20-00-0546-iv Foundations of Language Technology

Homework 4
Accessing Text Corpora

26. November 2020

In case your submission consists of several files, compress these to a `zip`-file. Indicate clearly which submission corresponds to which question. Include comments in your program code to make it easier readable. It is very important that you submit your solution as a Jupyter Notebook file (.ipnyb).The deadline for the homework is **Thursday, 03.12.20 18:00 CET**.

4.1 Homework

Homework 4.1 *(6 points)*
Implement a language guesser, i.e. a function that takes a given text and outputs the language it thinks the text is written in. The function should base its decision on the frequency of individual characters in each language.

(a) *Implement a function* `build_language_models(languages,words)` *which takes a list of languages and a dictionary of words as arguments and returns a conditional frequency distribution where:*

- *the languages are the conditions*
- *the values are the lower case characters found in* `words[language]`

Call the function as follows:

```
languages = ['English', 'German_Deutsch', 'Spanish']
# build the language models
# udhr contains the Universal Declaration of Human Rights in over 300
    languages
language_base = dict((language, udhr.words(language + '-Latin1')) for language
     in languages)
language_model_cfd = build_language_models(languages, language_base)

# print the models for visual inspection (you always should have a look at the
     data :)
for language in languages:
    for key in list(language_model_cfd[language].keys())[:10]:
        print(language, key, "->", language_model_cfd[language].freq(key))
```

(b) *Develop an algorithm which calculates the overall score of a given text based on the frequency of characters accessible by* `language_model_cfd[language].freq(character)`*. Explain how the algorithm works.*

(c) *Implement a function* `guess_language(language_model_cfd,text)` *that returns the most likely language for a given text according to your algorithm from the previous sub task.*

(d) *Test your implementation with the following data:*

```
text1 = "Peter had been to the office before they arrived."
text2 = "Si terminas tu tarea, te dare un caramelo."
text3 = "Das ist ein schon recht langes deutsches Beispiel."

# guess the language by comparing the frequency distributions
print('guess for english text is', guess_language(language_model_cfd, text1))
print('guess for spanish text is', guess_language(language_model_cfd, text2))
print('guess for german text is', guess_language(language_model_cfd, text3))
```

If your function does not detect the correct language for at least two of these sentences, improve your algorithm.

(e) *Discuss, why English and German texts are difficult to distinguish with the given approach.*

Homework 4.2 *(4 points)*
The previous language guesser was based on the frequency of characters. Implement alternative language guesser based on the following lexical units:

(a) *tokens*
(b) *character bigrams*
(c) *token bigrams*

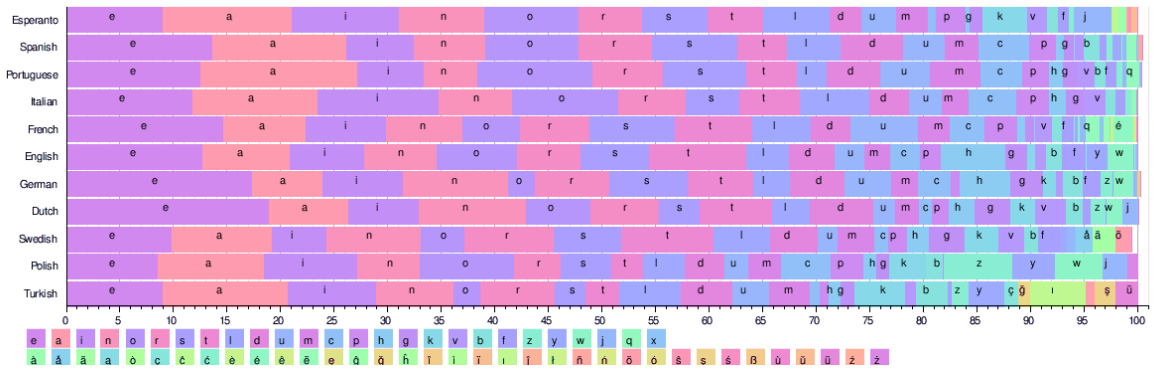

Figure 1: Character distribution for selected languages. (Source: Wikipedia)

(d) Discuss, which approach should work best theoretically. Is this reflected in the results?

Homework 4.3 *(This homework is not part of the bonus system. However, we recommend you to work it out. It will save you some time in the future.)*
Copy all functions implemented in the tasks and homeworks to one file and name it `UKP_Lib.py`*. You may easily access for examle the function* `word_freq` *of the previous tasks with the following statement:*[1]

```
1 from UKP_Lib import word_freq
```

You just implemented your first module. If you are familiar with another object oriented language, feel free to use classes and OO in the exercises. Make yourself familiar with syntax of OO-constructs in Python, e.g. consult http://docs.python.org/tutorial/classes.html

[1] See NLTK-book page chapter 2.4, page 59