IST 2018/2019

Processamento e Recuperação de Informação

Course Project

1 Description and Goal

The goal of this project is to implement an Information Search & Extraction System for the analysis of political discourse.

Your system will have access to a large set of documents containing the electoral manifestos of several political parties from different countries in the world. Using this data, the system should be able to provide the following functionalities.

(a) *Ad hoc* search on the collection of documents.

Given a query, represented by set of keywords, the system should **return all the manifestos containing such keywords**, ordered according to their relevance to the query. In addition, the system should **show some statistics**, such as:

- For each party, how many manifestos are in the results returned;
- How many times each party mentions each keyword;

as any others you may find interesting.

(b) Classification of documents according to their political affiliation.

Given any natural language text, the system should be able to **predict which political party is most likely to have produced such text**. In addition, the system should also be able **test the effectiveness of its predictions** by showing the values of precision, recall, and F1 that it is able to achieve with the available data.

(c) Statistical analysis of the subjects mentioned.

You document should be able to **discover all named entities mentioned in the manifestos** (e.g. people, companies, countries, etc.) and **provide some statistics on their usage**. These statistics should enable us to answer questions such as:

- What are the most mentioned entities for each party?
- What are the most mentioned entities globally?
- Which party is mentioned more times by the other parties?
- How many times does any given party mention other parties?

as any others you may find interesting.

2 Implementation

Your Information Search & Extraction System should be implemented using the **Python programming language**, version 2.7 or version 3.

There is no need for a user interface. You may implement one, if you wish, but it will not be taken into consideration for grading your work. If you do implement an interface, you can use any programming language (e.g. Javascript, etc.).

If you do not implement an interface, all the required functionalities should be available by **executing python scripts on the command line**.

Your code should be **easily readable and properly documented**.

3 Resources

Libraries

To implement the system, you can develop any code necessary and/or use any of the existing libraries or services available online. Examples of such libraries are:

- the *scikit-learn*[1] machine learning library;
- the *NLTK*[2] natural language processing library;
- the *Whoosh* search engine[3];

among others.

Data

A set of political manifestos to use as data for your project will be provided. This set has been extracted from the *Manifesto Project* dataset[4] and contains the textual content of each manifesto, together with additional information, such as the corresponding political party, the language, the date, etc. You are free to use this information in any way you see fit.

You can explore the Manifesto Project website, or any other source of information, to improve your solution. However, **there is no need to download data from the Manifesto Project website**. A pre-processed version of the documents will be provided on the PRI course page.

[1] http://scikit-learn.org/

[2] https://www.nltk.org/

[3] https://whoosh.readthedocs.io/

[4] https://manifesto-project.wzb.eu/

4 Deliverable

Your project should be delivered in a compressed *zip* file named `pri_project_group_XX.zip`, where `XX` should be replaced by your group number.

The *zip* file should contain:

- The project code, within a directory named `code`;
- A file named `readme.txt`, containing a succinct set of instructions on how to run your code;
- A *pdf* file named `pri_project_group_XX_report.pdf` (where `XX` should be replaced by your group number), containing your project report.

Your report should be formatted according to the ACM Proceedings template[5]. The report should have **at most 6 pages** and should contain:

- A short abstract (1 paragraph) summarizing the report;
- A short introduction (1 paragraph) describing the problem[6];
- One section describing **how you implemented your search solution**;
- One section describing **how you implemented your classification solution**;
- One section describing **how you implemented your entity analysis solution**;
- One section describing **your results**. This section should contain:
  - For the search solution, examples of successful and unsuccessful queries, with an explanation of why that happens;
  - For the classification solution, a confusion matrix and the corresponding precision, recall, and F1 results, with an explanation of the values found;
  - For the entity analysis solution, examples of successfully and unsuccessfully extracted entities, with an explanation of why that happens.

Describe your solutions **succinctly, but with enough detail** that they may be replicated by someone else. When describing the solutions, you should always **justify the decisions taken**, i.e. why you chose a certain classifier and not another, how and why you chose certain parameter values and not others.

You should also **prepare an oral presentation of the project**, to be delivered on the last week of classes. The presentation should be set for 10 minutes and include quick explanation and demo of the system functionalities. Strict timing will be enforced.

[5] Use the `sigconf` format, available at `https://www.acm.org/publications/proceedings-template`.

[6] No philosophy, please — just state what you are trying to do.

5 Grading

The project will be graded according to the following criteria. Each item will be scored on a scale of [0–5]. The final grade is the sum of all scores, converted to a [0–20] scale.

(a) **Originality** (5 = the solution is completely original vs. 0 = only existing libraries where applied, using the default parameters);

(b) **Appropriateness** (5 = each problem has an appropriate solution vs. 0 = the same hack was used everywhere);

(c) **Correctness** (5 = the solution was correctly applied, results were correctly measured, etc. vs. 0 = wrong application of the solution, erroneous results, etc.);

(d) **Exploration** (5 = sufficient effort was made to obtain the best results vs. 0 = the first result obtained was considered good enough);

(e) **Oral presentation** (5 = clear and succinct oral presentation vs. 0 = confusing or incomplete oral presentation)

(f) **Report quality** (5 = good report and code presentation vs. 0 = poorly written report and unreadable code).

All projects must be **submitted electronically before the deadline**. Delays will not be accepted.

A **printed version of the reported should also be delivered** during the first class reserved for the oral presentations.