QALD-Mini-Project

Lukas Blübaum
Nick Düsterhus
Ralf Keller

University of Paderborn

https://github.com/LukasBluebaum/QALD-Mini-Project

June 21, 2018

Overview

Task Description

- Building a Question Answering Engine that is able to get a F-measure of atleast 0.1
- Using Dbpedia as knowledge base

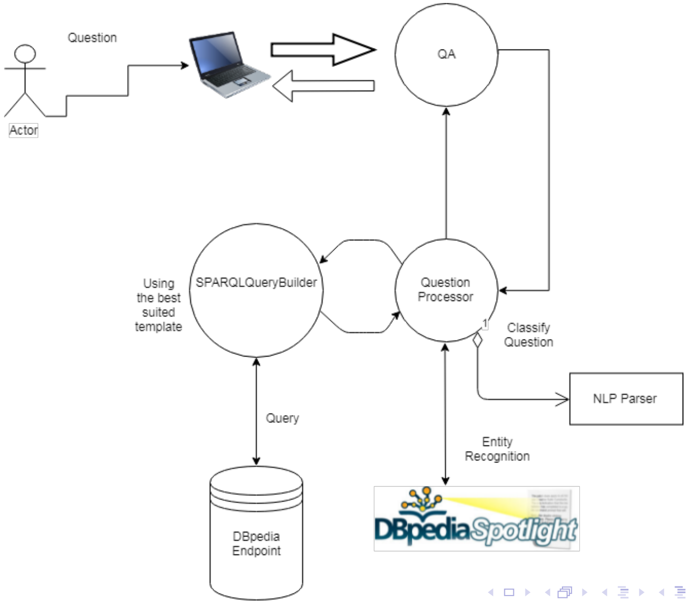
Given

- Library qa.annotation (finding entities, properties, classes) and qa.commons (load / store QAD Datasets)
- a wrapper to plug in GERBIL QA

Our Approach

- Template based
- Classify question types and apply natural language processing to get important keywords
- Find entities, classes, and properties that match the question
- Build SPARQL query templates for the most common types of questions

Simplified Procedure



Question Preprocessing

- Determining which method to build a sparql query should be applied
- Classifies questions by their starting word
  - e.g. When we can conclude from that, that the given result should be from the datatype date or year
  - distinguish between ASK and SELECT clause

Question Preprocessing

- Requesting Spotlight to get all named entities in the question
- Using the Stanford Core NLP to find keywords that give us information about the relations from the question
- findClasses/findProperties: using indexDBO_classes from the qa.annotation library on nouns, verbs and adjectives

Sparql Query Templates

Templates for different types of questions

- boolean questions such as: "Do Prince Harry and Prince William have the same parents?"
- list questions
- Who,Which,When,Where
- How (much/many)

- Further differentiation which template to use based on number of classes/entities and comparison words
- Request Dbpedia endpoint using Apache Jena Library

Sparql Query Templates
Example Query

Example (most basic query)

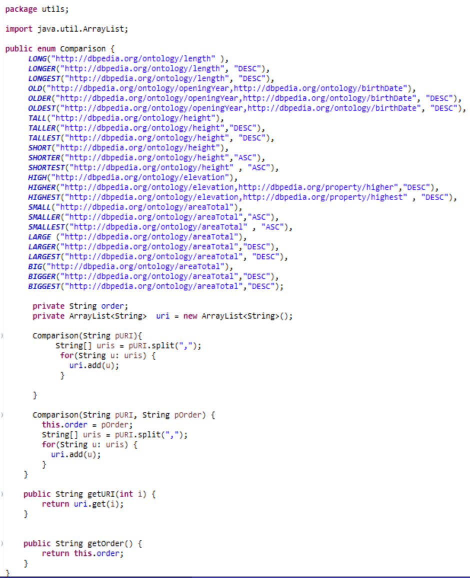
- Question: "Who was the doctoral supervisor of Albert Einstein"?
- One Entity: Albert Einstein
- doctoral supervisor maps to property dbo:doctoralAdvisor

Query ⇒ SELECT DISTINCT ?uri WHERE {
<http://dbpedia.org/resource/Albert_Einstein>
<http://dbpedia.org/ontology/doctoralAdvisor> ?uri . }

Sparql Query Templates

Comparison

- Predefined comparison enum for questions containg superlatives or comparatives



```
package utils;

import java.util.ArrayList;

public enum Comparison {
     LONG("http://dbpedia.org/ontology/length" ),
     LONGER("http://dbpedia.org/ontology/length", "DESC"),
     LONGEST("http://dbpedia.org/ontology/length", "DESC"),
     OLD("http://dbpedia.org/ontology/openingYear,http://dbpedia.org/ontology/birthDate"),
     OLDER("http://dbpedia.org/ontology/openingYear,http://dbpedia.org/ontology/birthDate", "DESC"),
     OLDEST("http://dbpedia.org/ontology/openingYear,http://dbpedia.org/ontology/birthDate", "DESC"),
     TALL("http://dbpedia.org/ontology/height"),
     TALLER("http://dbpedia.org/ontology/height","DESC"),
     TALLEST("http://dbpedia.org/ontology/height", "DESC"),
     SHORT("http://dbpedia.org/ontology/height"),
     SHORTER("http://dbpedia.org/ontology/height","ASC"),
     SHORTEST("http://dbpedia.org/ontology/height" , "ASC"),
     HIGH("http://dbpedia.org/ontology/elevation"),
     HIGHER("http://dbpedia.org/ontology/elevation,http://dbpedia.org/property/higher","DESC"),
     HIGHEST("http://dbpedia.org/ontology/elevation,http://dbpedia.org/property/highest" , "DESC"),
     SMALL("http://dbpedia.org/ontology/areaTotal"),
     SMALLER("http://dbpedia.org/ontology/areaTotal","ASC"),
     SMALLEST("http://dbpedia.org/ontology/areaTotal" , "ASC"),
     LARGE ("http://dbpedia.org/ontology/areaTotal"),
     LARGER("http://dbpedia.org/ontology/areaTotal","DESC"),
     LARGEST("http://dbpedia.org/ontology/areaTotal", "DESC"),
     BIG("http://dbpedia.org/ontology/areaTotal"),
     BIGGER("http://dbpedia.org/ontology/areaTotal","DESC"),
     BIGGEST("http://dbpedia.org/ontology/areaTotal","DESC");

      private String order;
      private ArrayList<String>  uri = new ArrayList<String>();

      Comparison(String pURI){
           String[] uris = pURI.split(",");
            for(String u: uris) {
              uri.add(u);
            }

      }

      Comparison(String pURI, String pOrder) {
        this.order = pOrder;
        String[] uris = pURI.split(",");
        for(String u: uris) {
          uri.add(u);
        }
    }

    public String getURI(int i) {
        return uri.get(i);
    }


    public String getOrder() {
        return this.order;
    }
}
```

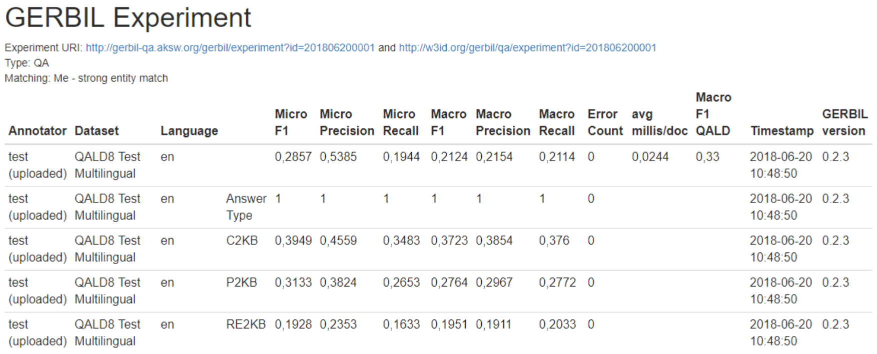
Sparql Query Templates

Example

- Example:

Benchmarking

QALD8 Test



GERBIL Experiment

Experiment URI: http://gerbil-qa.aksw.org/gerbil/experiment?id=201806200001 and http://w3id.org/gerbil/qa/experiment?id=201806200001
Type: QA
Matching: Me - strong entity match

Annotator	Dataset	Language		Micro F1	Micro Precision	Micro Recall	Macro F1	Macro Precision	Macro Recall	Error Count	avg millis/doc	Macro F1 QALD	Timestamp	GERBIL version
test (uploaded)	QALD8 Test Multilingual	en		0,2857	0,5385	0,1944	0,2124	0,2154	0,2114	0	0,0244	0,33	2018-06-20 10:48:50	0.2.3
test (uploaded)	QALD8 Test Multilingual	en	Answer Type	1	1	1	1	1	1	0			2018-06-20 10:48:50	0.2.3
test (uploaded)	QALD8 Test Multilingual	en	C2KB	0,3949	0,4559	0,3483	0,3723	0,3854	0,376	0			2018-06-20 10:48:50	0.2.3
test (uploaded)	QALD8 Test Multilingual	en	P2KB	0,3133	0,3824	0,2653	0,2764	0,2967	0,2772	0			2018-06-20 10:48:50	0.2.3
test (uploaded)	QALD8 Test Multilingual	en	RE2KB	0,1928	0,2353	0,1633	0,1951	0,1911	0,2033	0			2018-06-20 10:48:50	0.2.3

- http://gerbil-qa.aksw.org/gerbil/experiment?id=201806200001

Benchmarking
QALD8 Train

-