Group project in TDT4215
Web-intelligence

Alessio DE ANGELIS Martin HENGSTBERGER Simon STASTNY

April 20, 2013

Contents

1 Parsing ICD and ATC files

There are several ways to parse ICD and ATC files to extract information from them: you can consider those as normal XML files, use regular expressions and so on.

We have chosen a semantic web tool, Jena.[1] As we can read from the website,*Jena is an API for reading, processing and writing RDF data in XML, N-triples and Turtle formats; an ontology API for handling OWL and RDFS ontologies; a rule-based inference engine for reasoning with RDF and OWL data sources; stores to allow large numbers of RDF triples to be efficiently stored on disk.*

This tool may seems to be a little bit verbose to use, but let us handling in a very efficient way both the ontologies, the ICD that is an OWL format and the ATC that is a turtle, without the risk of losing information from them. Moreover, we can have easily access to the information about all the rdf graph: we can keep track for example of the hierarchies, the subclasses, the synonyms and so on. All this extra information may be used to improve our query engine, doing some refinements when the pure textual search is not accurate enough.

When we parse ICD we store in java beans the values of the following properties code_compacted (the code of the disease), label (description of the disease), underterm (extra information about the disease, not every instance has it), synonyms and subClassOf. When we parse ATC we store in java beans the values of the code, the label and subClassOf.

2 Tools for indexing and querying

For indexing and querying we are using an information retrieval approach, through a bag of words model. We are using Lucene, [2] (TODO: if you want you can describe a little bit about Lucene, reading from the website). At the beginning we were trying to create an our own information retrieval engine, but we came across several problems. First of all how to deal with Norwegian language, our background is very poor since we are all international students in the group, how to stem correctly words or divide properly the *aggregates* (four, five words merged all together in one). Lucene is a library made by experts and their implementation of the Norwegian analyzer (with the Porter stemming algorithm), even if maybe not as accurate as the English one, has been revealed very useful for our purposes. Moreover it natively supports the vector space model for a more accurate retrieval than boolean.

We keep two different indexes, one for the ATC documents and the second for the ICD documents. Since the indexing is very fast, we keep both the indexes in ram, for performance reasons during the queries (you don't need to access the disk that is usually time consuming). The ATC documents are composed by two fields: one for the atc_code, that just indexed without any further string processing; the second field, *label*, contains the description of the code and is

[1] http://jena.apache.org/

[2] http://lucene.apache.org/core/

the one used for extracting the documents. This field is indexed, stored, tokenized and calculates the term vector matrix.

The ICD documents are indexed according to three fields. The first one is for the property `icd:code_compacted`, to identify uniquely the codes. The second one is for the `label:` the field has the same properties as the ATC label. The third field contains all the terms of the label plus the other information for doing query expansion, such as the synonyms and the property `icd:underterm`. Adding those extra words let have a more accurate result when we submit the query to our engine. In fact at the moment we query the ICD index looking into the terms contained in the extra field.

3 Future work

Our system is accurate but in the future some refinements could be done to improve our query engine that we didn't manage to implement due to time constraints. For example an evaluation function that gives more importance to the hierarchy of the classes, for example if a code is subClassOf another code (E10, subclass and E10-14, the superclass in ICD for example). Since we use Jena to read the OWL ontologies, could be done through a simple graph traversal using suitable methods offered by the APIs, or using ad hoc SPARQL queries.

We could even taken better into account the synonyms: if two codes have the same synonyms, the may refer to the same disease.