EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Philosophische Fakultät
Seminar für Sprachwissenschaft

Data Science for Linguists

Session 5: Linguistic Preprocessing

Johannes Dellert

24 November, 2023

Table of Contents

Linguistic Preprocessing

- in the data science literature, **linguistic preprocessing** typically comprises
  - ▷ lemmatisation in order to reduce texts to standardised content words
  - ▷ topic modeling and sentiment detection in order to classify texts into categories
  - ▷ named entity recognition
  - ▷ information extraction from textual data (e.g. the places at which events occurred)
  - ▷ training simple sequence models or embeddings from corpora in order to generate simulated language data
- the literature is written from the perspective of people who are not primarily interested in answering questions about language, but about other domains, and NLP is mostly seen as a tool for extracting the actually relevant information from textual data
- as computational linguists, we are of course aware of much more sophisticated NLP tasks
- as general linguists, the questions which we want to answer based on our data are going to be focused on linguistic properties of the data, leading to a focus on annotations (part-of-speech tagging, dependency parses,)

Table of Contents

SpaCy

- SpaCy is one of the most popular libraries for NLP tasks in Python
- very large ecosystem, efficient implementation and streamlined interface
- markets itself as an industry standard (and it is difficult to argue with that)

SpaCy: General Design

- the configuration returns a pipeline method which you run once on a string of text, and the result is a very complex `Doc` object which contains annotations on various levels of description, organised into a hierarchy of containers:
  - sentences and other larger units are organised into `Span` objects which form a tree over `Token` objects
  - every `Token` contains information about the lemma, the syntactic category (UDPOS tags), the dependency label and the head (for UD dependency structures), as well as morphological features
- NB: for the actual string content, the relevant fields have an underscore suffix (`token.lemma` is just an integer ID, `token.lemma_` the actual string)
- it is a very large and very mature library, perusing the tutorials and the extensive documentation is very much worth your while!

Installing and Running a SpaCy model from Jupyter

- for `spacy download` in Jupyter, you need to use the `!` prefix for command execution:
  ```
  !python -m spacy download en_core_web_sm
  ```
- after installation, restart the kernel to make sure the installed package is found
- this is how you load model, and run the pipeline on some input for first experiments:
  ```
  import spacy
  nlp = spacy.load("en_core_web_sm")
  doc = nlp("This is a first example sentence.")
  ```

SpaCy: Self-Guided Tutorial

- if you have never worked with SpaCy, you might want to go through the following course in order to understand the basic concepts and usage patterns before tackling the assignment `https://course.spacy.io/en/chapter1`

Table of Contents

Assignment 5: Tasks (Part 1)

In this assignment, we are going to explore SpaCy's abilities by parsing a classic Mexican novel.
1) Install SpaCy and let it download the model `es_dep_news_trf`.
2) Create a new Jupyter notebook and import `spacy`. Load the contents of the file `azuela1920_los-de-abajo.txt` into a single string, and run the model on it. Extract the sentence spans, and store their contents into a text file.
3) Repeat the process, but this time, replace all newline characters in the text by a space before running it through the pipeline. Store the sentence spans in a file again, and investigate the differences. What do you notice? Use machine translation in case you don't understand enough. For the following tasks, we will stick to the version without newlines.
4) Familiarise yourself with the different properties of the `Token` object, and extract all pairs of full verbs (UD tag VERB) and their subjects (relation `nsubj`) in lemmatised form, storing them for later processing. In my solution, the first pair is `('decir','parte')` (which is actually wrong).
5) Extract the ten most common verbs occurring in your pairs, and the three most common subjects for the following verbs: *gritar* "to shout", *preguntar* "to ask", and *responder* "to answer". Who shouts the most, who asks the most questions, and who appears to answer them?

Assignment 5: Tasks (Part 2)

We are now going to analyse the difference between foreground and background events, using the distinction between the two tenses *indefinido* (which is said to be used for events which advance the storyline) and the *imperfecto* (used for background circumstances and events).

6) Repeat the extraction of verb-subject pairs for the verb forms in both tenses (they are distinguished in UD by the values `Past` and `Imp` of the feature `Tense`). Which verbs occur more than five times in this novel, but exclusively denote foreground or background events? Based on their translations, do the results make sense? What does the plot appear to be centered on?
7) Can you conclude from your data about the most frequent subjects who the protagonists of the plot are? Are there conspicuous differences in the rankings of people who are the agents in foreground and background events?

Preliminary Course Plan

1	**27/10**	**IPython and Jupyter**
2	**03/11**	**Introduction to NumPy**
3	**10/11**	**Pandas and Data Frames**
4	**17/11**	**Data Cleaning and Preparation**
5	**24/11**	**Linguistic Preprocessing**
6	01/12	Data Wrangling: Join, Combine, Reshape
7	08/12	Data Aggregation and Grouping
8	15/12	Visualisation with Seaborn
9	22/12	Modeling and Prediction
10	12/01	Classification
11	19/01	Clustering
12	26/01	Pattern Extraction and Density Estimation
13	02/02	Statistical Inference
14	09/02	Data Science Projects

Questions

Questions?

Comments?

Suggestions?