Laboratory Project

Introduction to Natural Language Processing – Master in Artificial Intelligence

Authors	Pablo Martínez
	Aleix Solanes
Course	2015/16
Date	January '16

Contents

Description of the content

In the following document, we describe the approach we made to the resolution of the problem proposed at GeoCLEF2007. This is a project proposed for the INLP subject from the Master in Artificial Intelligence in the UPC. The code that accompanies this document, is structured as follows: “data” folder (contains the xml and the grammars), “lib” (contains the functions of 3rd party used in this work), “scripts” (contains the main program, as well as the output xml with results).

Introduction

In this project, we were asked to face the problem from the GeoCLEF2007. Geographical Information Retrieval (GIR) is an augmentation of Information Retrieval by adding geographic information. GeoCLEF2007 was dedified to GIR, concretely by identifying a few properties of a set of queries.

We were provided by a set of 100 queries that were supposed to extract the information in it, and give as a result an xml with the queries and its properties correctly filled. The main properties that has to be analyzed are WHAT, WHAT-TYPE, GEO-RELATION and WHERE. Each of this properties contains an important information about the query.

In order to test, develop, improve and obtain results, a dataset “GC_Test_Solved_100_2.xml” is considered (it is included under the folder “data”).

Acknowledgements

In order to develop some functionalities, some code has been used from a previous task which each group had to develop a task that could be used afterwards.

In order to access the DBPedia, we have used the approach proposed by Amin, Kazancli and Kazimi. The file name is “dbpedia.py” under the “lib” folder.

To read an xml we have used the code provided by Brando and Roig, with a file “xmlToDict_functions”.

To save to an XML we have used the code from Castillo, Garza and Chen. The file is “saveIntoXML_functions”.

Finally, to obtain the results we have used the scorer from Duato, Fabregues and Selva. The name of the file is “scorer”.

Searching for attributes

The main basic algorithm to find the properties is as follows:

1- Load the XML

2- Iterate the queries in order to obtain the properties

   a. Obtain the GEO-RELATION with a rule system and by this use, determine a first approach to the WHAT and WHERE properties.

   b. Obtain the WHAT-TYPE, based on grammars

    c. If the place found before is not correct, look for a location inside the whole query.
        i. If Not, assign None to the properties
        ii. If Yes, recalculate the WHAT, the WHAT-TYPE.

3- Save the result to a XML

The process to identify each property is explained in the next sections more in detail.

GEO-RELATION

This property refers to the relation between the WHERE and the WHAT.

In order to find the geo-relation, a function called "getRelation(query)" is coded. This function looks for appearances of relation words (IN, TO, ON THE, OF, NEAR,WITHIN...) by using regular expressions, and returns the estimated relation.

First, looks for a simple relation (IN, TO, OF...) then, looks for a longer one like for example (WITHIN), and finally a compass relation (in the south of, in the north of, southwest to). The relation that keeps is the longer one. After that, the first separation is done between the WHAT and the WHERE.

WHERE AND WHAT

This property refers to the place that the query talks to. The where is determined two times. The first one is after a geo-relation is found, but this is only an assumption. After the geo-relation is found, we check whether the supposed WHERE is really a location by checking the DBPedia. If it is not a location, then, we look for a place in the whole query, and if it is found, we restructure the WHERE and the WHAT.

WHAT-TYPE

This property specifies which is the category of information that the user is looking for. There are three possibilities: Map type, Yellow Page type and Information Type.

In order to select a type, we use two grammars that checks if a query is a MAP type or a YELLOW PAGE type. It is a bunch of words that are normally related to each type. If these grammars do not let us classify the word we consider that it is a INFORMATION type query.

If in the sentence there is no WHAT we consider the type as MAP.

LOCAL

A query is LOCAL if it contains at least a WHERE component.

Results

In order to test the results, we have used the scorer provided from the first task by another group. This scorer is based on the evaluation that appears in the paper of the GeoCLEF2007, but it has some ambiguous descriptions of what they consider to be the Recall, the Precision and the F1-Score, so in order to show the results in such a way that let us evaluate how the performance of the algorithm was, we decided to provide both approaches, the one shown in the paper, and the formal formula for this terms.

First of all, an evaluation of the properties correctly classified has been done, in order to determine which of them are the most conflict.

By using the data set in “GC_Test_Solved_100_2.xml”.

Term	Correctly classified (%)
WHAT	79
WHAT-TYPE	60
WHERE	72
LOCAL	74
GEO-RELATION	94

Table 1. Precision of each property individually

It is easy to see in the last table, that the property that shows more conflicts is the property WHAT-TYPE. It is a property, that is defined thanks to a grammar for the “Yellow Pages” type and another one for “Map Type”.

The more reliable property is the GEO-RELATION one, as it has a 94% of correct values. The remaining errors are related to errors with the WHERE.

Next, we will show the results obtained by the use of the scorer with the use of the formulas in the paper:

Paper formulas	Result
Precision	0.55
Recall	0.71
F1-Score	0.62

Table 2. Evaluation results of the paper formulas

And in the next table the results for the formal formula of the terms:

Normal formulas	Result
Precision	0.60
Recall	0.48
F1-Score	0.54

Table 3. Evaluation results of the normal formulas

As it can be seen in the previous tables, we obtain a precision of the 60% (or the 55% in the paper formulas) which is not the best result, but that can classify more than half of the queries correctly, which is a decent approach.

There are a few corrections that could bring us a better result in the precision. Those improvements will be commented in the following section.

How to execute the code?

The main file is stored under the folder “scripts” and it is called “main”. Just executing the file main should let the user to choose between two options:

1- Execute code with GC_Test_solved_100_2.xml (process and save xml

    a. This will load an xml, extract the queries and save them to an OutFile.xml in the same folder.

2- Show the results of the OutFile.xml

    a. This will compare the results by analyzing the OutFile.xml and the GC_Test_Solved_100_2.xml

If another set wants to be tested, it is as simple as change the variables in the main function called “queries” and “goldens”, which contain the path to the xml that will be analyzed.

Analysis of the errors and ideas to solve them

Most of the errors from this approach come from the option chosen in order to determine the type of a WHAT. We are using two files with common words that refer to Yellow Page types and Map types. The validity of these files determines the validity of the output, as if the WHAT-TYPE is not correctly classified, the system classifies it as a wrong option.

The solution to this problem can be either to improve this file by better and more words, or the solution that would improve the system is by training a dataset(the bigger the training dataset, the greater the solution) and with machine learning algorithms determine the type of the query.

There are also several errors related to the situation where there are two relation words (IN, OF,...) in the same word but separated by words inbetween. The solution to this is to improve the reliability of the detection of the WHERE. As the relation type refers to the relation between the WHAT and the WHERE.

The WHERE had many errors, that were related to the DBPedia accessor. It uses SPARQL in order to make queries to DBPedia. The problem is that there are several places not located in the DBPedia, and this is caused because of the type of location considered when searching for the place. For instance, a town can be categorized in DBPedia as a Settlement, a LocatedPlace, a City,... and this makes it difficult to return an all-in-one solution to find a place.
The solution it should be to detect all possible categories of a city, and find better ways to find streets, cities, regions,...

Another error that makes a few queries to fail are the grammatical errors, like “bismark state college”. If an error of this type occurs, the algorithm will probably fail, because no mispelling corrector is included.
The solution could be to introduce a corrector, that could return the most probable word by considering distances between words and a corpus.

As extra improvements, we could consider the use of several natural language processors like for example Freeling or Stanford that would help us in the process of natural language analysis.

Conclusion

The analysis of a text it is not an easy task, as it has natural language properties that are not easy to interprete. In this project, we found problems in the categorization of sentences, mispelling errors, and other problems that let us see the complexity of the natural language processing. So, we made a first approach, but by using more tools apart from a grammar, like Freeling and Stanford, or a more complex grammar can help a lot in the facing of a project like this one.

The most interesting part about this Challenge, is that it is possible to reach conclusions and information from a vast quantity of queries by the use of NLP, and we have seen the application of the concepts explained during the course, as well as seen the possibilities this field gives to the retrieval of information from several sources like a searcher or social networks.