Berri Corpus Manager: A corpus analysis tool using MongoDB technology

H. Sanjurjo-González
University of Deusto (Spain)

ACTRES

Deusto
Universidad de Deusto
University of Deusto

INTRODUCTION

A corpus can be stored into a machine using different data models. From raw file texts to relational databases, this data model determines performance and usability of the available methods for analysing a corpus. E.g. concordances or quantitative and qualitative stats. Most popular data configurations for corpus analysis software are CQP/CWB, SQL and dedicated implementations using different programming languages.

However, a popular technology such as NoSQL has not been widely used for developing any corpus analysis software. Maybe the most relevant work using MongoDB technologies in corpus analysis is Coole, Rayson and Marian[1], which demonstrates that NoSQL databases are viable solutions for analysing extreme scale corpus using database clustering.

OBJECTIVES

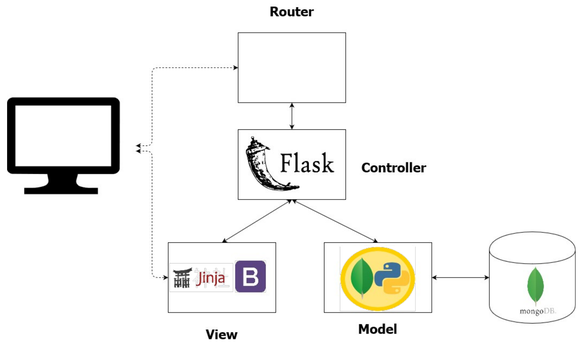
Our main objective is to answer if a NoSQL database can be a useful technology to develop corpus analysis software.

In order to achieve that, the following points are addressed:

- Getting to know MongoDB development environment.
- Proposing a software architecture to analyse monolingual and bilingual parallel corpora with grammatical annotation.
- Analysing suitability of the current MongoDB features for handling linguistic corpora.

[1] Coole M, Rayson P, Mariani J. Scaling out for extreme scale corpus data. In: Proceedings of the 2015 IEEE International Conference on Big Data (Big Data). 2015 29 Oct - Nov 1; Santa Clara, CA. IEEE Computer Society. p. 1643-1649.

SOFTWARE ARCHITECTURE

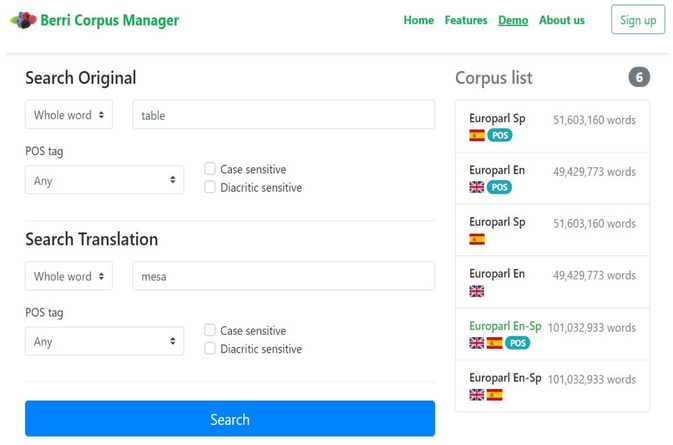


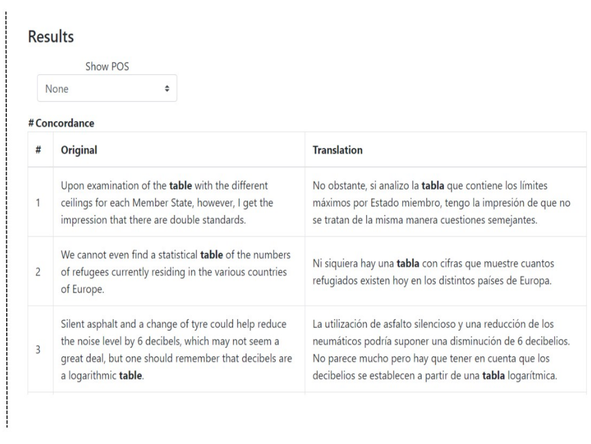
- Berri Corpus manager operates across a simple Model-View-Controller pattern. The software interacts with the corpus by means of PyMongo utilities.
- Python scripts are used to parse user queries into understandable MongoDB commands and execute natural language processing tasks.

```
{
    "corpus": "Europarl.English",
    "language": "English",
    "pos": 1,
    "parallel" : ["Europarl.Spanish"],
    "texts":[
        {
            "_id": 1,
            "sentence":
                "PROPN_Resumption  PROPN_of
                 PROPN_the PROPN_session SPACE_"
        }
    ]
}
```

- Schema-less architecture.
- Sentence alignment.
- Grammatical annotation.

EXPERIMENTS





RESULTS

- MongoDB offers a wide variety of drivers to develop in almost any programming language.
- Some tools and services like Compass, a GUI for the database, MongoDB shell, a Javascript interface, and Atlas, a cloud server, are very useful for a rapid development.
- Text indexes offer language specification, but they use stemming and remove stop words.
- Text indexes do not support partial word searches.
- Native count function returns the number of registers of the database that matches the query. This may be a problem depending on the employed corpus data modelling
- Aggregation pipeline offers a solid tool for grouping results from different documents of the database.

CONCLUSIONS

MongoDB has been successfully used as corpus storage. The proposed software allows users to query monolingual and bilingual parallel corpora with grammatical annotation.

Although it has been proved useful, it also entails several drawbacks: i) language specification of text indexes may not be useful in all circumstances as a consequence of the stemming, ii) text index does not support partial search, and iii), count of occurrences requires additional processing. However, it also provides an easy and powerful query language (e.g. aggregation pipeline), it has a very high performance when text indexes are used, and last, schema-less architecture may be very useful for corpus attribute specification as a consequence of its flexibility.

To sum up, if some of these problems are solved, specially partial search indexes, MongoDB may become an interesting option for corpus storage.