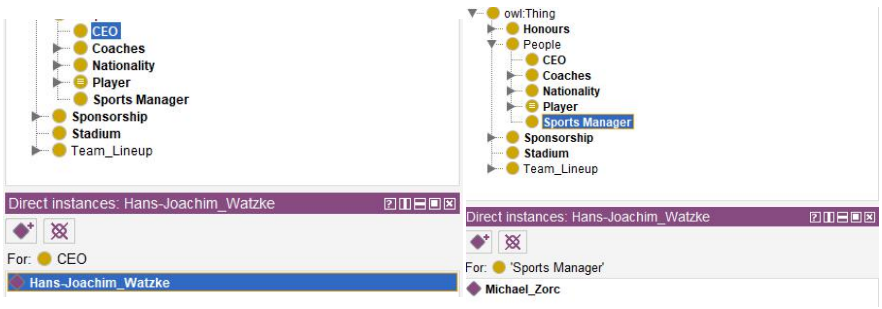

Figure 3 - Examples of my ontology

It shows Watzke and Zorc are the senior manager of the club, so they are both the instances of people. And this result matches with what I have defined in my ontology.

[3]. (A and related B)

This is the result:

There is no this pattern in my corpus.

By analyzing the results, I find Pattern-Based Approach is a really good taxonomy induction and some of the result can really match with my ontology. And I can also add some relationships between different classes in my ontology which I missed when I built up my ontology. But the disadvantage is that only a few results show the relationships with different instances and different classes. So I need to use other functions to extract useful data from the process of Pattern-Based approach to optimize my design.

5 References.

[1] Wikipedia contributors. "Borussia Dortmund." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 8 Apr. 2019. Web. 11 Apr. 2019. Retrieved from https://en.wikipedia.org/wiki/Borussia_Dortmund

[2] Flora-2 official website. Retrieved from https://www.bbc.com/news

[3] Translated Labs official website. Retrieved from https://labs.translated.net/terminology-extraction/

[4] Termine official website. Retrieved from

http://www.nactem.ac.uk/software/termine/